$\mathcal{H}^2$-Matrix-Arithmetik basierend auf Niedrigrangupdates

Dissertation
zur Erlangung des Doktorgrades der Naturwissenschaften
der Technischen Fakultät
der Christian-Albrechts-Universität zu Kiel

vorgelegt von
Knut Reimer

Kiel, 2015

Referent:	Prof. Dr. Steffen Börm
Korreferent:	Prof Dr. Malte Braack
Tag der mündlichen Prüfung:	17.06.2015
Zum Druck genehmigt:	17.06.2015

Zusammenfassung

Viele naturwissenschaftliche Phänomene lassen sich durch Differential- oder Integralgleichungen beschreiben. Bei typischen Diskretisierungen dieser Probleme entstehen große lineare Gleichungssysteme. Ein häufig verwendeter Ansatz zum Lösen dieser Gleichungssysteme sind iterative Verfahren, die schrittweise die näherungsweise Lösung verbessern. Dabei ist die Verbesserung, die in jedem Schritt erreicht wird, meistens von der Kondition der Systemmatrix abhängig. Deshalb werden zum effizienten Lösen solcher Probleme Vorkonditionierer verwendet, die die Inverse der Systemmatrix approximieren.

An dieser Stelle setzt diese Arbeit an, die einen neuen Ansatz für eine approximative Arithmetik für $\mathcal{H}^2$-Matrizen vorstellt, mit der robuste Vorkonditionierer konstruiert werden können. Die $\mathcal{H}^2$-Matrizen verwenden hierarchische Partitionen der Indexmengen der Matrix, eine hierarchische Blockpartition der Matrix und faktorisierte Niedrigrang-Darstellungen für gewisse Blöcke. Damit erreichen $\mathcal{H}^2$-Matrizen einen Speicheraufwand in $O(kn)$, wobei n die Matrix-Dimension und k den Rang der Niedrigrang-Darstellung beschreibt. Der Aufwand für die Matrix-Vektor-Multiplikation besitzt die gleiche Komplexität.

Die hierarchische Blockpartition erlaubt es, arithmetische Operationen wie die Inversion oder die Cholesky-Zerlegung auf die Matrix-Matrix-Multiplikation für $\mathcal{H}^2$-Matrizen zurückzuführen. Dabei werden die gleichen Ansätze verwendet wie für $\mathcal{H}$-Matrizen, die eine andere Niedrigrang-Darstellung nutzen und einen Speicheraufwand in $O(kn\log(n))$ haben. Die in dieser Arbeit präsentierte Matrix-Matrix-Multiplikation für $\mathcal{H}^2$-Matrizen stellt eine Adaption des $\mathcal{H}$-Algorithmus dar, der auf Niedrigrangupdates für $\mathcal{H}$-Matrizen basiert.

Die vorgestellten Niedrigrangupdates für $\mathcal{H}^2$-Matrizen verwenden eine Adaption eines Rekompression-Algorithmus für $\mathcal{H}^2$-Matrizen, die nur einen Teil der Matrix rekomprimiert. Wir untersuchen den Aufwand der Niedrigrangupdates und den entstehenden Approximationsfehler, der durch Fehlertoleranzen gesteuert werden kann. Mit Hilfe des Updates können wir die Arithmetik formulieren, deren Aufwand für Matrix-Produkt, Inversion und Cholesky-Zerlegung in $O(k^2 n\log(n))$ liegt. Damit sparen wir wie beim Speicheraufwand einen $\log(n)$-Faktor im Vergleich zu den $\mathcal{H}$-Matrizen ein. Zusätzlich stellen wir eine Variante für das Niedrigrangupdate und das Matrix-Produkt vor, die den Rechenaufwand für dreidimensionale Probleme deutlich reduziert.

Als numerische Experimente betrachten wir elliptische partielle Differentialgleichungen und Integralgleichungen. Die Probleme lösen wir mit dem cg-Verfahren, das wir mit Hilfe der approximierten Cholesky-Zerlegung im $\mathcal{H}^2$-Matrix-Format vorkonditionieren. Dabei zeigt sich der Vorkonditionierer robust bezüglich der betrachteten Geometrien und für die Differentialgleichung bezüglich springender Koeffizienten.

Abstract

Many processes in nature can be described by differential or integral equations. Typical discretisation schemes lead to large systems of linear equations. Often iterative solvers are used to improve the approximate solution of the system of equations. The improvement in every step depends on the condition of the system matrix. Thus preconditioners which approximate the inverse of the system matrix are applied to solve such problems efficiently.
This study presents a new approach for an approximate arithmetic for $\mathcal{H}^2$-matrices which allows the construction of robust preconditioners. The $\mathcal{H}^2$-matrices use a hierarchical partition of the index sets of the matrix, a block partition of the matrix and factorized low rank representations for certain blocks. Thereby the $\mathcal{H}^2$-matrices have a storage requirement in $O(kn)$ with matrix dimension n and rank k for the low rank representations. The matrix-vector multiplication has the same complexity.
The hierarchical block partition allows to reduce arithmetic operations like inversion or the Cholesky decomposition to the matrix-matrix multiplication for $\mathcal{H}^2$-matrices. The techniques for the reduction are the same as for $\mathcal{H}$-matrices that use another low rank representation and reach a storage requirement in $O(kn\log(n))$. The matrix-matrix multiplication for $\mathcal{H}^2$-matrices presented in this study is an adaption of the $\mathcal{H}$-matrix algorithms which are based on low rank updates for $\mathcal{H}$-matrices.
The presented low rank updates for $\mathcal{H}^2$-matrices are an adaption of a recompression algorithm for $\mathcal{H}^2$-matrices which recompresses only a part of the matrix. We will discuss the computing time and the resulting approximation error, which can be controlled by error tolerances. With the help of the updates we can define the arithmetic with computing time in $O(k^2n\log(n))$ for the matrix-matrix multiplikations, the inversions and the Cholesky decompositions. In comparison to $\mathcal{H}$-matrices, we save a factor $\log(n)$ for the storage requirement. In addition, we present a variant of the low rank update and the matrix-matrix multiplication which reduces the computing time for three dimensional problem essentially.
Numerical experiments are carried out for elliptic partial differential equations and integral equations. We solve the problems with the preconditioned cg-method and use approximated Cholesky decompositions in the $\mathcal{H}^2$-matrix format as preconditioners. In the experiments, the preconditioners are robust in respect of the considered geometries and for the differential equation relating to jumping coefficients.

Inhaltsverzeichnis

Symbolverzeichnis

$A_{|\mathbf{t}\times\mathbf{s}}$ ..... eingeschränkte Matrix (S. 13)
b^T ..... transponierter Block zu b (S. 53)
$C(V, \overline{V})$ ..... Basiswechsel von V zu $\overline{V}$ (S. 76)
$\operatorname{chil}(v)$ ..... Kinder des Knotens v (S. 37)
$\operatorname{col}(s) = \operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$ .... Blockspalte des Clusters s (S. 69)
$\overline{\operatorname{col}}(s) = \overline{\operatorname{col}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$ .... erweiterte Blockspalte des Clusters s (S. 69)
$\operatorname{depth}(V, \operatorname{chil})$ ..... Tiefe des Baums (V, chil) (S. 39)
$\operatorname{desc}(v)$ ..... Nachfahren des Knotens v (S. 40)
$\operatorname{diam}(\Omega_t)$ ..... Durchmesser der Menge Ω_t (S. 23)
$\operatorname{dist}(\Omega_t, \Omega_s)$ ..... Abstand der Mengen Ω_t und Ω_s (S. 23)
$\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ ..... $\mathcal{H}^2$-Matrix-Raum (S. 64)
$\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ .... $\mathcal{H}^2$-Matrix (S. 64)
$\mathcal{H}^2(H, R, \tilde{b})$ ..... erweiterte $\mathcal{H}^2$-Matrix (S. 129)
$\mathcal{I},\ \mathcal{J},\ \mathcal{K}$ ..... endliche Indexmengen
$I_{\mathbf{t}}$ ..... die Identität in $\mathbb{R}^{\mathbf{t}}$ bzw. $\mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$
$k_{\max}$ ..... maximaler Rang in den verwendeten Strukturen
$\operatorname{level}(v)$ ..... Level des Knotens v (S. 39)
$\mathcal{L}_{\mathcal{I}}$ ..... Blätter des Clusterbaums $\mathcal{T}_{\mathcal{I}}$ (S. 42)
$\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ ..... Blätter des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ (S. 48)
$\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ ..... zulässige Blätter des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ (S. 48)
$\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ ..... unzulässige Blätter des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ (S. 48)
$\mathbb{N}$ ..... Menge der natürlichen Zahlen mit der Null
$n_{\max}$ ..... maximale Mächtigkeit der Blattcluster
$\operatorname{par}(v)$ ..... Elter des Knotens v (S. 39)
$p_{\mathcal{I}}$ ..... Anzahl der Level des Clusterbaums $\mathcal{T}_{\mathcal{I}}$ (S. 42)
$\Pi_{\mathbf{t}}$ ..... Einschränkung auf die Menge $\mathbf{t}$ (S. 12)
$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},V,*}$ ..... lokale einseitige $\mathcal{H}^2$-Matrix-Projektion (S. 139)
$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},V,W}$ ..... lokale $\mathcal{H}^2$-Matrix-Projektion (S. 133)
$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,*}$ ..... einseitige $\mathcal{H}^2$-Matrix-Projektion (S. 82)
$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W}$ ..... $\mathcal{H}^2$-Matrix-Projektion (S. 81)
Π_Q ..... Projektion zur orthogonalen Matrix Q (S. 21)
$p_{\mathcal{I}\times\mathcal{J}}$ ..... Anzahl der Level des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$
$\operatorname{pred}(v)$ ..... Vorfahren des Knotens v (S. 40)
$\mathbb{R}$ ..... Menge der reellen Zahlen
$\mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$ ..... eingeschränkter Vektorraum (S. 13)

1 Einleitung

Viele naturwissenschaftliche Phänomene sind durch Differential- oder Integralgleichungen beschrieben. Um Vorhersagen zu treffen, müssen diese Gleichungen gelöst werden. Dies ist in vielen Fällen nicht analytisch möglich. Deshalb werden numerische Verfahren für die approximative Lösung der Gleichung benötigt.
Nach der Wahl der Problemmodellierung muss das Problem diskretisiert werden. Aus der Diskretisierung entsteht häufig ein lineares Gleichungssystem. Für eine zufriedenstellende Approximation der kontinuierlichen Gleichung muss die Diskretisierung hinreichend fein gewählt werden. Dies führt typischerweise zu einem großen Gleichungssystem, welches mit direkten Verfahren für vollbesetzte Matrizen wie der Gauß-Elimination aufgrund begrenzter Speicher- und Rechenkapazitäten nicht gelöst werden kann (zumindest nicht in vertretbarer Zeit).
Um diese Problematik zu bewältigen, gibt es verschiedene Ansätze. Einerseits werden auf der Seite der Diskretisierung vielfältige Methoden entwickelt, um die resultierenden Gleichungssysteme klein zu halten oder mit besonders einfacher Struktur zu versehen. Andererseits werden effiziente Lösungsverfahren für die Gleichungssysteme entwickelt, die auch große Systeme in annehmbarer Zeit lösen können. Dabei sind die beiden Bereiche nicht getrennt von einander zu sehen, sondern hängen wesentlich voneinander ab.
In dieser Arbeit beschäftigen wir uns mit der Entwicklung von schnellen Lösern für lineare Gleichungssysteme. Dazu verwenden wir die Techniken der hierarchischen Matrizen. Die in [31, 33] vorgestellten $\mathcal{H}$-Matrizen basieren auf einer hierarchischen Blockpartition der Systemmatrix und approximieren gewisse Matrixblöcke b durch eine Niedrigrangmatrix $A_b B_b^T$, bei der A_b und B_b nur wenige Spalten besitzen. Die Anzahl der Spalten bezeichnen wir auch als den lokalen Rang k. Dadurch reduzieren die $\mathcal{H}$-Matrizen den Speicheraufwand auf $\mathcal{O}(n \log(n) k)$ statt $\mathcal{O}(n^2)$ für vollbesetzte Matrizen im $\mathbb{R}^{n \times n}$.
Wir betrachten in dieser Arbeit die in [34] eingeführten $\mathcal{H}^2$-Matrizen, die in approximierten Matrixblöcken $b = \mathbf{t} \times \mathbf{s}$ eine Drei-Term-Darstellung $V_t S_b W_s^T$ verwenden, bei der V_t nur von den Zeilen und W_s nur von den Spalten abhängt. Um den $\log(n)$-Faktor der $\mathcal{H}$-Matrizen zu vermeiden, werden dabei zusätzliche Hierarchien in den Zeilen und Spalten für die Speicherung der Matrizen V_t und W_s genutzt. Damit wird ein Speicheraufwand in $\mathcal{O}(nk)$ erreicht.
Der lokale Rang kann dabei so gewählt werden, dass der entstehende Approximationsfehler in der Größenordnung des Diskretisierungsfehlers liegt. Allerdings muss der lokale Rang k dazu häufig in $\mathcal{O}(\log^\alpha(n))$ für ein $\alpha > 0$ gewählt werden, um sicherzustellen, dass sich der Approximationsfehler der hierarchischen Matrizen wie der Diskretisierungsfehler verhält. Beide Matrix-Formate ermöglichen die Matrix-Vektor-Multiplikation in der Komplexität des Speicheraufwand. Dies ermöglicht die Verwendung von iterativen Verfahren zum Lösen

der linearen Gleichungssysteme. Als Nächstes wollen wir kurz auf die beiden typischen Problemklassen eingehen, für die hierarchische Matrizen verwendet werden. Danach skizzieren wir kurz die Idee der in dieser Arbeit beschriebenen $\mathcal{H}^2$-Matrix-Arithmetik und geben eine kurze Gliederung der Arbeit. Zum Abschluss des Kapitels machen wir noch einige allgemeine Anmerkung zur Arbeit.

Integralgleichungen mit Randelementmethoden

Die eine Kategorie von Problemen, für die hierarchische Matrizen klassischerweise verwendet werden, sind mit Randelementmethoden (BEM) diskretisierte Integralgleichungen, bei denen der Integraloperator die Form

$$\mathcal{G}[u](x) \;=\; \int_\Gamma \kappa(x,y)u(y)dy$$

hat, wobei Γ der Rand eines Gebiets und κ eine Kernfunktion ist. Die Kernfunktion ist dabei vom jeweiligen Problem abhängig.
Typische Kernfunktionen κ haben bei $x = y$ eine Singularität und klingen inklusive der Ableitungen mit zunehmender Entfernung zur Singularität ab, ohne zu verschwinden (siehe [32, Abschnitt 4.2.4]). Bei der Diskretisierung mit BEM unter Verwendung stückweiser Polynome auf einem Gitter entsteht aufgrund der Nichtlokalität des Operators eine vollbesetzte Matrix. Um das Problem der vollbesetzten Matrizen zu umgehen, wurden eine Reihe von Verfahren entwickelt, die im Wesentlichen darauf basieren, die Matrix-Vektor-Multiplikation approximativ, aber schnell und ohne Speicherung der vollbesetzten Matrix auszuwerten und damit die effiziente Verwendung von iterativen Verfahren zu ermöglichen.
Eine Möglichkeit ist die Verwendung spezieller Basen. Diesen Ansatz verfolgen die Methoden der Wavelet-Basen (siehe [5, 36]). Diese setzen direkt bei der Diskretisierung an und verwenden spezielle Basisfunktionen, die dafür sorgen, dass ein großer Teil der Matrixeinträge sehr klein ist. Dadurch kann die Matrix durch eine schwachbesetzte Matrix approximiert werden, für die die Matrix-Vektor-Multiplikation schnell berechnet werden kann. Ein Problem dieser Methode besteht in der Konstruktion der Basen für komplizierte Geometrien (siehe [19]).
Ein anderer Ansatz ist die Verwendung der Standardbasen zusammen mit einer hierarchischen Blockpartition der Systemmatrix und einer Approximation der Kernfunktion in gewissen Matrixblöcken, die die Variablen separiert. Die klassischen Multipol-Methoden basieren auf speziellen problemabhängigen Reihenentwicklungen (siehe [29, 40]). Dies hat den Vorteil eines auf das Problem abgestimmten Verfahrens und den Nachteil, dass für jedes Problem eine Entwicklung gefunden und analysiert werden muss. Das Panel-Clustering basiert auf der Taylor-Entwicklung der Kernfunktion, um die Variablen zu separieren und so die Matrix in gewissen Teilblöcken durch niedrigen Rang zu approximieren (siehe [35, 41]). Dies hat den Vorteil, dass das Verfahren für eine große Klasse von Problemen verwendet werden kann, falls die Auswertungen der entsprechenden Ableitungen sichergestellt sind. Die Multilevel-Methode in [23] verwendet Tensor-Interpolation zur

Approximation der Kernfunktion und erfordert somit nur die Auswertung der Kernfunktion. Teilweise werden verschiedene Ansätze kombiniert, wie bei der in [24] vorgestellten Multipole-Methode, die zur Approximation der Kernfunktion Legendre-Polynome und Singulärwertzerlegung verwendet, um ein größeres Spektrum an Problemen behandeln zu können.
Die hierarchischen Matrizen können als algebraisches Gegenstück dieser Verfahren angesehen werden. Das Speichern als Matrix-Format hat den Vorteil, dass durch die Arithmetik, die bereits in [31] für $\mathcal{H}$-Matrizen in einer ersten Form beschrieben ist, Vorkonditionierer konstruiert werden können.

Elliptische Differentialgleichungen mit Finiten Elementen

Bei der Diskretisierung von partiellen Differentialgleichungen mit Finite Elemente Methoden (FEM) entstehen schwachbesetzte Matrizen, sodass der Speicheraufwand in üblichen Formaten lediglich $\mathcal{O}(n)$ beträgt. Die Speicherung der Matrizen ist also kein sinnvolles Anwendungsgebiet für hierarchische Matrizen. Allerdings sind die entstehenden Gleichungssysteme typischerweise schlecht konditioniert, so dass für das effiziente Lösen mit iterativen Verfahren ein Vorkonditionierer notwendig ist. Ein typischer Ansatz für die Konstruktion eines Vorkonditionierers ist eine effizient zu berechnende Approximation der Inversen. In [3] und [12] ist bewiesen, dass die Inverse der FE-Matrix einer elliptischen Differentialgleichung durch eine $\mathcal{H}$-Matrix bzw. $\mathcal{H}^2$-Matrix approximiert werden kann. Dies bildet die theoretische Basis, um mit Hilfe von hierarchischen Matrizen Vorkonditionierer für FEM-Matrizen zu berechnen. Dabei haben sich die hierarchischen Matrizen in der Praxis vor allem bei Problemen mit springenden Koeffizienten oder Anisotropien bewährt.
An dieser Stelle seien die hierarchischen semiseparablen Matrizen (HSS) erwähnt. Diese entsprechen $\mathcal{H}^2$-Matrizen zu einer simplen hierarchischen Blockstruktur, bei der nur die Diagonalblöcke unterteilt sind und alle Nebendiagonalblöcke durch eine Drei-Term-Darstellung dargestellt werden (siehe [18]). Durch die einfache Blockstruktur lassen sich vor allem Probleme behandeln, die von der Struktur her eindimensional sind. In [43] wird ein Löser für zweidimensionale partielle Differentialgleichungen vorgestellt, der eine Kombination aus der Multifrontal-Method, Nested Dissection und HSS-Matrizen verwendet. Dabei können die HSS-Matrizen genutzt werden, weil die Matrizen in den Separatoren aus der Nested Dissection von der Struktur her eindimensional sind.

Arithmetik basierend auf Niedrigrangupdates

Die Arithmetik für $\mathcal{H}$-Matrizen, die notwendig für die Konstruktion eines Vorkonditionierers ist, wird in [26] beschrieben. Wie bereits in [31] beschrieben, kann die Berechnung der Inversen für $\mathcal{H}$-Matrizen mit Hilfe der hierarchischen Blockstruktur und des Schur-Komplements auf die Berechnung der Inversen für vollbesetzte Matrizen und die $\mathcal{H}$-Matrix-Multiplikation in der Form $Z + XY$ reduziert werden. Auch die LR-Zerlegung

für $\mathcal{H}$-Matrizen lässt sich mit Hilfe des $\mathcal{H}$-Matrix-Produkts berechnen (siehe [1]). Weil die beiden Ansätze nicht auf der konkreten Darstellung der Niedrigrangmatrizen basieren, können sie direkt auf $\mathcal{H}^2$-Matrizen übertragen werden (siehe Kapitel 6).
Für die $\mathcal{H}^2$-Matrix-Multiplikation ist ein hoch effizienter Algorithmus für das $\mathcal{H}^2$-Matrix-Produkt in [8] beschrieben, für den allerdings die sogenannten Clusterbasen V_t und W_s der Drei-Term-Darstellung bekannt sein müssen. Weil diese typischerweise nicht a priori bekannt sind, benötigen wir einen Ansatz für das Matrix-Produkt, der die Clusterbasen adaptiv aufstellt. Adaptive Ansätze wie der in [12] haben bisher zu keinen befriedigenden Ergebnissen geführt. Diese Lücke soll mit dieser Arbeit geschlossen werden.
Dabei soll die $\mathcal{H}^2$-Matrix-Multiplikation durch Niedrigrangupdates realisiert werden. Die Idee hierfür stammt aus der Analyse der auf der hierarchischen Blockpartition basierenden $\mathcal{H}$-Matrix-Multiplikation, wie sie in [26] beschrieben ist. Falls alle Matrizen in eine 2×2-Blockstruktur unterteilt sind, erhalten wir für die Berechnung von $Z + XY$ die Gleichung

$$\begin{aligned}\begin{pmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{pmatrix} + \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix} \begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix} \\ = \begin{pmatrix} Z_{11} + X_{11}Y_{11} + X_{12}Y_{21} & Z_{12} + X_{11}Y_{12} + X_{12}Y_{22} \\ Z_{21} + X_{21}Y_{11} + X_{22}Y_{21} & Z_{22} + X_{21}Y_{12} + X_{22}Y_{22} \end{pmatrix}.\end{aligned}$$

Damit lässt sich die Multiplikation durch Rekursion berechnen. Falls X nicht weiter unterteilt ist, ist sie als Niedrigrangmatrix AB^T gespeichert oder die Matrix hat wenige Zeilen oder wenige Spalten und kann mit Hilfe der Identität als Niedrigrangmatrix $AB^T := XI^T$ bzw. $AB^T := I(X^T)^T$ dargestellt werden. Danach kann $\widehat{B} := Y^T B$ für die $\mathcal{H}$-Matrix Y und die vollbesetzte Matrix B spaltenweise durch die $\mathcal{H}$-Matrix-Vektor-Multiplikation aufgestellt werden. Dann gilt $XY = A\widehat{B}^T$ und es bleibt, die Niedrigrangmatrix zu Z zu addieren. Falls Y nicht weiter unterteilt ist, kann XY analog als Niedrigrangmatrix dargestellt werden. (Falls Z nicht weiter unterteilt ist, wird die Matrix erst künstlich unterteilt und später wieder zusammengefasst.)
Die Niedrigrangmatrix kann zur $\mathcal{H}$-Matrix addiert werden, indem sie aufgeteilt wird, bis Z nicht weiter unterteilt ist. Dann kann die Niedrigrangmatrix entweder direkt zur vollbesetzten Matrix addiert werden oder, falls Z als Niedrigrangmatrix CD^T gespeichert wird, die Summe als neue Niedrigrangmatrix

$$AB^T + CD^T = \begin{pmatrix} A & C \end{pmatrix} \begin{pmatrix} B & D \end{pmatrix}^T$$

dargestellt und mit Hilfe der Singulärwertzerlegung gekürzt werden.
Der Kern der $\mathcal{H}$-Matrix-Multiplikation ist also für gewisse Blöcke XY effizient als Niedrigrangmatrix AB^T darzustellen und anschließend per Niedrigrangupdate für $\mathcal{H}$-Matrizen zur Zielmatrix Z zu addieren. Die Berechnung der Niedrigrangmatrix kann für $\mathcal{H}^2$-Matrizen bei einem entsprechenden blockweisen Ansatz relativ leicht aus der bekannten Matrix-Vektor-Multiplikation konstruiert werden. Der schwierigere Teil ist die Entwicklung eines effizienten Niedrigrangupdates für $\mathcal{H}^2$-Matrizen. Hierfür werden wir die Algorithmen der $\mathcal{H}^2$-Matrix-Rekompression (siehe [12, Chapter 6]) so adaptieren, dass sich damit ein Niedrigrangupdate verwirklichen lässt, das die Matrix nur lokal verändert. Mit Hilfe dieser

Updates kann die Arithmetik dann analog zum $\mathcal{H}$-Matrix-Fall behandelt werden, wobei natürlich einige Anpassungen an die Besonderheiten der $\mathcal{H}^2$-Matrizen notwendig sind.

Gliederung der Arbeit

Die Arbeit beginnt in Kapitel 2 mit einigen mathematischen Grundlagen, die später in der Arbeit benötigt werden. Dies dient vor allem der Vorstellung von Notationen und Aussagen, die bekannt sind und später benötigt werden. In Kapitel 3 stellen wir die für diese Arbeit zentralen hierarchischen Strukturen vor und beweisen erste Eigenschaften. Insbesondere führen wir das Format der $\mathcal{H}^2$-Matrizen ein. Für die $\mathcal{H}^2$-Matrizen beschreiben wir in Kapitel 4 den Rekompressions-Algorithmus aus [12, Chapter 6]. Dabei betrachten wir auch verschiedene Eigenschaften der $\mathcal{H}^2$-Matrizen und beschreiben den Fehler der Rekompression. Weil wir in dieser Arbeit die lokalen Ränge anders als in [12] abschätzen, beweisen wir leicht abgewandelte Aufwandsschranken und geben Verweise auf die entsprechenden Aussagen in [12].
Für das Niedrigrangupdate verwenden wir in Kapitel 5 eine Adaption der Rekompression, die nur für einen Teil der Matrix Berechnungen vornimmt. Dabei zeigen wir, dass das Update in einem Matrixblock $\mathbf{b} = \mathbf{t} \times \mathbf{s}$ derart lokal berechnet werden kann, dass der Aufwand von $(\#\mathbf{t}+\#\mathbf{s})$ abhängt, aber nicht von der Größe der Matrix. Außerdem erhalten wir eine Fehlerabschätzung, die uns erlaubt, den Fehler wie bei der Rekompression über Parameter zu steuern. Das Update in Kapitel 5 ist zusätzlich derart gestaltet, dass es in verschiedenen Blöcken nacheinander berechnet werden kann, ohne zwischen den Updates zusätzlich Operationen zu benötigen.
Mit dem Niedrigrangupdate formulieren wir in Kapitel 6 die Arithmetik. Insbesondere sind wir an der Inversion und der Cholesky-Zerlegung interessiert. Beide Aufgaben lassen sich mit Hilfe der Matrix-Matrix-Multiplikation durch einen rekursiven Algorithmus bewältigen. Die Aufwandsabschätzungen zeigen, dass die Komplexität für die neue $\mathcal{H}^2$-Arithmetik unter geeigneten Voraussetzungen in $\mathcal{O}(n \log(n) k^2)$ liegt. Wir können also im Vergleich zur $\mathcal{H}$-Arithmetik aus [26] einen $\log(n)$-Faktor einsparen.
Das Niedrigrangupdate aus Kapitel 5 nutzt nicht aus, dass die Algorithmen aus Kapitel 6 die hierarchische Blockstruktur der Zielmatrix rekursiv durchlaufen. Deshalb definieren wir in Kapitel 7 eine Alternative für das Niedrigrangupdate, die die Reihenfolge der Updates ausnutzt. Diese erweist sich für dreidimensionale Probleme als effizienter. Zum Abschluss zeigen wir in Kapitel 8 einige numerische Ergebnisse und fassen die Ergebnisse der Arbeit kurz zusammen.

Anmerkungen zur Arbeit

Aufgrund der Komplexität des Themas müssen verschiedene Notationen eingeführt werden. Um dem Leser die Übersicht zu erleichtern, sind die wichtigsten Notationen im Symbolverzeichnis zusammengeführt.
Des Weiteren sind viele Objekte von anderen abhängig, z. B. muss vor der Definition

einer Matrix $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ bekannt sein, was sich hinter $\mathcal{I}$ und $\mathcal{J}$ verbirgt. Da teilweise sehr viele solcher Objekte notwendig sind, wird häufig die Bedeutung durch die Notation impliziert. So wird $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ definiert ohne $\mathcal{I}$ und $\mathcal{J}$ explizit einzuführen, da diese in diesem Kontext grundsätzlich endliche Indexmengen bezeichnen.
Für die Untersuchung des Aufwands der vorgestellten Algorithmen zählen wir den Rechenaufwand in arithmetischen Operationen wie der Summe oder dem Produkt zweier Zahlen. Dies ist dadurch motiviert, dass bei den vorgestellten Algorithmen bei Speicherzugriffen und Vergleichen typischerweise auch arithmetische Operationen berechnet werden.
Für die Beschreibung der Komplexität der Algorithmen verwenden wir wie bereits in der Einführung das Landau-Symbol $\mathcal{O}$. Um dieses zu verwenden benötigen wir eine Folge von Problemen. Dafür definieren wir später entsprechende Voraussetzungen, die typisch für Anwendungen sind.
Die numerischen Ergebnisse in dieser Arbeit wurden mit der am Lehrstuhl Scientific Computing der Christian-Albrechts-Universität zu Kiel entwickelten Bibliothek `H2Lib` berechnet (siehe [39]).

Danksagung

An dieser Stelle möchte ich mich bei meinem Betreuer Steffen Börm für die Möglichkeit bedanken, dieses interessante Thema als Teil seiner Arbeitsgruppe zu bearbeiten.

2 Grundlagen

In diesem Kapitel fassen wir einige mathematische Grundlagen zusammen, die wir später benötigen. Dabei handelt sich es um bekannte Aussagen, die wir hier in unserer Notation zusammenfassen, um sie im Verlauf der Arbeit besser verwenden zu können.
In Abschnitt 2.1 fassen wir kurz einige Aussagen über Projektionen in Hilberträumen zusammen. Im darauf folgenden Abschnitt 2.2 behandeln wir lineare Abbildungen und Matrizen. Die von uns verwendeten Normen stellen wir in 2.3 vor. Insbesondere gehen wir auf Projektionen in den betrachteten Vektor- und Matrixräumen ein. Als wichtige Hilfsmittel behandeln wir orthogonale Zerlegungen (QR-Zerlegung und Singulärwertzerlegung) in Abschnitt 2.4. Zum Abschluss stellen wir in Abschnitt 2.5 ein Modellproblem vor, welches wir an verschiedenen Stellen zur Illustration verwenden.

2.1 Hilberträume und Projektionen

Wir verwenden in dieser Arbeit drei Normen, euklidische, Spektral- und Frobeniusnorm, von denen die erste und dritte Hilbertnormen sind. Deshalb wollen wir in diesem Abschnitt einige wichtige Aussagen zu Hilberträumen sammeln (siehe [37, Definition II.4.6]). Weil wir in dieser Arbeit nur endlich-dimensionale Räume nutzen, betrachten wir endlich-dimensionale Hilberträume. Ein wichtiges Konzept in Hilberträumen ist die Orthogonalität von Vektoren.

Definition 2.1.1 (Orthogonalität)
Es sei H ein endlich-dimensionaler Hilbertraum mit Skalarprodukt $\langle ., . \rangle$. Dann heißen zwei Vektoren $x, y \in H$ *orthogonal* zueinander, $x \perp y$, falls $\langle x, y \rangle = 0$ gilt.
Für eine endliche Menge $\mathcal{I}$ heißt eine Familie von Vektoren $(x_i)_{i \in \mathcal{I}} \in H^{\mathcal{I}}$ *paarweise orthogonal*, falls $x_i \perp x_j$ für alle $i, j \in \mathcal{I}$ mit $i \neq j$ gilt.
Zwei Mengen $X, Y \subseteq H$ heißen *orthogonal* zueinander, $X \perp Y$, falls $x \perp y$ für alle $x \in X$ und $y \in Y$ gilt.

Für paarweise orthogonale Vektoren können wir die Norm der Summe der Vektoren durch die Summe der Normen ausdrücken. Dies werden wir später bei der Abschätzung des Fehlers ausnutzen.

Lemma 2.1.2
Es sei $(H, \|.\|)$ ein Hilbertraum, $\mathcal{I}$ eine endliche Menge und $(x_i)_{i \in \mathcal{I}} \in H^{\mathcal{I}}$ paarweise

orthogonal. Dann gilt

$$\left\| \sum_{i \in \mathcal{I}} x_i \right\|^2 = \sum_{i \in \mathcal{I}} \|x_i\|^2.$$

Beweis: Es sei $\langle ., . \rangle$ das Skalarprodukt von H. Dann gilt

$$\left\| \sum_{i \in \mathcal{I}} x_i \right\|^2 = \left\langle \sum_{i \in \mathcal{I}} x_i, \sum_{j \in \mathcal{I}} x_j \right\rangle = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} \langle x_i, x_j \rangle = \sum_{i \in \mathcal{I}} \langle x_i, x_i \rangle = \sum_{i \in \mathcal{I}} \|x_i\|^2.$$

■

Ein weiterer wichtiger Begriff ist die adjungierte Abbildung.

Definition 2.1.3 (Adjungierte Abbildung)
Es seien H_1, H_2 endlich-dimensionale Hilberträume mit Skalarprodukten $\langle ., . \rangle_1$ und $\langle ., . \rangle_2$ sowie $A : H_2 \mapsto H_1$ und $B : H_1 \mapsto H_2$ lineare Abbildungen. Falls

$$\langle Ax, y \rangle_1 = \langle x, By \rangle_2 \qquad \text{für alle } x \in H_2,\ y \in H_1$$

gilt, heißt B die *adjungierte Abbildung* von A. (Nach [37, Hauptsatz II.5.2] existiert zu A genau eine solche Abbildung B.) Diese bezeichnen wir mit $A^* := B$. Die Abbildung A nennen wir *selbstadjungiert*, falls $A = A^*$ gilt.

Der Algorithmus für das Niedrigrangupdate einer $\mathcal{H}^2$-Matrix berechnet zuerst adaptive Basen und anschließend die neue $\mathcal{H}^2$-Matrix per orthogonalen Projektionen. Deshalb fassen wir an dieser Stelle einige wichtige Eigenschaften der Projektion zusammen.

Definition 2.1.4 (Projektion)
Es sei H ein endlich-dimensionaler Hilbertraum. Eine Abbildung $\Pi : H \to H$ heißt *Projektion* auf $\mathrm{Bild}(\Pi)$, falls $\Pi^2 = \Pi$ gilt. Falls Π zusätzlich selbstadjungiert ist, so ist Π eine *orthogonale Projektion*.

Die definierende Eigenschaft der Projektion besitzt eine weitere äquivalente Formulierung, welche für manche Beweise geeigneter ist.

Bemerkung 2.1.5 (Charakterisierung von Projektionen)
Die Bedingung $\Pi = \Pi^2$ ist äquivalent zu $\Pi(x) = x$ für alle $x \in \mathrm{Bild}(\Pi)$.

Bei der Fehlerabschätzung erhalten wir auch Summen von orthogonalen Projektionen. In bestimmten Fällen können wir zeigen, dass diese Summen wieder orthogonale Projektionen sind.

Lemma 2.1.6
Es seien $(H, \langle ., . \rangle)$ ein reeller endlich-dimensionaler Hilbertraum, $\mathcal{I}$ eine endliche Menge und $\Pi_i : H \to H$, $i \in \mathcal{I}$, orthogonale Projektionen. Falls $\mathrm{Bild}(\Pi_j) \subseteq \mathrm{Kern}(\Pi_i)$ für alle $i, j \in \mathcal{I}$ mit $i \neq j$ gilt, ist $\Pi_\Sigma := \sum_{i \in \mathcal{I}} \Pi_i$ eine orthogonale Projektion.

Beweis: Es sei $x \in H$. Dann gilt, weil Π_i für alle $i \in \mathcal{I}$ eine Projektion ist,

$$\Pi_{\sum}^2 x \;=\; \left(\sum_{i\in\mathcal{I}} \Pi_i\right)^2 x \;=\; \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}} \Pi_i\Pi_j x \;=\; \sum_{i\in\mathcal{I}} \Pi_i^2 x \;=\; \sum_{i\in\mathcal{I}} \Pi_i x \;=\; \Pi_{\sum} x.$$

Also ist $\Pi_{\sum}$ eine Projektion. Weil $\Pi_i = \Pi_i^*$ für alle $i \in \mathcal{I}$ ist, gilt für alle $x, y \in H$

$$\langle \Pi_{\sum} x, y\rangle \;=\; \sum_{i\in\mathcal{I}} \langle \Pi_i x, y\rangle \;=\; \sum_{i\in\mathcal{I}} \langle x, \Pi_i y\rangle \;=\; \langle x, \Pi_{\sum} y\rangle.$$

■

Eine wichtige Eigenschaft der orthogonalen Projektionen Π ist, dass der Projektionsfehler orthogonal zum Bild(Π) ist. Dies ermöglicht uns später, mit Hilfe von Lemma 2.1.2 die Fehler abzuschätzen. Weil für den Projektionsfehler $x - \Pi x = (I - \Pi)x \in \text{Bild}(I - \Pi)$ gilt, zeigen wir im Lemma 2.1.7, dass die Bilder von Π und $I - \Pi$ orthogonal zueinander sind.

Lemma 2.1.7 (Orthogonalität des Projektionsfehlers)
Es sei $(H, \langle .\,, .\,\rangle)$ ein endlich-dimensionaler Hilbertraum und $\Pi : H \to H$ eine orthogonale Projektion. Dann gilt

$$\text{Bild}(I - \Pi) \;=\; \text{Kern}(\Pi) \quad \text{ und } \quad \text{Bild}(I - \Pi) \;\perp\; \text{Bild}(\Pi)$$

d. h. Projektionsfehler stehen senkrecht auf dem Bild der Projektion Π.

Beweis: Es sei $x \in \text{Bild}(I - \Pi)$. Dann existiert $y \in H$ mit $x = (I - \Pi)y$ und, weil Π eine Projektion ist, gilt

$$\Pi x \;=\; \Pi(I - \Pi)y \;=\; (\Pi - \Pi^2)y \;=\; (\Pi - \Pi)y \;=\; 0.$$

Es sei $x \in \text{Kern}(\Pi)$. Dann gilt

$$x \;=\; x - \Pi x \;=\; (I - \Pi)x \;\in \text{Bild}(I - \Pi).$$

Also gilt $\text{Kern}(\Pi) = \text{Bild}(I - \Pi)$.
Es seien $x, y \in H$. Dann gilt wegen $(I - \Pi)y \in \text{Bild}(I - \Pi) = \text{Kern}(\Pi)$

$$\langle \Pi x, (I - \Pi)y\rangle \;=\; \langle x, \Pi(I - \Pi)y\rangle \;=\; \langle x, 0\rangle \;=\; 0.$$

Somit ist $\text{Bild}(\Pi) \perp \text{Bild}(I - \Pi)$. ■

Es ist zu beachten, dass sich die Namensgebung der orthogonalen Projektionen auf die Eigenschaft aus Lemma 2.1.7 bezieht. Es existieren orthogonale Projektionen, die keinen vollen Rang besitzen (siehe Beispiel 2.3.17) und somit keine orthogonalen Abbildungen sind (z. B. die Nullabbildung).
Eine weitere wichtige Eigenschaft ist, dass die Operatornorm von orthogonalen Projektionen höchstens eins ist.

Korollar 2.1.8 (Norm orthogonaler Projektionen)
Es sei $(H, \langle .\,, .\rangle)$ ein reeller endlich-dimensionaler Hilbertraum mit Norm $\|.\|$ und $\Pi : H \to H$ eine orthogonale Projektion. Für alle $x \in H$ gilt

$$\|x\|^2 \;=\; \|(I-\Pi)x\|^2 + \|\Pi x\|^2 \;\geq\; \|\Pi x\|^2. \tag{2.1}$$

Beweis: Es sei $x \in H$. Nach Lemma 2.1.7 ist $(I-\Pi)x \perp \text{Bild}(\Pi)$. Wegen $\Pi x \in \text{Bild}(\Pi)$ folgt mit Lemma 2.1.2

$$\|x\|^2 \;=\; \|(I-\Pi)x + \Pi x\|^2 \;=\; \|(I-\Pi)x\|^2 + \|\Pi x\|^2 \;\geq\; \|\Pi x\|^2.$$

■

Als Nächstes zeigen wir, dass eine orthogonale Projektionen Π stets die Bestapproximation in Bild(Π) liefert.

Lemma 2.1.9 (Bestapproximation durch orthogonale Projektion)
Es sei $(H, \langle .\,, .\rangle)$ ein reeller endlich-dimensionaler Hilbertraum mit Norm $\|.\|$ und $\Pi : H \to H$ eine orthogonale Projektion. Dann gilt für $x \in H$

$$\|x - \Pi x\| \;=\; \inf_{y \in \text{Bild}(\Pi)} \|x - y\|. \tag{2.2}$$

Beweis: Es sei $x \in H$ und $y \in \text{Bild}(\Pi)$. Dann gilt

$$\begin{aligned} \|x-y\|^2 \;&=\; \|x - \Pi x + \Pi x - y\|^2 \;=\; \langle x - \Pi x + \Pi x - y,\, x - \Pi x + \Pi x - y\rangle \\ &=\; \|x - \Pi x\|^2 + 2\langle x - \Pi x,\, \Pi x - y\rangle + \|\Pi x - y\|^2. \end{aligned}$$

Wegen $\Pi x - y \in \text{Bild}(\Pi)$ folgt mit Lemma 2.1.7

$$\begin{aligned} \|x-y\|^2 \;&=\; \|x - \Pi x\|^2 + 2\langle x - \Pi x,\, \Pi x - y\rangle + \|\Pi x - y\|^2 \\ &=\; \|x - \Pi x\|^2 + \|\Pi x - y\|^2 \;\geq\; \|x - \Pi x\|^2. \end{aligned}$$

Also gilt

$$\|x - \Pi x\| \;\leq\; \inf_{y \in \text{Bild}(\Pi)} \|x - y\|.$$

Die umgekehrte Ungleichung folgt, da $\Pi x \in \text{Bild}(\Pi)$ gilt. ■

Wir finden durch orthogonale Projektionen also die bestmögliche Approximation in Bild(Π). In Abschnitt 2.3 stellen wir für unsere Arbeit wichtige Projektionen in den von uns betrachteten normierten Räumen vor. Diese nutzen wir bei der Rekompression von $\mathcal{H}^2$-Matrizen in Kapitel 4, um Projektionen auf gegebene $\mathcal{H}^2$-Matrix-Räume zu definieren.

2.2 Lineare Abbildungen und Matrizen

Im Folgenden werden wir vor allem endlich-dimensionale Räume betrachten. Dazu definieren wir sogenannte Indexmengen.

Definition 2.2.1 (Indexmenge)
Eine Menge $\mathcal{I}$ heißt *Indexmenge*, falls sie endlich ist.

Die Indexmengen sind i. A. nicht angeordnet. Allerdings finden wir natürlich eine Bijektion von $\mathcal{I}$ nach $\{1, \dots, \#\mathcal{I}\}$, so dass wir stets eine Anordnung konstruieren können (z. B. für Dreieckszerlegungen).

Bemerkung 2.2.2
Falls wir eine neue Indexmenge wählen, bei der nur die Mächtigkeit festgelegt ist, soll diese immer disjunkt zu allen vorherigen Indexmengen sein.

Weil wir häufig disjunkte Mengen betrachten, führen wir kurz eine Notation für disjunkte Vereinigungen ein.

Bezeichnungen 2.2.3
Die disjunkte Vereinigung von Mengen bezeichnen wir mit $\dot\cup$ bzw. $\dot\bigcup$.

Als Nächstes wollen wir uns mit linearen Abbildungen zwischen endlich-dimensionalen Vektorräumen beschäftigen.

Bemerkung 2.2.4
Es seien $\mathcal{I}, \mathcal{J}, \mathcal{K}$ Indexmengen. Dann identifizieren wir eine Matrix $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ einerseits mit der linearen Abbildung

$$A : \mathbb{R}^{\mathcal{J}} \to \mathbb{R}^{\mathcal{I}},\ x \mapsto Ax$$

und andererseits mit der Abbildung

$$A : \mathbb{R}^{\mathcal{J}\times\mathcal{K}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{K}},\ X \mapsto AX.$$

Da wir keine Basiswechsel in den Vektor-/Matrixräumen vornehmen, ist diese Identifikation jeweils eindeutig (siehe [6, Kapitel 3]). Allerdings existieren lineare Abbildungen zwischen den Matrixräumen, welche nicht auf obige Weise dargestellt werden können. Dies liegt daran, dass bei der Matrix-Matrix-Multiplikation immer alle Spalten auf gleiche Weise abgebildet werden.

Die erste Identifizierung ist die übliche. Die zweite verwenden wir, weil wir später Matrixräume betrachten, die mit der Spektralnorm bzw. der Frobeniusnorm ausgestattet sind. $\mathcal{H}^2$-Matrizen basieren auf einer Blockpartition der Indexmenge $\mathcal{I} \times \mathcal{J}$. Um die Rechnungen mit und Zugriffe auf Teilmatrizen elegant darzustellen, verwenden wir die Einschränkung auf Teilmatrizen. Zu diesem Zweck definieren wir folgende Diagonalmatrix.

Definition 2.2.5 (Einschränkung)
Es sei $\mathcal{I}$ eine Indexmenge und $\mathbf{t} \subseteq \mathcal{I}$. Es sei für alle $i \in \mathcal{I}$

$$\delta_{i,\mathbf{t}} \;=\; \sum_{j \in \mathbf{t}} \delta_{i,j} \;=\; \begin{cases} 1 & , \; i \in \mathbf{t} \\ 0 & , \; i \notin \mathbf{t} \end{cases} \tag{2.3}$$

eine Verallgemeinerung des *Dirac-Symbols* $\delta_{i,j}$. Dann definiert

$$\Pi_{\mathbf{t}} \;:=\; \operatorname{diag}(\delta_{i,\mathbf{t}}) \;\in\; \mathbb{R}^{\mathcal{I} \times \mathcal{I}} \tag{2.4}$$

die *Einschränkung auf die Teilmenge* $\mathbf{t}$.

Das Lemma 2.2.6 beschreibt, wie die Einschränkung auf Vektoren wirkt.

Lemma 2.2.6 (Einschränkung von Vektoren)
Es seien $\mathcal{I}$ eine Indexmenge und $\mathbf{t}, \mathbf{s} \subseteq \mathcal{I}$.

(i) Für alle $x \in \mathbb{R}^{\mathcal{I}}$ und alle $i \in \mathcal{I}$ gilt

$$(\Pi_{\mathbf{t}} x)_i \;=\; \begin{cases} x_i & , \text{ falls } i \in \mathbf{t} \\ 0 & , \text{ falls } i \notin \mathbf{t} \end{cases}.$$

(ii) Es gilt

$$\Pi_{\mathbf{t}}\Pi_{\mathbf{s}} \;=\; \Pi_{\mathbf{t}\cap\mathbf{s}} \;=\; \Pi_{\mathbf{s}}\Pi_{\mathbf{t}} \;=\; \begin{cases} \Pi_{\mathbf{t}} & , \text{ falls } \mathbf{t} \subseteq \mathbf{s} \\ \Pi_{\mathbf{s}} & , \text{ falls } \mathbf{s} \subseteq \mathbf{t} \\ 0 & , \text{ falls } \mathbf{t} \cap \mathbf{s} = \emptyset \\ \Pi_{\mathbf{t}\cap\mathbf{s}} & \text{sonst} \end{cases}.$$

Beweis: Es sei $x \in \mathbb{R}^{\mathcal{I}}$. Dann gilt für alle $i \in \mathcal{I}$

$$(\Pi_{\mathbf{t}} x)_i \;=\; \delta_{i,\mathbf{t}} x_i \;=\; \begin{cases} x_i & , \text{ falls } i \in \mathbf{t} \\ 0 & , \text{ falls } i \notin \mathbf{t} \end{cases}.$$

Es gilt

$$\Pi_{\mathbf{t}}\Pi_{\mathbf{s}} \;=\; \operatorname{diag}(\delta_{i,\mathbf{t}}) \operatorname{diag}(\delta_{i,\mathbf{s}}) \;=\; \operatorname{diag}(\delta_{i,\mathbf{t}}\delta_{i,\mathbf{s}}) \;=\; \operatorname{diag}(\delta_{i,\mathbf{t}\cap\mathbf{s}}) \;=\; \Pi_{\mathbf{t}\cap\mathbf{s}}$$

und damit

$$\Pi_{\mathbf{t}}\Pi_{\mathbf{s}} \;=\; \Pi_{\mathbf{t}\cap\mathbf{s}} \;=\; \Pi_{\mathbf{s}\cap\mathbf{t}} \;=\; \Pi_{\mathbf{t}}\Pi_{\mathbf{s}}.$$

■

Nach der Anwendung von Einschränkungen auf Vektoren untersuchen wir, wie die Einschränkungen auf Matrizen wirken.

Lemma 2.2.7 (Einschränkung von Matrizen)
Es seien $\mathcal{I}, \mathcal{J}$ Indexmengen. Dann gilt für alle $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathbf{t} \subseteq \mathcal{I}$ und $\mathbf{s} \subseteq \mathcal{J}$, $(i,j) \in \mathcal{I} \times \mathcal{J}$

$$(\Pi_{\mathbf{t}} A \Pi_{\mathbf{s}})_{i,j} = \begin{cases} A_{i,j} & , \text{ falls } (i,j) \in \mathbf{t} \times \mathbf{s} \\ 0 & , \text{ falls } (i,j) \notin \mathbf{t} \times \mathbf{s} \end{cases}.$$

Beweis: Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathbf{t} \subseteq \mathcal{I}$ und $\mathbf{s} \subseteq \mathcal{J}$. Dann gilt für alle $(i,j) \in \mathcal{I} \times \mathcal{J}$

$$(\Pi_{\mathbf{t}} A \Pi_{\mathbf{s}})_{(i,j)} = (\delta_{i,\mathbf{t}} A_{i,j} \delta_{j,\mathbf{s}})_{(i,j)} = \begin{cases} A_{i,j} & , \text{ falls } (i,j) \in \mathbf{t} \times \mathbf{s} \\ 0 & , \text{ falls } (i,j) \notin \mathbf{t} \times \mathbf{s} \end{cases}.$$

■

Mit obigem Lemma können wir Matrizen auf einen einzelnen Block einschränken. Dies ist später wichtig, da hierarchische Matrizen mit einer Blockpartition arbeiten. Um dann mit einzelnen Blöcken zu arbeiten, können wir die Einschränkungen nutzen. Wir definieren die zugehörigen Teilräume.

Definition 2.2.8 (Eingeschränkte Teilräume)
Es seien $\mathcal{I}, \mathcal{J}$ Indexmengen mit Teilmengen $\mathbf{t} \subseteq \mathcal{I}, \mathbf{s} \subseteq \mathcal{J}$. Dann definieren wir den *eingeschränkten Vektorraum*

$$\mathbb{R}^{\mathcal{I}}_{\mathbf{t}} := \{x \in \mathbb{R}^{\mathcal{I}} \,|\, \Pi_{\mathbf{t}} x = x\} = \{x \in \mathbb{R}^{\mathcal{I}} \,|\, \forall i \in \mathcal{I}\backslash\mathbf{t} : x_i = 0\}$$

und den *eingeschränkten Matrixraum*

$$\begin{aligned} \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}} &:= \{A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \,|\, \Pi_{\mathbf{t}} A \Pi_{\mathbf{s}} = A\} \\ &= \{A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \,|\, \forall (i,j) \in (\mathcal{I}\times\mathcal{J})\backslash(\mathbf{t}\times\mathbf{s}) : A_{i,j} = 0\}. \end{aligned}$$

Falls $\mathbf{s} = \mathcal{J}$ gilt schreiben wir auch $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}} := \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathcal{J}}$.

Da diese Einschränkungen häufig auftauchen, führen wir eine kurze Notation ein.

Bezeichnungen 2.2.9
Für Vektoren benutzen wir die Schreibweise

$$x_{|\mathbf{t}} := \Pi_{\mathbf{t}} x \in \mathbb{R}^{\mathcal{I}}_{\mathbf{t}} \quad \text{für alle } x \in \mathbb{R}^{\mathcal{I}}, \mathbf{t} \subseteq \mathcal{I}$$

und für Matrizen

$$A_{|\mathbf{t}\times\mathbf{s}} := \Pi_{\mathbf{t}} A \Pi_{\mathbf{s}} \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}} \quad \text{für alle } A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \mathbf{t} \subseteq \mathcal{I}, \mathbf{s} \subseteq \mathcal{J}.$$

Es kommt vor, dass wir eine Matrix nur bezüglich der Zeilen einschränken. Dann schreiben wir auch $A_{|\mathbf{t}} \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}}$ für $A_{|\mathbf{t}\times\mathcal{J}} \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathcal{J}}$, um die Ausdrücke möglichst kurz zu halten.

Für die effiziente Implementierung ist es wichtig, die Elemente der eingeschränkten Räume nicht vollständig zu speichern. Dies halten wir in folgender Bemerkung fest.

Bemerkung 2.2.10
Bei der Implementierung der Algorithmen speichern wir Vektoren $x \in \mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$ als Elemente von $\mathbb{R}^{\mathbf{t}}$. Analog behandeln wir Matrizen $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}}$ als Elemente von $\mathbb{R}^{\mathbf{t}\times\mathbf{s}}$.

2.3 Normen

Später wollen wir den Fehler der approximativen Arithmetik in Matrixnormen quantifizieren. Dazu verwenden wir die Spektralnorm und die Frobeniusnorm. Die wesentlichen Definitionen und Aussagen über diese Normen werden in diesem Abschnitt zusammengefasst.
Bevor wir die Matrixnormen definieren, führen wir die euklidische Norm ein.

Definition 2.3.1 (euklidische Norm)
Es sei $\mathcal{I}$ eine Indexmenge. Dann definiert

$$\langle .,. \rangle_2 : \mathbb{R}^{\mathcal{I}} \times \mathbb{R}^{\mathcal{I}} \to \mathbb{R},\ (x,y) \mapsto \langle x,y \rangle_2 = \sum_{i \in \mathcal{I}} x_i y_i \tag{2.5}$$

das euklidische Skalarprodukt (Standardskalarprodukt) und

$$\|.\|_2 : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}_{\geq 0},\ x \mapsto \|x\|_2 \ = \ (\langle x,y \rangle_2)^{\frac{1}{2}} \ = \ \left(\sum_{i \in \mathcal{I}} x_i^2 \right)^{\frac{1}{2}} \tag{2.6}$$

die euklidische Norm auf $\mathbb{R}^{\mathcal{I}}$.

Bemerkung 2.3.2 (Eigenschaften der euklidischen Norm)
Es sei $\mathcal{I}$ eine Indexmenge. Dann ist

(i) $\left(\mathbb{R}^{\mathcal{I}}, \langle .,. \rangle_2\right)$ ein endlich-dimensionaler Hilbertraum und

(ii) für alle $x \in \mathbb{R}^{\mathcal{I}}$

$$\|x\|_2 \ = \ \max_{\|y\|_2 = 1} \langle x,y \rangle_2.$$

Beweis: In (ii) folgt $\leq$ für $x \in \mathbb{R}^{\mathcal{I}} \backslash \{0\}$ mit $y := x/\|x\|_2$ und $\geq$ mit der Cauchy-Schwarz-Ungleichung (siehe [37, Hauptsatz II.4.3]). Für $x = 0$ gilt die Aussage offensichtlich. ∎

Auf der Basis des euklidischen Skalarprodukts können wir die Spektralnorm und die Frobeniusnorm auf den Matrixräumen definieren. Wir beginnen mit der Spektralnorm.

Definition 2.3.3 (Spektralnorm)
Es seien $\mathcal{I}, \mathcal{J}$ Indexmengen. Dann definiert

$$\|.\| : \mathbb{R}^{\mathcal{I} \times \mathcal{J}} \to \mathbb{R}_{\geq 0},\ A \mapsto \|A\|_S \ := \ \sup_{x \in \mathbb{R}^{\mathcal{I}} \backslash \{0\}} \frac{\|Ax\|_2}{\|x\|_2}$$

die *Spektralnorm* auf $\mathbb{R}^{\mathcal{I} \times \mathcal{J}}$.

Die Spektralnorm ist die durch die euklidische Norm induzierte Operatornorm. Die folgende Bemerkung fasst einige Eigenschaften der Spektralnorm zusammen.

Bemerkung 2.3.4 (Eigenschaften der Spektralnorm)
Es seien $\mathcal{I}$ und $\mathcal{J}$ Indexmengen.

(i) Die Spektralnorm ist eine Norm auf dem Matrixraum $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$.

(ii) Für $\mathcal{I}=\mathcal{J}$ und die Identität $I_{\mathcal{I}}:\mathcal{I}\to\mathcal{I}, x\mapsto x$ gilt $\|I_{\mathcal{I}}\|_2=1$.

(iii) Es gilt für alle $A\in\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$

$$\|A\|_S \;=\; \max_{\|x\|_2\leq 1}\|Ax\|_2 \;=\; \max_{\|x\|_2=1}\|Ax\|_2 \;=\; \max_{\|x\|_2=1=\|y\|_2}\langle Ax,y\rangle_2.$$

Beweis: Die Aussage (iii) folgt aus der Eigenschaft positiver Skalare, in die Norm gezogen werden zu können, und aus Bemerkung 2.3.2 (ii). ∎

Die zweite Matrixnorm, die wir betrachten, ist die Frobeniusnorm.

Definition 2.3.5 (Frobeniusnorm)
Es seien $\mathcal{I},\mathcal{J}$ Indexmengen. Dann definiert

$$\langle .,. \rangle_F :\; \mathbb{R}^{\mathcal{I}\times\mathcal{J}}\times\mathbb{R}^{\mathcal{I}\times\mathcal{J}}\to\mathbb{R},\; (A,B)\mapsto\langle A,B\rangle_F \;:=\; \sum_{(i,j)\in\mathcal{I}\times\mathcal{J}} A_{i,j}B_{i,j} \tag{2.7}$$

das *Frobenius-Skalarprodukt* und

$$\|.\|_F :\; \mathbb{R}^{\mathcal{I}\times\mathcal{J}}\to\mathbb{R}_{\geq 0},\; A\mapsto\|A\|_F \;:=\; (\langle A,A\rangle_F)^{\frac{1}{2}} \;=\; \left(\sum_{(i,j)\in\mathcal{I}\times\mathcal{J}} A_{i,j}^2\right)^{\frac{1}{2}} \tag{2.8}$$

die *Frobeniusnorm*.

Als Nächstes fassen wir erste Eigenschaften der Frobeniusnorm zusammen.

Bemerkung 2.3.6 (Eigenschaften der Frobeniusnorm)
Es seien $\mathcal{I}$ und $\mathcal{J}$ Indexmengen.

(i) Es ist $\left(\mathbb{R}^{\mathcal{I}\times\mathcal{J}},\|.\|_F\right)$ ein endlich-dimensionaler Hilbertraum mit Skalarprodukt $\langle .,. \rangle_F$.

(ii) Es seien e_j, $j\in\mathcal{J}$, die Einheitsvektoren in $\mathbb{R}^{\mathcal{J}}$. Dann gilt für $A\in\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$

$$\|A\|_F^2 \;=\; \sum_{j\in\mathcal{J}}\|Ae_j\|_2^2.$$

Beweis: Die erste Aussage folgt, indem wir die Matrizen in $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ als Vektoren auffassen. Dann entspricht die Frobeniusnorm der euklidischen Norm.
Es gilt

$$\|A\|_F^2 \;=\; \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} A_{i,j}^2 \;=\; \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}} (Ae_j)_i^2 \;=\; \sum_{j\in\mathcal{J}} \|Ae_j\|_2^2$$

und somit (ii). ∎

Beide betrachteten Matrixnormen haben eine enge Verbindung zur euklidischen Norm. Daraus folgt auch eine gewisse Verträglichkeit der Matrixnormen mit der euklidischen Norm.

Lemma 2.3.7 (Verträglichkeit mit der euklidischen Norm)
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $x \in \mathbb{R}^{\mathcal{J}}$. Dann sind die betrachteten Matrixnormen verträglich mit der euklidischen Norm

$$\|Ax\|_2 \;\leq\; \|A\|_S\|x\|_2 \quad \text{und} \quad \|Ax\|_2 \;\leq\; \|A\|_F\|x\|_2.$$

Beweis: Für den Fall, dass $x = 0$ ist, sind beide Seiten gleich null und die Aussage folgt direkt. Es sei also o. E. d. A. $x \neq 0$.
Nach Definition 2.3.3 der Spektralnorm gilt

$$\frac{\|Ax\|_2}{\|x\|_2} \;\leq\; \sup_{y\in\mathbb{R}^{\mathcal{J}}\setminus\{0\}} \frac{\|Ay\|_2}{\|y\|_2} \;=\; \|A\|_S$$

und somit die Aussage.
Für die Frobeniusnorm folgt mit der Cauchy-Schwarz-Ungleichung (siehe [37, Hauptsatz II.4.3])

$$\|Ax\|_2^2 \;=\; \sum_{i\in\mathcal{I}} \Big(\sum_{j\in\mathcal{J}} A_{i,j}x_j\Big)^2 \;\leq\; \sum_{i\in\mathcal{I}} \Big(\sum_{j\in\mathcal{J}} A_{i,j}^2\Big)\Big(\sum_{j\in\mathcal{J}} x_j^2\Big) \;=\; \|A\|_F^2\|x\|_2^2.$$

∎

Um bei späteren Fehlerabschätzungen die Produkte von Matrizen in der Norm auseinander zu ziehen, verwenden wir die sogenannte Submultiplikativität.

Lemma 2.3.8 (Submultiplikativität)
Es seien $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $B \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}}$. Dann gilt

$$\|AB\|_S \;\leq\; \|A\|_S\|B\|_S \quad \text{und} \quad \|AB\|_F \;\leq\; \|A\|_F\|B\|_F.$$

Beweis: Es gilt mit der Definition 2.3.3 der Spektralnorm und der Verträglichkeit mit der euklidischen Norm (siehe Lemma 2.3.7)

$$\begin{aligned}\|AB\|_S &= \sup_{x\in\mathbb{R}^{\mathcal{K}}\setminus\{0\}} \frac{\|ABx\|_2}{\|x\|_2} \leq \sup_{x\in\mathbb{R}^{\mathcal{K}}\setminus\{0\}} \frac{\|A\|_S\|Bx\|_2}{\|x\|_2} \\ &\leq \sup_{x\in\mathbb{R}^{\mathcal{K}}\setminus\{0\}} \frac{\|A\|_S\|B\|_S\|x\|_2}{\|x\|_2} = \|A\|_S\|B\|_S.\end{aligned}$$

Es seien e_k, $k \in \mathcal{K}$, die k-ten Einheitsvektoren in $\mathbb{R}^{\mathcal{K}}$. Dann gilt mit der Darstellung der Frobeniusnorm aus 2.3.6 (ii) und der Verträglichkeit mit der euklidischen Norm

$$\|AB\|_F^2 = \sum_{k\in\mathcal{K}} \|ABe_k\|_2^2 \leq \|A\|_F^2 \sum_{k\in\mathcal{K}} \|Be_k\|_2^2 = \|A\|_F^2\|B\|_F^2.$$

■

Wir haben gesehen, dass die betrachteten Matrixnormen mit der euklidischen Norm verträglich sind. Mit Lemma 2.3.9 können wir die Matrixnormen abschätzen, indem wir die Bilder von Teilmatrizen in der euklidischen Norm beschränken.

Lemma 2.3.9 (Matrixnormen abschätzen)
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathcal{K}$ eine Indexmenge und $A_k \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ für alle $k \in \mathcal{K}$. Falls

$$\|Ax\|_2^2 \leq \sum_{k\in\mathcal{K}} \|A_kx\|_2^2 \quad \text{für alle } x \in \mathbb{R}^{\mathcal{J}} \tag{2.9}$$

gilt, folgt

$$\|A\|_S^2 \leq \sum_{k\in\mathcal{K}} \|A_k\|_S^2 \quad \text{und} \quad \|A\|_F^2 \leq \sum_{k\in\mathcal{K}} \|A_k\|_F^2. \tag{2.10}$$

Falls in (2.9) Gleichheit gilt, folgt in (2.10) für die Frobeniusnorm auch Gleichheit.

Beweis: Es gelte (2.9). Dann folgt mit der Darstellung der Spektralnorm aus Bemerkung 2.3.4 (iii) und der Verträglichkeit mit der euklidischen Norm (siehe Lemma 2.3.7)

$$\begin{aligned}\|A\|_S^2 &= \left(\max_{\|x\|=1} \|Ax\|_2\right)^2 = \max_{\|x\|_2=1} \|Ax\|_2^2 \leq \max_{\|x\|_2=1} \sum_{k\in\mathcal{K}} \|A_kx\|_2^2 \\ &\leq \max_{\|x\|_2=1} \sum_{k\in\mathcal{K}} \|A_k\|_S^2\|x\|_2^2 = \sum_{k\in\mathcal{K}} \|A_k\|_S^2.\end{aligned}$$

Mit der Darstellung der Frobeniusnorm aus Bemerkung 2.3.6 (ii) folgt für die Einheitsvektoren $e_j \in \mathbb{R}^{\mathcal{J}}$, $j \in \mathcal{J}$, des $\mathbb{R}^{\mathcal{J}}$

$$\begin{aligned}\|A\|_F^2 &= \sum_{(i,j)\in\mathcal{I}\times\mathcal{J}} A_{i,j}^2 = \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}} A_{i,j}^2 = \sum_{j\in\mathcal{J}} \|Ae_j\|_2^2 \leq \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \|A_ke_j\|_2^2 \\ &= \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{J}} \|A_ke_j\|_2^2 = \sum_{k\in\mathcal{K}} \|A_k\|_F^2.\end{aligned}$$

Falls in (2.9) Gleichheit gilt, folgt analog (2.10) mit Gleichheit für die Frobeniusnorm. ■

Die von uns betrachteten hierarchischen Matrizen basieren auf einer Blockpartition. Deshalb wollen wir bei den Fehlerabschätzungen unter anderem den blockweisen Fehler untersuchen. Um daraus eine globale Fehlerabschätzung zu erhalten, nutzen wir das Lemma 2.3.10.

Lemma 2.3.10 (Lokale und globale Matrixnorm)
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{P}$ eine Blockpartition von $\mathcal{I} \times \mathcal{J}$, d. h. $\mathcal{P}$ ist eine Partition und für alle $p \in \mathcal{P}$ existieren $\mathbf{t} \subseteq \mathcal{I}$ und $\mathbf{s} \subseteq \mathcal{J}$ mit $p = \mathbf{t} \times \mathbf{s}$. Dann gilt

$$\|A\|_S \;\leq\; \left(\sum_{p\in\mathcal{P}} \|A_{|p}\|_S^2\right)^{\frac{1}{2}} \quad \text{und} \quad \|A\|_F \;=\; \left(\sum_{p\in\mathcal{P}} \|A_{|p}\|_F^2\right)^{\frac{1}{2}}.$$

Beweis:[siehe [12, Lemma 4.45 und Lemma 4.39]] ■

Wie wir in Lemma 2.3.10 sehen, ist die Frobeniusnorm für die Betrachtung von hierarchischen Matrizen besonders geeignet, weil sich die Frobeniusnorm einer Matrix einfach aus den Frobeniusnormen der Teilblöcke berechnet lässt. Deshalb bekommen wir typischerweise etwas bessere Aussagen für die Frobeniusnorm im Vergleich zur Spektralnorm. Andererseits sind die Ergebnisse in der Spektralnorm von besonderem Interesse, weil es sich um eine Operatornorm handelt.
Für Hilberträume haben wir adjungierte Abbildungen eingeführt. Für euklidische Vektorräume erhalten wir eine einfache Charakterisierung der adjungierten Abbildung.

Definition 2.3.11 (Transponierte Matrizen)
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Dann ist die *transponierte Matrix* zu A durch

$$A^T \;:=\; (A_{i,j})_{(j,i)\in\mathcal{J}\times\mathcal{I}} \;\in \mathbb{R}^{\mathcal{J}\times\mathcal{I}}$$

definiert.

Lemma 2.3.12
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{K}$ eine Indexmenge. Dann definiert A^T die adjungierte Abbildung zu A bez. der euklidischen Norm und bez. der Frobeniusnorm. D. h. , es gilt

$$\langle Ax, y\rangle_2 \;=\; \langle x, A^T y\rangle_2 \quad \text{für alle } x \in \mathbb{R}^{\mathcal{J}},\, y \in \mathbb{R}^{\mathcal{I}}$$

und

$$\langle AX, Y\rangle_F \;=\; \langle X, A^T Y\rangle_F \quad \text{für alle } X \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}},\, Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}.$$

Beweis: Es sei $x \in \mathbb{R}^{\mathcal{J}}$ und $y \in \mathbb{R}^{\mathcal{I}}$. Dann gilt

$$\langle Ax, y\rangle_2 \;=\; \sum_{i\in\mathcal{I}} \left(\sum_{j\in\mathcal{J}} A_{i,j}x_j\right) y_i \;=\; \sum_{j\in\mathcal{J}} x_j \left(\sum_{i\in\mathcal{I}} (A^T)_{j,i} y_i\right) \;=\; \langle x, A^T y\rangle_2.$$

Es sei $X \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}}$ und $Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$. Dann gilt

$$\begin{aligned}\langle AX, Y\rangle_F &= \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}}\Big(\sum_{j\in\mathcal{J}} A_{i,j}X_{jk}\Big)Y_{ik} = \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} X_{jk}\Big(\sum_{i\in\mathcal{I}}(A^T)_{j,i}Y_{ik}\Big)\\ &= \langle X, A^TY\rangle_F.\end{aligned}$$

■

Aus Lemma 2.3.12 folgt, dass insbesondere reelle Diagonalmatrizen selbstadjungiert sind, weil sie gleich ihrer transponierten Matrix sind. Somit sind die in Abschnitt 2.2 definierten Einschränkungen selbstadjungiert. Später werden wir noch weitere selbstadjungierte Matrizen betrachten.
Eine wichtige Eigenschaft für die Fehleranalyse ist, dass die betrachteten Matrixnormen unter Adjungieren invariant bleiben. Dadurch können wir später die Betrachtung des Fehlers der Projektion bezüglich der Spalten auf den Fall für die Zeilen zurückführen.

Lemma 2.3.13
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Dann gilt

$$\|A\|_S = \|A^T\|_S \quad \text{und} \quad \|A\|_F = \|A^T\|_F.$$

Für das Frobenius-Skalarprodukt gilt zusätzlich

$$\langle A, B\rangle_F = \langle A^T, B^T\rangle_F \quad \text{für alle } B \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}.$$

Beweis: Mit Bemerkung 2.3.4 (iii) folgt

$$\begin{aligned}\|A\|_S &= \max_{\|x\|_2=1=\|y\|_2}\langle Ax, y\rangle_2 = \max_{\|x\|_2=1=\|y\|_2}\langle x, A^Ty\rangle_2\\ &= \max_{\|y\|_2=1=\|x\|_2}\langle A^Ty, x\rangle_2 = \|A^T\|_2.\end{aligned}$$

Es sei $B \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Dann gilt für das Frobenius-Skalarprodukt

$$\langle A, B\rangle_F = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} A_{i,j}B_{i,j} = \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}(A^T)_{j,i}(B^T)_{j,i} = \langle A^T, B^T\rangle_F.$$

Nach Definition 2.3.5 der Frobeniusnorm folgt

$$\|A\|_F^2 = \langle A, A\rangle_F = \langle A^T, A^T\rangle_F = \|A^T\|_F^2.$$

■

Eine wichtige Klasse von Matrizen sind die orthogonalen Matrizen. Diese werden in der vorliegenden Arbeit viel verwendet, weil sie die betrachteten Normen invariant lassen und somit eine Möglichkeit bieten, die Problemdimension zu reduzieren. Außerdem lassen sich durch orthogonale Matrizen einfach orthogonale Projektionen definieren.

Definition 2.3.14 (Orthogonale Matrizen)
Eine Matrix $Q \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ heißt *orthogonal*, falls $Q^TQ = I_{\mathcal{J}}$ gilt, wobei $I_{\mathcal{J}} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}$ die Identität bezeichnet.

Die folgende Bemerkung fasst zwei einfache Aussagen über orthogonale Matrizen zusammen.

Bemerkung 2.3.15 (Eigenschaften orthogonaler Matrizen)

(i) Die Spalten einer orthogonalen Matrix Q bilden eine orthonormale Basis für den Raum Bild(Q). Insbesondere hat Q vollen Rang.

(ii) Es seien $\mathcal{K} = \{1, \dots, \kappa\}$ eine Indexmenge, $\mathbf{t}_k \subseteq \mathcal{I}$, $k \in \mathcal{K}$, paarweise disjunkte Teilmengen von $\mathcal{I}$ und $\mathcal{J}_k$, $k \in \mathcal{K}$, paarweise disjunkte Indexmengen. Weiter seien $Q_k \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}_k}_{\mathbf{t}_k\times\mathcal{J}_k}$ für alle $k \in \mathcal{K}$ orthogonal. Dann ist $\big(Q_1 \cdots Q_\kappa\big) \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ mit $\mathcal{J} := \dot{\bigcup}_{k\in\mathcal{K}} \mathcal{J}_k$ auch orthogonal.

Beweis: Wir zeigen (ii). Nach Voraussetzung gilt für alle $i, j \in \mathcal{K}$

$$Q_i^TQ_j \;=\; Q_i^T\Pi_{\mathbf{t}_i}\Pi_{\mathbf{t}_j}Q_j \;=\; \delta_{i,j}Q_i^TQ_j \;=\; \begin{cases} I_{\mathcal{J}_i} & \text{, falls } i = j \\ 0 & \text{, falls } i \neq j \end{cases}.$$

Damit ist das Produkt $\big(Q_1, \dots, Q_\tau\big)^T \big(Q_1, \dots, Q_\tau\big)$ gleich der Blockdiagonal-Matrix mit $I_{\mathcal{J}_i}$ als Diagonalblöcke und somit gleich der Identität $I_{\mathcal{J}}$ mit $\mathcal{J} = \dot{\bigcup}_{i=1}^{\tau} \mathcal{J}_i$. ∎

Eine besonders wichtige Eigenschaft der orthogonalen Matrizen ist, dass sie die von uns betrachteten Normen invariant lassen.

Lemma 2.3.16 (Norminvarianz unter orthogonaler Transformation)
Es seien $Q \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $A \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}}$ und $P \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$. Falls Q und P orthogonal sind, gelten

$$\|Qx\|_2 \;=\; \|x\|_2 \quad \text{für alle } x \in \mathbb{R}^{\mathcal{J}},$$

$$\|QA\|_S \;=\; \|A\|_S \quad \text{und} \quad \|QA\|_F \;=\; \|A\|_F$$

und

$$\|AP^T\|_S \;=\; \|A\|_S \quad \text{und} \quad \|AP^T\|_F \;=\; \|A\|_F.$$

Beweis: Es seien Q und P orthogonal. Dann gilt für alle $x \in \mathbb{R}^{\mathcal{J}}$

$$\|Qx\|_2^2 \;=\; \langle Qx, Qx\rangle_2 \;=\; \langle x, Q^TQx\rangle_2 \;=\; \langle x, x\rangle_2 \;=\; \|x\|_2^2.$$

Aus der Invarianz der euklidischen Norm folgt direkt die Aussage für die Spektralnorm (siehe Definition 2.3.3)

$$\|QA\|_S \;=\; \sup_{x\in\mathbb{R}^{\mathcal{J}}\setminus\{0\}} \frac{\|QAx\|_2}{\|x\|_2} \;=\; \sup_{x\in\mathbb{R}^{\mathcal{J}}\setminus\{0\}} \frac{\|Ax\|_2}{\|x\|_2} \;=\; \|A\|_S.$$

Die Aussage mit der orthogonalen Matrix auf der rechten Seite folgt dann aus der Invarianz bezüglich des Adjungierens (siehe Lemma 2.3.13)

$$\|AP^T\|_S \;=\; \|PA^T\|_S \;=\; \|A^T\|_S \;=\; \|A\|_S.$$

Für die Frobeniusnorm folgt die Aussage aus der Darstellung aus Bemerkung 2.3.6 (ii)

$$\|QA\|_F^2 \;=\; \sum_{j\in\mathcal{J}} \|QAe_j\|_2^2 \;=\; \sum_{j\in\mathcal{J}} \|Ae_j\|_2^2 \;=\; \|A\|_F^2.$$

Wie im Fall der Spektralnorm folgt der zweite Teil der Aussage mit Lemma 2.3.13

$$\|AP^T\|_F \;=\; \|PA^T\|_F \;=\; \|A^T\|_F \;=\; \|A\|_F.$$

■

Als Nächstes wenden wir uns den orthogonalen Projektionen in den von uns betrachteten Hilberträumen zu. Dazu sei angemerkt, dass eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ natürlich eine Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_S)$ darstellt. Allerdings ist $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_S)$ kein Hilbertraum, sodass wir die Eigenschaften orthogonaler Projektionen hierbei nicht ausnutzen können.
Als Erstes stellen wir zwei Beispiele orthogonaler Projektionen vor.

Beispiel 2.3.17
Es seien $\mathcal{I}, \mathcal{J}$ Indexmengen und $\mathbf{t} \subseteq \mathcal{I}$. Dann ist die Multiplikation mit der Matrix $\Pi_{\mathbf{t}}$ aus Definition 2.2.5 eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ mit $\mathrm{Bild}(\Pi_{\mathbf{t}}) = \mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$ und in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ mit $\mathrm{Bild}(\Pi_{\mathbf{t}}) = \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathcal{J}}$. Insbesondere sind die Identität und die Nullfunktion orthogonale Projektionen.

Beweis: Es gilt offensichtlich $\Pi_{\mathbf{t}}^2 = \Pi_{\mathbf{t}}$ und $\Pi_{\mathbf{t}}^T = \Pi_{\mathbf{t}}$. Also ist $\Pi_{\mathbf{t}}$ eine Projektion und nach Lemma 2.3.12 in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ und $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ selbstadjungiert. Nach Definition 2.1.4 ist $\Pi_{\mathbf{t}}$ in beiden Räumen eine orthogonale Projektion. ■

Ein weiteres Beispiel für orthogonale Projektionen, die wir in dieser Arbeit häufig verwenden werden, ist durch orthogonale Matrizen definiert.

Beispiel 2.3.18
Es sei $Q \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine orthogonale Matrix. Dann definieren wir $\Pi_Q := QQ^T$. Dies ist als Abbildung $\Pi_Q : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}^{\mathcal{I}}$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ mit $\mathrm{Bild}(\Pi_Q) = \mathrm{Bild}(Q)$. Für eine Indexmenge $\mathcal{K}$ ist die Abbildung $\Pi_Q : \mathbb{R}^{\mathcal{I}\times\mathcal{K}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{K}}, \|.\|_F)$ mit $\mathrm{Bild}(\Pi_Q A) \subseteq \mathrm{Bild}(Q)$ für alle $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$.

Beweis: Nach Definition 2.3.14 der orthogonalen Matrizen gilt $\Pi_Q^2 = QQ^TQQ^T = QQ^T = \Pi_Q$. Des Weiteren gilt $\Pi_Q^T = (QQ^T)^T = QQ^T = \Pi_Q$. Die Aussage folgt wie bei Beispiel 2.3.17. ■

Ein drittes Beispiel für orthogonale Projektionen bezüglich der Frobeniusnorm ist die Anwendung von orthogonalen Projektionen bezüglich der euklidischen Norm von links und von rechts auf eine Matrix. Diese Form nutzen wir später, um Projektionen auf $\mathcal{H}^2$-Matrix-Räume zu definieren.

Beispiel 2.3.19
Es sei $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, Π_ℓ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ mit $\mathrm{Bild}(\Pi_\ell) \subseteq \mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$ und Π_r eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$ mit $\mathrm{Bild}(\Pi_r) \subseteq \mathbb{R}^{J}_{\mathbf{s}}$. Dann definiert $\Pi_{\ell r} : \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $X \mapsto \Pi_\ell X \Pi_r$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ mit $\mathrm{Bild}(\Pi_{\ell r}) \subseteq \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}}$.

Beweis: Für alle $X, Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ gilt

$$\Pi_{\ell r}^2 X \;=\; \Pi_{\ell r}(\Pi_\ell X \Pi_r) \;=\; \Pi_\ell^2 X \Pi_r^2 \;=\; \Pi_\ell X \Pi_r \;=\; \Pi_{\ell r} X$$

und mit Lemma 2.3.12 und 2.3.13

$$\begin{aligned}\langle \Pi_{\ell r} X, Y\rangle_F \;&=\; \langle \Pi_\ell X \Pi_r, Y\rangle_F \;=\; \langle X\Pi_r, \Pi_\ell Y\rangle_F \;=\; \langle \Pi_r X^T, Y^T \Pi_\ell\rangle_F \\ &=\; \langle X^T, \Pi_r Y^T \Pi_\ell\rangle_F \;=\; \langle X, \Pi_\ell Y \Pi_r\rangle_F.\end{aligned}$$

Somit folgt die Behauptung. ∎

Nach Korollar 2.1.8 ist die Operatornorm von orthogonalen Projektionen höchstens eins. In Lemma 2.3.20 zeigen wir, dass orthogonale Projektionen bezüglich der euklidischen Norm die Spektralnorm nicht vergrößern.

Lemma 2.3.20
Es sei Π eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ und $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Dann gilt für alle $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und alle $B \in \mathbb{R}^{\mathcal{J}\times\mathcal{I}}$

$$\|\Pi A\| \;\leq\; \|A\| \quad \text{und} \quad \|B\Pi\| \;\leq\; \|B\|.$$

Beweis: Nach Korollar 2.1.8 gilt $\|\Pi A x\|_2 \leq \|Ax\|_2$ für alle $x \in \mathbb{R}^{\mathcal{J}}$. Mit Lemma 2.3.9 folgt $\|\Pi A\| \leq \|A\|$. Weil beide betrachteten Matrixnormen invariant unter Adjungieren sind (siehe Lemma 2.3.12), folgt

$$\|B\Pi\| \;=\; \|\Pi B^T\| \;\leq\; \|B^T\| \;=\; \|B\|.$$

∎

Mit Hilfe der transponierten Matrix können wir charakterisieren, wann die Bilder zweier Matrizen in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ bzw. in $(\mathbb{R}^{\mathcal{I}\times\mathcal{K}}, \|.\|_F)$ orthogonal zueinander sind.

Lemma 2.3.21
Es seien $A, B \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{K}$ eine Indexmenge. Falls $B^T A = 0 \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}$ gilt, folgt $\mathrm{Bild}(A) \perp \mathrm{Bild}(B)$ in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ und in $(\mathbb{R}^{\mathcal{I}\times\mathcal{K}}, \|.\|_F)$.

Beweis: Es gelte $B^T A = 0$. Es seien $x, y \in \mathbb{R}^{\mathcal{J}}$. Dann gilt

$$\langle Ax, By \rangle \;=\; \langle B^T Ax, y \rangle \;=\; \langle 0x, y \rangle \;=\; 0.$$

Es seien $X, Y \in \mathbb{R}^{\mathcal{J} \times \mathcal{K}}$

$$\langle AX, BY \rangle_F \;=\; \langle B^T AX, Y \rangle_F \;=\; \langle 0X, Y \rangle_F \;=\; 0.$$

■

Zum Abschluss dieses Abschnitts definieren wir noch Durchmesser und Abstand von Mengen. Diese benötigen wir, um später zu beurteilen, ob wir einen Matrixblock approximieren können. Wir verwenden dabei die euklidische Norm, weil diese bei vielen Fehlerabschätzungen auftaucht. Natürlich können Durchmesser und Abstand bez. anderer Metriken definiert werden.

Definition 2.3.22 (Durchmesser und Abstand)
Es seien $\Omega_t, \Omega_s \subseteq \mathbb{R}^{\mathcal{I}}$. Dann definiert

$$\operatorname{diam}(\Omega_t) \;:=\; \sup \{ \|x - y\|_2 \mid x, y \in \Omega_t \}$$

den *Durchmesser* einer Menge und

$$\operatorname{dist}(\Omega_t, \Omega_s) \;=\; \inf \{ \|x - y\|_2 \mid x \in \Omega_t,\, y \in \Omega_s \}$$

den *Abstand* von zwei Mengen.

2.4 Orthogonale Zerlegungen

Wie oben erwähnt, sind sowohl die Frobeniusnorm als auch die Spektralnorm invariant unter orthogonalen Transformationen (siehe Lemma 2.3.16). Deshalb ist die orthogonale Zerlegung ein wichtiges Hilfsmittel zum Reduzieren der Problemgröße.

Definition 2.4.1 (QR-Zerlegung)
Es seien $m, n \in \mathbb{N}$ und $A \in \mathbb{R}^{m \times n}$. Eine *volle QR-Zerlegung* besteht aus einer orthogonalen Matrix $Q \in \mathbb{R}^{m \times m}$ und einer oberen Dreiecksmatrix $R \in \mathbb{R}^{m \times n}$, für die $QR = A$ gilt. Es sei $k := \min\{m, n\}$, $\tilde{Q} \in \mathbb{R}^{m \times k}$ orthogonal und $\tilde{R} \in \mathbb{R}^{k \times n}$ obere Dreiecksmatrix mit $\tilde{Q}\tilde{R} = A$, so nennen wir dies eine *QR-Zerlegung*.

Häufig wird die QR-Zerlegung, wie wir sie definieren, auch als *dünne QR-Zerlegung* bezeichnet. Wir wählen die Bezeichnung QR-Zerlegung, da in dieser Arbeit grundsätzlich dünne QR-Zerlegungen verwendet werden.
Im nächsten Schritt beschäftigen wir uns mit der Berechnung der QR-Zerlegung.

Bemerkung 2.4.2 (Berechnung der QR-Zerlegung)
Es existiert eine Konstante $c_{qr} > 0$ derart, dass eine QR-Zerlegung einer Matrix $A \in \mathbb{R}^{m \times n}$ mit Hilfe von Householder-Spiegelungen in weniger als $c_{qr} \min\{m, n\}mn$ Operationen berechnet werden kann (siehe [22, Abschnitt 5.2]). Die entsprechende Routine nennen wir `QR-Zerlegung_einfach`.

Da wir mit Indexmengen arbeiten und teilweise die QR-Zerlegung von bezüglich der Zeilen eingeschränkten Matrizen benötigen, definieren wir einen Algorithmus zur Berechnung einer QR-Zerlegung einer Matrix $A \in \mathbb{R}^{\mathcal{I} \times \mathcal{J}}_{\mathbf{t} \times \mathcal{J}}$. Dabei ist der Faktor R nur bei entsprechender Anordnung der Indexmengen eine obere Dreiecksmatrix. Da wir diese Eigenschaft im Weiteren nicht benötigen, beeinflusst diese Einschränkung die Arbeit nicht.

Algorithmus 2.4.1 Der Algorithmus berechnet eine QR-Zerlegung $QR = A \in \mathbb{R}^{\mathcal{I} \times \mathcal{J}}_{\mathbf{t} \times \mathcal{J}}$ mit $Q \in \mathbb{R}^{\mathcal{I} \times \mathcal{K}}_{\mathbf{t} \times \mathcal{K}}$, $R \in \mathbb{R}^{\mathcal{K} \times \mathcal{J}}$ und $\#\mathcal{K} = \min\{\#\mathbf{t}, \#\mathcal{J}\}$.

function QR-Zerlegung(A, $\mathcal{I}$, $\mathbf{t}$, $\mathcal{J}$, Q, R, $\mathcal{K}$)
  $m \leftarrow \#\mathbf{t}$, $n \leftarrow \#\mathcal{J}$
  Wähle Bijektionen $\mu : \mathbf{t} \to \{1, \ldots, m\}$ und $\nu : \mathcal{J} \to \{1, \ldots, n\}$
  $\tilde{A} \in \mathbb{R}^{m \times n}$
  for $i \in \{1, ..., m\}, j \in \{1, ..., n\}$ **do**
    $\tilde{A}_{i,j} \leftarrow A_{\mu^{-1}(i), \nu^{-1}(j)}$
  end for
  QR-Zerlegung_einfach($\tilde{A}$, m, n, $\tilde{Q}$, $\tilde{R}$)
  Wähle Indexmenge $\mathcal{K}$ mit $\#\mathcal{K} = \min\{m, n\}$
  Wähle Bijektion $\kappa : \mathcal{K} \to \{1, \ldots, \min\{m, n\}\}$
  $Q \in \mathbb{R}^{\mathcal{I} \times \mathcal{K}}_{\mathbf{t} \times \mathcal{K}}$, $R \in \mathbb{R}^{\mathcal{K} \times \mathcal{J}}$
  for $k \in \mathcal{K}$ **do**
    for $i \in \mathbf{t}$ **do**
      $Q_{i,k} \leftarrow \tilde{Q}_{\mu(i), \kappa(k)}$
    end for
    for $j \in \mathcal{J}$ **do**
      $R_{k,j} \leftarrow \tilde{R}_{\kappa(k), \nu(j)}$
    end for
  end for
end function

Lemma 2.4.3 (Berechnung einer QR-Zerlegung)
Es sei $A \in \mathbb{R}^{\mathcal{I} \times \mathcal{J}}_{\mathbf{t} \times \mathcal{J}}$. Dann berechnet Algorithmus 2.4.1 eine Indexmenge $\mathcal{K}$ mit $\#\mathcal{K} = \min\{\#\mathbf{t}, \#\mathcal{J}\}$, eine orthogonale Matrix $Q \in \mathbb{R}^{\mathcal{I} \times \mathcal{K}}_{\mathbf{t} \times \mathcal{K}}$ und eine Matrix $R \in \mathbb{R}^{\mathcal{K} \times \mathcal{J}}$ mit $A = QR$ in weniger als $c_{qr} \min\{\#\mathbf{t}, \#\mathcal{J}\}\#\mathbf{t}\#\mathcal{J}$ Operationen.

Beweis: Lediglich im Aufruf von `QR-Zerlegung_einfach` werden arithmetische Operationen in Algorithmus 2.4.1 benötigt. Der Aufwand für diesen Aufruf lässt sich nach Bemerkung 2.4.2 durch $c_{qr} \min\{m, n\}mn = c_{qr} \min\{\#\mathbf{t}, \#\mathcal{J}\}\#\mathbf{t}\#\mathcal{J}$ beschränken.

Für die berechneten Matrizen Q und R erhalten wir unter Verwendung der Notation in Algorithmus 2.4.1 für alle $i \in \mathbf{t}$ und $j \in \mathcal{J}$

$$(QR)_{i,j} \;=\; \sum_{k \in \mathcal{K}} Q_{i,k} R_{k,j} \;=\; \sum_{k \in \mathcal{K}} \tilde{Q}_{\mu(i),\kappa(k)} \tilde{R}_{\kappa(k),\nu(j)} \;=\; \tilde{A}_{\mu(i),\nu(j)} \;=\; A_{i,j}$$

und für alle $k, l \in \mathcal{K}$

$$(Q^T Q)_{k,l} \;=\; \sum_{i \in \mathbf{t}} Q_{i,k} Q_{i,l} \;=\; \sum_{i \in \mathcal{I}} \tilde{Q}_{\mu(i),\kappa(k)} \tilde{Q}_{\mu(i),\kappa(l)} \;=\; \delta_{\kappa(k),\kappa(l)} \;=\; \delta_{k,l}.$$

Für alle $i \in \mathcal{I} \backslash \mathbf{t}$ und alle $k \in \mathcal{K}$ ist $Q_{i,k} = 0$ und somit $(QR)_{i,j} = 0 = A_{i,j}$ für alle $j \in \mathcal{J}$. Also gilt $QR = A$ und $Q^T Q = I$ (somit ist Q orthogonal). ■

Wir können also auch zu Matrizen mit Indexmengen eine QR-Zerlegung berechnen, da wir Bijektionen in endliche Teilmengen von $\mathbb{N}$ finden.
Als zweite orthogonale Zerlegung betrachten wir die Singulärwertzerlegung. Wie schon in der Einleitung beschrieben, sind die Teilmatrizen meistens nicht von niedrigem Rang, lassen sich aber durch Matrizen mit niedrigem Rang approximieren. Beim Approximieren der Problemmatrizen von diskretisierten Integraloperatoren durch $\mathcal{H}^2$-Matrizen lässt sich die Berechnung der Niedrigrang-Approximationen gut mit analytischen Methoden wie z. B. der Taylor-Entwicklung oder Interpolation lösen (siehe [12, Kapitel 2 bzw. Kapitel 4] und [15]).
Während der Arithmetik fehlen die analytischen Informationen, um solche Methoden anzuwenden. Deshalb verwenden wir in der Arithmetik die Singulärwertzerlegung, um Niedrigrang-Approximationen von gewissen Matrizen zu berechnen.

Lemma 2.4.4 (Volle Singulärwertzerlegung)
Es seien $m, n \in \mathbb{N}$ und $A \in \mathbb{R}^{m \times n}$. Dann existieren orthogonale Matrizen $Q \in \mathbb{R}^{m \times m}$ und $P \in \mathbb{R}^{n \times n}$ und eine Diagonalmatrix $\Sigma = \operatorname{diag}(\sigma_i) \in \mathbb{R}^{m \times n}$, $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_{\min\{m,n\}} \geq 0$, mit

$$A \;=\; Q \Sigma P^T.$$

Dies ist die sogenannte *volle Singulärwertzerlegung* und die σ_i werden *Singulärwerte* genannt. Die Spalten von Q sind die *linken Singulärvektoren* und die Spalten von P sind die *rechten Singulärvektoren.*

Beweis: Siehe [22, Theorem 2.5.2]. ■

Die Singulärwertzerlegung einer Matrix ist i. A. nicht eindeutig. Allerdings sind die Singulärwerte durch ihre abfallende Reihenfolge eindeutig bestimmt. Analog zur QR-Zerlegung definieren wir die *dünne Singulärwertzerlegung.*

Definition 2.4.5 (Dünne Singulärwertzerlegung)
Es seien $m, n \in \mathbb{N}$ und $A \in \mathbb{R}^{m \times n}$. Es sei $k := \min\{m, n\}$, $Q \in \mathbb{R}^{m \times k}$ und $P \in \mathbb{R}^{n \times k}$ orthogonal und $\Sigma = \text{diag}(\sigma_i) \in \mathbb{R}^{k \times k}$, $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_k \geq 0$, eine Diagonalmatrix mit

$$A \;=\; Q\Sigma P^T.$$

Dann ist $Q\Sigma P^T$ eine *dünne Singulärwertzerlegung* von A. Da wir im Folgenden immer die dünne Singulärwertzerlegung berechnen, sprechen wir einfach von der Singulärwertzerlegung.

Die Existenz folgt daraus, dass für die rechteckige Diagonalmatrix $\Sigma = \Pi_{\{1,...,k\}}\Sigma\Pi_{\{1,...,k\}}$ gilt. Daraus folgt

$$Q\Sigma P^T \;=\; Q\Pi_{\{1,...,k\}}\Sigma\Pi_{\{1,...,k\}}P^T \;=\; (Q\Pi_{\{1,...,k\}})(\Pi_{\{1,...,k\}}\Sigma\Pi_{\{1,...,k\}})(P\Pi_{\{1,...,k\}})^T.$$

Wir brauchen also nur die ersten k Zeilen von Q und P und den linken oberen $k \times k$-Block von Σ. Dies ist die dünne Singulärwertzerlegung.
Die Singulärwertzerlegung kann in zwei Schritten berechnet werden. Als Erstes wird die Matrix auf Bidiagonal-Gestalt gebracht (siehe [22, Abschnitt 5.4]) und dann werden per Golub-Kahan-Schritte die Singulärwerte berechnet (siehe [22, Abschnitt 8.6]).

Bemerkung 2.4.6 (Berechnung der Singulärwertzerlegung)
Wie auch in anderen Arbeiten (vgl. [12,25]) nehmen wir an, dass die Anzahl der Iterationen für die Berechnung der Singulärwertzerlegung sich durch eine Konstante abschätzen lässt (siehe [22]). Dann existiert eine Konstante $c_{svd} > 0$ so, dass die Berechnung der Singulärwertzerlegung von $A \in \mathbb{R}^{m \times n}$ höchstens $c_{svd}\, m\, n\, \min\{m, n\}$ Operationen benötigt (siehe [22, Abschnitt 5.4, 8.6]). Wir bezeichnen die entsprechende Funktion mit `SVD`.

Es sei $r := \text{Rang}(A)$ der Rang der Matrix $A \in \mathbb{R}^{n \times n}$. Dann sind die Singulärwerte $\sigma_{r+1}, ..., \sigma_n$ alle gleich null. D. h. , dass sich anhand der Singulärwerte schnell der Rang einer Matrix bestimmen lässt. In der Praxis führen Rundungsfehler dazu, dass die Berechnungen nicht exakt durchgeführt werden. Deshalb wird häufig der numerische Rang verwendet (siehe [22, Abschnitt 5.5.8]).
Das nächste Lemma zeigt, dass mit Hilfe der Singulärwertzerlegung eine Niedrigrang-Approximation einer Matrix berechnet werden kann, bei der der Fehler in der Spektralnorm und der Frobeniusnorm exakt angegeben werden kann.

Lemma 2.4.7 (Fehler der gekürzten Singulärwertzerlegung)
Es sei $A \in \mathbb{R}^{m \times n}$, $Q\Sigma P^T$ die Singulärwertzerlegung von A und $k \in \mathbb{N}_{\leq \min\{m,n\}}$. Weiter sei $\overline{\Sigma} = \Pi_{\{1,...,k\}}\Sigma\Pi_{\{1,...,k\}}$. Dann gilt

$$\|Q\Sigma P^T - Q\overline{\Sigma}P^T\|_2 \;=\; \sigma_{k+1} \quad \text{und} \quad \|Q\Sigma P^T - Q\overline{\Sigma}P^T\|_F \;=\; \left(\sum_{i=k+1}^{\min\{m,n\}} \sigma_i^2\right)^{\frac{1}{2}}.$$

Insbesondere erhalten wir für $k = 0$ eine Darstellungen für die Norm von A.

Beweis: Die Aussage gilt nach [32, Satz 2.4.1], wobei nach dem Satz, dass $Q\overline{\Sigma}P^T$ die Bestapproximation von A in der Menge der Matrizen mit Rang k ist. ■

Wir nutzen die Singulärwertzerlegung, um eine Niedrigrang-Approximation für eine Matrix zu berechnen. Dabei wählen wir den Rang mit Hilfe des Lemmas 2.4.7 so, dass der absolute/relative Fehler in der Spektral-/Frobeniusnorm unter einer vorgegebenen Schranke $\epsilon \geq 0$ bleibt. Algorithmus 2.4.2 bestimmt diesen Rang.

Algorithmus 2.4.2 [siehe [12, Algorithm 17]] Der Algorithmus berechnet den adaptiven Rang k aus den Singulärwerten in Σ, wobei *opt* die Fehlerart angibt und $\epsilon \geq 0$ die Fehlerschranke. Für die $\sigma_1 = 0$ ist A die Nullmatrix und wir setzen $k = 0$.

```
function ADAPTIVER_RANG((σ)_{i=1}^n, k, opt, ε)
    k ← 0
    if σ_1 ≠ 0 then
        if opt = Spektralnorm, absolut then
            while (k < n) ∧ (σ_{k+1} ≥ ε) do
                k ← k + 1
            end while
        else if opt = Spektralnorm, relativ then
            while (k < n) ∧ (σ_{k+1} ≥ εσ_1) do
                k ← k + 1
            end while
        else if opt = Frobeniusnorm, absolut then
            a ← σ_1^2
            while (k < n) ∧ (a ≥ ε^2) do
                k ← k + 1
                a ← a + σ_{k+1}^2
            end while
        else if opt = Frobeniusnorm, relativ then
            b ← 0
            for i ∈ {1, ..., n} do
                b ← b + σ_i^2
            end for
            a ← σ_1^2
            while (k < n) ∧ (a ≥ ε^2 b) do
                k ← k + 1
                a ← a + σ_{k+1}^2
            end while
        end if
    end if
end function
```

Bemerkung 2.4.8 (Berechnung des adaptiven Rangs)
Es seien die Singulärwerte $\sigma_1, ..., \sigma_n$, eine Fehlertoleranz $\epsilon \geq 0$ und eine Fehlerart *opt*

gegeben. Dann berechnet `adaptiver_Rang` den minimalen Rang, für den der Fehler $Q\Sigma P^T - Q\overline{\Sigma}P^T$ in der durch *opt* gegebenen Fehlerart kleiner als ϵ ist, in höchstens $4n$ Operationen. Die Fehlerabschätzung folgt aus Lemma 2.4.7.

Wir können also den Rang so wählen, dass der Fehler kontrollierbar ist. Für diese Arbeit ist es von besonderem Interesse, eine orthogonale Matrix Q zu einer gegebenen Matrix A zu finden, so dass der Projektionsfehler $(I - \Pi_Q)A$ nicht größer als eine gegebene Fehlerschranke ist. Diese Matrix können wir mit Hilfe von `SVD` und `adaptiver_Rang` berechnen.

Algorithmus 2.4.3 Der Algorithmus berechnet eine orthogonale Matrix $Q \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}_{\mathbf{t}\times\mathcal{K}}$ für die gegebene Matrix $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathcal{K}}$ derart, dass der Projektionsfehler $(I - \Pi_Q)A$ in der gegebenen Fehlerart *opt* kleiner als $\epsilon \geq 0$ ist.

function APAPTIVE_ORTHOGONALE_MATRIX(A, $\mathcal{I}$, $\mathbf{t}$, $\mathcal{J}$, Q, $\mathcal{K}$, *opt*, ϵ)
  $m \leftarrow \#\mathbf{t}$, $n \leftarrow \#\mathcal{J}$
  Wähle Bijektionen $\mu : \mathbf{t} \to \{1, \ldots, m\}$ und $\nu : \mathcal{J} \to \{1, \ldots, n\}$
  $\tilde{A} \in \mathbb{R}^{m\times n}$
  for $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$ **do**
    $\tilde{A}_{i,j} \leftarrow A_{\mu^{-1}(i),\nu^{-1}(j)}$
  end for
  SVD($\tilde{A}$, $\tilde{Q}$, $\tilde{\Sigma}$, $\tilde{P}$)
  ADAPTIVER_RANG($\tilde{\Sigma}$, k, *opt*, ϵ)
  Wähle Indexmenge $\mathcal{K}$ mit $\#\mathcal{K} = k$ und Bijektion $\kappa : \mathcal{K} \to \{1, \ldots, k\}$
  $Q \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$
  for $i \in \mathbf{t}$, $k \in \mathcal{K}$ **do**
    $Q_{i,k} \leftarrow \tilde{Q}_{\mu(i),\kappa(k)}$
  end for
end function

Lemma 2.4.9
Es existiert $c_{apr} > 0$ derart, dass für alle $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathcal{J}}$, $\epsilon > 0$ und Fehlerarten *opt* der Algorithmus 2.4.3 für die Berechnung der Matrix Q weniger als $c_{apr} \min\{\#\mathbf{t}, \#\mathcal{J}\}\#\mathbf{t}\#\mathcal{J}$ Operationen benötigt. Der Projektionsfehler $(I - \Pi_Q)A$ ist in der gegebenen Fehlerart *opt* kleiner als ϵ.

Beweis: Der Aufwand für die Berechnung der Singulärwertzerlegung mit `SVD` ist kleiner als $c_{svd} \min\{\#\mathbf{t}, \#\mathcal{J}\}\#\mathbf{t}\#\mathcal{J}$. Weil wir $\min\{\#\mathbf{t}, \#\mathcal{J}\}$ Singulärwerte haben, benötigt der Aufruf von `adaptiver_Rang` höchstens $4\min\{\#\mathbf{t}, \#\mathcal{J}\}$ Operationen (siehe Bemerkung 2.4.8). Also können wir $c_{apr} > 0$ so wählen, dass der Aufwand für den Aufruf der Funktion `apaptive_orthogonale_Matrix` kleiner als $c_{apr} \min\{\#\mathbf{t}, \#\mathcal{J}\}\#\mathbf{t}\#\mathcal{J}$ ist.
Um den Fehler zu untersuchen, betrachten wir zuerst die Projektion von $\tilde{A}$

$$\begin{aligned}\Pi_{\tilde{Q}\Pi_{\{1,\ldots,k\}}}\tilde{A} &= \tilde{Q}\Pi_{\{1,\ldots,k\}}\Pi_{\{1,\ldots,k\}}\tilde{Q}^T\tilde{Q}\tilde{\Sigma}\tilde{P}^T = \tilde{Q}\Pi_{\{1,\ldots,k\}}\tilde{\Sigma}\tilde{P}^T \\ &= \tilde{Q}\Pi_{\{1,\ldots,k\}}\tilde{\Sigma}\Pi_{\{1,\ldots,k\}}\tilde{P}^T = \tilde{Q}\overline{\Sigma}\tilde{P}^T,\end{aligned}$$

wobei $\overline{\Sigma}$ der Matrix aus Lemma 2.4.7 entspricht. Damit folgt, dass $(I - \Pi_{\tilde{Q}\Pi_{\{1,\dots,k\}}})\tilde{A}$ in der entsprechenden Fehlerart kleiner als ϵ ist. Weil für alle $i \in \mathbf{t}$ und alle $j \in \mathcal{J}$

$$((I - \Pi_{\tilde{Q}\Pi_{\{1,\dots,k\}}})\tilde{A})_{\mu(i),\nu(j)} \;=\; ((I - \Pi_Q)A)_{i,j}$$

gilt und für alle $i \in \mathcal{I}\backslash\mathbf{t}$ und alle $j \in \mathcal{J}$ die rechte Seite gleich null ist, folgt die Fehlerschranke auch für $(I - \Pi_Q)A$. ∎

Wie in der Einführung erwähnt, basiert unser Ansatz für die Arithmetik auf Niedrigrangupdates. Um diese effizient durchzuführen, benötigen wir eine effiziente Darstellung für Niedrigrangmatrizen.

Definition 2.4.10 (Rang-k-Matrizen)
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $k \in \mathbb{N}$. Dann ist X eine *Rang-k-Matrix*, falls $\mathrm{Rang}(X) \leq k$ gilt. Falls $X = AB^T$ für $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$ und $B \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}}$ mit $\#\mathcal{K} = k$ gilt, ist (A, B) eine *Rang-k-Darstellung* bzw. *Rang-$\mathcal{K}$-Darstellung* von X. Falls der Rang nicht näher bestimmt ist, bezeichnen wir diese Matrizen auch als Niedrigrangmatrizen.

Die Rang-k-Darstellung wird auch bei den $\mathcal{H}$-Matrizen verwendet, um die Matrixblöcke mit niedrigem Rang effizient zu speichern. Für eine Rang-k-Matrix lässt sich eine Rang-k-Darstellung zum Beispiel per Singulärwertzerlegung konstruieren.

Bemerkung 2.4.11
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ mit $\mathrm{Rang}(X) = k$ und $\mathcal{K}$ eine Indexmenge mit $\#\mathcal{K} = k$. Dann existieren $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$ und $B \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}}$ mit $X = AB^T$.

Später benötigen wir das Kürzen von Rang-k-Matrizen, d. h. die Berechnung einer Approximation mit niedrigerem Rang. Der entsprechende Algorithmus nutzt die Singulärwertzerlegung. Die Singulärwertzerlegung direkt von X zu berechnen, würde zu einem Aufwand von $c_{apr} \min\{\#\mathcal{I}, \#\mathcal{J}\}\#\mathcal{I}\#\mathcal{J}$ führen. Deshalb reduzieren wir das Problem mit Hilfe der QR-Zerlegungen $Q_A R_A = A$ und $Q_B R_B = B$. Für die Matrix $C := R_A R_B^T$ berechnen wir dann mit `adaptive_orthogonale_Matrix` eine orthogonale Matrix Q mit hinreichend kleinem Projektionsfehler. Damit erhalten wir

$$X \;=\; AB^T \;=\; Q_A R_A R_B^T Q_B^T \;=\; Q_A C Q_B^T \;\approx\; Q_A Q Q^T C Q_B^T \;=\; (Q_A Q)(Q_B C^T Q)^T.$$

Dieser Ansatz führt zu Algorithmus 2.4.4.

Lemma 2.4.12
Es sei $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ mit Rang-$\mathcal{K}$-Darstellung (A, B) und $k = \#\mathcal{K}$. Dann benötigt der Algorithmus 2.4.4 höchstens

$$(c_{qr} + 2)k^2(\#\mathcal{I} + \#\mathcal{J}) + (c_{apr} + 4)k^3$$

Operationen. Die Differenz $AB^T - \widetilde{A}\widetilde{B}^T$ ist in der gegebenen Fehlerart *opt* kleiner als ϵ.

Algorithmus 2.4.4 [siehe [26, Lemma 1.3]] Die Funktion berechnet aus einer Rang-κ_R-Matrix $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine gekürzte Rang-$\kappa_{\widetilde{R}}$-Matrix $\widetilde{R} = \widetilde{A}\widetilde{B}^T$, wobei die Differenz $R - \widetilde{R}$ in der vorgegebenen Fehlerart *opt* kleiner als eine vorgegebene Toleranz $\epsilon \geq 0$ ist.

function RkMatrix_Kürzen(R, $\widetilde{R}$, *opt*, ϵ)
  QR-Zerlegung(A, $\mathcal{I}$, $\mathcal{I}$, κ_R, Q_A, R_A, ρ_A)
  QR-Zerlegung(B, $\mathcal{I}$, $\mathcal{I}$, κ_R, Q_B, R_B, ρ_B) ▷ Algorithmus 2.4.1
  $C \leftarrow R_A R_B^T \in \mathbb{R}^{\rho_A\times\rho_B}$
  apaptive_orthogonale_Matrix(C, ρ_A, ρ_A, ρ_B, Q, $\kappa_{\widetilde{R}}$, *opt*, ϵ)
    ▷ Algorithmus 2.4.3
  $P \leftarrow C^T Q \in \mathbb{R}^{\rho_B\times\kappa_{\widetilde{R}}}$
  $\widetilde{A} \leftarrow Q_A Q \in \mathbb{R}^{\mathcal{I}\times\kappa_{\widetilde{R}}}$
  $\widetilde{B} \leftarrow Q_B P \in \mathbb{R}^{\mathcal{J}\times\kappa_{\widetilde{R}}}$
end function

Beweis:[siehe [26, Lemma 1.3]] Die Berechnung der QR-Zerlegungen von A und B benötigt höchstens

$$c_{qr}\#\mathcal{I}k\min\{\#\mathcal{I},k\} + c_{qr}\#\mathcal{J}k\min\{\#\mathcal{J},k\} \;\leq\; c_{qr}k^2(\#\mathcal{I}+\#\mathcal{J})$$

Operationen (siehe Lemma 2.4.3). Für das Aufstellen der Matrix C werden höchstens $2\#\rho_A\#\kappa\#\rho_B \leq 2k^3$ Operationen benötigt. Der Aufwand für die Berechnung der Matrix Q mit Algorithmus 2.4.3 ist nach Lemma 2.4.9 höchstens

$$c_{apr}\#\rho_A\#\rho_B\min\{\#\rho_A,\#\rho_B\} \;\leq\; c_{apr}k^3.$$

Wegen $\#\tilde{\mathcal{K}} \leq \#\mathcal{K} = k$ ist der Aufwand für das Aufstellen von P durch $2k^3$ beschränkt. Die Multiplikation von Q_A mit Q und von Q_B mit P benötigt höchstens $2\#\mathcal{I}\#\mathcal{K}\#\tilde{\mathcal{K}} \leq 2k^2\#\mathcal{I}$ bzw. $2\#\mathcal{J}\#\mathcal{K}\#\tilde{\mathcal{K}} \leq 2k^2\#\mathcal{J}$ Operationen. Zusammen ergibt sich folgende obere Schranke für die Berechnung der neuen Rang-$\tilde{\mathcal{K}}$-Matrix

$$(c_{qr}+2)k^2(\#\mathcal{I}+\#\mathcal{J}) + (c_{apr}+4)k^3.$$

Die Abschätzung des Fehlers folgt wegen der Orthogonalität der Matrizen Q_A und Q_B aus Lemma 2.3.16 und Lemma 2.4.9. ∎

Mit Hilfe des Kürzungsalgorithmus können wir direkt einen Algorithmus zur Addition zweier Niedrigrangmatrizen definieren. Dabei nutzen wir aus, dass die Summe zweier Matrizen $R = AB^T, \widetilde{R} = \widetilde{A}\widetilde{B}^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ mit Rang-κ/$\tilde{\kappa}$-Darstellung die Rang-$\kappa\,\dot\cup\,\tilde{\kappa}$-Darstellung

$$R + \widetilde{R} \;=\; AB^T + \widetilde{A}\widetilde{B}^T \;=\; \begin{pmatrix} A & \widetilde{A}\end{pmatrix}\begin{pmatrix} B & \widetilde{B}\end{pmatrix}^T$$

besitzt. Damit erhalten wir den Algorithmus 2.4.5, dessen Aufwand wir in Lemma 2.4.13 abschätzen.

Algorithmus 2.4.5 [vergleiche [26, Definition 1.4]] Die Funktion berechnet die approximierte Summe einer Rang-κ_R-Matrix $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und einer Rang-$\kappa_{\widetilde{R}}$-Matrix $\widetilde{R} = \widetilde{A}\widetilde{B}^T$, wobei R mit dem Ergebnis überschrieben wird.

function RkMatrix_Summe($\tilde{R}$, R, *opt*, ϵ)
  $\widehat{A} \leftarrow \begin{pmatrix} A & \widetilde{A} \end{pmatrix} \in \mathbb{R}^{\mathcal{I}\times(\kappa_R \dot\cup \kappa_{\widetilde{R}})}$
  $\widehat{B} \leftarrow \begin{pmatrix} B & \widetilde{B} \end{pmatrix} \in \mathbb{R}^{\mathcal{J}\times(\kappa_R \dot\cup \kappa_{\widetilde{R}})}$
  RkMatrix_Kürzen($(\widehat{A}, \widehat{B})$, R, *opt*, ϵ) ▷ Algorithmus 2.4.4
end function

Lemma 2.4.13
Es sei $R, \widetilde{R} \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ mit Rang-κ_R-Darstellung (A, B) bzw. mit Rang-$\kappa_{\widetilde{R}}$-Darstellung $(\widetilde{A}, \widetilde{B})$ und $k = \#\kappa_R + \#\kappa_{\widetilde{R}}$. Dann benötigt der Algorithmus 2.4.5 höchstens

$$(c_{qr} + 2)k^2(\#\mathcal{I} + \#\mathcal{J}) + (c_{apr} + 4)k^3$$

Operationen.

Beweis: [siehe [26, Bemerkung 1.5]] Arithmetische Operationen werden nur während des Aufrufs von Algorithmus 2.4.4 benötigt. Mit Lemma 2.4.12 folgt die Aufwandsabschätzung. ■

2.5 Modellproblem

Als Modellproblem verwenden wir die gleiche 1D-Integralgleichung wie in [12, Kapitel 2] oder [31]. Weil für diese Arbeit vor allem die algebraische Struktur der $\mathcal{H}^2$-Matrizen von Interesse ist, reicht uns dieses simple Beispiel, um die Prinzipien der hierarchischen Matrizen zu veranschaulichen.

1D-Modellproblem

Wir untersuchen die Integralgleichung

$$\mathcal{G}[u](x) := \int_0^1 -\log|x-y|u(y)dy \quad \text{für alle } x \in [0,1],$$

für Funktionen $u \in L^2[0,1]$. Diese Integralgleichung diskretisieren wir mit der Galerkin-Methode unter Verwendung von stückweise konstanten Basisfunktionen. Für $n \in \mathbb{N}$ ist der n-dimensionale Ansatzraum durch die Basisfunktionen $(\phi_i)_{i=1}^n$ mit

$$\phi_i : [0,1] \to \mathbb{R},\ x \mapsto \begin{cases} 1 & \text{, falls } x \in [\frac{i-1}{n}, \frac{i}{n}] \\ 0 & \text{, sonst} \end{cases}$$

für alle $i \in \{1, \ldots, n\} =: \mathcal{I}$. Dieser Ansatz führt zur Matrix $G \in \mathbb{R}^{n \times n}$ mit Einträgen

$$G_{i,j} := \int_0^1 \phi_i(x) \int_0^1 -\log|x-y|\phi_j(y)dy\,dx = \int_{\frac{i-1}{n}}^{\frac{i}{n}} \int_{\frac{j-1}{n}}^{\frac{j}{n}} -\log|x-y|dy\,dx$$

für alle $i, j \in \mathcal{I}$. Mit der Kernfunktion

$$g : [0,1]^2 \to \mathbb{R},\ (x,y) \mapsto \begin{cases} 0 & \text{, falls } x = y \\ -\log|x-y| & \text{, falls } x \neq y \end{cases}$$

bekommen die Matrixeinträge für alle $i, j \in \mathcal{I}$ folgende Gestalt

$$G_{i,j} = \int_0^1 \phi_i(x) \int_0^1 g(x,y)\phi_j(y)dy\,dx = \int_{\frac{i-1}{n}}^{\frac{i}{n}} \int_{\frac{j-1}{n}}^{\frac{j}{n}} g(x,y)dy\,dx.$$

Im Gegensatz zur Diskretisierung von partiellen Differentialgleichungen mit Finiten Elementen erhalten wir bei Integralgleichungen, wie der hier vorgestellten, eine vollbesetzte Matrix, weil die Kernfunktion nicht lokal ist. In dem konkreten Fall bedeutet dies, dass der Träger $\text{supp}(g) = [0,1]^2$ ist und $G_{i,j} \neq 0$ für alle $i, j \in \mathcal{I}$.
Weil wir typischerweise n sehr groß wählen müssen, um den entstehenden Diskretisierungsfehler hinreichend zu verkleinern, ergeben sich für die vollbesetzte Matrix zwei Probleme. Das erste betrifft den Speicherbedarf für das direkte Speichern, das zweite Problem ist das Lösen eines vollbesetzten Gleichungssystems.
Wie wir schon in der Einleitung erwähnt haben, lösen die hierarchischen Matrizen diese Probleme mit Hilfe von lokalen Niedrigrang-Approximationen. Diese ermöglichen eine datenschwache Speicherung und effiziente Arithmetik. Die Niedrigrang-Approximation konstruieren wir durch eine Approximation der Kernfunktion, welche die Variablen x und y separiert.

Approximation der Kernfunktion

Für unser Beispiel wählen wir die Taylor-Entwicklung. Da wir die eindimensionale Entwicklung anwenden wollen, definieren wir die Funktion

$$f : \mathbb{R}_{>0} \to \mathbb{R},\ z \mapsto -\log(z).$$

Mit f lässt sich die Kernfunktion g für alle $(x,y) \in [0,1]^2$ durch

$$g(x,y) = \begin{cases} 0 & \text{, falls } x = y, \\ f(|x-y|) & \text{, falls } x \neq y \end{cases}$$

darstellen. Wir wollen g in einer Umgebung von $(x_0, y_0) \in [0,1]^2$, $x_0 \neq y_0$, approximieren. (Dabei schließen wir den Fall $x_0 = y_0$ aus, weil dort eine Singularität liegt.) Zuerst

untersuchen wir den Fall $x_0 > y_0$. Dazu approximieren wir die Funktion f mit der Taylor-Entwicklung der Ordnung $m \in \mathbb{N}_{>0}$ um den Entwicklungspunkt $z_0 := x_0 - y_0 \in \mathbb{R}_{>0}$ und erhalten für $z \in \mathbb{R}$

$$f_{z_0,m}(z) \;:=\; \sum_{\alpha=0}^{m-1} f^{(\alpha)}(z_0)\frac{(z-z_0)^\alpha}{\alpha!}.$$

(Wir definieren die Taylor-Entwicklung auf ganz $\mathbb{R}$ und untersuchen später, in welchem Bereich gewisse Fehlerschranken gelten.) Mit Hilfe der Approximation von f erhalten wir auch eine Approximation von g. Wir definieren

$$g_{(x_0,y_0),m} : [0,1]^2,\ (x,y) \;\mapsto\; f_{z_0,m}(x-y).$$

Für diese Funktion gilt

$$\begin{aligned} g_{(x_0,y_0),m}(x,y) \;&=\; f_{z_0,m}(x-y) \;=\; \sum_{\alpha=0}^{m-1} f^{(\alpha)}(z_0)\frac{(x-y-z_0)^\alpha}{\alpha!} \\ &=\; \sum_{\alpha=0}^{m-1} f^{(\alpha)}(x_0-y_0)\frac{(x-y-x_0+y_0)^\alpha}{\alpha!}. \end{aligned}$$

Wir formen die Potenz mit Hilfe des binomischen Lehrsatzes und erhalten

$$\begin{aligned} g_{(x_0,y_0),m}(x,y) \;&=\; \sum_{\alpha=0}^{m-1} f^{(\alpha)}(x_0-y_0)\frac{(x-y-x_0+y_0)^\alpha}{\alpha!} \\ &=\; \sum_{\alpha=0}^{m-1} \frac{f^{(\alpha)}(x_0-y_0)}{\alpha!} \sum_{\mu=0}^{\alpha} \binom{\alpha}{\mu} (x-x_0)^{\alpha-\mu}(y_0-y)^\mu \\ &=\; \sum_{\alpha=0}^{m-1}\sum_{\mu=0}^{\alpha} \alpha!\, \frac{f^{(\alpha)}(x_0-y_0)}{\alpha!}\, \frac{(x-x_0)^{\alpha-\mu}}{(\alpha-\mu)!}\, (-1)^\mu \frac{(y-y_0)^\mu}{\mu!} \\ &=\; \sum_{\nu=0}^{m-1}\sum_{\mu=0}^{m-1-\nu} (-1)^\mu f^{(\nu+\mu)}(x_0-y_0)\, \frac{(x-x_0)^\nu}{\nu!}\, \frac{(y-y_0)^\mu}{\mu!}. \end{aligned}$$

Die Summanden in der Doppelsumme am Ende lassen sich in drei Faktoren aufteilen. Dabei ist der erste Faktor $(-1)^\mu f^{(\nu+\mu)}(x_0-y_0)$ unabhängig von x und y. Der zweite Faktor $(x-x_0)^\nu/\nu!$ und der dritte Faktor $(y-y_0)^\mu/\mu!$ hängen nur von x bzw. y ab. Wenn wir die approximierte Kernfunktion $g_{(x_0,y_0),m}$ statt des ursprünglichen g benutzen, erhalten wir somit folgende approximierte Matrixeinträge für alle $i,j \in \mathcal{I}$

$$\begin{aligned} \tilde{G}_{i,j} \;&:=\; \int_{\frac{i-1}{n}}^{\frac{i}{n}} \int_{\frac{j-1}{n}}^{\frac{j}{n}} g_{(x_0,y_0),m}(x,y)\,dy\,dx \\ &=\; \sum_{\nu=0}^{m-1}\sum_{\mu=0}^{m-1-\nu} \underbrace{(-1)^\mu f^{(\nu+\mu)}(x_0-y_0)}_{=:S_{\nu\mu}} \underbrace{\int_{\frac{i-1}{n}}^{\frac{i}{n}} \frac{(x-x_0)^\nu}{\nu!}dx}_{=:V_{i\nu}} \underbrace{\int_{\frac{j-1}{n}}^{\frac{j}{n}} \frac{(y-y_0)^\mu}{\mu!}dy}_{=:W_{j\nu}}. \end{aligned}$$

Die Approximation der Kernfunktion durch die Funktion $g_{(x_0,y_0),m}$, bei der die Variablen x und y separiert sind, ermöglicht uns die faktorisierte Darstellung

$$G \approx \tilde{G} = VSW^T,$$

wobei wir $S_{i,j} = 0$ für alle $i, j \in \{0, \ldots, m-1\}$ mit $\nu + \mu \geq m$ setzen. Wir können also die Matrix $\tilde{G}$ durch die Matrizen V, S und W darstellen. Für die vollbesetzte Darstellung der Matrix $\tilde{G}$ müssen wir n^2 Einträge speichern. Die Matrizen V und W haben jeweils nm Einträge und S sogar nur m^2. Also wird der Speicherbedarf von n^2 auf $2nm + m^2$ reduziert und somit der Speicheraufwand für $m \ll n$ deutlich verkleinert. Der Aufwand für die Matrix-Vektor-Multiplikation lässt sich von $2n^2$ auf $2nm + 2m^2$ Operationen reduzieren, indem wir nacheinander mit W^T, S und V multiplizieren.
Für den Fall $x_0 < y_0$ definieren wir $z_0 := y_0 - x_0$ und $g_{(x_0,y_0),m}(x, y) := f_{z_0,m}(y - x)$ für alle $x, y \in [0, 1]$. Die Matrizen V und W sind für diesen Fall identisch zu dem Fall $x_0 > y_0$. Die Matrix S ist durch die Einträge

$$S_{i,j} := (-1)^\nu f^{(\nu+\mu)}(y_0 - x_0)$$

für alle $i, j \in \{0, \ldots, m-1\}$ mit $\nu + \mu < m$ definiert. Dies folgt durch den Tausch der Rollen von x und x_0 mit y bzw. y_0.
Weil g in den Punkten (x, y) mit $x = y$ nicht differenzierbar ist, können wir nicht erwarten, dass die Approximation der gesamten Matrix G durch $\tilde{G}$ besonders gut ist. Allerdings werden wir zeigen, dass Teilblöcke sich gut approximieren lassen. Um zu beurteilen, welche Blöcke der Matrix G für eine Approximation geeignet sind, schätzen wir als Nächstes den Fehler der Taylor-Entwicklung ab.

Fehlerabschätzung

Durch das Ersetzen von g durch $g_{(x_0,y_0),m}$ entsteht ein Approximationsfehler, der den Fehler der Matrixapproximation bestimmt. Mit Hilfe der Fehlerdarstellung von Cauchy können wir den Fehler für die Taylor-Entwicklung von f abschätzen.

Lemma 2.5.1
Es seien $a, b \in \mathbb{R}$ mit $0 < a < b$. Wir setzen $z_0 := (a + b)/2$ und $\xi := 2a/(b - a)$. Dann gilt für alle $z \in [a, b]$ und alle $m \in \mathbb{N}_{>0}$

$$|f(z) - f_{z_0,m}(z)| \leq \log\left(\frac{1}{\xi} + 1\right)\left(\frac{1}{\xi + 1}\right)^{m-1}.$$

Beweis: [siehe [12] Lemma 2.2] ∎

Für ein abgeschlossenes Intervall $[a, b]$ können wir also den Fehler der Taylor-Entwicklung von f uniform in Abhängigkeit von ξ abschätzen. Dieses Ergebnis überträgt sich auf die Approximation der Kernfunktion g.

Lemma 2.5.2
Es seien $\eta \in \mathbb{R}_{>0}$ und $\Omega_t = [a_t, b_t], \Omega_s = [a_s, b_s] \subset \mathbb{R}$ nicht-triviale Intervalle ($a_t < b_t$ und $a_s < b_s$) mit

$$\operatorname{diam}(\Omega_t) + \operatorname{diam}(\Omega_s) \;\leq\; 2\eta \operatorname{dist}(\Omega_t, \Omega_s).$$

Weiter seien $x_0 := (a_t + b_t)/2$ und $y_0 := (a_s + b_s)/2$ die Mittelpunkte von Ω_t bzw. Ω_s. Dann gilt für $m \in \mathbb{N}$, alle $x \in \Omega_t$ und alle $y \in \Omega_s$

$$|g(x,y) - g_{(x_0,y_0),m}(x,y)| \;\leq\; \log(\eta+1)\left(\frac{\eta}{\eta+1}\right)^{m-1}.$$

Beweis: [siehe [12] Korollar 2.3] ∎

Wir erhalten also eine Schranke für den Approximationsfehler innerhalb des Rechtecks $\Omega_t \times \Omega_s$. Nach Voraussetzung des Lemmas 2.5.2 müssen die Intervalle Ω_t und Ω_s disjunkt sein. Dies bedeutet insbesondere, dass sich nicht die gesamte Matrix G auf einmal approximieren lässt.
Deshalb führen wir in Abschnitt 3.3 Strukturen zur Konstruktion von Blockpartitionen ein, bei denen ein großer Teil der Blöcke durch niedrigen Rang approximiert werden kann. Zuvor definieren wir noch für gegebene Blöcke die approximierenden Matrizen und beweisen eine Fehlerschranke.

Blockweise Approximation

Zuerst definieren wir für Blöcke $\mathbf{t} \times \mathbf{s} \subseteq \mathcal{I} \times \mathcal{I}$ Matrizen $G_{t,s,m} = V_{t,m} S_{t,s,m} W_{s,m}^T$ in faktorisierter Form, die die Matrix G im Block $\mathbf{t} \times \mathbf{s}$ per Taylor-Entwicklung der Ordnung m approximieren.

Definition 2.5.3
Es seien $\mathbf{t}, \mathbf{s} \subseteq \mathcal{I}$ nicht leer und $\Omega_t = [a_t, b_t], \Omega_s = [a_s, b_s] \subseteq \mathbb{R}$ disjunkte Intervalle mit

$$\left[\frac{i-1}{n}, \frac{i}{n}\right] \subseteq \Omega_t \quad \text{und} \quad \left[\frac{j-1}{n}, \frac{j}{n}\right] \subseteq \Omega_s \quad \text{für alle } i \in \mathbf{t},\, j \in \mathbf{s}.$$

Weiter seien $x_t := (a_t + b_t)/2$ und $y_s := (a_s + b_s)/2$ die Mittelpunke von Ω_t bzw. Ω_s. Dann definieren wir für $m \in \mathbb{N}$ und $\mathcal{K} := \{0, \ldots, m-1\}$ die Matrizen $V_{t,m} \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}_{\mathbf{t}\times\mathcal{K}}$, $S_{t,s,m} \in \mathbb{R}^{\mathcal{K}\times\mathcal{K}}$ und $W_{s,m} \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}_{\mathbf{s}\times\mathcal{K}}$ durch

$$(V_{t,m})_{i,\nu} := \begin{cases} 0 & \text{, falls } i \notin \mathbf{t} \\ \int_{\frac{i-1}{n}}^{\frac{i}{n}} \frac{(x-x_t)^\nu}{\nu!} dx & \text{, falls } i \in \mathbf{t} \end{cases},$$

$$(W_{s,m})_{j,\mu} := \begin{cases} 0 & \text{, falls } j \notin \mathbf{s} \\ \int_{\frac{j-1}{n}}^{\frac{j}{n}} \frac{(y-y_t)^\mu}{\mu!} dy & \text{, falls } j \in \mathbf{s} \end{cases}$$

und

$$(S_{t,s,m})_{\nu,\mu} := \begin{cases} 0 & \text{, falls } \nu+\mu \geq m \\ (-1)^{\mu} f^{(\nu+\mu)}(x_t - y_s) & \text{, falls } x_t > y_s,\ \nu+\mu < m \\ (-1)^{\nu} f^{(\nu+\mu)}(y_s - x_t) & \text{, falls } x_t < y_s,\ \nu+\mu < m \end{cases}$$

für alle $i, j \in \mathcal{I}$ und $\nu, \mu \in \mathcal{K}$. Des Weiteren definieren wir $G_{t,s,m} = V_{t,m} S_{t,s,m} W_{s,m}^T \in \mathbb{R}_{\mathbf{t}\times\mathbf{s}}^{\mathcal{I}\times\mathcal{I}}$.

Wegen $\Omega_t \cap \Omega_s = \emptyset$ folgt $x_t \neq y_s$ und somit die Wohldefiniertheit von S. Da wir Lemma 2.5.2 zur Fehlerabschätzung nutzen, stellt die Annahme, das $\Omega_t \cap \Omega_s = \emptyset$ disjunkt seien, keine weitere Einschränkung dar. Die Erweiterung der Matrizen um Nullen ist unserer Definition der Einschränkung geschuldet. Diese Nulleinträge werden in der Praxis natürlich nicht gespeichert.
Im nächsten Lemma schätzen wir den Fehler zwischen den Einträgen von $G_{|\mathbf{t}\times\mathbf{s}}$ und $G_{t,s,m}$ ab.

Lemma 2.5.4
Es seien $\mathbf{t}, \mathbf{s} \subseteq \mathcal{I}$ und $\Omega_t = [a_t, b_t], \Omega_s = [a_s, b_s] \subseteq \mathbb{R}$ disjunkte Intervalle mit

$$\left[\frac{i-1}{n}, \frac{i}{n}\right] \subseteq \Omega_t \quad \text{und} \quad \left[\frac{j-1}{n}, \frac{j}{n}\right] \subseteq \Omega_s \quad \text{für alle } i \in \mathbf{t},\ j \in \mathbf{s}.$$

und

$$\operatorname{diam}(\Omega_t) + \operatorname{diam}(\Omega_s) \ \leq \ 2\eta \operatorname{dist}(\Omega_t, \Omega_s).$$

Weiter seien $x_t := (a_t + b_t)/2$ und $y_s := (a_s + b_s)/2$ die Mittelpunke von Ω_t bzw. Ω_s. Dann gilt für alle $m \in \mathbb{N}$ und $(i,j) \in \mathbf{t} \times \mathbf{s}$

$$|G_{i,j} - (G_{t,s,m})_{i,j}| \ \leq \ \frac{1}{n^2} \log(1+\eta) \left(\frac{\eta}{\eta+1}\right)^{m-1}.$$

Beweis: [siehe [12, Abschnitt 2.4]] ∎

Wir können also die einzelnen Einträge der Differenz der Matrizen abschätzen. Mit Lemma 2.3.10 folgen entsprechende Abschätzungen für die Matrixnormen einzelner Matrixblöcke.

3 Hierarchische Strukturen

Die hierarchischen Matrizen sind durch Bäume strukturiert. Deshalb fangen wir damit an, in Abschnitt 3.1 Graphen und Bäume einzuführen. Danach betrachten wir in Abschnitt 3.2 die sogenannten Clusterbäume, deren Blätter eine Partition der zugrunde liegenden Indexmenge liefern. Basierend auf den Clusterbäumen führen wir in Abschnitt 3.3 Blockbäume ein, welche Clusterbäume für das kartesische Produkt zweier Indexmengen bilden. Diese liefern eine Blockpartition der Matrix, bei der mit Hilfe einer Zulässigkeitsbedingung entschieden wird, ob ein Blatt-Block in faktorisiertem oder in vollbesetztem Format gespeichert wird. Mit dieser hierarchisch strukturierten Blockpartition definieren wir in Abschnitt 3.4 die verschiedenen hierarchischen Matrizen.

3.1 Graphen und Bäume

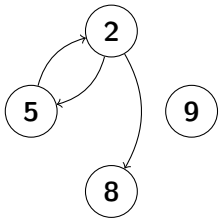
In diesem Abschnitt führen wir Graphen und Bäume kurz ein. Dabei konzentrieren wir uns auf die für diese Arbeit relevanten Begriffe und Aussagen. Zuerst führen wir gerichtete Graphen ein. Darauf aufbauend definieren wir dann Bäume.

Definition 3.1.1 (Graph)
Es sei V eine endliche Menge und eine Abbildung $\mathrm{chil} : V \to \mathcal{P}(V)$ in die Potenzmenge gegeben. Dann ist das Tupel (V, chil) ein *(gerichteter) Graph*. Dabei heißen die Elemente von V *Knoten* (engl.: vertices) und die Elemente von $\mathrm{chil}(v)$ *Kinder* (engl.: children).

Bemerkung 3.1.2
Häufig wird bei der Definition der gerichteten Graphen statt der Abbildung chil die Menge der Kanten (engl.: edges) $E \subseteq V \times V = \{(x, y) \,|\, x, y \in V\}$ verwendet. Über die Identifikation $\mathrm{chil}(v) := \{w \in V \,|\, (v, w) \in E\}$ bzw. $E := \{(v, w) \,|\, v \in V, w \in \mathrm{chil}(v)\}$ kann zwischen den Darstellungen gewechselt werden.

Wir nutzen die Definition über die Abbildung chil, da diese der Implementierung näher und für die Darstellung der rekursiven Algorithmen besser geeignet ist. Da wir ausschließlich gerichtete Graphen betrachten, ist mit Graph stets ein gerichteter Graph gemeint.

Beispiel 3.1.3
Es sei $V = \{2, 5, 8, 9\}$, $\mathrm{chil}(2) = \{5, 8\}$, $\mathrm{chil}(5) = \{2\}$ und $\mathrm{chil}(8) = \emptyset = \mathrm{chil}(9)$. Dann ist (V, chil) folgender Graph:



Die sogenannten Pfade beschreiben Wege über die Kanten durch den Graphen.

Definition 3.1.4 (Pfad)
Es sei (V, chil) ein Graph. Eine Folge von Knoten $(v_i)_{i=0}^m \in V^{m+1}$ heißt *Pfad* der *Länge* $m \in \mathbb{N}$ von v_0 nach v_m, falls der jeweils nächste Knoten ein Kind des aktuellen Knoten ist, d. h. $v_{i+1} \in \text{chil}(v_i)$ für alle $i \in \{0, \ldots, m-1\}$ gilt.

Definition 3.1.5 (Verkettung von Pfaden)
Es seien (V, chil) ein Graph und $p_1 := (v_i)_{i=0}^m$ und $p_2 := (w_i)_{i=0}^n$ zwei Pfade mit $v_m = w_0$. Dann ist die *Verkettung der Pfade* durch $(p_1, p_2) := (v_i)_{i=0}^{m+n}$ definiert, wobei $v_i := w_{i-m}$ für alle $i \in \{m+1, ..., m+n\}$ ist. Die so definierte Verkettung ist ein Pfad.

Mit Hilfe der Pfade können wir Bäume definieren.

Definition 3.1.6 (Baum)
Es sei (V, chil) ein Graph. Falls ein $r \in V$ derart existiert, dass für alle $v \in V$ genau ein Pfad von r nach v existiert, ist (V, chil) ein *Baum*. In diesem Fall ist $r =: \text{root}(V, \text{chil})$ die Wurzel (engl.: root) des Baums.

In Lemma 3.1.9 zeigen wir, dass die Wurzel und somit die Bezeichnung $\text{root}(V, \text{chil})$ eindeutig ist. Zuvor führen wir noch den Begriff des Zyklus ein.

Definition 3.1.7 (Zyklus)
Ein Pfad $(v_i)_{i=0}^m$ heißt *Zyklus*, falls er nicht trivial ist, d. h. $m \geq 1$, und $v_0 = v_m$ gilt.

Beispiel 3.1.8
In Beispiel 3.1.3 ist $(2, 5, 2)$ ein Zyklus.
Der kürzeste Zyklus ist (v, v) für einen Knoten v mit $v \in \text{chil}(v)$.

Lemma 3.1.9
Es sei (V, chil) ein Baum mit Wurzel r.

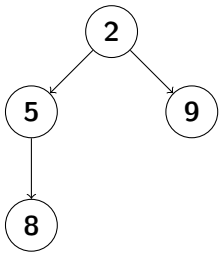
(i) Alle Pfade in (V, chil) sind eindeutig, d. h. , dass $p = q$ für alle Pfade $p = (v_i)_{i=0}^m$ und $q = (w_j)_{j=0}^n$ in (V, chil) mit $v_0 = w_0$ und $v_m = w_n$ folgt.

(ii) Es existieren keine Zyklen in (V, chil).

(iii) Die Wurzel r ist eindeutig.

Beweis: (i): Es seien $p = (v_i)_{i=0}^m$ und $q = (w_j)_{j=0}^n$ Pfade in (V, chil) mit $v_0 = w_0$ und $v_m = w_n$. Dann existiert ein eindeutiger Pfad p_0 von der Wurzel r zu v_0. Die Verkettungen (p_0, p) und (p_0, q) sind Pfade von der Wurzel r zu $v_m = w_n$. Wegen der Eindeutigkeit dieses Pfades folgt $(p_0, p) = (p_0, q)$ und somit $p = q$.
(ii): Es sei p ein Pfad von $v \in V$ zu v. Dann ist die Verkettung (p, p) wieder ein Pfad von v zu v. Wegen der Eindeutigkeit folgt $(p, p) = p$. Daraus folgt, dass p der triviale Pfad in v sein muss. Also existiert kein Zyklus.
(iii): Es sei $r' \in V$ so, dass zu jedem Knoten $v \in V$ ein Pfad von r' nach v existiert. Dann existieren Pfade $p_{r,r'}$ von r nach r' und $p_{r',r}$ von r' nach r. Die Verkettung $(p_{r,r'}, p_{r',r})$ ist ein Pfad von r nach r. Da es keinen Zyklus im Baum gibt, muss $(p_{r,r'}, p_{r',r})$ die Länge 0 haben und $r = r'$ sein. Also ist r eindeutig. ■

Beispiel 3.1.10
Der Graph aus Beispiel 3.1.3 ist kein Baum, da er einen Zyklus beinhaltet und darüber hinaus keine Pfade zu 9 führen.
Es sei $V = \{2, 5, 8, 9\}$, $\text{chil}(2) = \{5, 9\}$, $\text{chil}(5) = \{8\}$ und $\text{chil}(8) = \emptyset = \text{chil}(9)$. Dann ist (V, chil) folgender Baum:



Definition 3.1.11 (Level und Tiefe)
Es sei (V, chil) ein Baum mit Wurzel $r \in V$. Dann ist die *Level-Funktion* wie folgt definiert

$$\text{level} := \text{level}_{(V,\text{chil})} : V \to \mathbb{N},\ v \mapsto \text{Länge des Pfads von } r \text{ nach } v. \tag{3.1}$$

Die *Tiefe eines Baums* (engl.: depth) ist

$$\text{depth}(V, \text{chil}) := \max\{\text{level}(v) \mid v \in V\}. \tag{3.2}$$

Wegen der Eindeutigkeit der Wurzel und der Pfade zu jedem Knoten im Baum ist die Level-Funktion wohldefiniert.

Beispiel 3.1.12
Im Beispiel 3.1.10 gilt

$$\text{level}(2) = 0, \quad \text{level}(5) = 1 = \text{level}(9), \quad \text{level}(8) = 2 \quad \text{und} \quad \text{depth}(V, \text{chil}) = 2.$$

Lemma 3.1.13 (Elter)
Es sei (V, chil) ein Baum mit Wurzel $r \in V$. Dann existiert für alle $v \in V \backslash \{r\}$ genau ein $\acute{v}$ mit $v \in \text{chil}(\acute{v})$. Diesen Knoten $\acute{v}$ bezeichnen wir als *Elter* von v (engl.: parent). Wir verwenden die Notation $\text{par}(v) := \acute{v}$.

Beweis: Es sei $v \in V \backslash \{r\}$. Dann existiert ein Pfad $(v_i)_{i=0}^m$ von r nach v mit $m > 0$. Nach Definition der Pfade gilt dann $v \in \text{chil}(\acute{v})$ für $\acute{v} := v_{m-1}$.
Es sei $w \in V$ mit $v \in \text{chil}(w)$. Dann existiert ein Pfad $(w_i)_{i=0}^n$ von r nach w. Mit $w_{n+1} := v$ ist dann $(w_i)_{i=0}^{n+1}$ ein Pfad von r nach v. Wegen der Eindeutigkeit des Pfads von r nach v folgt $n = m - 1$ und $w = w_n = v_n = v_{m-1} = \acute{v}$. Also ist $\acute{v}$ eindeutig. ■

Definition 3.1.14 (Vorfahren und Nachfahren)
Es sei (V, chil) ein Baum mit Wurzel r. Für jeden Knoten $v \in V$ sind die Nachfahren (engl.: descendants) durch

$$\text{desc}(v) \;:=\; \{w \in V \mid \text{es existiert ein Pfad von } v \text{ nach } w\} \tag{3.3}$$

und die Vorfahren (Vorgänger, engl.: predecessor) durch

$$\text{pred}(v) \;:=\; \{w \in V \mid \text{es existiert ein Pfad von } w \text{ nach } v\} \tag{3.4}$$

definiert.

Da viele unserer Algorithmen rekursiv arbeiten, wollen wir an dieser Stelle eine rekursive Darstellung der Vorfahren und Nachfahren angeben.

Lemma 3.1.15 (Vorfahren und Nachfahren)
Es sei (V, chil) ein Baum mit Wurzel r. Dann gilt für alle $v \in V$

$$\text{desc}(v) \;=\; \{v\} \dot\cup \dot{\bigcup_{\breve{v} \in \text{chil}(v)}} \text{desc}(\breve{v})$$

und

$$\text{pred}(v) \;=\; \begin{cases} \{v\} & \text{, falls } v = r \\ \{v\} \dot\cup \text{pred}(\acute{v}) & \text{, falls } v \neq r,\ \acute{v} := \text{par}(v) \end{cases}.$$

Beweis: Wir beweisen die erste Aussage per Induktion über $\#\,\text{desc}(v)$. Es sei $v \in V$ mit $\#\,\text{desc}(v) = 1$. Dann ist $\text{chil}(v) = \emptyset$ und es gibt nur den trivialen Pfad von v aus. Somit gilt $\text{desc}(v) = \{v\}$.
Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $v \in V$ mit $\#\,\text{desc}(v) \leq n$ gilt. Es sei $v \in V$ mit $\#\,\text{desc}(v) = n + 1$.
($\subseteq$): Es sei $w \in \text{desc}(v)$. Falls $v = w$ gilt, ist w in der rechten Seite enthalten. Sei also $v \neq w$ und $(v_i)_{i=0}^m$ ein Pfad von v nach w. Wegen $v \neq w$ ist $m > 0$ und somit $v_1 \in \text{chil}(v)$. Dann ist $(v_i)_{i=1}^m$ ein Pfad von v_1 nach w, also $w \in \text{desc}(v_1)$.
($\supseteq$): Es sei w aus der rechten Seite. Falls $v = w$ ist, folgt $w \in \text{desc}(v)$. Es sei also $v \neq w$. Dann existiert $v_1 \in \text{chil}(v)$ mit $w \in \text{desc}(v_1)$. Es sei $(v_i)_{i=1}^m$ ein Pfad von v_1 nach w. Dann ist $(v_i)_{i=0}^m$ mit $v_0 := v$ ein Pfad von v nach w. Somit ist $w \in \text{desc}(v)$.
Die zweite Aussage beweisen wir per Induktion über $\#\,\text{pred}(v)$. Es sei $v \in V$ mit $\text{pred}(v) = 1$. Dann gilt $v = r$ (es gilt $\{r, v\} \subseteq \text{pred}(v)$). Somit gilt $\text{pred}(v) = \{v\}$.

Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $v \in V$ mit $\#\operatorname{desc}(v) \leq n$ gelte. Es sei $v \in V$ mit $\#\operatorname{desc}(v) = n+1$.
($\subseteq$): Es sei $w \in \operatorname{pred}(v)$. Falls $v = w$ gilt, ist w in der rechten Seite enthalten. Es sei also $v \neq w$. Es sei $(w_i)_{i=0}^m$ ein Pfad von w nach v. Wegen $v \neq w$ ist $m > 0$ und es gilt $v \in \operatorname{chil}(w_{m-1})$. Damit ist $(w_i)_{i=0}^{m-1}$ ein Pfad von w nach $w_{m-1} = \acute{v} := \operatorname{par}(v)$. Insbesondere ist $w \in \operatorname{pred}(\acute{v})$.
($\supseteq$): Es sei w aus der rechten Seite. Falls $v = w$ gilt, ist $w \in \operatorname{pred}(v)$. Es sei also $v \neq w$. Dann ist $w \in \operatorname{pred}(\acute{v})$. Es sei $(w_i)_{i=0}^m$ ein Pfad von w nach $\acute{v}$. Dann ist $(w_i)_{i=0}^{m+1}$ mit $w_{m+1} := v$ ein Pfad von w nach v. Somit folgt $w \in \operatorname{pred}(v)$. ∎

Später wollen wir Algorithmen, die auf Bäumen definiert sind, auch auf Teilen eines gegebenen Baums anwenden. Deshalb zeigen wir, dass die Nachfahren eines Knoten wieder einen Baum bilden.

Lemma 3.1.16 (Teilbaum)
Es sei (V, chil) ein Baum mit Wurzel r. Weiter sei $v \in V$. Dann definiert $(V, \operatorname{chil})_{|v} := (\operatorname{desc}(v), \operatorname{chil}_{|\operatorname{desc}(v)})$ einen Baum mit Wurzel v. Wir bezeichnen $(V, \operatorname{chil})_{|v}$ als den *Teilbaum* von (V, chil) zum Knoten v.

Beweis: Nach Definition der Nachfahren existiert für alle $w \in \operatorname{desc}(v)$ ein Pfad $(v_i)_{i=0}^m$ in (V, chil) von v nach w. Da für alle $j \in \{0, \ldots, m\}$ der Pfad $(v_i)_{i=0}^j$ von v nach v_j führt, ist $\{v_i \mid i \in \{0, \ldots, m\}\} \subseteq \operatorname{desc}(v)$. Also ist $(v_i)_{i=0}^m$ ein Pfad in $(V, \operatorname{chil})_{|v}$. Die Eindeutigkeit der Pfade folgt aus Lemma 3.1.9 (i). ∎

Zum Abschluss dieses Abschnitts zeigen wir eine kleine technische Aussage.

Lemma 3.1.17
Es sei (V, chil) ein Baum mit Wurzel r. Weiter sei $v \in V$ und $\check{v} \in \operatorname{desc}(v)$. Dann gilt für den Pfad $p = (v_i)_{i=0}^{\operatorname{level}(\check{v})}$ von r nach $\check{v}$, dass $v = v_{\operatorname{level}(v)}$ ist.

Beweis: Es sei $p = (v_i)_{i=0}^{\operatorname{level}(\check{v})}$ der Pfad von r nach $\check{v}$, $p' = (v'_i)_{i=0}^{\operatorname{level}(v)}$ der Pfad von r nach v und p'' der Pfad von v nach $\check{v}$. Dann ist die Verkettung (p', p'') ein Pfad von r nach $\check{v}$ und somit $p = (p', p'')$. Insbesondere gilt $v_{\operatorname{level}(v)} = v'_{\operatorname{level}(v)} = v$. ∎

3.2 Clusterbäume

Es seien endliche Indexmengen $\mathcal{I}, \mathcal{J}$ gegeben. Das Ziel ist, eine Blockpartition der Menge $\mathcal{I} \times \mathcal{J}$ zu konstruieren, die durch eine Baumstruktur organisiert ist. Diese liefert eine hierarchisch organisierte Partition der Matrix. Die hierarchische Struktur ermöglicht die Nutzung von rekursiven Algorithmen zum Durchlaufen der Matrixteile.

Für diese Zwecke muss jeder Knoten mit einer endlichen Indexmenge identifiziert werden. Dies könnte erreicht werden, indem jeder Knoten gleich der entsprechenden Menge gewählt würde (siehe [26]). Ein solches Vorgehen führt allerdings zu der Einschränkung, dass eine Menge im Baum nur einmal vorkommen kann. Dieser Effekt soll vermieden werden, weil damit bei den später beschriebenen Clusterbäumen ausgeschlossen wird, dass es nur ein Kind gibt. Deshalb wird hier das Konzept der beschrifteten Bäume genutzt (siehe [12]).

Definition 3.2.1 (Beschrifteter Baum)
Es sei (V, chil) ein Baum. Eine Abbildung b auf V heißt *Beschriftung* und (V, chil) zusammen mit b beschrifteter Baum.

Bemerkung 3.2.2
Da durch die Beschriftung Indexmengen beschrieben werden sollen, gilt in dieser Arbeit grundsätzlich $b : V \to \mathcal{P}(\mathcal{I})$ für eine geeignete Indexmenge $\mathcal{I}$.

Da eine Partition konstruiert werden soll, reicht es nicht aus, dass die Knoten mit Indexmengen beschriftet sind. Die Blätter des Clusterbaums sollen eine Partition der Indexmenge liefern. Daraus ergibt sich die Forderung, dass die Vereinigung der Beschriftungen der Kinder gleich der Beschriftung des Elter sind. Dies ist der Ansatz für die nächste Definition.

Definition 3.2.3 (Clusterbaum)
Es sei (V, chil) ein beschrifteter Baum, wobei für $v \in V$ die Beschriftung mit $\mathbf{v}$ bezeichnet werde. Falls

$$\mathbf{r}_{\mathcal{I}} \;=\; \mathcal{I} \quad \text{für die Wurzel } r_{\mathcal{I}} := \mathrm{root}(V, \mathrm{chil}) \tag{3.5}$$

und

$$\mathbf{t} \;=\; \dot{\bigcup_{\check{t} \in \mathrm{chil}(t)}} \check{\mathbf{t}} \quad \text{für alle } t \in V \text{ mit } \mathrm{chil}(t) \neq \emptyset \tag{3.6}$$

gilt, heißt (V, chil) *Clusterbaum* zur Indexmenge $\mathcal{I}$. Einen Clusterbaum zur Indexmenge bezeichnen wir typischerweise mit $\mathcal{T}_{\mathcal{I}}$. Die Elemente aus V heißen *Cluster* und $t \in \mathcal{T}_{\mathcal{I}}$ meint $t \in V$. Weiter bezeichnet

$$\mathcal{L}_{\mathcal{I}} \;:=\; \{t \in \mathcal{T}_{\mathcal{I}} \mid \mathrm{chil}(t) = \emptyset\}$$

die Menge der *Blätter* von $\mathcal{T}_{\mathcal{I}}$. Die Anzahl der Level, auf denen Cluster des Clusterbaums existieren, bezeichnen wir mit $p_{\mathcal{I}} := \mathrm{depth}(\mathcal{T}_{\mathcal{I}}) + 1$.

Die Bezeichnungen wie $r_{\mathcal{I}}$, $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{L}_{\mathcal{I}}$ mit der Indexmenge $\mathcal{I}$ ist dadurch motiviert, dass wir für eine Indexmenge $\mathcal{I}$ typischerweise nur einen Clusterbaum verwenden. Wir bezeichnen mit $p_{\mathcal{I}}$ nicht die Tiefe des Baums, weil bei späteren Aufwandsabschätzungen häufig die Summe über alle Level auftaucht. Diese können wir dann mit $\mathrm{depth}(\mathcal{T}_{\mathcal{I}})+1 = p_{\mathcal{I}}$ abschätzen und müssen somit keinen zusätzlichen Term $+1$ hinzufügen. Als Nächstes zeigen wir, wie ein Clusterbaum für das Modellproblem konstruiert werden kann.

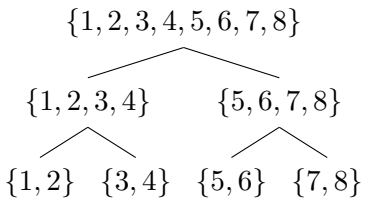


Abbildung 3.1: Clusterbaum für das Modellproblem für $n = 8$ und $n_{\max} = 2$

Beispiel 3.2.4
Wir nehmen an, dass die Anzahl der Basisfunktionen $n \in \mathbb{N}_{>0}$ im Modellproblem aus Abschnitt 2.5 eine Zweierpotenz ist, also $n = 2^\ell$ für ein $\ell \in \mathbb{N}$. Die Wurzel des Clusterbaums definieren wir als die Menge $\mathcal{I} := \{1, \ldots, n\}$ und die Kinder durch

$$\operatorname{chil}(\{i, \ldots, j\}) \;=\; \begin{cases} \emptyset & , \text{falls } j - i + 1 \leq n_{\max} \\ \{\{i, \ldots, \frac{i+j-1}{2}\}, \{\frac{i+j+1}{2}, \ldots, j\} & , \text{falls } j - i + 1 > n_{\max} \end{cases},$$

wobei $n_{\max} \in \mathbb{N}_{>0}$ die maximale Größe eines Blatt-Clusters angibt. Ein Beispiel für den entstehenden Baum ist in Abbildung 3.1 gezeigt. Mit der Beschriftung

$$b(\{i, \ldots, j\}) \;=\; \{i, \ldots, j\}$$

ergibt dies einen Clusterbaum zur Indexmenge $\mathcal{I}$.

Häufig wird der Clusterbaum geometrisch konstruiert (siehe [26, Konstruktion 4.2]). Dabei wird jeder Basisfunktion ein geometrischer Punkt zugewiesen und anschließend eine Box um alle Punkte gelegt. Falls in einer Box noch mehr als $n_{\max}$ Punkte liegen, wird die Box in weitere Boxen unterteilt und die Punkte werden den jeweiligen Boxen zugeordnet. Die neuen Boxen bilden die Kinder und über die geometrischen Punkte können die Basisfunktionen zugewiesen werden.
Eine weitere Möglichkeit ist das kardinalitätsbalancierte Clustern (siehe [26, Konstruktion 4.1]). Bei diesem wird darauf geachtet, das alle Kinder (beinahe) gleich viele Elemente enthalten. Da die Bedingung für die Approximationsgüte meistens von geometrischen Informationen abhängt (siehe 2.5.2), ist es schwer, Aussagen für diese Clusterstrategie zu beweisen.
Für die LR-Zerlegung von Finite Elemente Steifigkeitsmatrizen wird häufig eine Methode verwendet, die auf den Idee der Gebietszerlegung nutzt (siehe [28]). Dieser Ansatz hat den Vorteil, dass große Nullblöcke der Steifigkeitsmatrix bei der LR-Zerlegung erhalten bleiben.
Ein weiteren Ansatz für die LR-Zerlegung bei Finite Elementen, der versucht, große Nullblöcke zu erhalten, basiert auf dem Matrixgraphen der Steifigkeitsmatrix (siehe [27]). Dabei wird an Stelle des Gebiets versucht, den Matrixgraphen in möglichst unabhängige Teile zu zerlegen.
In Lemma 3.2.5 beschreiben wir einige Zusammenhänge zwischen Clustern und ihren Beschriftungen, die später benötigt werden.

Lemma 3.2.5
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum.

(i) Es seien $s, t \in \mathcal{T}_\mathcal{I}$ mit $\text{level}(s) = \text{level}(t)$. Dann gilt $s = t$ oder $\mathbf{s} \cap \mathbf{t} = \emptyset$.

(ii) Für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \text{desc}(t)$ gilt $\mathbf{s} \subseteq \mathbf{t}$. Insbesondere gilt $\mathbf{s} \subseteq r_\mathcal{I} = \mathcal{I}$ für alle $s \in \mathcal{T}_\mathcal{I}$.

(iii) Für alle $s, t \in \mathcal{T}_\mathcal{I}$ mit $\mathbf{s} \cap \mathbf{t} \neq \emptyset$ gilt $s \in \text{desc}(t)$ oder $t \in \text{desc}(s)$.

(iv) Für alle $s, t \in \mathcal{T}_\mathcal{I}$ mit $\mathbf{s} \cap \mathbf{t} \neq \emptyset$ gilt

$$\begin{cases} s \in \text{desc}(t) \text{ und } \mathbf{s} \subseteq \mathbf{t} & , \text{ falls } \text{level}(s) \geq \text{level}(t) \\ s = t \text{ und } \mathbf{s} = \mathbf{t} & , \text{ falls } \text{level}(s) = \text{level}(t) \\ t \in \text{desc}(s) \text{ und } \mathbf{s} \supseteq \mathbf{t} & , \text{ falls } \text{level}(s) \leq \text{level}(t) \end{cases}.$$

Beweis:
(i): Wir beweisen die Aussage induktiv über $\ell := \text{level}(s) = \text{level}(t)$. Für $\ell = 0$ gilt $s = r_\mathcal{I} = t$. Es gelte die Aussage für $\ell \in \mathbb{N}$ und es seien $s, t \in \mathcal{T}_\mathcal{I}$ mit $\text{level}(s) = \text{level}(t) = \ell + 1$. Weiter sei $s \neq t$. Falls $\acute{s} := \text{par}(s) = \text{par}(t) =: \acute{t}$ gilt, folgt aus (3.6) $\mathbf{s} \cap \mathbf{t} = \emptyset$. Falls $\acute{s} \neq \acute{t}$ ist, gilt $\text{level}(\acute{s}) = \text{level}(\acute{t}) = \ell$. Mit der Induktionsannahme und (3.6) folgt $\mathbf{s} \cap \mathbf{t} \subseteq \acute{\mathbf{s}} \cap \acute{\mathbf{t}} = \emptyset$.
(ii): Es sei $s \in \mathcal{T}_\mathcal{I}$ und $t \in \text{desc}(s)$. Dann existiert ein Pfad $(t_i)_{i=0}^m$ von s nach t. Für alle $i \in \{1, \ldots, m\}$ gilt $t_i \in \text{chil}(t_{i-1})$ und mit (3.6) folgt $\mathbf{t}_i \subseteq \mathbf{t}_{i-1}$. Daraus folgt $\mathbf{t} \subseteq \mathbf{s}$.
(iii): Es seien $s, t \in \mathcal{T}_\mathcal{I}$ mit $\mathbf{s} \cap \mathbf{t} \neq \emptyset$. Weiter seien $(s_i)_{i=0}^m$ und $(t_i)_{i=0}^n$ Pfade von $r_\mathcal{I}$ zu s bzw. t. O. E. d. A. sei $0 \leq m = \text{level}(s) \leq \text{level}(t) = n$. Mit (ii) folgt $\mathbf{s}_i \cap \mathbf{t}_i \supseteq \mathbf{s} \cap \mathbf{t} \neq \emptyset$ für alle $i \in \{0, ..., m\}$. Daraus folgt mit $s_1, t_1 \in \text{chil}(r_\mathcal{I})$ und (3.6) $s_1 = t_1$.
Induktiv folgt $s = s_m = t_m$. Somit ist $(t_i)_{i=m}^n$ ein Pfad von $s = t_m$ nach $t = t_n$ und $t \in \text{desc}(s)$.
(iv): Folgt aus (iii) und (ii). ∎

Korollar 3.2.6
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Dann gilt für alle $\ell \in \mathbb{N}$

$$\dot{\bigcup_{\substack{t \in \mathcal{T}_\mathcal{I} \\ \text{level}(t) = \ell}}} \mathbf{t} \subseteq \mathcal{I}.$$

Beweis: Die Inklusion gilt, da nach Lemma 3.2.5 (ii) $\mathbf{t} \subseteq \mathcal{I}$ ist. Dass die Mengen paarweise disjunkt sind, folgt aus Lemma 3.2.5 (i). ∎

Wir definieren den ab einem Level $\ell \in \mathbb{N}$ abgeschnittenen Clusterbaum, um zu zeigen, dass die Beschriftungen der Blätter eines Clusterbaums $\mathcal{T}_\mathcal{I}$ eine Partition der zugehörigen Indexmenge $\mathcal{I}$ liefern.

Definition 3.2.7
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Für $\ell \in \mathbb{N}$ sei

$$\mathcal{T}_\mathcal{I}^{(\ell)} := \{t \in \mathcal{T}_\mathcal{I} \mid \operatorname{level}(t) = \ell\} \dot\cup \{t \in \mathcal{L}_\mathcal{I} \mid \operatorname{level}(t) < \ell\}.$$

Das nächste Lemma zeigt, dass die Blätter eines Clusterbaums bzw. die zugehörigen Beschriftungen eine Partition der Indexmenge bilden. Dies ist die wesentliche Eigenschaft eines Clusterbaums.

Lemma 3.2.8 (Partition der Indexmenge)
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Dann gilt für alle $\ell \in \mathbb{N}$

$$\mathcal{I} = \dot\bigcup_{t \in \mathcal{T}_\mathcal{I}^{(\ell)}} \mathbf{t}. \tag{3.7}$$

Insbesondere gilt

$$\mathcal{I} = \dot\bigcup_{t \in \mathcal{L}_\mathcal{I}} \mathbf{t}. \tag{3.8}$$

Beweis: Wir zeigen die Aussage per Induktion. Es gilt $\mathcal{T}_\mathcal{I}^{(0)} = \{r_\mathcal{I}\}$ und somit (3.7) für $\ell = 0$.
Es gelte (3.7) für $\ell \in \mathbb{N}$. Dann gilt mit (3.6)

$$\begin{aligned}\mathcal{I} = \dot\bigcup_{t \in \mathcal{T}_\mathcal{I}^{(\ell)}} \mathbf{t} &= \dot\bigcup_{\substack{t \in \mathcal{L}_\mathcal{I} \\ \operatorname{level}(t) \le \ell}} \mathbf{t} \;\dot\cup \dot\bigcup_{\substack{s \in \mathcal{T}_\mathcal{I} \setminus \mathcal{L}_\mathcal{I} \\ \operatorname{level}(s) = \ell}} \mathbf{s} = \dot\bigcup_{\substack{t \in \mathcal{L}_\mathcal{I} \\ \operatorname{level}(t) \le \ell}} \mathbf{t} \;\dot\cup \dot\bigcup_{\substack{s \in \mathcal{T}_\mathcal{I} \setminus \mathcal{L}_\mathcal{I} \\ \operatorname{level}(s) = \ell}} \dot\bigcup_{\check{s} \in \operatorname{chil}(s)} \check{\mathbf{s}} \\ &= \dot\bigcup_{\substack{t \in \mathcal{L}_\mathcal{I} \\ \operatorname{level}(t) < \ell+1}} \mathbf{t} \;\dot\cup \dot\bigcup_{\substack{\check{s} \in \mathcal{T}_\mathcal{I} \\ \operatorname{level}(s) = \ell+1}} \check{\mathbf{s}} = \dot\bigcup_{t \in \mathcal{T}_\mathcal{I}^{(\ell+1)}} \mathbf{t}.\end{aligned}$$

Also gilt (3.7).
Aus (3.7) mit $\ell = p_\mathcal{I}$ folgt (3.8). ∎

Viele Algorithmen werden mit einem Cluster aufgerufen, bei dem es sich nicht um die Wurzel handelt. Um für diese die gleiche Theorie verwenden zu können, definieren wir Teilbäume.

Definition 3.2.9 (Teilbaum)
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum und $t \in \mathcal{T}_\mathcal{I}$. Dann bezeichne $\mathcal{T}_\mathbf{t}$ den Teilbaum von $\mathcal{T}_\mathcal{I}$ mit der Wurzel t (siehe Lemma 3.1.16), wobei $\mathcal{T}_\mathbf{t}$ die Beschriftung von $\mathcal{T}_\mathcal{I}$ übernimmt.

Korollar 3.2.10
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum und $t \in \mathcal{T}_\mathcal{I}$. Dann ist $\mathcal{T}_\mathbf{t}$ ein Clusterbaum und es gilt

$$\mathbf{t} = \dot\bigcup_{\check{t} \in \mathcal{L}_\mathbf{t}} \check{\mathbf{t}}.$$

Beweis: $\mathcal{T}_{\mathbf{t}}$ ist ein Clusterbaum zur Indexmenge $\mathbf{t}$, da sich alle Eigenschaften des Clusterbaums auf den Teilbaum vererben.
Aus Lemma 3.2.8 folgt der zweite Teil der Aussage. ∎

Lemma 3.2.11 (Aufwandsabschätzungen für Clusterbäume)
Es sei $\mathcal{I}$ eine endliche Indexmenge und $\mathcal{T}_{\mathcal{I}}$ ein zugehöriger Clusterbaum. Dann gilt

$$\sum_{t \in \mathcal{T}_{\mathcal{I}}} \#\mathbf{t} \;\leq\; p_{\mathcal{I}} \#\mathcal{I}. \tag{3.9}$$

Falls zusätzlich jeder Nicht-Blatt-Cluster mindestens $m > 1$ Kinder hat, d. h. $\#\operatorname{chil}(t) \geq m$ für alle $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ gilt, folgt

$$\#\mathcal{T}_{\mathcal{I}} \;\leq\; \frac{m}{m-1} \#\mathcal{L}_{\mathcal{I}}. \tag{3.10}$$

Beweis: Es gilt mit Korollar 3.2.6

$$\sum_{t \in \mathcal{T}_{\mathcal{I}}} \#\mathbf{t} \;=\; \sum_{\ell=0}^{\operatorname{depth}(\mathcal{T}_{\mathcal{I}})} \sum_{\substack{t \in \mathcal{T}_{\mathcal{I}} \\ \operatorname{level}(t)=\ell}} \#\mathbf{t} \;\leq\; \sum_{\ell=0}^{\operatorname{depth}(\mathcal{T}_{\mathcal{I}})} \#\mathcal{I} \;=\; p_{\mathcal{I}} \#\mathcal{I}.$$

Dies zeigt die erste Abschätzung.
Da jeder Cluster außer der Wurzel Kind genau eines Nicht-Blatt-Clusters ist (siehe Lemma 3.1.13), gilt

$$\begin{aligned} \#\mathcal{T}_{\mathcal{I}} - 1 \;&=\; \#\left(\mathcal{T}_{\mathcal{I}} \backslash \{r_{\mathcal{I}}\}\right) \;=\; \#\left(\dot{\bigcup_{t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}}} \operatorname{chil}(t) \right) \;=\; \sum_{t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}} \#\operatorname{chil}(t) \\ &\geq \sum_{t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}} m \;=\; m \#\left(\mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}\right) \;=\; m\#\mathcal{T}_{\mathcal{I}} - m\#\mathcal{L}_{\mathcal{I}}. \end{aligned}$$

Durch Umstellen und Division mit $m - 1$ erhalten wir

$$\frac{m}{m-1} \#\mathcal{L}_{\mathcal{I}} \;\geq\; \#\mathcal{T}_{\mathcal{I}} + \frac{1}{m-1} \;\geq\; \#\mathcal{T}_{\mathcal{I}}.$$

∎

In späteren Abschätzungen für den Aufwand kommen häufig sowohl $\#\mathcal{I}$ als auch $\#\mathcal{T}_{\mathcal{I}}$ vor. Wir zeigen als Nächstes, dass sich $\#\mathcal{T}_{\mathcal{I}}$ in bestimmten Fällen durch $\#\mathcal{I}$ abschätzen lässt.

Korollar 3.2.12
Es sei $\mathcal{T}_{\mathcal{I}}$ ein Clusterbaum zur Indexmenge $\mathcal{I}$ und jeder Nicht-Blatt-Cluster $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ habe mindestens 2 Kinder, d. h. $\operatorname{chil}(t) \geq 2$. Es existiere $k \in \mathbb{N}_{>0}$ mit $\#\mathbf{t} \geq k$ für alle $t \in \mathcal{L}_{\mathcal{I}}$. Dann gilt

$$\#\mathcal{T}_{\mathcal{I}} \;\leq\; 2\frac{\#\mathcal{I}}{k}.$$

Insbesondere gilt $\#\mathcal{T}_{\mathcal{I}} \leq 2\#\mathcal{I}$, falls die Beschriftungen für alle Cluster nicht-leer sind.

Beweis: Weil $\#\mathbf{t} \geq k$ für alle $t \in \mathcal{L}_\mathcal{I}$ gilt, ist

$$\#\mathcal{I} \overset{(3.8)}{=} \sum_{t \in \mathcal{L}_\mathcal{I}} \#t \;\geq\; \sum_{t \in \mathcal{L}_\mathcal{I}} k \;=\; k\#\mathcal{L}_\mathcal{I}$$

und es folgt $\#\mathcal{L}_\mathcal{I} \leq \#\mathcal{I}/k$. Zusammen mit Lemma 3.2.11 für $m = 2$ folgt

$$\#\mathcal{T}_\mathcal{I} \;\leq\; \frac{m}{m-1}\#\mathcal{L}_\mathcal{I} \;=\; 2\#\mathcal{L}_\mathcal{I} \;\leq\; 2\frac{\#\mathcal{I}}{k}.$$

■

Das Korollar 3.2.12 können wir folgendermaßen auf die Clusterstrategie des Modellproblems anwenden.

Beispiel 3.2.13
Es sei $n = 2^\ell$, $\ell \in \mathbb{N}$, und $\mathcal{T}_\mathcal{I}$ der für das Modellproblem nach Beispiel 3.2.4 mit $n_{\max} = 4k$, $k \in \mathbb{N}_{>0}$, konstruierte Clusterbaum. Dann gilt $\#\operatorname{chil}(t) = 2$ für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ und $\#\mathbf{t} \geq 2k$ für alle $t \in \mathcal{L}_\mathcal{I}$. Mit Korollar 3.2.12 folgt $\#\mathcal{T}_\mathcal{I} \leq \#\mathcal{I}/k$.

Nach (3.9) führt linearer Aufwand in jedem Cluster $t \in \mathcal{T}_\mathcal{I}$ (d. h. in $\#\mathbf{t}$) zu einem Gesamtaufwand in $\mathcal{O}(p_\mathcal{I}\#\mathcal{I})$. Da sich die Tiefe des Clusterbaums bei konstanter Blattgröße und konstanter Anzahl der Kinder mindestens logarithmisch zu $\#\mathcal{I}$ verhält, erhalten wir in diesem Fall wenigstens log-linearen Gesamtaufwand.
Andererseits zeigt Korollar 3.2.12, dass für Algorithmen mit konstantem Aufwand in jedem Cluster $t \in \mathcal{T}_\mathcal{I}$ der Gesamtaufwand linear bleibt ($\mathcal{O}(\#\mathcal{I})$). Nach Lemma 3.2.8 darf in jedem Blatt $t \in \mathcal{L}_\mathcal{I}$ zusätzlich ein linearer Aufwand ($\mathcal{O}(\#\mathbf{t})$) betrieben werden, ohne die Asymptotik des Gesamtaufwands zu erhöhen.
Diese beiden Betrachtungen geben uns eine Orientierung, welcher Aufwand in Clusterbäumen erlaubt ist, um einen Gesamtaufwand in $\mathcal{O}(\#\mathcal{I})$ zu erhalten.

3.3 Blockbäume

Für die Partition der Menge $\mathcal{I} \times \mathcal{J}$ wird aus Clusterbäumen $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$ ein neuer Baum konstruiert, der sogenannte Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Da dieser wieder ein Clusterbaum ist, bilden seine Blätter eine Partition von $\mathcal{I} \times \mathcal{J}$ (siehe Lemma 3.2.8).
Bevor wir diese definieren, führen wir noch eine vereinfachende Schreibweise ein.

Definition 3.3.1
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Dann sei für alle $t \in \mathcal{T}_\mathcal{I}$

$$\operatorname{chil}^*(t) \;=\; \begin{cases} \{t\} & ,\; t \in \mathcal{L}_\mathcal{I} \\ \operatorname{chil}(t) & ,\; t \notin \mathcal{L}_\mathcal{I} \end{cases}. \tag{3.11}$$

In Definition 3.3.2 führen wir den Blockbaum ein. Dies ist der zentrale Begriff für die hierarchische Partition von Matrizen.

Definition 3.3.2 (Blockbaum)
Es seien $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$ Clusterbäume der Mengen $\mathcal{I}$ bzw. $\mathcal{J}$. Dann ist ein beschriftete Baum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \subseteq \mathcal{T}_\mathcal{I} \times \mathcal{T}_\mathcal{J}$ ein *Blockbaum* zur Indexmenge $\mathcal{I} \times \mathcal{J}$ mit zugehörigen Clusterbäumen $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$, falls für die Wurzel

$$r_{\mathcal{I}\times\mathcal{J}} = (r_\mathcal{I}, r_\mathcal{J}) \tag{3.12}$$

gilt, für alle $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ die Menge der Kinder durch

$$\operatorname{chil}(b) \;=\; \operatorname{chil}^*(t) \times \operatorname{chil}^*(s) \tag{3.13}$$

gegeben ist und für alle $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ die Beschriftung

$$\mathbf{b} \;=\; \mathbf{t} \times \mathbf{s} \tag{3.14}$$

ist. Die Menge der *Blätter* sei

$$\mathcal{L}_{\mathcal{I}\times\mathcal{J}} \;:=\; \{b \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \mid \operatorname{chil}(b) = \emptyset\}\,.$$

Bemerkung 3.3.3
Es sei darauf hingewiesen, dass in (3.13) der Fall

$$\operatorname{chil}(b) \;=\; \{t\} \times \{s\} \;=\; \{(t, s)\} \;=\; \{b\}$$

ausgeschlossen ist, weil $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Baum ist.

Das nächste Lemma zeigt, dass jeder Blockbaum wieder ein Clusterbaum ist.

Lemma 3.3.4
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum. Dann ist $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Clusterbaum für die Indexmenge $\mathcal{I} \times \mathcal{J}$. Insbesondere gilt

$$\dot{\bigcup_{b\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}}} \mathbf{b} \;=\; \mathcal{I} \times \mathcal{J}. \tag{3.15}$$

Beweis:[siehe [12, Lemma 3.14 und Corollary 3.15]] ∎

Zusammen mit Lemma 3.2.11 können wir die Mächtigkeit eines Blockbaums gegen die Anzahl seiner Blätter abschätzen. Auf naive Weise erhalten wir mit Korollar 3.2.12 $\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \leq c\#(\mathcal{I} \times \mathcal{J})$. Diese Abschätzung ist zu grob, um später die gewünschte Ergebnisse zu beweisen. Dies liegt daran, dass Korollar 3.2.12 nur dann gute Abschätzungen liefert, wenn die Blätter ähnliche Größen haben.
Wie bereits angedeutet werden wir aber später sehr unterschiedliche Blöcke haben. Um trotzdem vernünftige Aufwandsabschätzungen zu erhalten, definieren wir die zusätzliche Eigenschaft der Schwachbesetztheit für Blockbäume, wie sie in [25, 26] eingeführt wird.

Definition 3.3.5 (schwachbesetzte Blockbäume)
Ein Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ heißt c_{sp}-*schwachbesetzt*, falls eine Konstante $c_{sp} > 0$ mit

$$c_{sp} \geq \max\left\{ \max_{t\in\mathcal{T}_{\mathcal{I}}} \#\left\{s' \in \mathcal{T}_{\mathcal{J}} \mid b = (t,s') \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\right\}, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\left\{t' \in \mathcal{T}_{\mathcal{I}} \mid b = (t',s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\right\} \right\} \tag{3.16}$$

existiert. In diesem Fall wird $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ als c_{sp}-schwachbesetzter Blockbaum bezeichnet.

Wir werden uns später noch damit beschäftigen, inwieweit diese Eigenschaft für die von uns betrachteten Blockbäume gilt. Zuerst nutzen wir die Definition, um für schwachbesetzte Blockbäume zwei Abschätzungen zu beweisen, die wir später für Aufwandsabschätzungen benötigen.

Lemma 3.3.6 (Aufwandsabschätzungen für Blockbäume)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein c_{sp}-schwachbesetzter Blockbaum. Dann gilt

$$\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \leq c_{sp} \min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}\} \tag{3.17}$$

und

$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} (\#\mathbf{t} + \#\mathbf{s}) \leq c_{sp}\,(p_{\mathcal{I}}\#\mathcal{I} + p_{\mathcal{J}}\#\mathcal{J})\,. \tag{3.18}$$

Beweis: Um die Verwendung der c_{sp}-schwachbesetzten Blockbäume vorzustellen, zeigen wir hier nochmals die Schlussweise. Aus (3.16) folgt (3.17) wie in [32, Lemma 6.3.4]:

$$\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} = \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\{s \in \mathcal{T}_{\mathcal{J}} \,|\, (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\} \leq \sum_{t\in\mathcal{T}_{\mathcal{I}}} c_{sp} = c_{sp}\#\mathcal{T}_{\mathcal{I}}.$$

Analog folgt $\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \leq c_{sp}\#\mathcal{T}_{\mathcal{J}}$. Also gilt (3.17).
Die Schlussweise für (3.18) wird in [26, Lemma 2.4] für die Speicherabschätzung der $\mathcal{H}$-Matrizen verwendet. Wir betrachten den Teil für die Zeilencluster und erhalten

$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathbf{t} = \sum_{t\in\mathcal{T}_{\mathcal{I}}} \sum_{\substack{s\in\mathcal{T}_{\mathcal{J}}\\(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}}} \#\mathbf{t} = \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\mathbf{t} \sum_{\substack{s\in\mathcal{T}_{\mathcal{J}}\\(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}}} 1 \leq c_{sp} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\mathbf{t}.$$

Mit (3.9) folgt

$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathbf{t} \leq c_{sp} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\mathbf{t} \leq c_{sp} p_{\mathcal{I}} \#\mathcal{I}.$$

Analog folgen die Abschätzung für die Spaltencluster und somit die Aussage. ∎

Wie bei Clusterbäumen führt linearer Aufwand in jedem Block eines schwachbesetzten Blockbaums nach Lemma 3.3.6 zu log-linearem Gesamtaufwand. Dies ist der typische Fall

für die einfachen $\mathcal{H}$-Matrix-Anwendungen. Für einfache Anwendungen der $\mathcal{H}^2$-Matrizen, die eine zusätzlich Hierarchie nutzen, ist der Aufwand in jedem Block konstant. Mit Lemma 3.3.6 führt der konstante Aufwand in jedem Block eines schwachbesetzten Blockbaums zu linearem Gesamtaufwand. Die Resultate entsprechen denen in Abschnitt 3.2 beschriebenen. Damit haben wir eine grundsätzliche Einschätzung für den Aufwand.
Der Aufwand entsteht teilweise nur in den Blättern des Blockbaums. Dies kann offensichtlich gegen einen Aufwand in allen Blöcken abgeschätzt werden. Falls jeder Nicht-Blatt-Block mindestens zwei Kinder hat, folgt aus Lemma 3.2.11, dass damit zumindest die Mächtigkeit des Blockbaums nicht wesentlich überschätzt wird.
Das Ziel ist, eine Matrixpartition zu konstruieren, bei der ein erheblicher Teil der Blöcke faktorisiert dargestellt werden kann, um so Speicher zu sparen. So verwenden z. B. die später vorgestellten $\mathcal{H}$-Matrizen eine Zwei-Term-Darstellung AB^T, bei der A und B jeweils nur wenige Spalten (k) haben. Dadurch reduziert sich der Speicheraufwand in jedem Block von quadratisch auf linear. Da der lineare Speicheraufwand den Faktor k beinhaltet, muss k wesentlich kleiner sein als die Anzahl der Zeilen und Spalten, um Speicher einzusparen. Deshalb dürfen die Blöcke nicht zu klein sein. Auf der anderen Seite dürfen die Blöcke nicht beliebig gewählt werden, weil sonst das k sehr groß gewählt werden muss, um eine hinreichend gute Approximation zu erreichen. (Siehe dazu die Fehlerabschätzung für das Modellproblem in Lemma 2.5.2.) Ein sehr großes k würde wiederum den Speicheraufwand vergrößern. Also wird eine Funktion benötigt, die beurteilt, ob ein Block gut approximiert werden kann oder nicht.

Definition 3.3.7 (Zulässigkeitsbedingung)
Es seien $\mathcal{T}_\mathcal{I}, \mathcal{T}_\mathcal{J}$ Clusterbäume. Dann ist eine Funktion

$$\mathrm{adm} : \mathcal{T}_\mathcal{I} \times \mathcal{T}_\mathcal{J} \to \{\mathrm{true}, \mathrm{false}\}$$

eine *Zulässigkeitsbedingung* auf $\mathcal{T}_\mathcal{I} \times \mathcal{T}_\mathcal{J}$ (engl.: admissibility).

Für das Modellproblem leitet sich die Zulässigkeitsbedingung aus der Fehlerabschätzung in Lemma 2.5.2 ab.

Beispiel 3.3.8 (Zulässigkeitsbedingung)
Es sei $\ell \in \mathbb{N}$, $n = 2^\ell$, $\mathcal{I} := \{1, \dots, n\}$ und $\mathcal{T}_\mathcal{I}$ der Clusterbaum für das Modellproblem aus Beispiel 3.2.4 zu $n_{\max} \in \mathbb{N}_{>0}$. Es sei $\Omega_i := \mathrm{supp}(\phi_i)$ für alle $i \in \mathcal{I}$ der Träger der Basisfunktion und $\Omega_t := \bigcup_{i \in \mathbf{t}} \Omega_i$ für alle $t \in \mathcal{T}_\mathcal{I}$ die Vereinigung der Träger in dem Cluster t. Dann definieren wir die Zulässigkeitsbedingung

$$\begin{aligned}\mathrm{adm}_{1d} :& \mathcal{T}_\mathcal{I} \times \mathcal{T}_\mathcal{I} \to \{\mathrm{true}, \mathrm{false}\},\\ &(t, s) \mapsto \left(\{\mathrm{diam}(\Omega_t) + \mathrm{diam}(\Omega_s)\} \leq \eta \,\mathrm{dist}(\Omega_t, \Omega_s)\right),\end{aligned}$$

wobei η ein Parameter zur Wahl der Stärke der Zulässigkeitsbedingung ist. Eine typische Wahl ist $\eta = 1$.

Der Vergleich von Durchmessern und Abständen wie in Beispiel 3.3.8 ist typisch für Zulässigkeitsbedingung im Kontext der hierarchischen Matrizen. Häufig wird bei $\mathcal{H}^2$-Matrix-Anwendungen das Maximum der Durchmesser $\mathrm{diam}(\Omega_t)$ und $\mathrm{diam}(\Omega_s)$ statt der

Summe verwendet. Und für $\mathcal{H}$-Matrizen wird typischerweise das Minimum statt der Summe verwendet (siehe [32, Kapitel 5]). Dies liegt daran, dass es für die $\mathcal{H}$-Matrizen ausreicht, bezüglich eines der beiden Cluster t oder s zu approximieren.
Mit Hilfe der Zulässigkeitsbedingung können wir die zulässigen Blockbäume definieren. Diese geben uns nicht nur eine Blockpartition der Matrix, sondern auch eine Einteilung in Blöcke mit niedrigem Rang (zulässige Blöcke) und Blöcke mit hohem Rang (unzulässige Blöcke).

Definition 3.3.9 (Zulässiger Blockbaum)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum und adm eine Zulässigkeitsbedingung auf $\mathcal{T}_{\mathcal{I}} \times \mathcal{T}_{\mathcal{J}}$. Dann heißt $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ *zulässig* bezüglich adm, falls für alle $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gilt

$$b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \;\Leftrightarrow\; ((\mathrm{adm}(b) = \mathrm{true}) \vee (t \in \mathcal{L}_{\mathcal{I}} \vee s \in \mathcal{L}_{\mathcal{J}}))$$

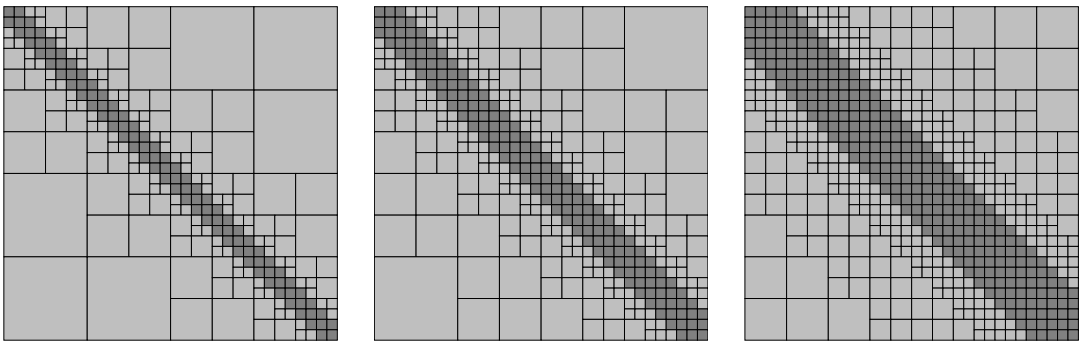
bzw. *strikt zulässig* bezüglich adm, falls für alle $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gilt

$$b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \;\Leftrightarrow\; ((\mathrm{adm}(b) = \mathrm{true}) \vee (t \in \mathcal{L}_{\mathcal{I}} \wedge s \in \mathcal{L}_{\mathcal{J}}))\,.$$

$\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ wird dann auch *(strikt) zulässiger Blockbaum* genannt. Die Blätter werden in zulässige und unzulässige unterteilt:

$$\mathcal{L}^{+}_{\mathcal{I}\times\mathcal{J}} := \{b \in \mathcal{L}_{\mathcal{I}} \mid \mathrm{adm}(b) = \mathrm{true}\} \quad \text{bzw.} \tag{3.19}$$
$$\mathcal{L}^{-}_{\mathcal{I}\times\mathcal{J}} := \{b \in \mathcal{L}_{\mathcal{I}} \mid \mathrm{adm}(b) = \mathrm{false}\}\,. \tag{3.20}$$

Starke Zulässigkeitsbedingungen (kleineres η, siehe Bild 3.2 (c)) führen zu einem Blockbaum mit mehr Blöcken und kleineren zulässigen Blöcken. Umgekehrt hat ein Blockbaum zu einer schwachen Zulässigkeitsbedingung (größeres η, siehe Bild 3.2 (a)) weniger Blöcke und größere zulässige Blöcke. Typischerweise werden dafür aber höhere Ränge für die Niedrigrangmatrizen in den zulässigen Blöcken benötigt.



(a) $\eta = 1.0$ (b) $\eta = 0.5$ (c) $\eta = 0.25$

Abbildung 3.2: Blockbäume des Modellproblems zu unterschiedlichen η: Unzulässige Blöcke sind dunkelgrau und zulässige Blöcke sind hellgrau markiert.

Die Definition 3.3.9 lässt zu, dass auch Nicht-Blatt-Blöcke zulässig sind. Diese Situation ist für die Theorie nicht relevant, solange der Blockbaum c_{sp}-schwachbesetzt bleibt. In der Praxis wollen wir, dass zulässige Blöcke stets auch Blätter sind, weil es effizienter ist, möglichst große Blöcke faktorisiert darzustellen. Dies können wir durch die Konstruktion in Algorithmus 3.3.1 sicherstellen.

Algorithmus 3.3.1 Die Funktion konstruiert den minimalen strikt zulässigen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ zu den Clusterbäumen $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$ und der Zulässigkeitsbedingung adm

```
function BLOCKBAUM(b, T_I, T_J, adm)
    (t, s) = b
    b = t × s
    if adm(t, s) = true then
        b ∈ L^+_{I×J}
        chil(b) = ∅
    else
        if (chil(t) = ∅ ∧ chil(s) = ∅) then
            b ∈ L^-_{I×J}
            chil(b) = ∅
        else
            b ∈ T_{I×J} \ L_{I×J}
            chil(b) = chil*(t) × chil*(s)
            for b' ∈ chil(b) do
                BLOCKBAUM(b', T_I, T_J, adm)
            end for
        end if
    end if
end function
```

Bemerkung 3.3.10
Der Algorithmus 3.3.1, aufgerufen mit $b = (r_{\mathcal{I}}, r_{\mathcal{J}})$, konstruiert einen strikt zulässigen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ zu den Clusterbäumen $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$ und der Zulässigkeitsbedingung adm. Für alle Blöcke $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gilt $\mathrm{adm}(t, s) = \mathrm{true} \Leftrightarrow b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.

Den minimalen zulässigen Blockbaum können wir konstruieren, indem wir die zweite `If`-Abfrage in Algorithmus 3.3.1 durch $(\mathrm{chil}(t) = \emptyset \vee \mathrm{chil}(s) = \emptyset)$ ersetzen. Der minimale zulässige Blockbaum hat höchstens so viele Blöcke wie der entsprechende minimale strikt zulässige Blockbaum. Auch sind beim minimalen zulässigen Blockbaum für alle Blöcke $b = (t, s)$ die Cluster t und s auf demselben Level.
Dafür können zulässigen Blockbäumen Blätter haben, bei denen ein Cluster noch sehr groß ist. Die zulässigen Blockbäume werden für $\mathcal{H}$-Matrizen benutzt und die strikt zulässigen für $\mathcal{H}^2$-Matrizen. Dass diese Wahl jeweils passend ist, erläutern wir später.
Für zulässige Blockbäume ist die c_{sp}-Schwachbesetztheit z. B. in [26] und [32, Abschnitt 6.4] untersucht. Dabei wird von minimalen zulässigen Blockbäumen ausgegangen, die den Vorteil haben, dass für jeden Block der Zeilen- und der Spaltencluster auf dem selben

Level liegen. Diese Eigenschaft kann für strikt zulässige Blockbäume im Allgemeinen nicht sicher gestellt werden, weshalb zusätzlich der Levelunterschied berücksichtigt werden muss. Allerdings fehlen an dieser Stelle entsprechende Veröffentlichungen.
Wie bereits erwähnt, entscheidet die Zulässigkeitsbedingung darüber, ob ein Block faktorisiert dargestellt wird oder nicht. D. h., dass die zulässigen Blätter in $\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ in faktorisierter Form und die unzulässigen Blätter in $\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ in vollbesetzter Form dargestellt werden.
Bevor wir auf Basis der zulässigen Blockbäume die hierarchischen Matrizen in Abschnitt 3.4 einführen, stellen wir noch den transponierten Blockbaum vor. Dieser ermöglicht uns, die Transponierte einer hierarchischen Matrix wieder als hierarchische Matrix darzustellen. Damit können wir später Betrachtungen für die Zeilencluster auf die Spaltencluster übertragen (und umgekehrt).

Definition 3.3.11
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum und adm eine Zulässigkeitsbedingung für $\mathcal{T}_\mathcal{I} \times \mathcal{T}_\mathcal{J}$. Dann ist der *transponierte Blockbaum* $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ durch die Blöcke

$$(s,t) \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}} \;\Leftrightarrow (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \tag{3.21}$$

und die Kinder

$$\mathrm{chil}^T : \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}} \to \mathcal{P}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}),\ (s,t) \mapsto \{(\check{s},\check{t}) \,|\, (\check{t},\check{s}) \in \mathrm{chil}(t,s)\} \tag{3.22}$$

definiert. Die Beschriftung eines Blocks $(s,t) \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ sei $\mathbf{s} \times \mathbf{t}$. Die *transponierte Zulässigkeitsbedingung* ist durch

$$\mathrm{adm}^T : \mathcal{T}_\mathcal{J} \times \mathcal{T}_\mathcal{I} \to \{\mathrm{true}, \mathrm{false}\},\ (s,t) \mapsto \mathrm{adm}(t,s) \tag{3.23}$$

gegeben.

Lemma 3.3.12
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein (strikt) zulässiger Blockbaum zu $\mathcal{I} \times \mathcal{J}$ mit Zulässigkeitsbedingung adm. Dann ist $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ein (strikt) zulässiger Blockbaum zu $\mathcal{J} \times \mathcal{I}$ mit Zulässigkeitsbedingung adm^T.

Beweis: Nach (3.12) müssen wir zeigen, dass $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ein Baum mit $r^T_{\mathcal{I}\times\mathcal{J}} = (r_\mathcal{J}, r_\mathcal{I})$ ist. Da $(r_\mathcal{I}, r_\mathcal{J}) = r_{\mathcal{I}\times\mathcal{J}} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gilt, ist $r^T_{\mathcal{I}\times\mathcal{J}} \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$.
Es sei $(s,t) \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$. Wir müssen zeigen, dass ein eindeutiger Pfad von $r^T_{\mathcal{I}\times\mathcal{J}}$ zu (s,t) existiert. Dazu zeigen wir, dass $(t_i,s_i)_{i=0}^m$ genau dann ein Pfad in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ist, wenn $(s_i,t_i)_{i=0}^m$ ein Pfad in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ist.
Es sei $(t_i,s_i)_{i=0}^m$ ein Pfad in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Wegen $(t_i,s_i) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und (3.21) gilt $(s_i,t_i) \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ für alle $i \in \{0,\dots,m\}$. Und aus $(t_i,s_i) \in \mathrm{chil}(t_{i-1},s_{i-1})$ und (3.22) folgt $(s_i,t_i) \in \mathrm{chil}^T((s_{i-1},t_{i-1}))$ für alle $i \in \{1,\dots,m\}$. Also ist $(s_i,t_i)_{i=0}^m$ ein Pfad in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$. Die umgekehrte Implikation folgt analog.
Nach Definition des Baums existiert ein Pfad $(t_i,s_i)_{i=0}^m$ in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $(t_0,s_0) = r_{\mathcal{I}\times\mathcal{J}} = (r_\mathcal{I}, r_\mathcal{J})$ und $(t_m,s_m) = (t,s)$. Dann ist $(s_i,t_i)_{i=0}^m$ ein Pfad in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ mit $(s_0,t_0) = (r_\mathcal{J}, r_\mathcal{I}) =$

$r^T_{\mathcal{I}\times\mathcal{J}}$ und $(s_m, t_m) = (s, t)$. Also existiert ein Pfad in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ von $r^T_{\mathcal{I}\times\mathcal{J}}$ zu (s, t).
Es seien $p_1 = (s^1_i, t^1_i)^m_{i=0}$ und $p_2 = (s^2_i, t^2_i)^n_{i=0}$ zwei Pfade in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ von $r^T_{\mathcal{I}\times\mathcal{J}}$ zu (s, t). Dann sind $(t^1_i, s^1_i)^m_{i=0}$ und $(t^2_i, s^2_i)^n_{i=0}$ Pfade in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ von $r_{\mathcal{I}\times\mathcal{J}}$ zu (t, s). Da $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Baum ist, müssen beide Pfade gleich sein und somit muss auch $p_1 = p_2$ gelten. Also ist der Pfad in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ von $r^T_{\mathcal{I}\times\mathcal{J}}$ zu (s, t) eindeutig und $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ein Baum.
Mit (3.23) folgt für alle $(s, t) \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ die Bedingung (3.13) an die Kinder

$$\begin{aligned}\mathrm{chil}^T(s,t) &= \{(\breve{s},\breve{t}) \,|\, (\breve{t},\breve{s}) \in \mathrm{chil}(t,s)\} = \{(\breve{s},\breve{t}) \,|\, \mathrm{chil}^*(t) \times \mathrm{chil}^*(s)\} \\ &= \mathrm{chil}^*(s) \times \mathrm{chil}^*(t).\end{aligned}$$

Die Beschriftung von $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ erfüllt per Definition die Bedingung (3.14). Somit ist $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum.
Um zu zeigen, dass $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ zulässig ist, müssen wir nach Definition 3.3.9 die Blätter betrachten. Aufgrund der Definition von chil^T gilt für die Blätter von $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$

$$\mathcal{L}^T_{\mathcal{I}\times\mathcal{J}} = \{(s,t) \,|\, (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}\}.$$

Dann gilt für alle $(s, t) \in \mathcal{L}^T_{\mathcal{I}\times\mathcal{J}}$ wegen $(t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ (siehe Definition 3.3.9)

$$((\mathrm{adm}(t,s) = \mathrm{true}) \vee (t \in \mathcal{L}_{\mathcal{I}} \vee s \in \mathcal{L}_{\mathcal{J}})) \Leftrightarrow \left((\mathrm{adm}^T(s,t) = \mathrm{true}) \vee (s \in \mathcal{L}_{\mathcal{J}} \vee t \in \mathcal{L}_{\mathcal{I}})\right).$$

Also ist $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum mit Zulässigkeitsbedingung adm^T. Die strikte Zulässigkeit von $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ folgt analog, falls $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ strikt zulässig ist. ■

Wie wir im Beweis sehen, ist es möglich, die Eigenschaften des Blockbaums getrennt von der Zulässigkeit zu zeigen. Allerdings benötigen wir die Aussagen im Folgenden nur zusammenhängend.
Für spätere Untersuchungen des Aufwands ist es wichtig, dass der transponierte Blockbaum die gleiche c_{sp}-Konstante wie der ursprüngliche Baum hat.

Lemma 3.3.13
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein c_{sp}-schwachbesetzter Blockbaum. Dann ist der transponierte Blockbaum $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ ebenfalls c_{sp}-schwachbesetzt.

Beweis: Wir betrachten die Mengen aus (3.16). Nach (3.21) gilt für alle $s \in \mathcal{T}_{\mathcal{I}}$

$$\#\left\{t' \in \mathcal{T}_{\mathcal{I}} \mid b = (s,t') \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}\right\} = \#\left\{t' \in \mathcal{T}_{\mathcal{I}} \mid b = (t',s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\right\} \leq c_{sp}$$

und für alle $t \in \mathcal{T}_{\mathcal{I}}$

$$\#\left\{s' \in \mathcal{T}_{\mathcal{J}} \mid b = (s',t) \in \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}\right\} = \#\left\{s' \in \mathcal{T}_{\mathcal{J}} \mid b = (t,s') \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\right\} \leq c_{sp}.$$

Damit ist $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ nach Definition 3.3.5 c_{sp}-schwachbesetzt. ■

Zum Abschluss dieses Abschnitts zeigen wir noch einige technische Aussagen zum Zusammenhang zwischen Blöcken und ihren Clustern.

Lemma 3.3.14
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum und $b = (t, s), b' = (t', s') \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $(t, s) \neq (t', s')$.

(i) Dann gilt $\text{level}(b) = \max\{\text{level}(t), \text{level}(s)\}$.

(ii) Für $\text{level}(t) < \text{level}(b)$ gilt $t \in \mathcal{L}_{\mathcal{I}}$ und für $\text{level}(s) < \text{level}(b)$ gilt $s \in \mathcal{L}_{\mathcal{J}}$.

(iii) Es gilt $b' \in \text{pred}(b)$ genau dann, wenn $t' \in \text{pred}(t)$ und $s' \in \text{pred}(s)$ ist.

(iv) Für alle $\check{t} \in \text{desc}(t)$ und $\hat{s} \in \text{pred}(s)\backslash\{s\}$ mit $(\check{t}, \hat{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gilt, dass $t = \check{t} \in \mathcal{L}_{\mathcal{I}}$ und $(\check{t}, \hat{s}) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ ist.

(v) Aus $b' = (t', s') \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ folgt $t' \notin \text{pred}(t)$ oder $s' \notin \text{pred}(s)$.

Beweis: Zuerst untersuchen wir die Cluster zu einem Pfad im Blockbaum. Dazu sei $m \in \mathbb{N}$, $(t_i)_{i=0}^m \in \mathcal{T}_{\mathcal{I}}^{m+1}$ mit $t_i \in \text{chil}^*(t_{i-1})$ für alle $i \in \{0, \dots, m\}$. Wir definieren

$$m_t := \min\{i \in \{0, \dots, m\} \mid t_i \in \mathcal{L}_{\mathcal{I}} \vee i = m\},$$

wobei die Bedingung $i = m$ sicherstellt, dass das Minimum nicht über die leere Menge genommen wird. Dann ist $t_i \notin \mathcal{L}_{\mathcal{I}}$ und somit $\text{chil}^*(t_i) = \text{chil}(t_i)$ für alle $i \in \{0, \dots, m_t - 1\}$. Also ist $(t_i)_{i=0}^{m_t}$ ein Pfad. Des Weiteren folgt für alle $i \in \{m_t, \dots, m-1\}$ induktiv $\text{chil}^*(t_i) = \{t_i\}$ und somit $t_{i+1} = t_i$. Also ist $(t_i)_{i=0}^{m_t}$ ein Pfad von t_0 zu $t_{m_t} = t_m$.
(i): Es sei $(b_i)_{i=0}^m = ((t_i, s_i))_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu b. Weiter seien m_t wie oben für $(t_i)_{i=0}^m$ und analog m_s für $(s_i)_{i=0}^m$ definiert. Dann ist $(t_i)_{i=0}^{m_t}$ ein Pfad von $r_{\mathcal{I}}$ zu t und damit $\text{level}(t) = m_t$. Genauso gilt $\text{level}(s) = m_s$. Nach Bemerkung 3.3.3 gilt $t_i \notin \mathcal{L}_{\mathcal{I}}$ oder $s_i \notin \mathcal{L}_{\mathcal{J}}$ für alle $i \in \{0, \dots, m-1\}$. Also gilt $\{m_t, \dots, m-1\} \cap \{m_s, \dots, m-1\} = \emptyset$ und somit

$$\max\{\text{level}(t), \text{level}(s)\} \;=\; \max\{m_t, m_s\} \;=\; m \;=\; \text{level}(b).$$

(ii): Wir verwenden die Bezeichnungen aus dem Beweis zu (i). Es sei $\text{level}(t) < \text{level}(b)$. Dann gilt $m_t = \text{level}(t) < \text{level}(b) = m$. Damit folgt $t = t_m = t_{m_t} \in \mathcal{L}_{\mathcal{I}}$. Der Fall $\text{level}(s) < \text{level}(b)$ folgt analog.
(iii): ($\Rightarrow$): Es sei $b' \in \text{pred}(b)$. Dann existiert ein Pfad $(b_i)_{i=0}^m = ((t_i, s_i))_{i=0}^m$ von b' zu b. Es sei m_t für $(t_i)_{i=0}^m$ wie in der Vorbetrachtung oben definiert. Dann ist $(t_i)_{i=0}^{m_t}$ ein Pfad von $t_0 = t'$ zu $t_{m_t} = t_m = t$ und somit $t' \in \text{pred}(t)$. Analog folgt $s' \in \text{pred}(s)$.
($\Leftarrow$): Es sei $t' \in \text{pred}(t)$ und $s' \in \text{pred}(s)$. Es sei $(b_i)_{i=0}^m = ((t_i, s_i))_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu b und $(b'_i)_{i=0}^{m'} = ((t'_i, s'_i))_{i=0}^{m'}$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu b'. Wegen $t' \in \text{pred}(t)$ und $s' \in \text{pred}(s)$ gilt nach (i)

$$m \;=\; \text{level}(b) \;=\; \max\{\text{level}(t), \text{level}(s)\} \;\geq\; \max\{\text{level}(t'), \text{level}(s')\} \;=\; \text{level}(b') \;=\; m'.$$

Wir zeigen induktiv $b_n = b'_n$ für alle $n \in \{0, \dots, m'\}$. Es gilt $b_0 = r_{\mathcal{I}\times\mathcal{J}} = b'_0$.
Es sei $n \in \{0, \dots, m'-1\}$ mit $b_n = b'_n$. Dann gilt $t_{n+1} \in \text{pred}(t)$ und $t'_{n+1} \in \text{pred}(t') \subseteq \text{pred}(t)$. Wegen der Eindeutigkeit des Pfades von $t_n = t'_n$ zu t folgt $t_{n+1} = t'_{n+1}$. Analog folgt $s_{n+1} = s'_{n+1}$ und somit $b_{n+1} = b'_{n+1}$. Also gilt $b_{m'} = b'_{m'} = b'$ und damit $b' \in \text{pred}(b)$.

(iv): Es sei $\check{t} \in \operatorname{desc}(t)$ und $\hat{s} \in \operatorname{pred}(s)\backslash\{s\}$ mit $(\check{t}, \hat{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. (Falls diese nicht existieren, folgt die Aussage direkt.) Dann gilt $\hat{s} \notin \mathcal{L}_{\mathcal{J}}$. Mit (i) und (ii) folgt

$$\operatorname{level}(t) \;\leq\; \operatorname{level}(\check{t}) \;\leq\; \operatorname{level}(\check{t}, \hat{s}) \;=\; \operatorname{level}(\hat{s}) \;<\; \operatorname{level}(s) \;\leq\; \operatorname{level}(t, s).$$

Aus (ii) folgt $t \in \mathcal{L}_{\mathcal{I}}$ und somit $\check{t} = t$.
Es bleibt zu zeigen, dass $(t, \hat{s}) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ ist. Mit $t \in \operatorname{pred}(t)$ und $\hat{s} \in \operatorname{pred}(s)\backslash\{s\}$ folgt $(t, \hat{s}) \in \operatorname{pred}(t, s)\backslash\{(t, s)\} \subseteq \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ aus (iii).
(v): Wegen $b' \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $b' \neq b$ folgt $b' \notin \operatorname{pred}(b)$. Mit (iii) folgt die Aussage. ∎

Als Nächstes betrachten wir, wie aus der durch die Blockbäume konstruierten Blockpartitionen datenschwache Darstellungen für Matrizen definiert werden können.

3.4 Hierarchische Matrizen

Wir stellen in diesem Abschnitt die hierarchischen Matrizen (sowohl $\mathcal{H}$- als auch $\mathcal{H}^2$-Matrizen) vor. Dabei führen wir auch die $\mathcal{H}$-Matrizen kurz ein, um die Struktur der $\mathcal{H}^2$-Matrizen besser erläutern zu können. Insbesondere soll herausgestellt werden, welcher Aufwand an gewissen Stellen erlaubt ist, um lineare bzw. log-lineare Komplexität zu erreichen.

Definition 3.4.1
Es sei $\mathcal{T}_{\mathcal{I}}$ ein Clusterbaum und $\kappa = (\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ eine Familie von paarweise disjunkten Indexmengen κ_t. Dann nennen wir κ eine *Rangverteilung* zu $\mathcal{T}_{\mathcal{I}}$.

Bemerkung 3.4.2
Falls ein $k \in \mathbb{N}$ mit $\#\kappa_t = k$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ existiert, bezeichnen wir k auch als den lokalen Rang.

Da jeder Blockbaum insbesondere ein Clusterbaum ist, sind Rangverteilungen auch auf Blockbäumen definiert. Damit können wir die Menge der $\mathcal{H}$-Matrizen zu einem zulässigen Blockbaum und einer Rangverteilung definieren.

Definition 3.4.3 ($\mathcal{H}$-Matrizen)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und $\kappa := (\kappa_b)_{b\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}}$ eine Rangverteilung. Dann ist

$$\mathcal{H}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \kappa\right) \;:=\; \left\{X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \mid \forall b \in \mathcal{L}^{+}_{\mathcal{I}\times\mathcal{J}} : \operatorname{Rang}\left(X_{|b}\right) \leq \#\kappa_b\right\} \tag{3.24}$$

die Menge der $\mathcal{H}$-Matrizen mit Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ zur Rangverteilung κ.

Die in Definition 2.4.10 eingeführte Rang-k-Darstellung für Matrizen mit Rang k ermöglicht die Reduzierung des Speicherbedarfs auf lineare $(k(\#\mathcal{I} + \#\mathcal{J}))$ statt des quadratischen $(\#\mathcal{I}\#\mathcal{J})$ für vollbesetzte Matrizen. Dies motiviert die folgende Darstellung für $\mathcal{H}$-Matrizen.

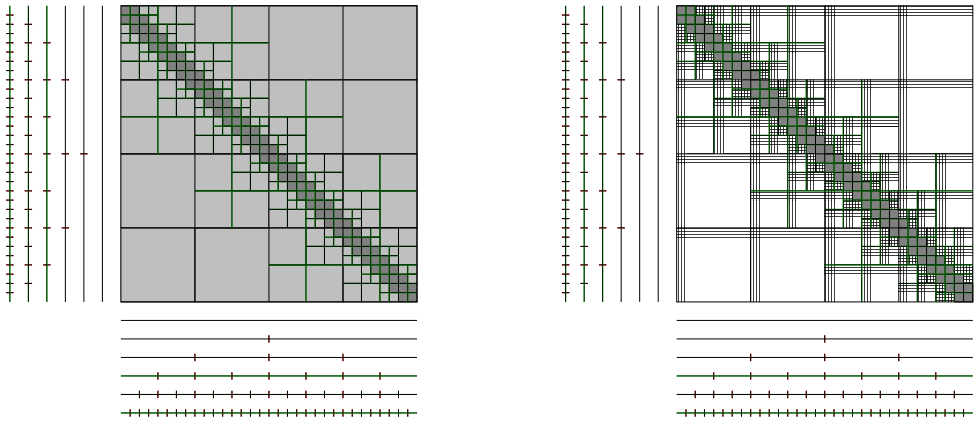


Abbildung 3.3: Vergleich vollbesetzte Blockmatrix zu $\mathcal{H}$-Matrix: Die vollbesetzten Matrizen in den zulässigen (hellgrauen) Blöcken links werden rechts durch Niedrigrangmatrizen AB^T (gestreift) ersetzt.

Definition 3.4.4 ($\mathcal{H}$-Matrix-Darstellung)
Es sei $X \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \kappa)$. Es sei $(A, B) := (A_b, B_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$, wobei (A_b, B_b) für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ eine Rang-κ_b-Darstellung von $X_{|\mathbf{b}}$ sei (siehe Definition 2.4.10). Weiter sei $N := (N_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} := (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}$. Dann ist

$$(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\ \kappa,\ N,\ (A, B))$$

eine *$\mathcal{H}$-Matrix-Darstellung* von X. Die zu der Darstellung gehörige $\mathcal{H}$-Matrix wird mit

$$\mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\ \kappa,\ N,\ (A, B)) \ =\ X$$

bezeichnet.

Die Verwendung von N_b für die Matrizen $X_{|\mathbf{b}}$, $b \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$, stammt von dem Begriff der Nahfeld-Matrizen. Diese Bezeichnung erklärt sich durch Zulässigkeitsbedingungen wie aus Beispiel 3.3.8, weil bei diesen, nah (im Verhältnis zu ihrer Größe) beieinander liegende Cluster unzulässig sind. Für das Modellproblem können wir die Rang-k-Matrizen definieren.

Beispiel 3.4.5
Um eine $\mathcal{H}$-Matrix-Approximation für das Modellproblem aus Abschnitt 2.5 zu definieren, fixieren wir ein $m \in \mathbb{N}$ und definieren für jeden zulässigen Block $b = (t, s)$ die Rang-m-Darstellung (A_b, B_b) durch

$$A_b \ :=\ V_{t,m} \quad \text{ und } B_b \ :=\ W_{s,m}S^*_{b,m},$$

wobei $V_{t,m}$, $W_{s,m}$ und $S_{b,m}$ die Matrizen aus Definition 2.5.3 seien. Falls wir die Zulässigkeitsbedingung adm_{1d} aus Beispiel 3.3.8 verwenden, liefert uns Lemma 2.5.4 eine obere Schranke für den Fehler.

Als Nächstes betrachten wir den Speicheraufwand für $\mathcal{H}$-Matrizen. Im Falle eines c_{sp}-schwachbesetzten Blockbaums können wir den Speicheraufwand folgendermaßen abschätzen.

Lemma 3.4.6 (Speicheraufwand für $\mathcal{H}$-Matrizen)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger c_{sp}-schwachbesetzter Blockbaum,

$$X = \mathcal{H}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\ \kappa,\ N,\ (A,B)\right).$$

eine $\mathcal{H}$-Matrix, $n_{\max} := \max\limits_{r\in\mathcal{L}_{\mathcal{I}}\cup\mathcal{L}_{\mathcal{J}}} \#\mathbf{r}$ und $k_{\max} := \max\limits_{b\in\mathcal{L}^{+}_{\mathcal{I}\times\mathcal{J}}} \#\kappa_b$. Dann ist der Speicheraufwand für die $\mathcal{H}$-Matrix-Darstellung von X

$$\mathcal{N}_{st}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\ \kappa,\ N,\ (A,B)\right) \ \leq\ c_{sp}\max\left\{n_{\max},k_{\max}\right\}\left(p_{\mathcal{I}}\#\mathcal{I}+p_{\mathcal{J}}\#\mathcal{J}\right).$$

Beweis:[siehe [26, Lemma 2.4]] ∎

Im Beweis wird ausgenutzt, dass sich der Speicheraufwand für jedes Blatt $(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ durch $\max\{n_{\max},k_{\max}\}(\#\mathbf{t}+\#\mathbf{s})$ abschätzen lässt. Die Abschätzung für den gesamten Speicheraufwand folgt dann wie in Lemma 3.3.6.
Um die Aufwandsschranken einfach zu vergleichen, verwenden wir das Landau-Symbol $\mathcal{O}$. Dabei wollen wir die Entwicklung in der Problemgröße n und dem lokalen Rang k beschreiben. Dazu machen wir einige grundsätzliche Annahmen.

Bemerkung 3.4.7
Die Problemdimension bezeichnen wir mit $n\in\mathbb{N}$ und den lokalen Rang mit $k\in\mathbb{N}$, wobei dieser von n abhängen kann. Wir gehen davon aus, dass die Dimension der Problemmatrix sich wie n verhält, d. h. $\#\mathcal{I}$, $\#\mathcal{J}\in\mathcal{O}(n)$. Für die Clusterbäume gelte $\#\mathcal{T}_{\mathcal{I}}$, $\#\mathcal{T}_{\mathcal{J}}\in\mathcal{O}(n/k)$, $p_{\mathcal{I}}$, $p_{\mathcal{J}}\in\mathcal{O}(\log(n))$ und $\#\mathbf{t}$, $\#\mathbf{s}\in\mathcal{O}(k)$ für alle $t\in\mathcal{L}_{\mathcal{I}}$ und $s\in\mathcal{L}_{\mathcal{J}}$. Der Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ sei c_{sp}-schwachbesetzt mit von n unabhängiger Konstante c_{sp}. Außerdem seien alle lokalen Ränge in $\mathcal{O}(k)$.

Die Annahme zur Mächtigkeit der Clusterbäume ist durch Korollar 3.2.12 motiviert. Auch wird typischerweise angenommen, dass die Clusterbäume nicht entartet sind, d. h. die Tiefe sich wie $\log(n)$ verhält. Dies ist bei geometrischen Clusterstrategien für gleichmäßig verteilte Freiheitsgrade oder für kardinalitätsbalancierten Clusterstrategien erfüllt. Die Größe der Blattcluster kann normalerweise vom Anwender geeignet gewählt werden. Die Schwachbesetztheit der Blockbäume ist eine typische Annahme in Kontext der hierarchischen Matrizen (siehe [12, 32]).
Im Falle der Annahme aus Bemerkung 3.4.7 folgt, dass der Speicheraufwand für $\mathcal{H}$-Matrizen in $\mathcal{O}(n\log(n)k)$ liegt. Der Rechenaufwand der Matrix-Vektor-Multiplikation für $\mathcal{H}$-Matrizen lässt sich gegen den zweifachen Speicheraufwand abschätzen (siehe [26, Lemma 2.5]).

Bemerkung 3.4.8
Nicht-zulässige Blätter eines zulässigen aber nicht strikt zulässigen Blockbaums können aus einem Blatt-Cluster und einem Nicht-Blatt-Cluster bestehen. Da $k_{\max}$ und $n_{\max}$ typischerweise ähnlich groß sind, bringt eine weitere Aufteilung solcher Blöcke bei $\mathcal{H}$-Matrizen keine wesentlichen Speicherersparnisse. Deshalb werden bei $\mathcal{H}$-Matrizen typischerweise zulässige Blockbäume genutzt.

Als ersten Schritt zur Reduzierung des Speicherbedarfs betrachten wir die uniformen $\mathcal{H}$-Matrizen. Für diese nutzen wir eine Drei-Term-Faktorisierung $V_b S_b W_b^T$ mit einer (kleinen) Kopplungsmatrix $S_b \in \mathbb{R}^{\kappa_t \times \lambda_s}$ und lokalen Rängen κ_t und λ_s. In dieser Form wird der Speicheraufwand nicht reduziert, da die Matrizen V_b und W_b die gleiche Größe wie A_b bzw. B_b haben. Wir verwenden die Matrix V_b allerdings nicht nur für den Block $b = (t, s)$, sondern auch für alle Blöcke $\tilde{b} = (t, \tilde{s})$ mit Zeilencluster t. Deshalb bezeichnen wir die Matrix mit V_t. Dieses Vorgehen verlagert einen Teil des Speicheraufwands von den Blöcken in die Cluster. Darauf basiert die Definition der uniformen $\mathcal{H}$-Matrizen.

Definition 3.4.9 (Uniforme $\mathcal{H}$-Matrizen)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und $\kappa := (\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $\lambda := (\lambda_s)_{s\in\mathcal{T}_\mathcal{J}}$ Rangverteilungen. Weiter seien $V_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t}$, $t \in \mathcal{T}_\mathcal{I}$, und $W_s \in \mathbb{R}^{\mathcal{J}\times\lambda_s}_{\mathbf{s}\times\lambda_s}$, $s \in \mathcal{T}_\mathcal{J}$ gegeben. Dann definiert

$$\begin{aligned}&\mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)\\ &\quad := \{X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \mid \forall b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \exists S_b \in \mathbb{R}^{\kappa_t\times\lambda_s} : X_{\mathbf{b}} = V_t S_b W_s^*\}\end{aligned}$$

die Menge der *uniformen $\mathcal{H}$-Matrizen* zum Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und den Matrixfamilien $(V_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $(W_s)_{s\in\mathcal{T}_\mathcal{J}}$.

Offensichtlich sind die uniformen $\mathcal{H}$-Matrizen $\mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)$ eine Teilmenge der $\mathcal{H}$-Matrizen $\mathcal{H}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \tilde{\kappa}\right)$, wobei die Rangverteilung $\tilde{\kappa}$ für alle $(t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ durch $\tilde{\kappa}_{(t,s)} := \kappa_t$ gegeben sei. (Dazu wähle $A_b := V_t$ und $B_b := W_s S_b^T$ für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.)
Bei der Definition der uniformen $\mathcal{H}$-Matrizen wird die Darstellung der zulässigen Blöcke durch die Festlegung der Matrizen V_t und W_s deutlich eingeschränkt. Dies hat den Vorteil, dass die uniformen $\mathcal{H}$-Matrizen einen Teilraum des $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ bilden.

Lemma 3.4.10 (Raum der uniformen $\mathcal{H}$-Matrizen)
Die Menge der uniformen $\mathcal{H}$-Matrizen $\mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)$ bildet einen Teilraum des $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$.

Beweis: Der Beweis verläuft analog zur Aussage für $\mathcal{H}^2$-Matrizen in [12, Remark 3.37]. Es seien $A, B \in \mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)$ und $\alpha \in \mathbb{R}$. Um die Aussage zu beweisen, zeigen wir $A + \alpha B \in \mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)$. Dazu betrachten wir ein zulässiges Blatt $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Nach Definition existieren Matrizen $S_{A,b}$ und $S_{B,b}$ mit

$$(A + \alpha B)_{|\mathbf{b}} \;=\; A_{|\mathbf{b}} + \alpha B_{|\mathbf{b}} \;=\; V_t S_{A,b} W_s^* + \alpha V_t S_{B,b} W_s^* \;=\; V_t (S_{A,b} + \alpha S_{B,b}) W_s^*.$$

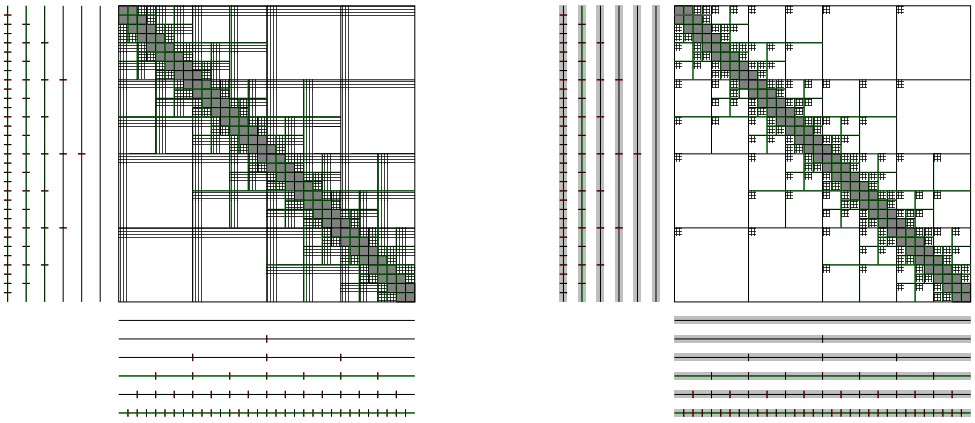


Abbildung 3.4: Vergleich $\mathcal{H}$-Matrix zu uniformer $\mathcal{H}$-Matrix: Es werden nur noch $k \times k$-Matrizen in den zulässigen Blöcken gespeichert. Dafür müssen die Matrizen V_t bzw. W_s in den Clustern gespeichert werden.

Somit folgt die Aussage. ∎

Als Nächstes definieren wir die Darstellung der uniformen $\mathcal{H}$-Matrizen.

Definition 3.4.11 (Darstellung von uniformen $\mathcal{H}$-Matrizen)
Es sei $X \in \mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)$. Für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ seien $S_b \in \mathbb{R}^{\kappa_t\times\lambda_s}$ mit $X_{|\mathbf{b}} = V_t S_b W_s^T$. Dann ist das Tupel

$$\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\, (V_t)_{t\in\mathcal{T}_\mathcal{I}},\, (W_s)_{s\in\mathcal{T}_\mathcal{J}},\, (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}},\, (S_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}\right)$$

eine *uniforme $\mathcal{H}$-Matrix-Darstellung* von X und die Matrix wird mit

$$X \;=\; \mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\, (V_t)_{t\in\mathcal{T}_\mathcal{I}},\, (W_s)_{s\in\mathcal{T}_\mathcal{J}},\, (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}},\, (S_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}\right)$$

bezeichnet.

Nach Bemerkung 2.2.10 werden die Matrizen $V_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t}$ und $W_s \in \mathbb{R}^{\mathcal{J}\times\lambda_s}_{\mathbf{s}\times\lambda_s}$ natürlich nur als Matrizen in $\mathbb{R}^{\mathbf{t}\times\kappa_t}$ bzw. $\mathbb{R}^{\mathbf{s}\times\lambda_s}$ gespeichert. Bevor wir den Speicheraufwand betrachten, wollen wir ein Beispiel für eine uniforme $\mathcal{H}$-Matrix angeben.

Beispiel 3.4.12
Die uniforme $\mathcal{H}$-Matrix-Approximation zu Rang $m \in \mathbb{N}$ für das Modellproblem aus Abschnitt 2.5 definieren wir mit Hilfe der Matrizen $(V_{t,m})_{t\in\mathcal{T}_\mathcal{I}}$, $(W_{s,m})_{s\in\mathcal{T}_\mathcal{J}}$ und $(S_{b,m})_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ aus Definition 2.5.3. Falls der zugrunde liegende Blockbaum die Zulässigkeitsbedingung adm_{1d} aus Beispiel 3.3.8 verwendet, liefert uns Lemma 2.5.4 eine Fehlerkontrolle.

Als Nächstes schätzen wir den Speicherbedarf ab, wobei wir den Speicherbedarf für die Darstellung des Raums durch die Matrizen V_t und W_s und für die Darstellung eines Elements durch die Matrizen $X_{|\mathbf{b}}$ und S_b einzeln betrachten.

Lemma 3.4.13 (Speicheraufwand für uniforme $\mathcal{H}$-Matrizen)
Es sei $\mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right)$ ein Raum von uniformen $\mathcal{H}$-Matrizen mit strikt zulässigem c_{sp}-schwachbesetztem Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und Rangverteilungen $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $(\lambda_s)_{s\in\mathcal{T}_\mathcal{J}}$. Weiter sei

$$n_{\max} := \max_{r\in\mathcal{L}_\mathcal{I}\cup\mathcal{L}_\mathcal{J}} \#\mathbf{r} \quad \text{und} \quad k_{\max} := \max\left\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\right\}.$$

Dann ist der Speicheraufwand für die Darstellung des Raums durch

$$\mathcal{N}_{st}((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}) \;\leq\; k_{\max}(p_\mathcal{I}\#\mathcal{I} + p_\mathcal{J}\#\mathcal{J})$$

beschränkt. Falls die Darstellung des Raums gespeichert ist, benötigt jedes Element $X = \mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}, (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}, (S_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}\right)$ einen Speicheraufwand von höchstens

$$\mathcal{N}_{st}\left((X_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}, (S_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}\right) \;\leq\; c_{sp}\max\left\{n_{\max}^2, k_{\max}^2\right\}\min\left\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\right\}.$$

Beweis: Der Beweis ähnelt der Abschätzung für $\mathcal{H}^2$-Matrizen in [12, Lemma 3.38].
Für die Matrizen in den Clusterbäumen folgt aus (3.9)

$$\mathcal{N}_{st}\left((V_t)_{t\in\mathcal{T}_\mathcal{I}}\right) \;=\; \sum_{t\in\mathcal{T}_\mathcal{I}} \kappa_t\#\mathbf{t} \;\leq\; k_{\max}p_\mathcal{I}\#\mathcal{I}$$

und

$$\mathcal{N}_{st}\left((W_t)_{s\in\mathcal{T}_\mathcal{J}}\right) \;=\; \sum_{s\in\mathcal{T}_\mathcal{J}} \lambda_s\#\mathbf{s} \;\leq\; k_{\max}p_\mathcal{J}\#\mathcal{J}.$$

Beides zusammengesetzt ergibt die erste Aufwandsabschätzung.
Um den Aufwand für ein Element abzuschätzen, betrachten wir die Blätter des Blockbaums. Es sei $b = (t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ ein unzulässiges Blatt. Weil $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ strikt zulässig ist, folgt $t \in \mathcal{L}_\mathcal{I}$ und $s \in \mathcal{L}_\mathcal{J}$. Also gilt

$$\mathcal{N}_{st}\left(X_{|b}\right) \;=\; \#\mathbf{t}\#\mathbf{s} \;=\; n_{\max}^2.$$

Für ein zulässiges Blatt $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gilt

$$\mathcal{N}_{st}\left(S_b\right) \;=\; \kappa_t\lambda_s \;\leq\; k_{\max}^2.$$

Aus der c_{sp}-Schwachbesetztheit von $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ folgt mit (3.17)

$$\begin{aligned}\sum_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \mathcal{N}_{st}\left(X_{|b}\right) + \sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \mathcal{N}_{st}\left(S_b\right) \;&=\; \sum_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} n_{\max}^2 + \sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} k_{\max}^2 \\ &=\; c_{sp}\max\left\{n_{\max}^2, k_{\max}^2\right\}\min\left\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\right\}.\end{aligned}$$

Somit gilt die Behauptung. ∎

Falls die Annahmen aus Bemerkung 3.4.7 gelten, erhalten wir, dass der Speicheraufwand für ein einzelnes Element in $\mathcal{O}(nk)$ liegt. Allerdings besitzt der Speicheraufwand zum Darstellen des Raums der uniformen $\mathcal{H}$-Matrizen die Komplexität $\mathcal{O}(n\log(n)k)$. Um diese noch weiter zu verringern, nutzen wir die Hierarchie der Clusterbäume aus. Die Matrizen V_t speichern wir nur in den Blättern direkt. Für alle Nicht-Blatt-Cluster verwenden wir sogenannte Transfermatrizen, mit denen sich V_t aus den Matrizen für die Kinder berechnen lässt. Diese Idee fassen wir mit dem Begriff der sogenannten Clusterbasen zusammen.

Definition 3.4.14 (Clusterbasen)
Es sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum und $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Rangverteilung. Dann ist $((V_t)_{t\in\mathcal{T}_\mathcal{I}},(E_t)_{t\in\mathcal{T}_\mathcal{I}})$ mit

$$V_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t} \quad \text{und} \quad E_t \in \mathbb{R}^{\kappa_t\times\kappa_{t'}} \quad \text{für alle } t\in\mathcal{T}_\mathcal{I}, t' := \operatorname{par}(t)$$

eine *Clusterbasis*, falls für alle $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ gilt

$$V_t \;=\; \sum_{\breve{t}\in\operatorname{chil}(t)} V_{\breve{t}} E_{\breve{t}}. \tag{3.25}$$

Bemerkung 3.4.15
Für $t_1,t_2\in\operatorname{chil}(t)$, $t_1\neq t_2$, gilt $\mathbf{t}_1\cap\mathbf{t}_2=\emptyset$. Und wegen $V_{\breve{t}}\in\mathbb{R}^{\mathcal{I}\times\kappa_{\breve{t}}}_{\breve{\mathbf{t}}\times\kappa_{\breve{t}}}$ gilt $\Pi_{\breve{t}}V_{\breve{t}}=V_{\breve{t}}$ für alle $\breve{t}\in\operatorname{chil}(t)$. Deshalb ist (3.25) äquivalent zu

$$V_{t|\breve{t}} \;=\; \Pi_{\breve{\mathbf{t}}}V_t \;=\; \sum_{r\in\operatorname{chil}(t)} \Pi_{\breve{\mathbf{t}}}V_rE_r \;=\; \sum_{r\in\operatorname{chil}(t)} \Pi_{\breve{\mathbf{t}}}\Pi_{\mathbf{r}}V_rE_r \;=\; V_{\breve{t}}E_{\breve{t}}\,. \tag{3.26}$$

Als Erstes wollen wir eine Clusterbasis für unser Modellproblem angeben.

Beispiel 3.4.16
Es sei $m\in\mathbb{N}$. Für $t\in\mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$, $\breve{t}\in\operatorname{chil}(t)$ definieren wir die Transfermatrix $E\in\mathbb{R}^{\mathcal{K}\times\mathcal{K}}$, $\mathcal{K}:=\{0,\dots,m-1\}$,

$$(E_{\breve{t}})_{\mu,\nu} \;:=\; \begin{cases} \frac{(x_{\breve{t}}-x_t)^{\nu-\mu}}{(\nu-\mu)!} & \text{, falls } \mu\le\nu \\ 0 & \text{, falls } \mu>\nu \end{cases},$$

wobei x_t, $x_{\breve{t}}$ die Mittelpunkte der Cluster t bzw. $\breve{t}$ sind. (Mit dem Mittelpunkt eines Cluster t ist der Mittelpunkt x_t der Vereinigung der Träger der Basisfunktionen Ω_t aus Beispiel 3.3.8 gemeint.) Dann ist $((V_{t,m})_{t\in\mathcal{T}_\mathcal{I}},(E_{t,m})_{t\in\mathcal{T}_\mathcal{I}})$ mit $V_{t,m}$ aus Definition 2.5.3 eine Clusterbasis.

Beweis:[siehe [12, Abschnitt 2.8]] ∎

Im Beispiel 3.4.16 haben wir vernachlässigt, dass die Rangverteilungen eigentlich paarweise disjunkt sein sollen. Natürlich können wir einfach Indexmengen κ_t mit $\#\kappa_t = m$ statt $\mathcal{K}$ wählen und ansonsten genauso wie oben vorgehen.
Bevor wir den Speicheraufwand abschätzen, machen wir noch eine Anmerkung zur Implementierung.

Bemerkung 3.4.17
Die Matrizen V_t werden in den Algorithmen nur für die Blätter gespeichert. Auch wird die Matrix $E_{r_\mathcal{I}}$ nicht gespeichert. Wir haben diese Matrizen in die Definition mit aufgenommen, um die Theorie einfacher formulieren zu können.
Nach Bemerkung 2.2.10 werden die Matrizen $V_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t}$ als Matrix in $\mathbb{R}^{\mathbf{t}\times k_t}$ gespeichert. Auch hier hat die Definition als Matrix in $\mathbb{R}^{\mathcal{I}\times\kappa_t}$ rein formale Gründe.

Das nächste Lemma zeigt, dass mit Hilfe der in (3.25) beschriebenen Hierarchie der Speicheraufwand für die Clusterbasis unter Berücksichtigung von Bemerkung 3.4.17 auf lineare Komplexität reduziert wird.

Lemma 3.4.18 (Speicheraufwand von Clusterbasen)
Es sei ein Clusterbaum $\mathcal{T}_\mathcal{I}$, eine zugehörige Rangverteilung $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ und eine Clusterbasis $((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ gegeben. Weiter sei $k_{\max} := \max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t$. Dann gilt unter Berücksichtigung von Bemerkung 3.4.17

$$\mathcal{N}_{st}\left(((V_t)_{t\in\mathcal{L}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})\right) \;\leq\; k_{\max}\#\mathcal{I} + k_{\max}^2\#\mathcal{T}_\mathcal{I}.$$

Beweis:[vergleiche [12, Lemma 3.38]]
Es gilt für alle $t \in \mathcal{T}_\mathcal{I}$ und $\acute{t} := \operatorname{par}(t)$

$$\mathcal{N}_{st}(V_t) \;=\; \#\mathbf{t}\kappa_t \;\leq\; k_{\max}\#\mathbf{t} \quad \text{und} \quad \mathcal{N}_{st}(E_t) \;=\; \kappa_t\kappa_{\acute{t}} \;\leq\; k_{\max}^2.$$

Aus (3.8) folgt für den Speicheraufwand

$$\begin{aligned}\mathcal{N}_{st}\left((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}}\right) &= \sum_{t\in\mathcal{L}_\mathcal{I}} \mathcal{N}_{st}(V_t) + \sum_{t\in\mathcal{T}_\mathcal{I}} \mathcal{N}_{st}(E_t) \;=\; \sum_{t\in\mathcal{L}_\mathcal{I}} k_{\max}\#\mathbf{t} + \sum_{t\in\mathcal{T}_\mathcal{I}} k_{\max}^2 \\ &= k_{\max}\#\mathcal{I} + k_{\max}^2\#\mathcal{T}_\mathcal{I}.\end{aligned}$$

■

Bemerkung 3.4.19
Falls die Annahmen aus Bemerkung 3.4.7 gelten, haben die Clusterbasen einen Speicheraufwand in $\mathcal{O}(kn)$. Der Speicheraufwand ist also linear im lokalen Rang k und in der Problemdimension n.

Die Clusterbasen ermöglichen, den log-linearen Aufwand, der bei den uniformen $\mathcal{H}$-Matrizen im Clusterbaum entsteht, auf lineare Komplexität zu reduzieren. Aus der Kombination der uniformen $\mathcal{H}$-Matrizen mit Clusterbasen entsteht die Struktur der $\mathcal{H}^2$-Matrizen.

Definition 3.4.20 ($\mathcal{H}^2$-Matrizen)
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und $V := (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $W := (W_S)_{s\in\mathcal{T}_\mathcal{J}}$ Clusterbasen mit Rangverteilungen $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ bzw. $(\lambda_s)_{s\in\mathcal{T}_\mathcal{J}}$. Dann ist

$$\mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W\right) \;:=\; \left\{X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \mid \forall b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \exists S_b \in \mathbb{R}^{\kappa_t\times\lambda_s} : X_{|\mathbf{b}} = V_t S_b W_s^T\right\}$$

die Menge der $\mathcal{H}^2$-*Matrizen* zum Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, der Zeilenclusterbasis V und der Spaltenclusterbasis W.

Es sei darauf hingewiesen, dass in der Definition der $\mathcal{H}^2$-Matrizen wie bei den uniformen $\mathcal{H}$-Matrizen bereits feste Clusterbasen vorausgesetzt werden. Dies hat den Vorteil, dass die Menge der $\mathcal{H}^2$-Matrizen aus Definition 3.4.20 einen Teilraum von $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ bildet.

Lemma 3.4.21 (Raum der $\mathcal{H}^2$-Matrizen)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und V und W Clusterbasen. Die Menge

$$\mathcal{H}^2 \;:=\; \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}}), ((W_s)_{s\in\mathcal{T}_\mathcal{J}}, (F_s)_{s\in\mathcal{T}_\mathcal{J}})\right)$$

bildet einen Teilraum von $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Insbesondere bildet $(\mathcal{H}^2, \|.\|_F)$ einen Hilbertraum.

Beweis: Nach Definition 3.4.9 der uniformen $\mathcal{H}$-Matrizen und der Definition 3.4.20 der $\mathcal{H}^2$-Matrizen gilt

$$\begin{aligned}&\mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}}), ((W_s)_{s\in\mathcal{T}_\mathcal{J}}, (F_s)_{s\in\mathcal{T}_\mathcal{J}})\right)\\ &\quad= \mathcal{H}_{\text{uni}}\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V_t)_{t\in\mathcal{T}_\mathcal{I}}, (W_s)_{s\in\mathcal{T}_\mathcal{J}}\right).\end{aligned}$$

Damit folgt die Behauptung aus Lemma 3.4.10. ∎

Die $\mathcal{H}^2$-Matrizen sind also eine Spezialisierung der uniformen $\mathcal{H}$-Matrizen, die wiederum eine Teilmenge der entsprechenden $\mathcal{H}$-Matrizen bilden. Wie bei den uniformen $\mathcal{H}$-Matrizen liefert bereits die Definition der $\mathcal{H}^2$-Matrizen eine Darstellung für die einzelnen Elemente.

Definition 3.4.22 (Darstellung von $\mathcal{H}^2$-Matrizen)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und V und W Clusterbasen zu $\mathcal{T}_\mathcal{I}$ bzw. $\mathcal{T}_\mathcal{J}$. Weiter sei $X \in \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W\right)$. Dann nennen wir

$$\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}, (S_{|b})_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}\right)$$

eine $\mathcal{H}^2$-*Matrix-Darstellung* von X und die zugehörige $\mathcal{H}^2$-Matrix bezeichnen wir mit

$$X \;=\; \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}, (S_{|b})_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}\right),$$

wobei die sogenannten *Kopplungsmatrizen* S_b die Eigenschaft aus der Definition 3.4.20 erfüllen. Die Drei-Term-Darstellung $V_t S_b W_s^T$ für einen zulässigen Block $b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ bezeichnen wir auch als *uniforme Matrix*.

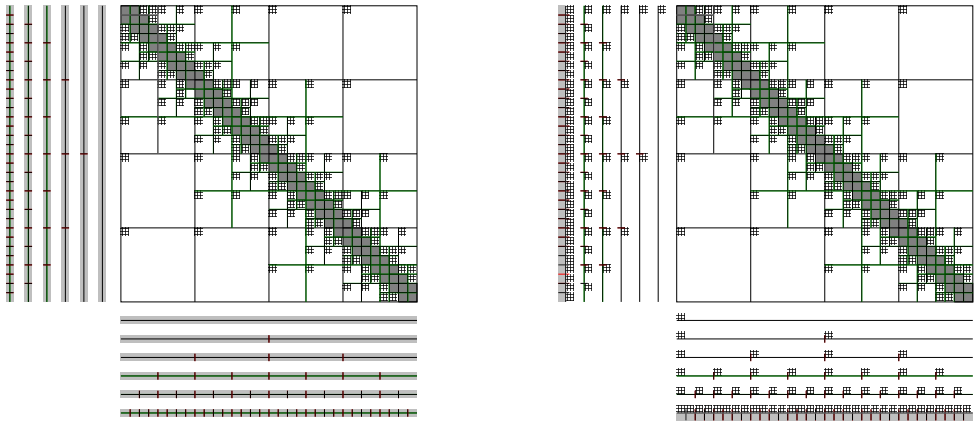


Abbildung 3.5: Vergleich uniforme $\mathcal{H}$-Matrix zu $\mathcal{H}^2$-Matrix: Der Speicheraufwand in den Clustern wird durch die geschachtelte Darstellung mit Hilfe der Transfermatrizen der Clusterbasen reduziert.

Weil wir im Folgenden häufig $\mathcal{H}^2$-Matrizen betrachten, führen wir an dieser Stelle noch einige Bezeichnungen ein.

Bezeichnungen 3.4.23
Die sogenannten Nahfeld-Matrizen in den unzulässigen Blättern werden mit

$$N := (N_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} := (X_{|\mathbf{b}})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}$$

und die Kopplungsmatrizen werden mit $S := (S_b)_{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ bezeichnet.
In dieser Notation hat eine $\mathcal{H}^2$-Matrix X die Darstellung

$$X \;=\; \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S).$$

An dieser Stelle sei nochmals darauf hingewiesen, dass die Notation für den $\mathcal{H}^2$-Matrix-Raum $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ impliziert, dass $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und V und W Clusterbasen sind. Falls wir die Transfermatrizen oder Rangverteilungen benötigen, werden wir sie extra benennen.

Für das Modellproblem können wir eine $\mathcal{H}^2$-Matrix folgendermaßen definieren.

Beispiel 3.4.24
Es sei $m \in \mathbb{N}$. Weiter seien $\mathcal{T}_{\mathcal{I}}$ der Clusterbaum aus Beispiel 3.2.4, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum zur Zulässigkeitsbedingung aus Beispiel 3.3.8, $V = ((V_{t,m})_{t\in\mathcal{T}_{\mathcal{I}}}, (E_{t,m})_{t\in\mathcal{T}_{\mathcal{I}}}) = W$ die Clusterbasis aus Beispiel 3.4.16 und $S = (S_{b,m})_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}$ die Kopplungsmatrizen wie in Beispiel 3.4.12. Dann ist $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix-Approximation des Modellproblems, wobei der Fehler durch Lemma 2.5.4 beschränkt ist.

Als Nächstes schätzen wir den Speicheraufwand ab. Dabei nutzen wir aus, dass wir den Speicheraufwand für die Elemente von uniformen $\mathcal{H}$-Matrix-Räumen und für Clusterbasen bereits abgeschätzt haben.

Lemma 3.4.25 (Speicheraufwand für $\mathcal{H}^2$-Matrizen)
Es sei $X = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ mit strikt zulässigem, c_{sp}-schwachbesetztem Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und Clusterbasen V und W mit Rangverteilungen $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ bzw. $(\lambda_s)_{s\in\mathcal{T}_\mathcal{J}}$. Weiter seien

$$n_{\max} := \max_{r\in\mathcal{L}_\mathcal{I}\cup\mathcal{L}_\mathcal{J}} \#\mathbf{r} \quad \text{und} \quad k_{\max} := \max\left\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\right\}.$$

Dann gilt

$$\begin{aligned}\mathcal{N}_{st}\left(\mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S\right)\right) \;\leq\; & \frac{c_{sp}+2}{2}\max\left\{n_{\max}^2, k_{\max}^2\right\}\left(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}\right) \\ & + \; k_{\max}(\#\mathcal{I} + \#\mathcal{J}).\end{aligned}$$

Beweis:[vergleiche [12, Lemma 3.38]]
Weil $X = \mathcal{H}_{\text{uni}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ gilt, folgt aus Lemma 3.4.13

$$\mathcal{N}_{st}(N, S) \;\leq\; c_{sp}\max\left\{n_{\max}^2, k_{\max}^2\right\}\min\left\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\right\}.$$

Für den Speicheraufwand der Clusterbasen gilt nach Lemma 3.4.18

$$\mathcal{N}_{st}(V, W) \;=\; \mathcal{N}_{st}(V) + \mathcal{N}_{st}(W) \;\leq\; k_{\max}(\#\mathcal{I} + \#\mathcal{J}) + k_{\max}^2(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}).$$

Mit $\min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\} \leq (\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J})/2$ folgt die Aussage. ∎

Falls die Annahme aus Bemerkung 3.4.7 gilt, ist der Speicheraufwand in $\mathcal{O}(nk)$, also linear in den lokalen Rängen und in der Problemdimension.
Wie wir später erläutern werden, lässt sich diese Komplexität auf die Matrix-Vektor-Multiplikation übertragen (siehe Lemma 6.1.6). Falls die Clusterbasen der Ergebnismatrix bereits bekannt sind, lässt sich das optimale Verhalten in der Problemdimension sogar auf komplexere arithmetische Operationen wie die Matrix-Matrix-Multiplikation übertragen (siehe [8]). Der Übergang vom $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ zu festen $\mathcal{H}^2$-Matrix-Teilräumen reduziert den Aufwand also erheblich.
Das Problem besteht bei der Arithmetik darin, dass die Clusterbasen und somit die richtigen $\mathcal{H}^2$-Matrix-Teilräume i. A. nicht bekannt sind. Wir wollen also eine Methode finden, die es uns ermöglicht, während der Arithmetik adaptiv eine gute Clusterbasis zu berechnen. Dazu müssen wir die Struktur der $\mathcal{H}^2$-Matrix näher untersuchen. Insbesondere soll beschrieben werden, welche Teile einer Matrix durch die Clusterbasen V und W darstellbar sein müssen, um sie als $\mathcal{H}^2$-Matrix zu V und W zu repräsentieren.
Bei dieser Untersuchung wollen wir uns vor allem auf die Zeilenclusterbasis konzentrieren. Um die Ergebnisse und Algorithmen auch auf die Spaltenclusterbasis zu übertragen, nutzen wir den transponierten Blockbaum (siehe Definition 3.3.11). Mit dessen Hilfe können wir die Adjungierte einer $\mathcal{H}^2$-Matrix wieder als $\mathcal{H}^2$-Matrix darstellen.

Lemma 3.4.26
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix. Dann gilt

$$H^T \;=\; \mathcal{H}^2(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, W, V, S^T, N^T),$$

wobei $S^T := (S^T_{(t,s)})_{(s,t)\in(\mathcal{L}^T_{\mathcal{I}\times\mathcal{J}})^+}$ und $N^T := (N^T_{(t,s)})_{(s,t)\in(\mathcal{L}^T_{\mathcal{I}\times\mathcal{J}})^-}$ sei.

Beweis: Es sei $(s,t) \in \mathcal{L}^T_{\mathcal{I}\times\mathcal{J}}$. Dann ist nach Lemma 3.3.12 $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Falls (s,t) in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ unzulässig ist, gilt dies auch für (t,s) in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und es folgt

$$(H^T)_{|\mathbf{s}\times\mathbf{t}} \;=\; (H_{|\mathbf{t}\times\mathbf{s}})^T \;=\; (N_{(t,s)})^T \;=\; (N^T)_{(s,t)}.$$

Falls andererseits (s,t) in $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ zulässig ist, gilt dies auch für (t,s) in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und es folgt

$$(H^T)_{|\mathbf{s}\times\mathbf{t}} \;=\; (H_{|\mathbf{t}\times\mathbf{s}})^T \;=\; (V_t S_{(t,s)} W_s^T)^T \;=\; W_s (S_{(s,t)})^T V_t^T \;=\; W_s (S^T)_{(s,t)} V_t^T.$$

Damit folgt die Behauptung. ∎

Also können wir Algorithmen, die für die Zeilenbasis konstruiert sind, einfach mit der Spaltenbasis und der adjungierten Matrix durchführen. Dies erlaubt es uns, im nächsten Kapitel vor allem die Zeilenbasis zu betrachten.

4 Rekompression von $\mathcal{H}^2$-Matrizen

In diesem Kapitel untersuchen wir, wie eine $\mathcal{H}^2$-Matrix rekomprimiert werden kann. Wir suchen also für eine gegebene $\mathcal{H}^2$-Matrix eine $\mathcal{H}^2$-Matrix-Darstellung mit niedrigeren lokalen Rängen und kontrollierbarem Approximationsfehler. Techniken für die Approximation von Matrizen durch $\mathcal{H}^2$-Matrizen sind in diversen Arbeiten entstanden, siehe [7, 9, 10, 16]. Wie orientieren uns in diesem Kapitel bei den Algorithmen für die Rekompression von $\mathcal{H}^2$-Matrizen an der in [12, Kapitel 6] vorgestellten Version.
In Abschnitt 4.1 untersuchen wir, welche Matrizen durch gegebene Clusterbasen darstellbar sind. Insbesondere beschreiben wir durch die sogenannten totalen Clusterbasen, welche Teile durch die jeweiligen Clusterbasen repräsentiert werden müssen. Danach beschreiben wir in Abschnitt 4.2 die orthogonalen Clusterbasen. Diese ermöglichen uns, eine orthogonale Projektion (bez. der Frobeniusnorm) auf den Raum der zugehörigen $\mathcal{H}^2$-Matrizen zu definieren, die wir effizient berechnen können. Nachdem wir die Gestalt der $\mathcal{H}^2$-Matrix-Räume näher untersucht und eine Möglichkeit zur Berechnung einer konkreten Darstellung gefunden haben, können wir in Abschnitt 4.3 den entstehenden Projektionsfehler abschätzen. Die Form der Fehlerabschätzung liefert die Idee für die Konstruktion von adaptiven Clusterbasen in Abschnitt 4.4. Zum Abschluss des Kapitels stellen wir in Abschnitt 4.5 den Rekompressions-Algorithmus vor, den wir in Kapitel 5 adaptieren, um ein effizientes Niedrigrangupdate zu definieren.

4.1 Struktur der $\mathcal{H}^2$-Matrizen

Wir haben bereits festgestellt, dass $\mathcal{H}^2$-Matrizen linearen Speicheraufwand haben und die $\mathcal{H}^2$-Matrizen zu festen Clusterbasen einen Teilraum des $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ bilden. In diesem Abschnitt untersuchen wir den $\mathcal{H}^2$-Matrix-Raum zu gegebenen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und Clusterbasen V, W näher. Insbesondere untersuchen wir, welche Matrizen des $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ Elemente des Teilraums $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ sind.
Der wesentliche Unterschied der $\mathcal{H}^2$-Matrizen zu den uniformen $\mathcal{H}$-Matrizen ist, dass eine zusätzliche Hierarchie im Clusterbaum in Form der Clusterbasen genutzt wird. Diese sorgt dafür, das nicht nur die zulässigen Blöcke mit Zeilencluster t von V_t dargestellt werden müssen, sondern auch die zu t gehörigen Anteile der zulässigen Blätter, die einen Vorfahren von t als Zeilencluster haben. Um dies formal zu beschreiben, führen wir einige neue Begriffe ein.

Definition 4.1.1 (Blockzeile/-spalte)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Dann sei für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$

$$\begin{aligned}\operatorname{row}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t) &= \left\{\tilde{s} \in \mathcal{T}_{\mathcal{J}} \mid (t, \tilde{s}) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\right\} \quad \text{bzw.}\\ \operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s) &= \left\{\tilde{t} \in \mathcal{T}_{\mathcal{I}} \mid (\tilde{t}, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\right\}.\end{aligned}$$

Wir bezeichnen $\operatorname{row}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t)$ als die *Blockzeile* von t und $\operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$ als die *Blockspalte* von s. Falls der Blockbaum im Kontext eindeutig ist, verwenden wir auch $\operatorname{row}(t) := \operatorname{row}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t)$ und $\operatorname{col}(s) := \operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$. Für spätere Zwecke definieren wir für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ zusätzlich die Indexmengen

$$\xi_t := \xi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},t} := \dot{\bigcup_{s\in\operatorname{row}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},t)}} \mathbf{s} \quad \text{und} \quad \xi_s := \xi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},s} := \dot{\bigcup_{t\in\operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},s)}} \mathbf{t}.$$

Die Blockzeile $\operatorname{row}(t)$ beschreibt alle Cluster s, die zusammen mit t ein zulässiges Blatt bilden (siehe Abbildung 4.1). Die zugehörigen Matrixblöcke $\mathbf{t} \times \mathbf{s}$ werden bei $\mathcal{H}^2$-Matrizen direkt mit Hilfe der Clusterbasis V_t dargestellt. Um später Aussagen von der Blockzeile auf die Blockspalte übertragen zu können, betrachten wir den transponierten Blockbaum.

Bemerkung 4.1.2
Es sei $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ der transponierte Blockbaum zum zulässigen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ (siehe Definition 3.3.11). Dann ist $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ nach Lemma 3.3.12 ein zulässiger Blockbaum und es gilt für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$

$$\operatorname{row}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t) := \operatorname{col}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, t) \quad \text{und} \quad \operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s) := \operatorname{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, s).$$

In der Literatur wird häufig die Blockzeile anhand aller Blöcke ($\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$) und nicht nur der zulässigen Blätter ($\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$) definiert (siehe [12, Definition 3.29]). Wir benötigen allerdings meistens nur die zulässigen Blätter, weshalb wir die obige Definition nutzen.
Durch die geschachtelte Darstellung der Clusterbasen müssen auch die Vorfahren eines Clusters durch die Clusterbasis beschrieben werden. Deshalb erweitern wir den Begriff der Blockzeile.

Definition 4.1.3 (erweiterte Blockzeile/-spalte)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Dann definieren wir die *erweiterte Blockzeile/-spalte* für alle $t \in \mathcal{T}_{\mathcal{I}}$ und alle $s \in \mathcal{T}_{\mathcal{J}}$ durch

$$\begin{aligned}\overline{\operatorname{row}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t) &:= \dot{\bigcup_{\hat{t}\in\operatorname{pred}(t)}} \operatorname{row}(\hat{t}) = \left\{\tilde{s} \in \mathcal{T}_{\mathcal{J}} \mid \exists \hat{t} \in \operatorname{pred}(t) : (\hat{t}, \tilde{s}) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\right\} \quad \text{bzw.}\\ \overline{\operatorname{col}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s) &:= \dot{\bigcup_{\hat{s}\in\operatorname{pred}(s)}} \operatorname{col}(\hat{s}) = \left\{\tilde{t} \in \mathcal{T}_{\mathcal{I}} \mid \exists \hat{s} \in \operatorname{pred}(s) : (\tilde{t}, \hat{s}) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\right\}.\end{aligned}$$

Falls der Blockbaum im Kontext eindeutig ist, nutzen wir wie für die Blockzeile/-spalte die Notationen $\overline{\operatorname{row}}(t) := \overline{\operatorname{row}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t)$ und $\overline{\operatorname{col}}(s) = \overline{\operatorname{col}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$. Für spätere Zwecke definieren wir für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ zusätzlich die Indexmengen

$$\overline{\xi}_t := \overline{\xi}_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},t} := \dot{\bigcup_{s\in\overline{\operatorname{row}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},t)}} \mathbf{s} \quad \text{und} \quad \overline{\xi}_s := \overline{\xi}_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},s} := \dot{\bigcup_{t\in\overline{\operatorname{col}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},s)}} \mathbf{t}.$$

Bemerkung 4.1.4
Nach Lemma 3.3.14 (v) und (iv) folgt, dass die Beschriftungen der Cluster in der Blockzeile $\overline{\text{row}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t)$ für $t \in \mathcal{T}_{\mathcal{I}}$ bzw. in der Blockspalte $\overline{\text{col}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$ für $s \in \mathcal{T}_{\mathcal{J}}$ paarweise disjunkt sind.

Bemerkung 4.1.5
Analog zur Blockzeile/-spalte (siehe Bemerkung 4.1.2) gilt für die erweiterte Blockzeile/-spalte und den transponierten Blockbaum $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ aus Definition 3.3.11

$$\overline{\text{row}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, t) \;=\; \overline{\text{col}}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, t) \quad \text{und} \quad \overline{\text{col}}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s) \;=\; \overline{\text{row}}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, s)$$

für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$.

Die Menge $\overline{\text{row}}(t)$ beschreibt die Spaltencluster, die zusammen mit einem Vorfahren von $t \in \mathcal{T}_{\mathcal{I}}$ einen zulässigen Block bilden, und die Menge $\overline{\text{col}}(s)$ die entsprechenden Zeilencluster für $s \in \mathcal{T}_{\mathcal{J}}$. Hierbei ist zu beachten das $t \in \text{pred}(t)$ gilt, d. h. jeder Cluster nach Definition zu seinen Vorfahren gehört.
Mit dieser Hilfsgröße können wir die sogenannten totalen Clusterbasen definieren. Diese beschreiben für einen Cluster, welche Teile einer Matrix X durch die Clusterbasis dargestellt werden muss. Dabei konzentrieren wir uns ab jetzt auf die Zeilencluster. Die Diskussion für die Spaltencluster verläuft analog (wir können einfach zu X^T und $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$ übergehen).

Definition 4.1.6 (totale Clusterbasis)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum und $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Es sei für alle $t \in \mathcal{T}_{\mathcal{I}}$

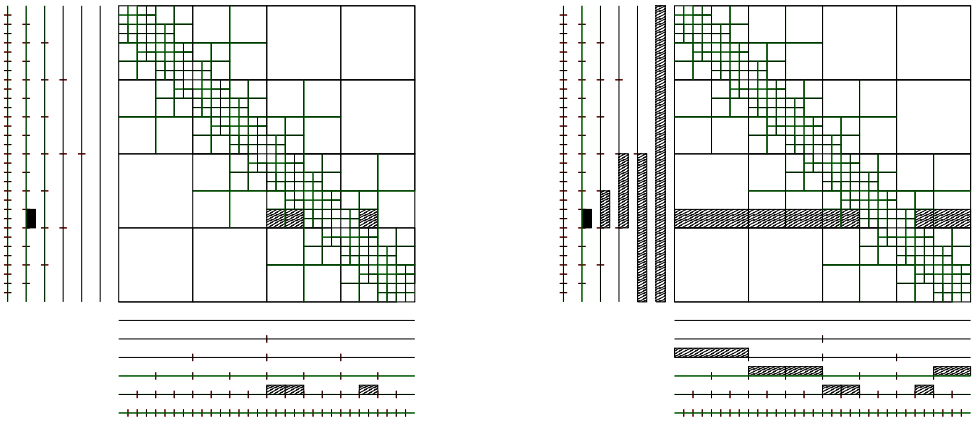
$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t \;:=\; \Pi_{\mathbf{t}} X \sum_{s\in\overline{\text{row}}(t)} \Pi_{\mathbf{s}} \;\in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\bar{\xi}_t}$$

und für $t \neq r_{\mathcal{I}}$ mit $\acute{t} := \text{par}(t)$

$$E(\mathcal{T}_{\mathcal{I}\times\mathcal{J}})_t \;:=\; \sum_{s\in\overline{\text{row}}(\acute{t})} \Pi_{\mathbf{s}} \;\in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\bar{\xi}_t\times\bar{\xi}_{\acute{t}}}.$$

Dann definiert $(X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X), E(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}))$ die *totale (Zeilen-)Clusterbasis* zu X. Falls der Blockbaum und die Matrix im Kontext eindeutig sind, verwenden wir auch die Schreibweisen $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t$ und $E_t := E(\mathcal{T}_{\mathcal{I}\times\mathcal{J}})_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}$.

Die Einschränkung $\Pi_{\mathbf{t}}$ erklärt sich durch die Eigenschaft $\Pi_{\mathbf{t}} V_t = V_t$ aus der Definition 3.4.14 der Clusterbasis. Die Einschränkungen $\Pi_{\mathbf{s}}$ auf der rechten Seite von X fassen alle Cluster s zusammen, die mit t oder mit einem Vorfahren von t einen zulässigen Block bilden. In Abbildung 4.1 ist ein Beispiel für die totale Clusterbasis eines Clusters visualisiert. In Lemma 4.1.12 zeigen wir, dass die totale Clusterbasis geeignet ist, $\mathcal{H}^2$-Matrizen zu charakterisieren.
Zuvor betrachten wir die totalen Clusterbasen etwas genauer. Als Erstes stellen wir fest, dass die totale Clusterbasis eine Clusterbasis ist.



(a) Blockzeile (b) totale Clusterbasis

Abbildung 4.1: Die Bilder visualisieren die Blockzeile bzw. totale Clusterbasis des schwarz markierten Zeilenclusters t. Zusätzlich sind für die Blockzeile die Matrixblöcke und für die totale Clusterbasis die beteiligten Cluster markiert.

Lemma 4.1.7
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum und $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine Matrix. Dann ist die totale Clusterbasis $(X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X), E(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}))$ aus Definition 4.1.6 eine Clusterbasis.

Beweis:[siehe [12, Lemma 6.12]] ∎

Bemerkung 4.1.8
Formal müssten wir für alle $t \in \mathcal{T}_{\mathcal{I}}$ eine Indexmenge κ_t mit $\#\kappa_t = \#\left(\dot{\bigcup}_{s\in\overline{\mathrm{row}}(t)} \mathbf{s}\right)$ und mit den zugehörigen Bijektionen arbeiten. Wir verzichten auf die Einführung einer solchen Rangverteilung, um die Betrachtung übersichtlicher zu halten.

Die totale Clusterbasis beschreibt, welcher Teil einer Matrix durch eine Clusterbasis im jeweiligen Cluster (approximativ) dargestellt werden muss.
Bevor die totale Clusterbasis einer $\mathcal{H}^2$-Matrix elegant dargestellt werden kann, werden die sogenannten allgemeinen Transfermatrizen definiert.

Definition 4.1.9 (allgemeine Transfermatrizen)
Es sei $((V_t)_{t\in\mathcal{T}_{\mathcal{I}}}, (E_t)_{t\in\mathcal{T}_{\mathcal{I}}})$ eine Clusterbasis. Wir definieren für alle $t \in \mathcal{T}_{\mathcal{I}}$ und alle $\check{t} \in \mathrm{desc}(t)$

$$E_{\check{t},t} = \begin{cases} I & , \text{falls } t = \check{t} \\ E_{\check{t}} E_{r,t} & , \text{falls } r := \mathrm{par}(\check{t}) \in \mathrm{desc}(t) \end{cases}$$

die *allgemeinen Transfermatrizen.*

Das nächste Lemma liefert eine Darstellung der Einschränkung von Clusterbasen auf ihre Nachfahren mit Hilfe der allgemeinen Transfermatrizen. Diese Darstellung verallgemeinert die Eigenschaft aus (3.26).

Lemma 4.1.10
Es sei $((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine Clusterbasis. Dann gilt für alle $t \in \mathcal{T}_\mathcal{I}$ und alle $\check{t} \in \operatorname{desc}(t)$

$$\Pi_{\check{t}} V_t \;=\; V_{\check{t}} E_{\check{t},t}.$$

Beweis:[siehe [12, Lemma 6.13]] ∎

Als Nächstes können wir die totalen Clusterbasen für $\mathcal{H}^2$-Matrizen elegant mit Hilfe der allgemeinen Transfermatrizen darstellen.

Lemma 4.1.11
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix. Dann gilt für die totale Clusterbasis

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t \;=\; V_t \left(\sum_{\hat{t}\in\operatorname{pred}(t)} E_{t,\hat{t}} \sum_{s\in\operatorname{row}(\hat{t})} S_{(\hat{t},s)} W_s^T \right) \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}.$$

Beweis: Es sei $t \in \mathcal{T}_\mathcal{I}$. Wir schreiben $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t$. Wegen $\mathbf{t} \subseteq \hat{\mathbf{t}}$ für alle $\hat{t} \in \operatorname{pred}(t)$ gilt

$$X_t \;=\; \sum_{s\in\overline{\operatorname{row}}(t)} \Pi_\mathbf{t} X \Pi_\mathbf{s} \;=\; \sum_{\hat{t}\in\operatorname{pred}(t)} \sum_{s\in\operatorname{row}(\hat{t})} \Pi_\mathbf{t} X \Pi_\mathbf{s} \;=\; \sum_{\hat{t}\in\operatorname{pred}(t)} \sum_{s\in\operatorname{row}(\hat{t})} \Pi_\mathbf{t} \Pi_{\hat{\mathbf{t}}} X \Pi_\mathbf{s}.$$

Weil $(\hat{t}, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ für alle $\hat{t} \in \operatorname{pred}(t)$ und $s \in \operatorname{row}(\hat{t})$ gilt, folgt

$$X_t \;=\; \sum_{\hat{t}\in\operatorname{pred}(t)} \sum_{s\in\operatorname{row}(\hat{t})} \Pi_\mathbf{t} V_{\hat{t}} S_{(\hat{t},s)} W_s^T \;=\; \sum_{\hat{t}\in\operatorname{pred}(t)} \Pi_\mathbf{t} V_{\hat{t}} \sum_{s\in\operatorname{row}(\hat{t})} S_{(\hat{t},s)} W_s^T.$$

Mit Lemma 4.1.10 stellen wir $V_{\hat{t}}$ über die erweiterten Transfermatrizen dar und erhalten

$$X_t \;=\; \sum_{\hat{t}\in\operatorname{pred}(t)} V_t E_{t,\hat{t}} \sum_{s\in\operatorname{row}(\hat{t})} S_{(\hat{t},s)} W_s^T \;=\; V_t \left(\sum_{\hat{t}\in\operatorname{pred}(t)} E_{t,\hat{t}} \sum_{s\in\operatorname{row}(\hat{t})} S_{(\hat{t},s)} W_s^T \right).$$

∎

Somit haben wir eine Darstellung für die totale Clusterbasis als Produkt, wobei V_t den vorderen Faktor bildet. Dies ermöglicht die Charakterisierung der Elemente eines gegebenen $\mathcal{H}^2$-Matrix-Raums mit Hilfe der totalen Clusterbasis.

Lemma 4.1.12
Es sei $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ ein $\mathcal{H}^2$-Matrix-Raum und $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Die totalen Clusterbasen bezeichnen wir mit $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t$, $t \in \mathcal{T}_{\mathcal{I}}$, und $X_s^T := X(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, X^T)_s$, $s \in \mathcal{T}_{\mathcal{J}}$, (siehe Definition 4.1.6).
Es gilt $X \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ genau dann, wenn $\operatorname{Bild}(X_t) \subseteq \operatorname{Bild}(V_t)$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $\operatorname{Bild}(X_s^T) \subseteq \operatorname{Bild}(W_s)$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ gilt.

Beweis:[vergleiche [12, Lemma 6.14 und 6.15]]
"⇒" Es sei $X \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$. Dann folgt mit Lemma 4.1.11

$$\operatorname{Bild}(X_t) \;=\; \operatorname{Bild}\left(V_t \left(\sum_{\hat{t}\in\operatorname{pred}(t)} E_{t,\hat{t}} \sum_{s\in\operatorname{row}(\hat{t})} S_{(\hat{t},s)} W_s^T\right)\right) \;\subseteq\; \operatorname{Bild}(V_t)$$

für alle $t \in \mathcal{T}_{\mathcal{I}}$. Nach Lemma 3.4.26 ist $X^T \in \mathcal{H}^2(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, W, V)$ und wie oben folgt mit Lemma 4.1.11 $\operatorname{Bild}(X_s^T) \subseteq \operatorname{Bild}(W_s)$ für alle $s \in \mathcal{T}_{\mathcal{J}}$.
"⇐" Es sei $\operatorname{Bild}(X_t) \subseteq \operatorname{Bild}(V_t)$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $\operatorname{Bild}(X_s^T) \subseteq \operatorname{Bild}(W_s)$ für alle $s \in \mathcal{T}_{\mathcal{J}}$. Es sei $(t, s) = b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Wir zeigen, dass eine Kopplungsmatrix S_b mit $V_t S_b W_s^T = \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}}$ existiert. Wegen $s \in \operatorname{row}(t)$ folgt aus Definition 4.1.6

$$\operatorname{Bild}(\Pi_{\mathbf{t}} X \Pi_{\mathbf{s}}) \;\subseteq\; \operatorname{Bild}(X_t) \;\subseteq\; \operatorname{Bild}(V_t)$$

und es existiert $B_b \in \mathbb{R}^{\mathcal{J}\times\kappa_t}$ mit $V_t B_b^T = \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}}$. Weiter ist $t \in \operatorname{col}(s) = \operatorname{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, s)$ und mit Definition 4.1.6 folgt

$$\operatorname{Bild}(\Pi_{\mathbf{s}} X^T \Pi_{\mathbf{t}}) \subseteq \operatorname{Bild}(X_s^T) \subseteq \operatorname{Bild}(W_s).$$

Also existiert $A_b \in \mathbb{R}^{\lambda_s\times\mathcal{I}}$ mit $W_s A_b^T = \Pi_s X^T \Pi_t$.
Um die beiden Darstellungen zusammenzubringen nutzen wir die Moore-Penrose-Inverse V_t^+ von V_t (siehe [22, Abschnitt 5.5.4]). Für diese gilt $V_t V_t^+ V_t = V_t$. Wir definieren $S_b := V_t^+ A_b$ und erhalten

$$\Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} \;=\; V_t B_b^T \;=\; V_t V_t^+ V_t B_b^T \;=\; V_t V_t^+ \Pi_t X \Pi_s \;=\; V_t V_t^+ A_b W_s^T \;=\; V_t S_b W_s^T.$$

Also ist $X \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$. ∎

Nach Lemma 4.1.12 beschreiben die totalen Clusterbasen also die Teile einer Matrix, die durch die Clusterbasen dargestellt werden müssen. Für allgemeine Clusterbasen kann es schwierig sein, die $\mathcal{H}^2$-Matrix-Darstellung anzugeben. Zusätzlich haben wir in den meisten Fällen Matrizen, die sich nicht exakt, sondern nur approximativ in einem gegebenen $\mathcal{H}^2$-Matrix-Raum darstellen lassen. Deshalb betrachten wir die spezielle Klasse der orthogonalen Clusterbasen.

4.2 Orthogonale Clusterbasen

Nach Lemma 3.4.21 bilden die $\mathcal{H}^2$-Matrizen zu gegebenen Blockbaum und Clusterbasen, zusammen mit der Frobeniusnorm, einen Hilbertraum. Wie in Lemma 2.1.9 beschrieben, geben uns orthogonale Projektionen die Möglichkeit, die Bestapproximation zu berechnen. Wir haben in Beispiel 2.3.18 gesehen, dass für Vektorräume orthogonale Projektionen einfach über orthogonale Matrizen definiert werden können. Auf ähnliche Weise können wir orthogonale Projektionen auf einen $\mathcal{H}^2$-Matrix-Raum definieren. Dazu benötigen wir die sogenannten orthogonalen Clusterbasen.

Definition 4.2.1 (orthogonale Clusterbasis)
Eine Clusterbasis $((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ heißt orthogonal, falls alle Matrizen V_t orthogonal sind, d. h.

$$V_t^T V_t = I \quad \text{für alle } t \in \mathcal{T}_\mathcal{I}.$$

Bevor wir in Definition 4.2.13 die orthogonale Projektion angeben, wollen wir die orthogonalen Clusterbasen näher untersuchen. Als Erstes geben wir ein Kriterium für orthogonale Clusterbasen an, das die Matrizen V_t nur in den Blättern und für alle anderen Cluster die Transfermatrizen E_t benutzt.

Lemma 4.2.2 (Kriterium für orthogonale Clusterbasen)
Eine Clusterbasis $((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ ist genau dann orthogonal, falls

$$V_t^T V_t = I \quad \text{für alle } t \in \mathcal{L}_\mathcal{I} \tag{4.1}$$

und

$$\sum_{\check{t}\in\operatorname{chil}(t)} E_{\check{t}}^T E_{\check{t}} = I \quad \text{für alle } t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I} \tag{4.2}$$

gelten.

Beweis:[siehe Lemma [12, Lemma 5.3]] ∎

Wir wollen später vor allem mit orthogonalen Clusterbasen arbeiten. Deshalb ist es wichtig, dass wir aus einer Clusterbasis V_t eine orthogonale Clusterbasis berechnen können, die $\operatorname{Bild}(V_t) \subseteq \operatorname{Bild}(\overline{V}_t)$ für alle $t \in \mathcal{T}_\mathcal{I}$ erfüllt und höchstens so viele Spalten wie V_t hat. Mit Lemma 4.1.12 folgt aus der ersten Bedingung $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W) \subseteq \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, W)$. Die Matrizen, die durch V darstellbar sind, lassen sich also auch mit $\overline{V}$ repräsentieren. Die zweite Bedingung beschränkt den Aufwand für $\overline{V}$ durch den für V. Um die orthogonale Clusterbasis zu definieren, führen wir zuvor folgende Hilfsmatrizen ein.

Definition 4.2.3
Es seien $V = ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ und $\overline{V} = ((\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}, (\overline{E}_t)_{t\in\mathcal{T}_\mathcal{I}})$ Clusterbasen. Wir definieren den *Basiswechsel* von V zu $\overline{V}$ durch

$$C(V,\overline{V})_t \;:=\; \overline{V}_t^T V_t \quad \text{für alle } t \in \mathcal{T}_\mathcal{I}.$$

Die in den Kindern bezüglich $\overline{V}$ projizierte Clusterbasis V ist für alle $t \in \mathcal{T}_\mathcal{I}$ durch

$$\widehat{V}(V,\overline{V})_t \;:=\; \begin{cases} V_t & \text{, falls } t \in \mathcal{L}_\mathcal{I} \\ \begin{pmatrix} C(V,\overline{V})_{t_1} E_{t_1} \\ \vdots \\ C(V,\overline{V})_{t_\tau} E_{t_\tau} \end{pmatrix} & \text{, falls } \operatorname{chil}(t) =: \{t_1,\dots,t_\tau\} \neq \emptyset \end{cases}$$

gegeben. Für den Spezialfall $V = \overline{V}$ definieren wir $\acute{V}(V)_t := \widehat{V}(V,V)_t$ für alle $t \in \mathcal{T}_\mathcal{I}$. Zusätzlich fassen wir die Clusterbasen der Kinder für alle $t \in \mathcal{T}_\mathcal{I}$ durch

$$\breve{V}(\overline{V})_t \;:=\; \begin{cases} I & \text{, falls } t \in \mathcal{L}_\mathcal{I} \\ \begin{pmatrix} \overline{V}_{t_1} & \cdots & \overline{V}_{t_\tau} \end{pmatrix} & \text{, falls } \operatorname{chil}(t) = \{t_1,\dots,t_\tau\} \neq \emptyset \end{cases}$$

zusammen.

Der Basiswechsel ermöglicht uns später, die Kopplungsmatrizen effizient anzupassen. Die Matrizen $\widehat{V}$ sind für die Konstruktion neuer Clusterbasen wichtig. Die Matrizen $\acute{V}$ und $\breve{V}$ vereinfachen später die Schreibweisen, insbesondere bei der Fehleranalyse.
Zuerst zeigen wir, dass $\widehat{V}(V,\overline{V})_t$ durch Projizieren von V mit $\breve{V}(\overline{V})_t$ entsteht.

Lemma 4.2.4
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix und $\overline{V}$ eine orthogonale Clusterbasis. Dann gilt mit den Bezeichnungen aus Definition 4.2.3

$$\breve{V}(\overline{V})_t^T V_t \;=\; \widehat{V}(V,\overline{V})_t \quad \text{für alle } t \in \mathcal{T}_\mathcal{I}.$$

Beweis: Für alle $t \in \mathcal{L}_\mathcal{I}$ gilt $\breve{V}(\overline{V})_t^T V_t = V_t = \widehat{V}(V,\overline{V})_t$.
Es sei $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ mit $\operatorname{chil}(t) =: \{t_1,\dots,t_\tau\}$. Für alle $i,j \in \{1,\dots,\tau\}$ und den Basiswechsel $C(V,\overline{V})$ aus Definition 4.2.3 gilt $\overline{V}_{t_i}^T V_{t_j} = \overline{V}_{t_i}^T \Pi_{\mathbf{t}_i}^T \Pi_{\mathbf{t}_j} V_{t_j} = \delta_{i,j} \overline{V}_{t_i}^T V_{t_i} = \delta_{i,j} C(V,\overline{V})_{t_i}$. Daraus folgt

$$\begin{aligned} \breve{V}(\overline{V})_t^T V_t \;&=\; \begin{pmatrix} \overline{V}_{t_1} & \cdots & \overline{V}_{t_\tau} \end{pmatrix}^T \sum_{\check{t}\in\operatorname{chil}(t)} V_{\check{t}} E_{\check{t}} \;=\; \begin{pmatrix} \overline{V}_{t_1}^T V_{t_1} E_{t_1} \\ \vdots \\ \overline{V}_{t_\tau}^T V_{t_\tau} E_{t_\tau} \end{pmatrix} \;=\; \begin{pmatrix} C(V,\overline{V})_{t_1} E_{t_1} \\ \vdots \\ C(V,\overline{V})_{t_\tau} E_{t_\tau} \end{pmatrix} \\ &=\; \widehat{V}(V,\overline{V})_t. \end{aligned}$$

■

Für den Fall, dass $\overline{V} = V$ orthogonal ist, erhalten wir eine vereinfachte Darstellung für die projizierte Clusterbasis.

Bemerkung 4.2.5
Es sei $V = (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine orthogonale Clusterbasis. Dann gilt für alle $t \in \mathcal{T}_\mathcal{I}$

$$\acute{V}(V)_t \;=\; \widehat{V}(V,V)_t \;=\; \begin{cases} V_t & \text{, falls } t \in \mathcal{L}_\mathcal{I} \\ \begin{pmatrix} E_{t_1} \\ \vdots \\ E_{t_\tau} \end{pmatrix} & \text{, falls } \operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\} \neq \emptyset \end{cases}.$$

Beweis: Weil V orthogonal ist, folgt für den Basiswechsel für alle $t \in \mathcal{T}_\mathcal{I}$

$$C(V,V)_t \;=\; V_t^T V_t \;=\; I.$$

Dies setzen wir in die Definition von $\widehat{V}(V,V)_t$ ein und erhalten die Behauptung. ∎

Mit den Matrizen aus Definition 4.2.3 können wir rekursiv eine orthogonale Clusterbasis $\overline{V}$ zur gegebenen Clusterbasis V definieren (vergleiche [12, Abschnitt 5.4]). Dabei erfüllt $\overline{V}$ die von uns verlangten Bedingungen.

Definition 4.2.6
Es sei $V = ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine Clusterbasis. Dann definieren wir rekursiv von den Blättern aus für alle $t \in \mathcal{T}_\mathcal{I}$

$$\overline{V}(V)_t \;:=\; \breve{V}(\overline{V}(V))_t \widehat{Q}_t \in \mathbb{R}^{\mathcal{I}\times\rho_t}_{\mathbf{t}\times\rho_t} \qquad \text{mit einer QR-Zerlegung } \widehat{Q}_t O_t \;=\; \widehat{V}(V,\overline{V})_t.$$

Weiter sei für alle $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ und $\breve{t} \in \operatorname{chil}(t)$

$$\overline{E}(V)_{\breve{t}} \;:=\; \widehat{Q}_{t|\rho_{\breve{t}}\times\rho_t}.$$

Bemerkung 4.2.7
Für die Blätter $t \in \mathcal{L}_\mathcal{I}$ gilt $\widehat{V}(V,\overline{V})_t = V_t$ und $\breve{V}(\overline{V}(V))_t = I$. Also ist $\overline{V}(V)_t$ wohldefiniert. Für einen Nicht-Blatt-Cluster $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ benötigen wir für das Aufstellen von $\widehat{V}(V,\overline{V})_t$ und $\breve{V}(\overline{V}(V))_t$ die Matrizen $\overline{V}(V)_{\breve{t}}$ nur für $\breve{t} \in \operatorname{chil}(t)$. Damit ist in diesem Fall $\overline{V}(V)_t$ über die Rekursion wohldefiniert.

In folgendem Lemma zeigen wir, dass die Matrizen $((\overline{V}(V)_t)_{t\in\mathcal{T}_\mathcal{I}}, (\overline{E}(V)_t)_{t\in\mathcal{T}_\mathcal{I}})$ aus Definition 4.2.6 eine neue orthogonale Clusterbasis definieren.

Lemma 4.2.8
Es sei $V = ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine Clusterbasis mit Rangverteilung $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$. Dann bilden die Matrizen $\overline{V} = ((\overline{V}(V)_t)_{t\in\mathcal{T}_\mathcal{I}}, (\overline{E}(V)_t)_{t\in\mathcal{T}_\mathcal{I}})$ aus Definition 4.2.6 eine orthogonale Clusterbasis mit Rangverteilung $(\rho_t)_{t\in\mathcal{T}_\mathcal{I}}$. Für alle $t \in \mathcal{T}_\mathcal{I}$ gilt $V_t = \overline{V}_t O_t$, für den Basiswechsel $C(V,\overline{V}) = O_t$ und für die Hilfsmatrix $\acute{V}(\overline{V})_t = \widehat{Q}_t$ mit $\widehat{Q}_t$ aus Definition 4.2.6. Außerdem gilt $\operatorname{Bild}(V_t) \subseteq \operatorname{Bild}(\overline{V}_t)$ und $\#\rho_t \leq \#\kappa_t$ für alle $t \in \mathcal{T}_\mathcal{I}$.

Beweis:[vergleiche [12, Abschnitt 5.4]]
Als Erstes zeigen wir die geschachtelte Darstellung unter Verwendung der Notationen $\overline{V}_t = \overline{V}(V)_t$ und $\overline{E}_t = \overline{E}(V)_t$ für alle $t \in \mathcal{T}_\mathcal{I}$. Für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ gilt

$$\overline{V}_t \;=\; \check{V}(\overline{V})_t \widehat{Q}_t \;=\; \begin{pmatrix} \overline{V}_{t_1} & \cdots & \overline{V}_{t_\tau} \end{pmatrix} \begin{pmatrix} \overline{E}_{t_1} \\ \vdots \\ \overline{E}_{t_\tau} \end{pmatrix} \;=\; \sum_{\check{t} \in \operatorname{chil}(t)} \overline{V}_{t_1} \overline{E}_{t_1}.$$

Also ist $((\overline{V}_t)_{t \in \mathcal{T}_\mathcal{I}}, (\overline{E}_t)_{t \in \mathcal{T}_\mathcal{I}})$ eine Clusterbasis.
Als Nächstes beweisen wir die Orthogonalität der Clusterbasis. Für alle $t \in \mathcal{L}_\mathcal{I}$ gilt, dass $\overline{V}_t = \check{V}(\overline{V})_t \widehat{Q}_t = \widehat{Q}_t$ orthogonal ist.
Für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ ist für $\operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\}$

$$\begin{pmatrix} \overline{E}_{t_1} \\ \vdots \\ \overline{E}_{t_\tau} \end{pmatrix} \;=\; \widehat{Q}_t$$

orthogonal. Nach dem Kriterium aus Lemma 4.2.2 ist $\overline{V}$ somit eine orthogonale Clusterbasis. Nach Bemerkung 4.2.5 folgt $\acute{V}(\overline{V})_t = \widehat{Q}_t$ für alle $t \in \mathcal{T}_\mathcal{I}$.
Wir zeigen $V_t = \overline{V}_t O_t$ und $C(V, \overline{V})_t = O_t$ per Induktion über $\#\operatorname{desc}(t) \in \mathbb{N}_{>0}$. Es sei $n = 1$ und $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{desc}(t) = 1$. Dann ist $t \in \mathcal{L}_\mathcal{I}$ und es gilt

$$V_t \;=\; \widehat{V}(V, \overline{V})_t \;=\; \widehat{Q}_t O_t \;= \check{V}(\overline{V})_t \widehat{Q}_t O_t \;=\; \overline{V}_t O_t.$$

Insbesondere folgt mit der Orthogonalität von $\overline{V}_t$

$$C(V, \overline{V})_t \;=\; \overline{V}_t^T V_t \;=\; \overline{V}_t^T \overline{V}_t O_t \;=\; O_t.$$

Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{desc}(t) \leq n$ gelte. Es sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{desc}(t) = n + 1$. Dann ist $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ und die Aussage gilt für alle $\check{t} \in \operatorname{chil}(t)$. Damit folgt

$$\begin{aligned} V_t \;&=\; \sum_{\check{t} \in \operatorname{chil}(t)} V_{\check{t}} E_{\check{t}} \;=\; \sum_{\check{t} \in \operatorname{chil}(t)} \overline{V}_{\check{t}} O_{\check{t}} E_{\check{t}} \;=\; \begin{pmatrix} \overline{V}_{t_1} & \cdots & \overline{V}_{t_\tau} \end{pmatrix} \begin{pmatrix} C(V, \overline{V})_{t_1} E_{t_1} \\ \vdots \\ C(V, \overline{V})_{t_\tau} E_{t_\tau} \end{pmatrix} \\ &=\; \check{V}(\overline{V})_t \widehat{V}(V, \overline{V})_t \;=\; \check{V}(\overline{V})_t \widehat{Q}_t O_t \;=\; \overline{V}_t O_t. \end{aligned}$$

Wie oben folgt $C(V, \overline{V})_t = O_t$. Aus $V_t = \overline{V}_t O_t$ folgt zusätzlich $\operatorname{Bild}(V_t) \subseteq \operatorname{Bild}(\overline{V}_t)$. Nach Definition 2.4.1 der (dünnen) QR-Zerlegung gilt für die Spaltenindexmenge von O_t für $t \in \mathcal{L}_\mathcal{I}$ die obere Schranke $\#\rho_t = \min\{\#t, \#\kappa_t\} \leq \#\kappa_t$ und für $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ die obere Schranke $\#\rho_t = \min\left\{\#\dot{\bigcup}_{\check{t} \in \operatorname{chil}(t)} \kappa_{\check{t}},\, \kappa_t\right\} \leq \#\kappa_t$. ∎

Bevor wir mit Hilfe von Lemma 4.2.8 den Algorithmus 4.2.1 zur Berechnung einer orthogonalen Clusterbasis definieren, führen wir noch die orthogonalen Gewichte ein.

Definition 4.2.9 (orthogonale Gewichte)
Es sei $(V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis, $(\rho_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Rangverteilung und $O_t \in \mathbb{R}^{\rho_t\times\kappa_t}$ für alle $t \in \mathcal{T}_\mathcal{I}$. Falls für alle $t \in \mathcal{T}_\mathcal{I}$ eine orthogonale Matrix $Q_t \in \mathbb{R}^{\mathcal{I}\times\rho_t}_{\mathbf{t}\times\rho_t}$ derart existiert, dass $\Pi_\mathbf{t} Q_t = Q_t$ und $V_t = Q_t O_t$ gilt, nennen wir die Matrizen O_t *orthogonale Gewichte*.

Nach Lemma 4.2.8 sind die Matrizen O_t aus Definition 4.2.6 orthogonale Gewichte. Diese sind später wichtig, da die orthogonalen Matrizen die von uns betrachteten Normen invariant lassen. Dies ermöglicht uns später teilweise die Clusterbasen V_t durch die orthogonalen Gewichte O_t zu ersetzen. Falls $\#\mathbf{t} \ll \#\rho_t$ gilt, können wir dadurch den Aufwand erheblich reduzieren.
Mit Lemma 4.2.8 ergibt sich der Algorithmus 4.2.1 zum Berechnen einer orthogonalen Clusterbasis aus einer bestehenden Clusterbasis. Dieser berechnet die neue orthogonale Clusterbasis $\overline{V}$ und die orthogonalen Gewichte O.

Algorithmus 4.2.1 [siehe [12, Algotrithm 16]] Die Funktion berechnet zur Clusterbasis V eine orthogonale Clusterbasis $\overline{V}$ mit orthogonalen Gewichten O und Rangverteilung ρ

```
function ORTHOGONALE_CLUSTERBASIS(t, V, E, κ, V̄, Ē, O, ρ)
    if chil(t) == ∅ then
        QR-ZERLEGUNG(V_t, I, t, κ_t, V̄_t, O_t, ρ_t)        ▷ siehe Algorithmus 2.4.1
    else
        ℓ ← ∅
        for ť ∈ chil(t) do
            ORTHOGONALE_CLUSTERBASIS(ť, V, E, κ, V̄, Ē, O, ρ)        ▷ bottom-up
            ℓ ← ℓ ∪̇ ρ_ť
        end for
        V̂_t ∈ R^{ℓ×κ_t}
        for ť ∈ chil(t) do
            V̂_{t|ρ_ť×κ_t} ← O_ť E_ť
        end for
        QR-ZERLEGUNG(V̂_t, ℓ, ℓ, κ_t, Q̂_t, O_t, ρ_t)        ▷ siehe Algorithmus 2.4.1
        for ť ∈ chil(t) do
            Ē_ť ← Q̂_{t|ρ_ť×ρ_t}
        end for
    end if
end function
```

Der Algorithmus hat eine Bottom-up-Struktur, da die Basiswechsel der Kinder zum Aufstellen der Matrix $\widehat{V}_t$ gebraucht werden. Die Matrizen $\overline{V}_t$ werden nur für die Blattcluster berechnet und für Nicht-Blattcluster nicht benötigt, da die Matrizen $\breve{V}_t$ im Algorithmus nicht aufgestellt werden. Dies ist ein typisches Beispiel, bei dem große orthogonale Matrizen nur in der Theorie, aber nicht in der Praxis benötigt werden. Auf diese Weise kann der Aufwand in den Algorithmen erheblich reduziert werden.

Lemma 4.2.10
Es sei $V := ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine Clusterbasis mit Rangverteilung $\kappa := (\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$. Dann braucht der Algorithmus 4.2.1 aufgerufen mit $t = r_\mathcal{I}$ höchstens

$$c_{qr}k_{\max}^2\#\mathcal{I} \;+\; (c_{qr}+2)k_{\max}^3\#\mathcal{T}_\mathcal{I}.$$

Operationen, wobei $k_{\max} := \max\limits_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t$ ist.

Beweis:[vergleiche [12, Lemma 5.18]]
Es sei $t \in \mathcal{T}_\mathcal{I}$. Falls $t \in \mathcal{L}_\mathcal{I}$ gilt, wird die QR-Zerlegung von $V_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t}$ berechnet. Dies benötigt nicht mehr als

$$c_{qr}\#\mathbf{t}\#\kappa_t \min\{\#\mathbf{t}, \#\kappa_t\} \;\leq\; c_{qr}\#\mathbf{t}\#\kappa_t^2 \leq\; c_{qr}k_{\max}^2\#\mathbf{t}$$

Operationen (siehe Lemma 2.4.3).
Es sei $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$. Für das Aufstellen der Matrix $\widehat{V}_t$ werden wegen $\#\rho_{\check{t}} \leq \#\kappa_{\check{t}} \leq k_{\max}$ maximal

$$\sum_{\check{t}\in\operatorname{chil}(t)} 2\#\rho_{\check{t}}\#\kappa_{\check{t}}\#\kappa_t \;\leq\; 2\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

Operationen verwendet. Der Aufwand der QR-Zerlegung von $\widehat{V}_t$ ist höchstens

$$\begin{aligned} c_{qr}\sum_{\check{t}\in\operatorname{chil}(t)} \#\rho_{\check{t}}\#\kappa_t \min\left\{\sum_{\check{t}\in\operatorname{chil}(t)} \#\rho_{\check{t}}, \#\kappa_t\right\} &\leq\; c_{qr}\sum_{\check{t}\in\operatorname{chil}(t)} \#\rho_{\check{t}}\#\kappa_t^2 \\ &\leq \sum_{\check{t}\in\operatorname{chil}(t)} c_{qr}k_{\max}^3. \end{aligned}$$

Zusammen mit Lemma 3.2.8 ergibt sich für den Aufwand von Algorithmus 4.2.1 die obere Schranke

$$\begin{aligned} &\sum_{t\in\mathcal{L}_\mathcal{I}} c_{qr}k_{\max}^2\#\mathbf{t} \;+\; \sum_{t\in\mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}}\sum_{\check{t}\in\operatorname{chil}(t)} (c_{qr}+2)k_{\max}^3 \\ &\qquad\leq\; c_{qr}k_{\max}^2\sum_{t\in\mathcal{L}_\mathcal{I}}\#\mathbf{t} \;+\; \sum_{t\in\mathcal{T}_\mathcal{I}}(c_{qr}+2)k_{\max}^3 \\ &\qquad\leq\; c_{qr}k_{\max}^2\#\mathcal{I} \;+\; (c_{qr}+2)k_{\max}^3\#\mathcal{T}_\mathcal{I}. \end{aligned}$$

■

Bemerkung 4.2.11
Die in Algorithmus 4.2.1 berechneten Matrizen $(O_t)_{t\in\mathcal{T}_\mathcal{I}}$ sind sowohl der Basiswechsel von V zu $\overline{V}$ als auch orthogonale Gewichte für V.

Zu gegebenen Clusterbasen können wir also orthogonale Clusterbasen konstruieren, die das Bild der alten Clusterbasen darstellen können. Nach Lemma 4.1.12 ist somit der $\mathcal{H}^2$-Matrix-Raum zu den alten Clusterbasen im $\mathcal{H}^2$-Matrix-Raum für die neuen Clusterbasen enthalten. Falls die Annahmen aus Bemerkung 3.4.7 gelten, können wir die orthogonalen Clusterbasen in $\mathcal{O}(nk^2)$ berechnen.
Bevor wir die orthogonalen Projektionen einführen, zeigen wir in Lemma 4.2.12, dass sich orthogonale Clusterbasen mit Hilfe der Matrizen aus Definition 4.2.3 in zwei orthogonale Teile zerlegen lassen. Diese Zerlegung nutzen wir später bei der Fehleranalyse für die Projektionen in $\mathcal{H}^2$-Matrix-Räume.

Lemma 4.2.12
Es sei $V = ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine orthogonale Clusterbasis. Dann gilt für die Matrizen aus Definition 4.2.3

$$V_t \;=\; \acute{V}(V)_t \check{V}(V)_t \quad \text{für alle } t \in \mathcal{T}_\mathcal{I}.$$

Außerdem sind die Matrizen $\acute{V}(V)_t$ und $\check{V}(V)_t$ für alle $t \in \mathcal{T}_\mathcal{I}$ orthogonal.

Beweis: Für $t \in \mathcal{L}_\mathcal{I}$ gilt

$$\check{V}(V)_t \acute{V}(V)_t \;=\; IV_t \;=\; V_t.$$

Und für $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ gilt mit der vereinfachten Form von $\acute{V}(V)_t$ aus Bemerkung 4.2.5

$$\check{V}(V)_t \acute{V}(V)_t \;=\; \begin{pmatrix} \overline{V}_{t_1} & \cdots & \overline{V}_{t_\tau} \end{pmatrix} \begin{pmatrix} \overline{E}_{t_1} \\ \vdots \\ \overline{E}_{t_\tau} \end{pmatrix} \;=\; \sum_{\check{t}\in\operatorname{chil}(t)} \overline{V}_{\check{t}} \overline{E}_{\check{t}} \;=\; \overline{V}_t.$$

Nach Lemma 4.2.2 folgt aus der Orthogonalität der Clusterbasis V, dass die Matrix $\acute{V}(V)_t$ für alle $t \in \mathcal{T}_\mathcal{I}$ orthogonal ist. Mit Bemerkung 2.3.15 (ii) folgt die Orthogonalität der Matrizen $\check{V}(V)_t$ für alle $t \in \mathcal{T}_\mathcal{I}$. ∎

Als Nächstes untersuchen wir, wie andere Matrizen in einem gegebenen $\mathcal{H}^2$-Matrix-Raum dargestellt werden können. Dabei gehen wir davon aus, dass die gegebenen Clusterbasen orthogonal sind. Diese Voraussetzung können wir mit Hilfe von Algorithmus 4.2.1 erfüllen. Mit Hilfe der Einschränkungen aus Definition 2.2.5 und den zu orthogonalen Matrizen Q gehörigen Projektionen Π_Q aus Beispiel 2.3.18 definieren wir die Projektion auf einen $\mathcal{H}^2$-Matrix-Raum.

Definition 4.2.13 (orthogonale Projektion in den $\mathcal{H}^2$-Matrix-Raum)
Es seien ein zulässiger Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und orthogonale Clusterbasen $V := (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $W := (W_s)_{s\in\mathcal{T}_\mathcal{J}}$ gegeben. Dann ist

$$\begin{aligned} \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W} : \mathbb{R}^{\mathcal{I}\times\mathcal{J}} &\to \mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \\ X &\mapsto \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{V_t} X \Pi_{W_s} + \sum_{(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}}. \end{aligned} \tag{4.3}$$

die $\mathcal{H}^2$-*Matrix-Projektion* zu $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, V und W.

Nach Lemma 4.2.14 ist die oben definierte Abbildung eine orthogonale Projektion bezüglich der Frobeniusnorm mit Bild($\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W}$) = $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W)$. Mit Lemma 2.1.9 folgt daraus, dass das Bild einer Matrix X unter der Abbildung $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W}$ gleich der Bestapproximation der Matrix X in $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W)$ bezüglich der Frobeniusnorm ist.

Lemma 4.2.14
Es seien ein zulässiger Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und orthogonale Clusterbasen $V := (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $W := (W_s)_{s\in\mathcal{T}_\mathcal{J}}$ gegeben. Dann gilt für alle $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W}[X] \;=\; \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W,(X_{|\mathbf{t}\times\mathbf{s}})_{(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}},(V_t^T X_{|\mathbf{t}\times\mathbf{s}}W_s)_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}) \qquad (4.4)$$

und $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W} :\ \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ ist die orthogonale Projektion bezüglich der Frobeniusnorm mit Bild($\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W}$) = $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W)$.

Beweis:[siehe [12, Lemma 5.5]] ∎

Der Beweis [12, Lemma 5.5] betrachtet die Projektion blockweise und nutzt aus, dass die angewendeten Abbildungen Π_{V_t}, Π_{W_s}, $\Pi_\mathbf{t}$ und $\Pi_\mathbf{s}$ orthogonale Projektionen sind. Um Projektionsfehler getrennt nach Zeilen- und Spaltenbasis betrachten zu können, verwenden wir die einseitige Projektion.

Definition 4.2.15 (einseitige Projektion)
Es seien ein zulässiger Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und orthogonale Clusterbasen $V := (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $W := (W_s)_{s\in\mathcal{T}_\mathcal{J}}$ gegeben. Dann definieren wir die *einseitigen Projektionen* durch

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,*} :\ \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}},\ X \mapsto \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{V_t} X \Pi_\mathbf{s} + \sum_{(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_\mathbf{t} X \Pi_\mathbf{s}$$

und

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,W} :\ \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}},\ \mapsto \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_\mathbf{t} X \Pi_{W_s} + \sum_{(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_\mathbf{t} X \Pi_\mathbf{s}.$$

Lemma 4.2.16
Es seien ein zulässiger Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und orthogonale Clusterbasen $V := (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $W := (W_s)_{s\in\mathcal{T}_\mathcal{J}}$ gegeben.

(i) Dann sind $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,*} :\ \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,W} :\ \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ orthogonale Projektionen bezüglich der Frobeniusnorm.

(ii) Es gilt $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W} = \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,*}\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,W} = \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,W}\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,*}$.

(iii) Es gilt $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,W}X = (\Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},W,*}X^T)^T$ für alle $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$.

Beweis:[siehe [12, Lemma 6.4 und Lemma 6.6]] ∎

Mit Hilfe von Lemma 4.2.16 zeigen wir in Lemma 4.3.1, dass wir den Projektionsfehler von $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W}$ gegen die Fehler von $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,*}$ und $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,W}$ abschätzen können.
Zuvor wenden wir uns der konkreten Berechnung der Projektion zu. Weil für diese Arbeit lediglich die Projektion einer Matrix im $\mathcal{H}^2$-Format in einen neuen $\mathcal{H}^2$-Matrix-Raum verwendet wird, untersuchen wir nur diese. Dazu betrachten wir die Basiswechsel aus Definition 4.2.3. Ein Beispiel für Basiswechsel haben wir bei der Orthogonalisierung von Clusterbasen kennengelernt.
Der Vollständigkeit halber geben wir eine rekursive Darstellung der Basiswechsel im allgemeinen Fall an. Weil wir in dieser Arbeit zu den orthogonalen Clusterbasen auch stets die Basiswechsel berechnen, verzichten wir an dieser Stelle auf die Angabe eines Algorithmus und eine entsprechende Aufwandsabschätzung.

Lemma 4.2.17
Es seien $V := ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ und $\overline{V} := \big((\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}, (\overline{E}_t)_{t\in\mathcal{T}_\mathcal{I}}\big)$ Clusterbasen. Dann gilt für den Basiswechsel

$$C(V,\overline{V})_t \;:=\; \begin{cases} \overline{V}_t^T V_t & \text{, falls } t \in \mathcal{L}_\mathcal{I} \\ \sum\limits_{\check{t}\in\operatorname{chil}(t)} \overline{E}_{\check{t}} C(V,\overline{V})_{\check{t}} E_{\check{t}} & \text{, falls } t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}. \end{cases}$$

Beweis:[siehe [12, Abschnitt 5.3]] ∎

Mit Hilfe der Basiswechsel können wir die $\mathcal{H}^2$-Matrix-Darstellung einer projizierten $\mathcal{H}^2$-Matrix einfach definieren.

Lemma 4.2.18
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix sowie $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ orthogonale Clusterbasen. Wir definieren mit den Basiswechseln aus Definition 4.2.3

$$\overline{S}_b \;:=\; C(V,\overline{V})_t S_b C(W,\overline{W})_s^T \quad \text{für alle } b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$$

und $\overline{N}_b := N_b = X_{|\mathbf{t}\times\mathbf{s}}$ für alle $b = (t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$. Dann gilt für die Projektion der $\mathcal{H}^2$-Matrix H auf die neuen Clusterbasen

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[H] = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{S}, \overline{N}).$$

Beweis: Für den Beweis zeigen wir, dass sich alle Blätter entsprechend der Definition 3.4.22 darstellen lassen. Nach (4.4) gilt für alle $b=(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$

$$(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[H])_{|\mathbf{t}\times\mathbf{s}} \;=\; H_{|\mathbf{t}\times\mathbf{s}} \;=\; N_b \;=\; \overline{N}_b.$$

Mit der Darstellung der Projektion in (4.4) folgt für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$

$$\begin{aligned}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[H])_{|\mathbf{t}\times\mathbf{s}} &= \overline{V}_t\overline{V}_t^T H_{|\mathbf{t}\times\mathbf{s}}\overline{W}_s\overline{W}_s^T = \overline{V}_t\overline{V}_t^T V_t S_b W_s^T\overline{W}_s\overline{W}_s^T \\ &= \overline{V}_t C(V,\overline{V})_t S_b C(W,\overline{W})_s^T\overline{W}_s^T = \overline{V}_t\overline{S}_b\overline{W}_s^T.\end{aligned}$$

■

Mit Lemma 4.2.18 können wir die Kopplungsmatrizen der projizierten $\mathcal{H}^2$-Matrix berechnen, wobei die Basiswechsel genutzt werden. Das Nahfeld kann übernommen werden, weil der ursprüngliche Blockbaum wiederverwendet wird. Damit erhalten wir den Algorithmus 4.2.2 zur Berechnung der projizierten Matrix.

Algorithmus 4.2.2 [vergleiche [12, Algorithm 14]] Die Funktion berechnet die Kopplungsmatrizen $\overline{S}_b$ und Nahfeldmatrizen $\overline{N}$ für die neuen, orthogonalen Clusterbasen $\overline{V}$ und $\overline{W}$, wobei die Basiswechsel C und D genutzt werden.

```
function H2-Matrix-Projektion(b, N, S, C, κ̄, D, λ̄, N̄, S̄)
    (t, s) ← b
    if b ∈ L⁺_{I×J} then
        S̄_b ← C_t S_b D_s^T ∈ R^{κ̄_t × λ̄_s}
    else if b ∈ L⁻_{I×J} then
        N̄_b ← N_b ∈ R^{I×J}_{t×s}
    else                                              ▷ b ∈ T_{I×J} \ L_{I×J}
        for b̆ ∈ chil(b) do
            H2-Matrix-Projektion(b̆, N, S, C, κ̄, D, λ̄, N̄, S̄)
        end for
    end if
end function
```

Die Matrix $\overline{N}_b$ ist in dem Algorithmus der Vollständigkeit halber aufgenommen. Die Basiswechsel sind als bekannt vorausgesetzt, weil diese meistens beim Berechnen der neuen Clusterbasis mit aufgestellt werden können (siehe zum Beispiel die Orthogonalisierung in Algorithmus 4.2.1). In Lemma 4.2.19 schätzen wir den Aufwand des Algorithmus 4.2.2 ab.

Lemma 4.2.19
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit Rangverteilungen κ bzw. λ und c_{sp}-schwachbesetztem Blockbaum. Weiter seien $\overline{V}$ bzw. $\overline{W}$ die neuen orthogonalen Clusterbasen mit Rangverteilungen $\overline{\kappa}$ bzw. $\overline{\lambda}$. Wir definieren

$$k_{\max} := \max\left\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s, \max_{t\in\mathcal{T}_\mathcal{I}} \#\overline{\kappa}_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\overline{\lambda}_s\right\}.$$

Dann gilt für den Aufwand zur Berechnung der neuen Kopplungsmatrizen mit Algorithmus 4.2.2 folgende obere Schranke

$$4c_{sp}k_{\max}^3 \min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\}.$$

Beweis:[vergleiche [12, Lemma 5.14]]
Für jedes zulässige Blatt $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ wird $\overline{S}_b = C_t S_b D_s$ berechnet. Das benötigt höchstens

$$2\#\overline{\kappa}_t\#\kappa_t\#\lambda_s \; + 2\#\overline{\kappa}_t\#\lambda_s\#\overline{\lambda}_s \leq \; 4k^3_{\max}$$

Operationen. In den unzulässigen Blättern werden keine arithmetischen Operationen benötigt. Zusammen mit Lemma 3.3.6 lässt sich der Gesamtaufwand durch

$$\sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} 4k^3_{\max} \;\leq\; 4k^3_{\max}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \;\leq\; 4c_{sp}k^3_{\max}\min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}\}$$

beschränken. ∎

Also können wir eine bestehende $\mathcal{H}^2$-Matrix effizient in einen anderen $\mathcal{H}^2$-Matrix-Raum projizieren. Häufig stellt sich die Wahl der richtigen Clusterbasis als deutlich komplizierter dar. Für die Konstruktion von adaptiven Clusterbasen für eine gegebene Matrix benötigen wir eine geeignete Darstellung für den Projektionsfehler. Diesen untersuchen wir im Abschnitt 4.3.
Zuvor wollen wir noch die in diesem Abschnitt vorgestellten Algorithmen zusammenfassen, um aus einer beliebigen $\mathcal{H}^2$-Matrix eine $\mathcal{H}^2$-Matrix mit orthogonalen Clusterbasen zu konstruieren.

Algorithmus 4.2.3 Die Funktion berechnet zu einer $\mathcal{H}^2$-Matrix eine Darstellung mit orthogonalen Clusterbasen.

---

function ORTHOGONALE_H2MATRIX($\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, V, κ, W, λ, N, S, $\overline{V}$, $\overline{\kappa}$, $\overline{W}$, $\overline{\lambda}$, $\overline{N}$, $\overline{S}$)
  ORTHOGONALE_CLUSTERBASIS($r_{\mathcal{I}}$, V, κ, $\overline{V}$, C, $\overline{\kappa}$) ▷ Algorithmus 4.2.1
  ORTHOGONALE_CLUSTERBASIS($r_{\mathcal{J}}$, W, κ, $\overline{W}$, D, $\overline{\lambda}$) ▷ Algorithmus 4.2.1
  H2-MATRIX-PROJEKTION($r_{\mathcal{I}\times\mathcal{J}}$, N, S, C, $\overline{\kappa}$, D, $\overline{\lambda}$, $\overline{N}$, $\overline{S}$) ▷ Algorithmus 4.2.2
end function

In Lemma 4.2.20 schätzen wir den Aufwand für das Orthogonalisieren einer $\mathcal{H}^2$-Matrix durch Algorithmus 4.2.3 ab.

Lemma 4.2.20
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen κ und λ. Wir definieren den maximalen lokalen Rang $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}}\#\kappa_t, \max_{s\in\mathcal{T}_{\mathcal{J}}}\#\lambda_s\}$. Dann benötigt Algorithmus 4.2.3 zur Berechnung von $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S}) = H$ mit orthogonalen Clusterbasen $\overline{V}$ und $\overline{W}$ höchstens

$$c_{qr}k^2_{\max}(\#\mathcal{I} + \#\mathcal{J}) \;+\; (2c_{sp} + c_{qr} + 2)k^3_{\max}(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}})$$

Operationen.

Beweis: Es seien $\overline{\kappa}$ und $\overline{\lambda}$ die Rangverteilungen der aus der Orthogonalisierung mit Algorithmus 4.2.1 entstandenen Clusterbasen $\overline{V}$ und $\overline{W}$. Dann gilt nach Lemma 4.2.8 $\overline{\kappa}_t \leq \kappa_t$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $\overline{\lambda}_s \leq \lambda_s$ für alle $s \in \mathcal{T}_\mathcal{J}$. Der Aufwand für den Aufruf von Algorithmus 4.2.1 für V und W ist nach Lemma 4.2.10 durch

$$c_{qr} k_{\max}^2 (\#\mathcal{I} + \#\mathcal{J}) \ + \ (c_{qr} + 2) k_{\max}^3 (\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J})$$

beschränkt. Die dabei berechneten Familien von Matrizen $(C_t)_{t \in \mathcal{T}_\mathcal{I}}$ bzw. $(D_s)_{s \in \mathcal{T}_\mathcal{J}}$ sind die entsprechenden Basiswechsel. Somit berechnet Algorithmus 4.2.2 die zugehörigen Kopplungs- und Nahfeldmatrizen. Dies benötigt nach Lemma 4.2.19 höchstens

$$4 c_{sp} k_{\max}^3 \min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\}$$

Operationen. Zusammen mit $\min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\} \leq \frac{1}{2}(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J})$ erhalten wir für den Aufwand von Algorithmus 4.2.3 die obere Schranke

$$\begin{aligned} c_{qr} k_{\max}^2 (\#\mathcal{I} + \#\mathcal{J}) \ + \ (c_{qr} + 2) k_{\max}^3 (\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}) \ + \ 4 c_{sp} k_{\max}^3 \min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\} \\ \leq \ c_{qr} k_{\max}^2 (\#\mathcal{I} + \#\mathcal{J}) \ + \ (2 c_{sp} + c_{qr} + 2) k_{\max}^3 (\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}). \end{aligned}$$

Nach Lemma 4.2.8 ist $\operatorname{Bild}(V_t) \subseteq \operatorname{Bild}(\overline{V}_t)$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $\operatorname{Bild}(W_s) \subseteq \operatorname{Bild}(\overline{W}_s)$ für alle $s \in \mathcal{T}_\mathcal{J}$. Zusammen mit Lemma 4.1.12 folgt $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W})$. Da Algorithmus 4.2.2 eine Projektion auf $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W})$ ist, folgt $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}}[H] = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{S}, \overline{N}) = H$. ∎

Falls die Annahmen aus Bemerkung 3.4.7 gelten, liegt der Aufwand in $\mathcal{O}(nk^2)$. Wir können also eine orthogonale Darstellung für eine beliebige $\mathcal{H}^2$-Matrix in linearer Komplexität (bezüglich der Anzahl der Freiheitsgrade) berechnen. Wie bereits erwähnt, wollen wir später eine adaptive Clusterbasis berechnen, die möglichst kleinen Rang besitzt und für die der Projektionsfehler eine gewisse Schranke nicht überschreitet. Als ersten Schritt untersuchen wir im nächsten Abschnitt den Projektionsfehler der orthogonalen Projektion aus Definition 4.2.13.

4.3 Projektionsfehler

In diesem Abschnitt werden wir den Projektionsfehler der Projektion aus Definition 4.2.13 untersuchen. Dabei verfolgen wir zuerst einen clusterweisen Ansatz, der den Projektionsfehler mit Hilfe der totalen Clusterbasen gut beschreibt.
Der zweite Ansatz ist eine blockweise Fehlerbetrachtung. Diese Betrachtungsweise kann zwar für die Spektralnorm zu einer größeren Überschätzung des Fehlers führen, hat allerdings den Vorteil, dass wir den Fehler in den einzelnen Blöcken kontrollieren können. Bei Matrizen, deren Teilblöcke besonders unterschiedlich große Normen aufweisen, hilft uns dieser Ansatz dabei, Blöcke mit kleiner Norm hinreichend gut zu approximieren. Dies ist für die Arithmetik von Vorteil, weil sie auf der blockweisen Berechnung basiert.

Wir beginnen mit der clusterweisen Betrachtung und konzentrieren uns dabei auf die Zeilenbasis. Um dies zu legitimieren, zeigen wir im folgenden Lemma, dass sich der Projektionsfehler durch den Fehler bezüglich der Zeilenbasis und der Spaltenbasis aufteilen lässt.

Lemma 4.3.1
Es seien $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\overline{V}$ und $\overline{W}$ orthogonale Clusterbasen und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Dann gilt

$$\begin{aligned}&\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2\\ &\quad\leq\ 2\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 + 2\|(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T,\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T]\|_S^2\end{aligned}$$

und

$$\begin{aligned}\max\{\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_F^2, \|X^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{W},*}[X^T]\|_F^2\}&\\ \leq\ \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_F^2&\\ \leq\ \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_F^2 + \|X^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{W},*}[X^T]\|_F^2.&\end{aligned}$$

Beweis: Für die Spektralnorm verwenden wir im Gegensatz zu [12, Lemma 6.7] eine Abschätzung mit quadratischen Termen. Es gilt

$$\begin{aligned}\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2 &\leq \Big(\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S + \|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S\Big)^2\\ &\leq 2\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 + 2\|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2.\end{aligned}$$

Wir schreiben den letzten Term mit Hilfe von Lemma 4.2.16 als

$$\begin{aligned}\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X] &= \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},*,\overline{W}}\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\\ &= \Big((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T,\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T]\Big)^T.\end{aligned}$$

Nach Lemma 2.3.13 ist die Spektralnorm unter Adjungieren invariant und es folgt

$$\begin{aligned}\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2 &\leq 2\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 + 2\|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2\\ &= 2\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 + 2\|(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T,\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T]\|_S^2.\end{aligned}$$

Als Nächstes betrachten wir die Frobeniusnorm. Die obere Fehlerschranke gilt nach [12, Theorem 6.9] und die untere nach [12, Remark 6.10]. Dabei werden die Eigenschaften der einseitigen Projektionen aus Lemma 4.2.16 ausgenutzt und verwendet, dass die $\mathcal{H}^2$-Matrix-Projektionen im Hilbertraum $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ orthogonale Projektionen sind. ■

Die untere Schranke für die Frobeniusnorm zeigt, dass der Fehler durch die obere Schranke maximal um den Faktor 2 überschätzt wird. Bei der weiteren Betrachtung des Projektionsfehlers können wir uns vorerst auf die Zeilenbasis konzentrieren. Mit Lemma 4.3.1

folgt dann eine Fehlerabschätzung für die gesamte Projektion.
Für die Spektralnorm können wir i. A. nicht davon ausgehen, dass $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}$ als Operator von $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_S)$ nach $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_S)$ eine Norm kleiner als 1 hat. Deshalb taucht $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}$ in der Abschätzung für die Spektralnorm auch im zweiten Term auf. Um dieses Problem werden wir uns später gesondert kümmern.
Im nächsten Schritt werden wir den Fehler der Projektion bezüglich der Zeilenbasis näher untersuchen. Dabei werden wir den Fehler im ersten Schritt in Teile für jeden Zeilencluster aufteilen. Dafür nutzen wir die Fehlerterme aus Definition 4.3.2 (vergleiche D_t aus [12, Lemma 5.25]). Diese verwenden statt $(I_{\mathbf{t}} - \Pi_{\overline{V}_t})$ die Abbildung $(I_{\mathbf{t}} - \Pi_{\overline{V}_t})\Pi_{\check{V}(\overline{V})_t}$, weil $\overline{V}_t$ durch die geschachtelte Darstellung der Clusterbasis nur das repräsentieren kann, was durch $\check{V}(\overline{V})_t$ darstellbar ist. Bei der Konstruktion der adaptiven Clusterbasen in Abschnitt 4.4 tauchen Terme auf, die den Fehlertermen $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t$ entsprechen.

Definition 4.3.2 (Fehlerterme für orthogonale Projektion)
Es sei $(\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ eine Clusterbasis. Dann definieren wir die Matrizen

$$\Pi_{D,\overline{V},t} \;:=\; \left(I_{\mathbf{t}} - \Pi_{\overline{V}_t}\right)\Pi_{\check{V}(\overline{V})_t} \;\in\; \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\mathbf{t}\times\mathbf{t}} \quad \text{für alle } t \in \mathcal{T}_{\mathcal{I}},$$

wobei $I_{\mathbf{t}} := \Pi_{\mathbf{t}}$ die Identität in $\mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$ bzw. $\mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\mathbf{t}\times\mathbf{t}}$ sei. Es sei zusätzlich $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Dann definieren wir den clusterweisen Fehlerterm

$$D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t \;:=\; \Pi_{D,\overline{V},t} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t \quad \text{für alle } t \in \mathcal{T}_{\mathcal{I}}$$

über die totale Clusterbasis und den blockweisen Fehlerterm

$$D(X, \overline{V})_{t,s} \;:=\; \Pi_{D,\overline{V},t} X_{|\mathbf{t}\times\mathbf{s}} \quad \text{für alle } t \in \mathcal{T}_{\mathcal{I}}, s \in \mathcal{T}_{\mathcal{J}}.$$

Wir verwenden die Notation $I_{\mathbf{t}}$ statt $\Pi_{\mathbf{t}}$, um zu betonen, dass wir an dieser Stelle Projektionsfehler betrachten. Wegen $\left(I_{\mathbf{t}} - \Pi_{\overline{V}_t}\right)\Pi_{\check{V}(\overline{V})_t} = \left(I_{\mathcal{I}} - \Pi_{\overline{V}_t}\right)\Pi_{\check{V}(\overline{V})_t}\Pi_{\mathbf{t}}$ könnten wir für die Fehlerterme auch die Identität auf $\mathbb{R}^{\mathcal{I}}$ nutzen. Die Verwendung von $I_{\mathbf{t}}$ hat den Grund, dass wir die Matrix $\Pi_{D,\overline{V},t}$ zuerst alleine untersuchen wollen. Für die beiden beteiligten Projektionen $\Pi_{\overline{V}_t}$ und $\Pi_{\check{V}_t}$ gilt folgender Zusammenhang.

Bemerkung 4.3.3
Es sei $(\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ eine orthogonale Clusterbasis. Dann gilt für alle $t \in \mathcal{T}_{\mathcal{I}}$ mit den Bezeichnungen $\acute{V}_t := \acute{V}(\overline{V})_t$ und $\check{V}_t := \check{V}(\overline{V})_t$

$$\begin{aligned}\Pi_{\overline{V}_t} \;&=\; \overline{V}_t\overline{V}_t^T \;=\; \overline{V}_t\acute{V}_t^T\check{V}_t^T \;=\; \overline{V}_t\acute{V}_t^T\check{V}_t^T\check{V}_t\check{V}_t^T \;=\; \Pi_{\overline{V}_t}\Pi_{\check{V}_t} \\ &=\; \overline{V}_t\overline{V}_t^T \;=\; \check{V}_t\acute{V}_t\overline{V}_t^T \;=\; \check{V}_t\acute{V}_t^T\check{V}_t\acute{V}_t\overline{V}_t^T \;=\; \Pi_{\check{V}_t}\Pi_{\overline{V}_t}.\end{aligned} \tag{4.5}$$

Mit dieser Umformung können wir zeigen, dass sich $I_{\mathbf{t}} - \Pi_{\overline{V}_t}$ in die Abbildungen $\Pi_{D,\overline{V},\check{t}}$, $\check{t} \in \operatorname{desc}(t)$, deren Bilder orthogonal zueinander sind, aufteilen lässt. Wir zeigen zusätzlich, dass die Abbildungen $\Pi_{D,\overline{V},\check{t}}$ orthogonale Projektionen sind.

Lemma 4.3.4
Es sei $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine orthogonale Clusterbasis. Für alle $t \in \mathcal{T}_\mathcal{I}$ sei $\acute{V}_t := \acute{V}(\overline{V})_t$ und $\breve{V}_t := \breve{V}(\overline{V})_t$. Es seien $t, t' \in \mathcal{T}_\mathcal{I}$ mit $t \neq t'$. Dann ist $\Pi_{D,\overline{V},t}$ eine orthogonale Projektion mit $\mathrm{Bild}(\Pi_{D,\overline{V},t}) = \mathrm{Bild}(I_\mathbf{t} - \Pi_{\overline{V}_t}) \cap \mathrm{Bild}(\Pi_{\breve{V}_t})$ und es gilt

$$I_\mathbf{t} - \Pi_{\overline{V}_t} = \sum_{\breve{t}\in\mathrm{desc}(t)} \Pi_{D,\overline{V},\breve{t}} \quad \text{und} \tag{4.6}$$

$$\Pi^T_{D,\overline{V},t'}\Pi_{D,\overline{V},t} = 0. \tag{4.7}$$

Falls $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ ist, gilt zusätzlich $\mathrm{Bild}(\Pi_{D,\overline{V},t}) \subseteq \mathrm{Bild}(\Pi_{\breve{V}_t}) = \dot{\bigcup}_{\breve{t}\in\mathrm{chil}(t)} \mathrm{Bild}(\overline{V}_{\breve{t}})$.

Beweis: Mit (4.5) folgt

$$\Pi_{D,\overline{V},t} = \left(I_\mathbf{t} - \Pi_{\overline{V}_t}\right)\Pi_{\breve{V}_t} = \Pi_{\breve{V}_t} - \Pi_{\overline{V}_t}\Pi_{\breve{V}_t} = \Pi_{\breve{V}_t} - \Pi_{\breve{V}_t}\Pi_{\overline{V}_t} = \Pi_{\breve{V}_t}\left(I_\mathbf{t} - \Pi_{\overline{V}_t}\right).$$

Also ist $\Pi_{D,\overline{V},t}$ Produkt zweier kommutierender orthogonaler Projektionen. Damit ist $\Pi_{D,\overline{V},t}$ nach [37, Satz II.5.13] eine orthogonale Projektion, wobei für das Bild

$$\mathrm{Bild}(\Pi_{D,\overline{V},t}) = \mathrm{Bild}(I_\mathbf{t} - \Pi_{\overline{V}_t}) \cap \mathrm{Bild}(\Pi_{\breve{V}_t}) \subseteq \mathrm{Bild}(\breve{V}_t)$$

gilt. Falls $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ ist, gilt $\mathrm{Bild}(\breve{V}_t) = \sum_{\breve{t}\in\mathrm{chil}(t)} \mathrm{Bild}(\overline{V}_{\breve{t}})$.
(4.6): [vergleiche [12, Lemma 5.29]] Wir zeigen die Aussage für alle $t \in \mathcal{T}_\mathcal{I}$ per Induktion über $\#\,\mathrm{desc}(t) \in \mathbb{N}_{>0}$. Für $t \in \mathcal{T}_\mathcal{I}$ mit $\#\,\mathrm{desc}(t) = 1$ gilt $t \in \mathcal{L}_\mathcal{I}$ und

$$I_\mathbf{t} - \Pi_{\overline{V}_t} = (I_\mathbf{t} - \Pi_{\overline{V}_t})\Pi_{\breve{V}_t} = \Pi_{D,\overline{V},\breve{t}} = \sum_{\breve{t}\in\mathrm{desc}(t)} \Pi_{D,\overline{V},\breve{t}}.$$

Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $t \in \mathcal{T}_\mathcal{I}$ mit $\#\,\mathrm{desc}(t) \leq n$ gelte. Es sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\,\mathrm{desc}(t) = n + 1$. Dann ist $\#\,\mathrm{desc}(\breve{t}) \leq n$ für alle $\breve{t} \in \mathrm{chil}(t)$ und die Induktionsannahme gilt für alle Kinder von t. Mit (4.5) folgt

$$\begin{aligned} I_\mathbf{t} - \Pi_{\overline{V}_t} &= (I_\mathbf{t} - \Pi_{\overline{V}_t})\Pi_{\breve{V}_t} + (I_\mathbf{t} - \Pi_{\overline{V}_t})(I_\mathbf{t} - \Pi_{\breve{V}_t}) \\ &= \Pi_{D,\overline{V},t} + I_\mathbf{t} - \Pi_{\overline{V}_t} - \Pi_{\breve{V}_t} + \Pi_{\overline{V}_t}\Pi_{\breve{V}_t} \\ &= \Pi_{D,\overline{V},t} + I_\mathbf{t} - \Pi_{\overline{V}_t} - \Pi_{\breve{V}_t} + \Pi_{\overline{V}_t} = \Pi_{D,\overline{V},t} + I_\mathbf{t} - \Pi_{\breve{V}_t}. \end{aligned}$$

Aufgrund der Definition 4.2.3 von $\Pi_{\breve{V}_t}$ folgt mit $\mathrm{chil}(t) =: \{t_1, \ldots, t_\tau\}$

$$\begin{aligned} I_\mathbf{t} - \Pi_{\breve{V}_t} &= I_\mathbf{t} - \begin{pmatrix}\overline{V}_{t_1} & \ldots & \overline{V}_{t_\tau}\end{pmatrix}\begin{pmatrix}\overline{V}_{t_1} & \ldots & \overline{V}_{t_\tau}\end{pmatrix}^T \\ &= I_\mathbf{t} - \sum_{\breve{t}\in\mathrm{chil}(t)} \overline{V}_{\breve{t}}\overline{V}^T_{\breve{t}} = \sum_{\breve{t}\in\mathrm{chil}(t)} I_{\breve{\mathbf{t}}} - \Pi_{\overline{V}_{\breve{t}}}. \end{aligned}$$

Wir setzen dies oben ein und erhalten mit der Induktionsvoraussetzung

$$\begin{aligned} I_\mathbf{t} - \Pi_{\overline{V}_t} &= \Pi_{D,\overline{V},t} + \sum_{\breve{t}\in\mathrm{chil}(t)} I_{\breve{\mathbf{t}}} - \Pi_{\overline{V}_{\breve{t}}} \\ &= \Pi_{D,\overline{V},t} + \sum_{\breve{t}\in\mathrm{chil}(t)} \sum_{\breve{t}\in\mathrm{desc}(\breve{t})} \Pi_{D,\overline{V},\breve{t}} = \sum_{\breve{t}\in\mathrm{desc}(t)} \Pi_{D,\overline{V},\breve{t}}. \end{aligned}$$

(4.7): [vergleiche [12, Lemma 5.26]] Für $\mathbf{t} \cap \mathbf{t}' = \emptyset$ gilt wegen $\mathrm{Bild}(\Pi_{D,\overline{V},t}) \subseteq \mathrm{Bild}(\check{V}_t) \subseteq \mathbb{R}^{\mathcal{I}}_{\mathbf{t}}$ und $\mathrm{Bild}(\Pi_{D,\overline{V},t'}) \subseteq \mathbb{R}^{\mathcal{I}}_{\mathbf{t}'}$

$$\Pi^T_{D,\overline{V},t_1}\Pi_{D,\overline{V},t_2} \;=\; \Pi^T_{D,\overline{V},t'}\Pi_{\mathbf{t}'}\Pi_{\mathbf{t}}\Pi_{D,\overline{V},t} \;=\; \Pi^T_{D,\overline{V},t'}0\Pi_{D,\overline{V},t} \;=\; 0.$$

Für $\mathbf{t} \cap \mathbf{t}' \neq \emptyset$ ist $t' \in \mathrm{desc}(t)$ oder $t \in \mathrm{desc}(t')$ (siehe Lemma 3.2.5 (iii)). Wegen $t \neq t'$ sei o. E. d. A. $t' \in \mathrm{desc}(t)\backslash\{t\}$ (ansonsten transponieren wir die Gleichung). Dann existiert genau ein $\check{t} \in \mathrm{chil}(t)$ mit $t' \in \mathrm{desc}(\check{t})$. Mit der allgemeinen Transfermatrix $E_{t',\check{t}}$ und Lemma 4.1.10 erhalten wir

$$\mathrm{Bild}(\Pi_{\mathbf{t}'}\Pi_{D,\overline{V},t}) \;\subseteq\; \mathrm{Bild}(\Pi_{\mathbf{t}'}\Pi_{\check{\mathbf{t}}}\check{V}_t) \;=\; \mathrm{Bild}(\Pi_{\mathbf{t}'}\overline{V}_{\check{t}}) \;=\; \mathrm{Bild}(\overline{V}_{t'}E_{t',\check{t}}) \;\subseteq\; \mathrm{Bild}(\overline{V}_{t'}).$$

Mit Lemma 2.1.7 folgt $\mathrm{Bild}((I_{\mathbf{t}'} - \Pi_{\overline{V}_{t'}}))\perp \mathrm{Bild}(\Pi_{\overline{V}_{t'}})$ und somit

$$\Pi^T_{D,\overline{V},t'}\Pi_{D,\overline{V},t} \;=\; \Pi^T_{D,\overline{V},t'}\Pi_{\mathbf{t}'}\Pi_{D,\overline{V},t} \;=\; \Pi_{\check{V}_{t'}}(I_{\mathbf{t}'} - \Pi_{\overline{V}_{t'}})\Pi_{D,\overline{V},t} \;=\; 0.$$

■

Das Theorem 4.3.5 gibt eine exakte Darstellung des einseitigen Projektionsfehlers durch die clusterweisen Fehlerterme aus 4.3.2. Die Eigenschaft $D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t^T D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_{t'} = 0$ für $t \neq t'$ entspricht dem Kriterium für Orthogonalität aus Lemma 2.3.21. Zusammen mit Lemma 2.1.2 können wir dann eine Fehlerabschätzung in der Norm folgern. Für diese Aussage zitieren wir [9]. Die Aussagen aus Lemma 4.3.4 verwenden wir später für den blockweisen Fehler und in Kapitel 5 für die Fehlerabschätzungen des Niedrigrangupdates

Theorem 4.3.5 (orthogonale Zerlegung des einseitigen Projektionsfehlers)
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $(\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ eine orthogonale Clusterbasis und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Es bezeichne $D_t := D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}$. Dann gilt

$$X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X] \;=\; \sum_{t\in\mathcal{T}_{\mathcal{I}}} D_t.$$

Außerdem gilt $D_{t'}^T D_t = 0$ für alle $t, t' \in \mathcal{T}_{\mathcal{I}}$ mit $t \neq t'$.

Beweis: Für alle $t \in \mathcal{T}_{\mathcal{I}}$ gilt mit (4.5), $\acute{V}_t := \acute{V}(\overline{V})_t$ und $\check{V}_t := \check{V}(\overline{V})_t$

$$\Pi_{D,\overline{V},t} \;=\; (I_{\mathbf{t}} - \Pi_{\overline{V}_t})\Pi_{\check{V}_t} \;=\; \Pi_{\check{V}_t} - \Pi_{\overline{V}_t}\Pi_{\check{V}_t} \;=\; \Pi_{\check{V}_t} - \Pi_{\overline{V}_t}. \tag{4.8}$$

Damit folgt, dass die von uns verwendeten Fehlerterme denen in [9, Theorem 3.11] entsprechen, und die Aussage folgt aus dem Beweis zu [9, Theorem 3.11]. ■

Wir zeigen als Nächstes, dass sich die Fehlerterme $\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})\|_S$ durch $\|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})\|_S$ beschränken lassen. Danach können wir mit Hilfe von Lemma 4.3.1 und Theorem 4.3.5 eine globale Fehlerschranke beweisen, die nur die Terme $D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})$ und $D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})$ enthält.

Lemma 4.3.6
Es seien $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\overline{V}$ und $\overline{W}$ orthogonale Clusterbasen und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Dann gilt für alle $s \in \mathcal{T}_{\mathcal{J}}$

$$\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S \;\leq\; \|D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S.$$

Beweis: Für die totalen Clusterbasen führen wir $X_{\Pi,s} := X(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, (\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T)_s$ und $X_{T,s} := X(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, X^T)_s$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ als Schreibweisen ein (siehe Definition 4.1.6 der totalen Clusterbasis). Dann gilt nach Definition 4.1.3 der erweiterten Blockzeile

$$\begin{aligned}
X_{\Pi,s} &= \Pi_{\mathbf{s}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T \sum_{t\in\overline{\mathrm{row}}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},s)} \Pi_{\mathbf{t}} \\
&= \sum_{\hat{s}\in\mathrm{pred}(s)} \sum_{t\in\mathrm{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\hat{s})} \Pi_{\mathbf{s}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T\Pi_{\mathbf{t}} \\
&= \Pi_{\mathbf{s}} \sum_{\hat{s}\in\mathrm{pred}(s)} \sum_{t\in\mathrm{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\hat{s})} \Pi_{\hat{\mathbf{s}}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T\Pi_{\mathbf{t}} \\
&= \Pi_{\mathbf{s}} \sum_{\hat{s}\in\mathrm{pred}(s)} \sum_{t\in\mathrm{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\hat{s})} (\Pi_{\mathbf{t}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])\Pi_{\hat{\mathbf{s}}})^T.
\end{aligned} \tag{4.9}$$

Weil $(t,\hat{s}) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ für alle $\hat{s} \in \mathrm{pred}(s)$ und $t \in \mathrm{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \hat{s}) = \mathrm{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \hat{s}) = \mathrm{col}(\hat{s})$ gilt, folgt

$$\begin{aligned}
X_{\Pi,s} &= \Pi_{\mathbf{s}} \sum_{\hat{s}\in\mathrm{pred}(s)} \sum_{t\in\mathrm{col}(\hat{s})} (\Pi_{\mathbf{t}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])\Pi_{\hat{\mathbf{s}}})^T \\
&= \Pi_{\mathbf{s}} \sum_{\hat{s}\in\mathrm{pred}(s)} \sum_{t\in\mathrm{col}(\hat{s})} (\Pi_{\overline{V}_t} X \Pi_{\hat{\mathbf{s}}})^T \;=\; \sum_{t\in\overline{\mathrm{col}}(s)} (\Pi_{\overline{V}_t} X \Pi_{\mathbf{s}})^T.
\end{aligned}$$

Wir verwenden $\overline{\xi}_s = \dot{\bigcup}_{t\in\overline{\mathrm{col}}(s)} \mathbf{t}$ und definieren die Matrix

$$\Pi_{\overline{\mathrm{col}}(s)} \;:=\; \sum_{t\in\overline{\mathrm{col}}(s)} \Pi_{\overline{V}_t} \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\overline{\xi}_s\times\overline{\xi}_s}.$$

Weil $\mathrm{Bild}(\Pi_{\overline{V}_t}) \subseteq \mathrm{Bild}(\Pi_{\mathbf{t}})$ gilt und die Mengen $\mathbf{t}$ für alle $t \in \overline{\mathrm{col}}(s)$ paarweise disjunkt sind, ist $\Pi_{\overline{\mathrm{col}}(s)}$ eine orthogonale Projektion (siehe Lemma 2.1.6). Für diese gilt

$$\begin{aligned}
X_{\Pi,s} &= \sum_{t\in\overline{\mathrm{col}}(s)} (\Pi_{\overline{V}_t} X \Pi_{\mathbf{s}})^T \;=\; \left(\sum_{t'\in\overline{\mathrm{col}}(s)} \Pi_{\overline{V}_{t'}} \sum_{t\in\overline{\mathrm{col}}(s)} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}}\right)^T \\
&= \left(\Pi_{\mathbf{s}} X^T \sum_{t\in\overline{\mathrm{col}}(s)} \Pi_{\mathbf{t}}\right) \sum_{t'\in\overline{\mathrm{col}}(s)} \Pi_{\overline{V}_{t'}} \;=\; X_{T,s}\Pi_{\overline{\mathrm{col}}(s)}.
\end{aligned}$$

Wir nutzen für alle $s \in \mathcal{T}_{\mathcal{J}}$ die Notation $\breve{W}_s := \breve{V}(\overline{W})_s$ (siehe Definition 4.2.3). Mit der Definition 4.3.2 für die Fehlerterme und der Invarianz der Spektralnorm unter Adjungieren (siehe Lemma 2.3.13) folgt

$$\begin{aligned}\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S &= \|\Pi_{D,\overline{W},s} X_{\Pi,s}\|_S = \|X^T_{\Pi,s}\Pi_{D,\overline{W},s}\|_S \\ &= \|\Pi_{\overline{\mathrm{col}(s)}} X^T_{T,s}\Pi_{D,\overline{W},s}\|_S.\end{aligned}$$

Weil $\Pi_{\overline{\mathrm{col}(s)}}$ eine orthogonale Projektion ist, folgt mit Lemma 2.3.20

$$\begin{aligned}\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S &= \|\Pi_{\overline{\mathrm{col}(s)}} X^T_{T,s}\Pi_{D,\overline{W},s}\|_S \leq \|X^T_{T,s}\Pi_{D,\overline{W},s}\|_S \\ &= \|\Pi_{D,\overline{W},s} X_{T,s}\|_S = \|D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S.\end{aligned}$$

■

Mit Hilfe von Lemma 4.3.1 und Theorem 4.3.5 folgt eine allgemeine Fehlerabschätzung. Dabei nutzen wir Lemma 4.3.6, um die Abschätzung so zu gestalten, dass nur die Fehlerterme $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})$ und $D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})$ benutzt werden.

Theorem 4.3.7
Es seien $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\overline{V}$ und $\overline{W}$ orthogonale Clusterbasen und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Dann gilt

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2 \leq 2\left(\sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|_S^2 + \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S^2\right)$$

und

$$\begin{aligned}&\max\left\{\sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|_F^2, \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_F^2\right\} \\ &\leq \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_F^2 \\ &\leq \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|_F^2 + \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_F^2.\end{aligned}$$

Beweis: Zuerst zeigen wir die Aussage für die Spektralnorm. Wir betrachten die Projektion bezüglich $\overline{V}$ mit der Notation $D_t := D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t$. Es sei $x \in \mathbb{R}^{\mathcal{J}}$. Für alle $t, t' \in \mathcal{T}_{\mathcal{I}}$, $t \neq t'$, gilt nach Theorem 4.3.5, dass $D_{t'}^T D_t = 0$ ist, und mit Lemma 2.3.21 folgt $\langle D_t x, D_{t'} x\rangle_2 = 0$. Also sind die Vektoren $D_t x$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ paarweise orthogonal und mit Theorem 4.3.5 sowie Lemma 2.1.2 folgt

$$\|(X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])x\|_2^2 = \|\sum_{t\in\mathcal{T}_{\mathcal{I}}} D_t x\|_2^2 = \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D_t x\|_2^2 = \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t x\|_2^2.$$

Nach Lemma 2.3.9 lässt sich die Abschätzung auf die Spektralnorm übertragen

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 \;\leq\; \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_S^2.$$

Analog erhalten wir für die Projektion $\Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W},*}$ bezüglich der Spaltenbasis $\overline{W}$

$$\begin{aligned}\|(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T - \Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T]\|_S^2 \\ \leq \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_S^2.\end{aligned}$$

Mit Lemma 4.3.6 können wir auf der rechten Seite die einseitige Projektion $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}$ entfernen und erhalten

$$\|(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T - \Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T]\|_S^2 \;\leq\; \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_S^2.$$

Nach Lemma 4.3.1 müssen wir nur noch die beiden Terme addieren und erhalten

$$\begin{aligned}&\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 \\ &\quad\leq\; 2\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X]\|_S^2 + 2\|(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T - \Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])^T]\|_S^2 \\ &\quad=\; 2\left(\sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_S^2 + \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_S^2\right).\end{aligned}$$

Nun betrachten wir die Frobeniusnorm. Wir verwenden wieder die Bezeichnung $D_t := D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t$. Nach Theorem 4.3.5 gilt $D_{t'}^T D_t = 0$ für alle $t,t' \in \mathcal{T}_{\mathcal{I}}$, $t \neq t'$. Mit Lemma 2.3.21 folgt daraus $\langle D_t, D_{t'}\rangle_F = 0$. Also sind die Matrizen D_t für alle $t \in \mathcal{T}_{\mathcal{I}}$ paarweise orthogonal und mit Lemma 2.1.2 folgt

$$\|(X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},*}[X])\|_F^2 \;=\; \|\sum_{t\in\mathcal{T}_{\mathcal{I}}} D_t\|_F^2 \;=\; \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D_t\|_F^2 \;=\; \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_F^2.$$

Analog folgt für die Projektion $\Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W},*}$ bezüglich der Spaltenbasis $\overline{W}$

$$\|(X^T - \Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W},*}[X^T])\|_F^2 \;=\; \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_t\|_F^2.$$

Mit Lemma 4.3.1 folgt

$$\begin{aligned}&\max\left\{\sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_F^2, \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_F^2\right\} \\ &\quad\leq\; \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_F^2 \\ &\quad\leq\; \sum_{t\in\mathcal{T}_{\mathcal{I}}} \|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_F^2 + \sum_{s\in\mathcal{T}_{\mathcal{J}}} \|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_F^2.\end{aligned}$$

Dabei ist für die untere Schranke wichtig, dass der Projektionsfehler bezüglich $\overline{V}$ und $\overline{W}$ jeweils mit Gleichheit umgeformt wurde. ∎

Nach Theorem 4.3.7 müssen wir bei der Konstruktion der adaptiven Clusterbasen in Abschnitt 4.4 darauf achten, dass die Normen der Fehlerterme $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t$ und $D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s$ hinreichend klein sind. Dies erreichen wir mit Hilfe der gekürzten Singulärwertzerlegung (siehe Algorithmus 2.4.3).
Zuvor betrachten wir den Fehler der Projektion blockweise. Dazu zeigen wir zuerst, dass der Projektionsfehler sich durch die blockweisen Fehler darstellen lässt. Danach spalten wir den blockweisen Fehler in Anteile bezüglich der Zeilenbasis und bezüglich der Spaltenbasis auf.

Lemma 4.3.8
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $\overline{W} = (\overline{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ orthogonale Clusterbasen und $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Dann gilt sowohl für die Spektralnorm als auch für die Frobeniusnorm

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|^2 \leq \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|^2,$$

wobei im Falle der Frobeniusnorm Gleichheit gilt.

Beweis: Nach (3.15) gilt $\dot{\bigcup}_{b\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \mathbf{b} = \mathcal{I}\times\mathcal{J}$. Damit folgt

$$\begin{aligned}
X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X] &= X - \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{\overline{V}_t} X \Pi_{\overline{W}_s} + \sum_{(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} \\
&= \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} - \Pi_{\overline{V}_t} X \Pi_{\overline{W}_s} + \sum_{(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} - \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} \\
&= \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} - \Pi_{\overline{V}_t} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} \Pi_{\overline{W}_s} \\
&= \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}.
\end{aligned}$$

Weil die Beschriftungen $\mathbf{b}$ der Blöcke $b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ eine Blockpartition von $\mathcal{I}\times\mathcal{J}$ bilden, folgt die Aussage nach Lemma 2.3.10. Dabei gilt für die Frobeniusnorm Gleichheit. ∎

Das Theorem 4.3.9 gibt uns die Möglichkeit, aus den Fehlern in den einzelnen Blöcken eine globale Fehlerschranke zu berechnen. Dabei trennen wir den Fehler wie in Theorem 4.3.7 in einen Anteil bezüglich der Zeilenbasis und einen Anteil bezüglich der Spaltenbasis.

Theorem 4.3.9
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum und $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $\overline{W} = (\overline{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ orthogonale Clusterbasen. Weiter sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Dann gilt für die Spektralnorm die Fehlerschranke

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_S^2 \leq \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_S^2 + \|X^T_{|\mathbf{b}} - \Pi_{\overline{W}_s} X^T_{|\mathbf{b}}\|_S^2.$$

Für die Frobeniusnorm gelten die untere und obere Fehlerschranken

$$\begin{aligned}\sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} &\max\left\{\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2,\ \|X^T_{|\mathbf{b}} - \Pi_{\overline{W}_s} X^T_{|\mathbf{b}}\|_F^2\right\}\\ &\leq \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W}}[X]\|_F^2\\ &\leq \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 + \|X^T_{|\mathbf{b}} - \Pi_{\overline{W}_s} X^T_{|\mathbf{b}}\|_F^2.\end{aligned}$$

Beweis: Wir nutzen Lemma 4.3.8 und untersuchen die Terme der rechten Seite genauer. Zuerst betrachten wir die Spektralnorm. Für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $x \in \mathbb{R}^\mathcal{J}$ gilt

$$\|(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s})x\|_2^2 = \|(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}})x + \Pi_{\overline{V}_t}(X_{|\mathbf{b}} - X_{|\mathbf{b}}\Pi_{\overline{W}_s})x\|_2^2.$$

Nach Beispiel 2.3.18 ist $\Pi_{\overline{V}_t}$ eine orthogonale Projektion bezüglich des euklidischen Skalarprodukt. Wegen $y := \Pi_{\overline{V}_t}(X_{|\mathbf{b}} - X_{|\mathbf{b}}\Pi_{\overline{W}_s})x \in \text{Bild}(\Pi_{\overline{V}_t})$ sind nach Lemma 2.1.7 y und $(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}})x$ orthogonal zueinander und es gilt

$$\|(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s})x\|_2^2 = \|(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}})x\|_2^2 + \|\Pi_{\overline{V}_t}(X_{|\mathbf{b}} - X_{|\mathbf{b}}\Pi_{\overline{W}_s})x\|_2^2.$$

Weil $\Pi_{\overline{V}_t}$ eine orthogonale Projektion ist, gilt nach Korollar 2.1.8

$$\|(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s})x\|_2^2 \leq \|(X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}})x\|_2^2 + \|(X_{|\mathbf{b}} - X_{|\mathbf{b}}\Pi_{\overline{W}_s})x\|_2.$$

Mit Lemma 2.3.9 überträgt sich die Ungleichung auf die Spektralnorm und wir erhalten

$$\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_S^2 \leq \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_S^2 + \|X_{|\mathbf{b}} - X_{|\mathbf{b}}\Pi_{\overline{W}_s}\|_S^2.$$

Da $\Pi_{\overline{W}_s}$ eine orthogonale Projektion ist und die Spektralnorm nach Lemma 2.3.13 unter Adjungieren invariant ist, folgt

$$\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_S^2 \leq \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_S^2 + \|X^T_{|\mathbf{b}} - \Pi_{\overline{W}_s} X^T_{|\mathbf{b}}\|_S^2.$$

Als Zweites betrachten wir die Frobeniusnorm. Es gilt

$$\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_F^2 = \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} + \Pi_{\overline{V}_t} X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_F^2.$$

Die Abbildung $\Pi_{\overline{V}_t}$ von $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ nach $\mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ ist eine orthogonale Projektion bezüglich des Frobenius-Skalarprodukts (siehe Beispiel 2.3.18). Dann folgt mit Lemma 2.1.7, dass $X_{|\mathbf{b}} -$

$\Pi_{\overline{V}_t} X_{|\mathbf{b}}$ und $\Pi_{\overline{V}_t} X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}$ orthogonal bezüglich des Frobenius-Skalarprodukts sind. Also gilt

$$\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_F^2 \;=\; \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 \;+\; \|\Pi_{\overline{V}_t}(X_{|\mathbf{b}} - X_{|\mathbf{b}} \Pi_{\overline{W}_s})\|_F^2. \tag{4.10}$$

Da $\Pi_{\overline{V}_t}$ eine orthogonale Projektion ist, folgt mit Korollar 2.1.8

$$\begin{aligned}
\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_F^2 &= \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 \;+\; \|\Pi_{\overline{V}_t}(X_{|\mathbf{b}} - X_{|\mathbf{b}} \Pi_{\overline{W}_s})\|_F^2 \\
&\leq \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 \;+\; \|X_{|\mathbf{b}} - X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_F^2 \\
&= \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 \;+\; \|X_{|\mathbf{b}}^T - \Pi_{\overline{W}_s} X_{|\mathbf{b}}^T\|_F^2.
\end{aligned}$$

Die letzte Gleichheit gilt, weil die Frobeniusnorm unter Adjungieren invariant bleibt (siehe Lemma 2.3.13). Aus der Gleichheit in (4.10) folgt die untere Schranke

$$\begin{aligned}
\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 &\leq \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}}\|_F^2 \;+\; \|\Pi_{\overline{V}_t}(X_{|\mathbf{b}} - X_{|\mathbf{b}} \Pi_{\overline{W}_s})\|_F^2 \\
&= \|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} \Pi_{\overline{W}_s}\|_F^2.
\end{aligned}$$

Analog ist $\|X_{|\mathbf{b}}^T - \Pi_{\overline{W}_s} X_{|\mathbf{b}}^T\|_F^2$ eine untere Schranke für den Fehler im Block b. Mit Lemma 4.3.8 folgen dann die Abschätzungen in der Behauptung. ∎

Falls wir durch unsere Konstruktion sicherstellen, dass der Fehler in jedem Block beschränkt ist, liefert Theorem 4.3.9 eine globale Fehlerschranke. Als Nächstes folgern wir aus Theorem 4.3.5 eine entsprechende Aussage für den blockweisen Fehler und die blockweisen Fehlerterme aus Definition 4.3.2.

Lemma 4.3.10
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine orthogonale Clusterbasis und $\mathcal{T}_\mathcal{J}$ ein Clusterbaum. Dann gilt für alle $t, t' \in \mathcal{T}_\mathcal{I}$ mit $t' \neq t$ und $s \in \mathcal{T}_\mathcal{J}$

$$X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} \;=\; \sum_{\check{t}\in\operatorname{desc}(t)} D(X, \overline{V})_{\check{t},s}.$$

und $D(X, \overline{V})_{t',s}^T D(X, \overline{V})_{t,s} = 0$.

Beweis: Es seien $t, t' \in \mathcal{T}_\mathcal{I}$ mit $t' \neq t$ und $s \in \mathcal{T}_\mathcal{I}$. Dann gilt mit Lemma 4.3.4 (4.6)

$$X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} \;=\; (I_\mathbf{t} - \Pi_{\overline{V}_t}) X_{|\mathbf{t}\times\mathbf{s}} \;=\; \sum_{\check{t}\in\operatorname{desc}(t)} \Pi_{D,\overline{V},t} X_{|\mathbf{t}\times\mathbf{s}} \;=\; \sum_{\check{t}\in\operatorname{desc}(t)} D(X, \overline{V})_{t,s}$$

und mit (4.7)

$$D(X, \overline{V})_{t',s}^T D(X, \overline{V})_{t,s} \;=\; X_{|\mathbf{t}'\times\mathbf{s}}^T \Pi_{D,\overline{V},t'}^T \Pi_{D,\overline{V},t} X_{|\mathbf{t}\times\mathbf{s}} \;=\; X_{|\mathbf{t}'\times\mathbf{s}}^T 0 X_{|\mathbf{t}\times\mathbf{s}} \;=\; 0.$$

∎

Mit Hilfe von Lemma 4.3.10 können wir in Lemma 4.3.11 den blockweisen Fehler durch die Fehlerterme aus Definition 4.3.2 abschätzen.

Lemma 4.3.11
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine orthogonale Clusterbasis. Dann gilt für alle $t \in \mathcal{T}_\mathcal{I}$, $s \in \operatorname{row}(t)$

$$\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}}\|^2 \leq \sum_{\check{t}\in\operatorname{desc}(t)} \|D(X,\overline{V})_{\check{t},s}\|^2,$$

wobei für die Frobeniusnorm Gleichheit gilt.

Beweis: Es sei $t \in \mathcal{T}_\mathcal{I}$ und $s \in \operatorname{row}(t)$. Nach Lemma 4.3.10 gilt

$$X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} = \sum_{\check{t}\in\operatorname{desc}(t)} D(X,\overline{V})_{\check{t},s}.$$

und zusammen mit Lemma 2.3.21 folgt $\langle D(X,\overline{V})_{t_1,s}x, D(X,\overline{V})_{t_2,s}x\rangle = 0$ für alle $t_1, t_2 \in \operatorname{desc}(t)$ und $x \in \mathbb{R}^\mathcal{J}$. Mit Lemma 2.1.2 gilt

$$\|X_{|\mathbf{b}} - \Pi_{\overline{V}_t} X_{|\mathbf{b}} x\|_2^2 = \|\sum_{t\in\operatorname{desc}(t)} D(X,\overline{V})_{\check{t},s}x\|_2^2 = \sum_{t\in\operatorname{desc}(t)} \|D(X,\overline{V})_{\check{t},s}x\|_2^2.$$

Mit Lemma 2.3.9 erhalten wir daraus die Abschätzungen für die Matrixnormen, wobei wegen der Gleichheit für die Frobeniusnorm Gleichheit gilt. ∎

Bevor wir mit der Konstruktion der adaptiven Clusterbasis fortfahren, gehen wir kurz auf zwei Eigenschaften unserer Fehlerabschätzungen ein.

Bemerkung 4.3.12
Die unteren Schranken bei den Abschätzungen des Projektionsfehlers in der Frobeniusnorm zeigen, dass die oberen Schranken in Theorem 4.3.7 und Theorem 4.3.9 den Fehler maximal um den Faktor 2 überschätzen.

Bemerkung 4.3.13
Sowohl in Theorem 4.3.9 als auch in Theorem 4.3.7 erhalten wir für die Projektionsfehler eine Fehlerschranke, die sich als euklidische Norm von Teilfehlern darstellt. Beim simplen Ansatz durch die Dreiecksungleichung würden wir eine Fehlerschranke erhalten, die sich als Betragssumme von Teilfehlern darstellt. Da im $\mathbb{R}^n$ für die Betragssumme $\|.\|_1$ und die euklidische Norm die Äquivalenz $\|.\|_1 \leq \|.\|_2 \leq \sqrt{n}\|.\|_1$ gilt, sparen wir bei gleichmäßiger Fehlerverteilung einen Faktor $\sqrt{\#\mathcal{T}_\mathcal{I}}$ bzw. $\sqrt{\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}}$ bei der Fehlerschranke.

4.4 Adaptive Clusterbasen

In diesem Abschnitt stellen wir die Konstruktion adaptiver Clusterbasen für $\mathcal{H}^2$-Matrizen vor, wie sie in [12, Abschnitt 6.6] beschrieben ist. Diese sollen erstens eine vorgegebene Fehlerschranke einhalten und zweitens effizient berechenbar sein. Der erste Ansatz folgt aus der clusterweisen Fehlerabschätzung und der zweite aus der Fehlerabschätzung für

Blöcke.
Wir müssen bei der Konstruktion der adaptiven Clusterbasen nach Theorem 4.3.7 dafür sorgen, dass die Normen der Fehlerterme $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t$ und $D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s$ hinreichend klein sind. Weil beide Terme die gleiche Struktur besitzen, konzentrieren wir uns auf die Zeilenbasis $\overline{V}$. Wir schränken unsere Betrachtung auf die Konstruktion adaptiver Clusterbasen für eine $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ ein.
Neben der Einhaltung einer gegebenen Fehlerschranke soll der Algorithmus lineare Komplexität bezüglich der Anzahl der Freiheitsgrade haben. Um dies zu erreichen, stellen wir sicher, dass der Aufwand in den Blättern $t \in \mathcal{L}_{\mathcal{I}}$ in $\mathcal{O}(\#\mathbf{t})$ und in den anderen Clustern $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$ in $\mathcal{O}(1)$ liegt (siehe Lemma 3.2.11 und Korollar 3.2.12).
Zuerst wollen wir die Idee erläutern. Es sei $t \in \mathcal{T}_{\mathcal{I}}$. Wir nehmen an, dass für die Nachfahren $\check{t} \in \operatorname{desc}(t)\backslash\{t\}$ von t die neue orthogonale Clusterbasis $\overline{V}$ bereits konstruiert ist. Dann ist die Matrix $\check{V}_t := \check{V}(\overline{V})_t$ definiert und für den Fehlerterm aus Definition 4.3.2 in t gilt mit $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t$

$$D_t \;:=\; D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t \;=\; \left(I_{\mathbf{t}} - \Pi_{\overline{V}_t}\right) \Pi_{\check{V}_t} X_t.$$

Der erste Ansatz ist die Berechnung von $\overline{V}_t$ durch eine gekürzte Singulärwertzerlegung der Matrix $\Pi_{\check{V}_t} X_t$ (siehe Algorithmus 2.4.3). Weil die Matrix $\Pi_{\check{V}_t} X_t$ Nicht-null-Einträge in $\#\mathbf{t}$ Zeilen und $\sum_{s\in\overline{\operatorname{row}}(t)} \#\mathbf{s}$ Spalten hat, wäre der Aufwand dafür zu groß (siehe Lemma 2.4.9).
Aufgrund der geschachtelten Struktur sind wir vor allem an den Transfermatrizen $E_{\check{t}}$, $\check{t} \in \operatorname{chil}(t)$ interessiert. Weil $\overline{V}$ orthogonal konstruiert werden soll, werden diese in der Matrix $\acute{V}_t := \acute{V}(\overline{V})_t$ zusammengefasst (siehe Definition 4.2.3 und Bemerkung 4.2.5). Aus $\overline{V}_t = \check{V}_t \acute{V}_t$ (siehe Lemma 4.2.12 und (4.8)) folgt

$$D_t \;=\; (\Pi_{\check{V}_t} - \Pi_{\overline{V}_t}) X_t \;=\; (\check{V}_t \check{V}_t^T - \check{V}_t \acute{V}_t \acute{V}_t^T \check{V}_t^T) X_t \;=\; \check{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \check{V}_t^T X_t. \tag{4.11}$$

Weil die Matrix $\check{V}_t$ orthogonal ist, können wir die Norm des Fehlerterms mit Lemma 2.3.16 umschreiben zu

$$\|D_t\| \;=\; \|\check{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \check{V}_t^T X_t\| \;=\; \| \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \check{V}_t^T X_t\|. \tag{4.12}$$

Damit hat sich die Aufgabe auf das Berechnen von $\acute{V}_t$ durch das Anwenden von Algorithmus 2.4.3 auf die Matrix $\check{V}_t^T X_t$ reduziert. Allerdings hat die Matrix $\check{V}_t^T X_t$ noch immer $\sum_{s\in\overline{\operatorname{row}}(t)} \#\mathbf{s}$ Spalten, was eine Berechnung zu aufwändig machen würde.
Weil wir die $\mathcal{H}^2$-Matrix H betrachten, ist $\operatorname{Bild}(X_t) \subseteq \operatorname{Bild}(V_t)$ und es existiert ein Y_t mit $X_t = V_t Y_t$ (siehe Lemma 4.1.12). Wir nehmen an, dass wir zusätzlich eine QR-Zerlegung $P_t Z_t^T = Y_t^T$ hätten. Dann können wir aufgrund der Orthogonalität von P_t die Gleichung (4.12) weiter reduzieren zu

$$\begin{aligned} \|D_t\| \;&=\; \| \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \check{V}_t^T X_t\| \;=\; \| \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \check{V}_t^T V_t Z_t P_t^*\| \\ &=\; \| \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \check{V}_t^T V_t Z_t\|. \end{aligned} \tag{4.13}$$

Nach Lemma 4.2.4 gilt $\breve{V}_t^T V_t = \widehat{V}(V,\overline{V})_t$, wobei sich $\widehat{V}(V,\overline{V})_t$ effizient mit Hilfe der Basiswechsel $C(V,\overline{V})$ aufstellen lässt (siehe Definition 4.2.3). Die Basiswechsel wiederum können wir mit Hilfe der rekursiven Darstellung aus Lemma 4.2.17 effizient berechnen. Dabei ist es wichtig, dass die Rekursion für die Basiswechsel, genauso wie die Konstruktion der neuen Clusterbasis, von den Blättern zur Wurzel verläuft. Damit lässt sich (4.13) umformen zu

$$\|D_t\| \;=\; \| \left(I_{\mathbf{t}} - \Pi_{\breve{V}_t} \right) \breve{V}_t^T V_t Z_t \| \;=\; \| \left(I_{\mathbf{t}} - \Pi_{\breve{V}_t} \right) \widehat{V}(V,\overline{V})_t Z_t \|. \tag{4.14}$$

Die Matrix $\widehat{V}(V,\overline{V})_t Z_t$ hat höchstens $\sum_{\check{t}\in\mathrm{chil}(t)} \#\kappa_{\check{t}}$ Zeilen und höchstens $\#\kappa_t$ Spalten. Somit können wir den Aufwand für den Aufruf von Algorithmus 2.4.3 für die Matrix $\widehat{V}(V,\overline{V})_t Z_t$ durch Größen des lokalen Rangs beschränken. Gleichzeitig können wir die Norm des Fehlerterms durch Algorithmus 2.4.3 steuern.
Dies bildet den Ansatz, den wir zur Konstruktion von $\overline{V}$ verfolgen. Weil $\widehat{V}(V,\overline{V})_t$ und Z_t jeweils "klein" sind, kann das Produkt effizient berechnet werden. Somit besteht die Aufgabe vor allem darin, die Matrizen $\widehat{V}(V,\overline{V})_t$ und Z_t aufzustellen. Als Erstes wenden wir uns den Matrizen Z_t zu und definieren die sogenannten (reduzierten) Gewichte.

Definition 4.4.1 (Gewichte für Clusterbasen)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$. Dann definieren die Matrizen

$$Y_t \;:=\; Y(H,V)_t \;:=\; \sum_{\hat{t}\in\mathrm{pred}(t)} \sum_{s\in\mathrm{row}(\hat{t})} E_{t,\hat{t}} S_{(\hat{t},s)} W_s^T$$

für alle $t \in \mathcal{T}_{\mathcal{I}}$ die *totalen Gewichte* zur Zeilenclusterbasis.
Eine Matrix $Z_t := Z(H,V)_t$ heißt *reduziertes Gewicht*, falls für alle $t \in \mathcal{T}_{\mathcal{I}}$ eine orthogonale Matrix P_t mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t = V_t Z_t P_t^T$ existiert. Die Spaltenindexmenge von Z_t bezeichnen wir typischerweise mit ζ_t.

Den Namen *reduziertes Gewicht* haben wir gewählt, weil Z_t typischerweise deutlich weniger Zeilen als das totale Gewicht Y_t besitzt.
Um die reduzierten Gewichte effizient berechnen zu können, wollen wir eine rekursive Darstellung angeben. Dazu benötigen wir eine rekursive Darstellung der totalen Clusterbasis.

Lemma 4.4.2
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix. Dann gilt für die totale Clusterbasis

$$X_t \;:=\; X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t \;=\; \begin{cases} \sum_{s\in\mathrm{row}(t)} V_t S_{(t,s)} W_s^T & \text{, falls } t = r_{\mathcal{I}} \\ \sum_{s\in\mathrm{row}(t)} V_t S_{(t,s)} W_s^T + X_{\acute{t}|\mathbf{t}} & \text{, falls } t \neq r_{\mathcal{I}},\ \acute{t} := \mathrm{par}(t) \end{cases}$$

für alle $t \in \mathcal{T}_{\mathcal{I}}$.

Beweis: Es sei $t = r_{\mathcal{I}}$. Dann gilt $\overline{\mathrm{row}}(t) = \mathrm{row}(t)$. Daraus folgt

$$X_t \;=\; \Pi_{\mathbf{t}} X \sum_{s\in\overline{\mathrm{row}}(t)} \Pi_{\mathbf{s}} \;=\; \Pi_{\mathbf{t}} X \sum_{s\in\mathrm{row}(t)} \Pi_{\mathbf{s}} \;=\; \sum_{s\in\mathrm{row}(t)} V_t S_{(t,s)} W_s^T.$$

Es sei $t \in \mathcal{T}_\mathcal{I} \backslash \{r_\mathcal{I}\}$ mit $\acute{t} := \operatorname{par}(t)$. Dann gilt

$$\begin{aligned} X_t &= \Pi_\mathbf{t} X \sum_{s \in \overline{\operatorname{row}}(t)} \Pi_\mathbf{s} = \Pi_\mathbf{t} X \sum_{s \in \operatorname{row}(t)} \Pi_\mathbf{s} + \Pi_\mathbf{t} \Pi_{\acute{\mathbf{t}}} X \sum_{s \in \overline{\operatorname{row}}(\acute{t})} \Pi_\mathbf{s} \\ &= \sum_{s \in \operatorname{row}(t)} V_t S_{(t,s)} W_s^T + X_{\acute{t}|\mathbf{t}}. \end{aligned}$$

■

Anschaulich besteht die totale Clusterbasis $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t$ für $t \in \mathcal{T}_\mathcal{I} \backslash \{r_\mathcal{I}\}$ aus dem Anteil der $\mathcal{H}^2$-Matrix H, der direkt durch V_t beschrieben wird, und einem Anteil, der bereits vom Elter $\acute{t} := \operatorname{par}(t)$ dargestellt wird (siehe Abbildung 4.1). Der erste Teil wird durch die Summe dargestellt. Der zweite Teil ist die Einschränkung der totalen Clusterbasis des Elters auf den aktuellen Cluster t. Dieser zweite Teil muss aufgrund der geschachtelten Clusterbasis auch durch V_t darstellbar sein.
Wie bereits erwähnt, sind die Matrizen X_t zu groß, um sie effizient berechnen oder speichern zu können. Deshalb beweisen wir in Lemma 4.4.3 eine rekursive Darstellung für reduzierte Gewichte, welche eine effiziente Berechnung ermöglicht.

Lemma 4.4.3
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix. Weiter seien für die Clusterbasis W orthogonale Gewichte $(O_s)_{s\in\mathcal{T}_\mathcal{J}}$ mit zugehörigen orthogonalen Matrizen $(P_s)_{s\in\mathcal{T}_\mathcal{J}}$ gegeben. Dann werden durch die QR-Zerlegung von $\widehat{Z}_t^T$ rekursiv von der Wurzel zu den Blättern

$$Z_t \widehat{P}_t^T = \widehat{Z}_t := \begin{cases} \begin{pmatrix} S_{(t,s_1)} O_{s_1}^T & \cdots & S_{(t,s_\sigma)} O_{s_\sigma}^T \end{pmatrix} & , \text{falls } t = r_\mathcal{I} \\ \begin{pmatrix} S_{(t,s_1)} O_{s_1}^T & \cdots & S_{(t,s_\sigma)} O_{s_\sigma}^T & E_t Z_{\acute{t}} \end{pmatrix} & , \text{falls } t \neq r_\mathcal{I},\ \acute{t} := \operatorname{par}(t) \end{cases},$$

für alle $t \in \mathcal{T}_\mathcal{I}$ reduzierte Gewichte $Z(H,V)_t := Z_t$ definiert.

Beweis:[vergleiche [12, Abschnitt 6.6]]
Es ist zu zeigen, dass eine orthogonale Matrix P_t mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t = V_t Z_t P_t^T$ für alle $t \in \mathcal{T}_\mathcal{I}$ existiert. Wir zeigen die Aussage per Induktion über $\#\operatorname{pred}(t)$.
Es sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{pred}(t) = 1$. Dann ist $t = r_\mathcal{I}$ und mit der Darstellung der totalen Clusterbasis aus Lemma 4.4.2 folgt

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t &= \sum_{s \in \operatorname{row}(t)} V_t S_{(t,s)} W_s^T = V_t \sum_{s \in \operatorname{row}(t)} S_{(t,s)} O_s^T P_s^T \\ &= V_t \underbrace{\begin{pmatrix} S_{(t,s_1)} O_{s_1}^T & \cdots & S_{(t,s_\sigma)} O_{s_\sigma}^T \end{pmatrix}}_{:=\widehat{Z}_t} \underbrace{\begin{pmatrix} P_{s_1} & \cdots & P_{s_\sigma} \end{pmatrix}^T}_{=:\widehat{P}_t^T} \\ &= V_t \widehat{Z}_t \widetilde{P}_t^T = V_t Z_t \widehat{P}_t^T \widetilde{P}_t^T =: V_t Z_t P_t^T. \end{aligned}$$

Die Matrix $\widetilde{P}_t$ ist nach Bemerkung 2.3.15 orthogonal. Somit ist auch $P_t = \widetilde{P}_t \widehat{P}_t$ orthogonal und Z_t definiert ein reduziertes Gewicht.

Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{pred}(t) \leq n$ gilt. Weiter sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{pred}(t) = n+1 > 1$. Dann gilt $t \neq r_\mathcal{I}$ und es existiert $\acute{t} := \operatorname{par}(t)$. Es gilt $\#\operatorname{pred}(\acute{t}) = n$ und somit ist $Z_{\acute{t}}$ ein reduziertes Gewicht, für das eine orthogonale Matrix $P_{\acute{t}}$ mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_{\acute{t}} = V_{\acute{t}} Z_{\acute{t}} P_{\acute{t}}^T$ existiert. Mit der rekursiven Darstellung der totalen Clusterbasis aus Lemma 4.4.2 folgt

$$\begin{aligned}
X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t &= \sum_{s\in\operatorname{row}(t)} V_t S_{(t,s)} W_s^T + X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_{\acute{t}|\mathbf{t}} \\
&= V_t \sum_{s\in\operatorname{row}(t)} S_{(t,s)} O_s^T P_s^T + V_{\acute{t}|\mathbf{t}} Z_{\acute{t}} P_{\acute{t}}^T \\
&= V_t \begin{pmatrix} S_{(t,s_1)} O_{s_1}^T & \cdots & S_{(t,s_\sigma)} O_{s_\sigma}^T & E_t Z_{\acute{t}} \end{pmatrix} \begin{pmatrix} P_{s_1} & \cdots & P_{s_\sigma} & P_{\acute{t}} \end{pmatrix}^T \\
&=: V_t \widehat{Z}_t \widetilde{P}_t^T = V_t Z_t \widehat{P}_t^T \widetilde{P}_t^T =: V_t Z_t P_t^T.
\end{aligned}$$

Wie oben ist die Matrix $\widetilde{P}_t$ nach Lemma 2.3.15 orthogonal. Also ist auch $P_t = \widetilde{P}_t \widehat{P}_t$ orthogonal und Z_t ein reduziertes Gewicht. ∎

Bemerkung 4.4.4
Nach Definition 4.4.1 bilden die Matrizen $\widehat{Z}_t$ reduzierte Gewichte. Allerdings würde die Anzahl der Spalten ohne zusätzliche QR-Zerlegung von Level zu Level wachsen. Dies würde zu einem zusätzlichen logarithmischen Faktor im Aufwand des Algorithmus führen.

Weil wir später eine Verallgemeinerung der reduzierten Gewichte und einen zugehörigen Algorithmus zur Berechnung dieser definieren, verzichten wir an dieser Stelle auf die Angabe eines Algorithmus zur Berechnung der reduzierten Gewichte.
Mit den reduzierten Gewichten können wir eine rekursive Formel für die adaptive Clusterbasis angeben. Dazu nutzen wir eine allgemeinere Formulierung für Gewichte, weil wir für die blockweise Fehlerabschätzung und beim Niedrigrangupdate modifizierte reduzierte Gewichte nutzen.

Definition 4.4.5
Es sei $(V)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis mit Rangverteilung κ. Dann bezeichnen wir eine Familie von Matrizen $(Z_t)_{t\in\mathcal{T}_\mathcal{I}}$ als Gewichte zu V mit Rangverteilung $\zeta = (\zeta_t)_{t\in\mathcal{T}_\mathcal{I}}$, falls $Z_t \in \mathbb{R}^{\kappa_t \times \zeta_t}$ für alle $t \in \mathcal{T}_\mathcal{I}$ gilt.

Bemerkung 4.4.6
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix. Dann sind die reduzierten Gewichte $Z(H, V)$ stets auch Gewichte zu V.

Als Nächstes zeigen wir, wie sich adaptive Clusterbasen für gegebene Gewichte berechnen lassen. Dazu führen wir eine rekursive Definition für die adaptive Clusterbasis ein, welche eng verwandt mit der rekursiven Berechnungsvorschrift für orthogonale Matrizen aus Definition 4.2.6 ist.

Definition 4.4.7 (Adaptive Clusterbasis)
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $(Z_t)_{t\in\mathcal{T}_\mathcal{I}}$ Gewichte zu V mit Rangverteilung ζ, *opt* eine Fehlerart und $\epsilon := (\epsilon_t)_{t\in\mathcal{T}_\mathcal{I}} \in \mathbb{R}^{\mathcal{T}_\mathcal{I}}_{\geq 0}$ eine Familie von Fehlerschranken. Dann definieren wir rekursiv von den Blättern aus für alle $t \in \mathcal{T}_\mathcal{I}$

$$\overline{V}(V,Z)_t \;:=\; \overline{V}(V,Z,opt,\epsilon)_t \;:=\; \check{V}(\overline{V}(V,Z))_t\widehat{Q}_t \in \mathbb{R}^{\mathcal{I}\times\overline{\kappa}_t},$$

wobei die orthogonale Matrix $\widehat{Q}_t \in \mathbb{R}^{\ell\times\overline{\kappa}_t}$, $\ell := \bigcup_{\check{t}\in\mathrm{chil}(t)} \kappa_{\check{t}}$, durch Aufruf des Algorithmus 2.4.3 für die Matrix $\widehat{V}(V,\overline{V}(V,Z))_t Z_t$ mit Fehlerart *opt* und der Fehlerschranke ϵ_t berechnet wird. Weiter sei für alle $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ und alle $\check{t} \in \mathrm{chil}(t)$

$$\overline{E}(V,Z)_t \;:=\; \overline{E}(V,Z,opt,\epsilon)_{\check{t}} \;:=\; \widehat{Q}_{t|\overline{\kappa}_{\check{t}}\times\overline{\kappa}_t}.$$

Bemerkung 4.4.8
Für ein Blatt $t \in \mathcal{L}_\mathcal{I}$ sind $\widehat{V}(V,\overline{V}(V,Z))_t = V_t$ und $\check{V}(\overline{V}(V,Z))_t = I$. Also sind für die Blätter $t \in \mathcal{T}_\mathcal{I}$ die Matrizen $\widehat{Q}_t$, $\overline{V}(V,Z)_t$ und $\overline{E}(V,Z)_t$ wohldefiniert.
Für $t \in \mathcal{T}_\mathcal{I}$ hängen die Matrizen $\widehat{V}(V,\overline{V}(V,Z))_t$ und $\check{V}(\overline{V}(V,Z))_t\ \overline{V}$ nur in den Kindern $\check{t} \in \mathrm{chil}(t)$ von der neuen Clusterbasis ab. Damit sind $\widehat{Q}_t$, $\overline{V}(V,Z)_t$ und $\overline{E}(V,Z)_t$ für $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ durch die Rekursion definiert.

Wir verwenden aufgrund der ähnlichen Definition für die orthogonale Clusterbasis $\overline{V}(V)$ und die adaptive Clusterbasis $\overline{V}(V,Z)$ sehr ähnliche Bezeichnungen. Anstatt der QR-Zerlegung von $\widehat{V}(V,\overline{V}(V))_t$ verwenden wir für die adaptive Clusterbasis die gekürzte Singulärwertzerlegung von $\widehat{V}(V,\overline{V}(V,Z))_t$. Im nächsten Lemma fassen wir einige wichtige Aussagen zu den Matrizen aus Definition 4.4.17.

Lemma 4.4.9
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ die Rangverteilung von V, $(Z_t)_{t\in\mathcal{T}_\mathcal{I}}$ Gewichte zu V mit Rangverteilung ζ, *opt* eine Fehlerart und $\epsilon := (\epsilon_t)_{t\in\mathcal{T}_\mathcal{I}} \in \mathbb{R}^{\mathcal{T}_\mathcal{I}}_{\geq 0}$ eine Familie von Fehlerschranken.

(i) Dann ist $\overline{V} := ((\overline{V}(V,Z,opt,\epsilon)_t)_{t\in\mathcal{T}_\mathcal{I}}, (\overline{E}(V,Z,opt,\epsilon)_t)_{t\in\mathcal{T}_\mathcal{I}})$ mit den Matrizen aus Definition 4.4.7 eine orthogonale Clusterbasis mit Rangverteilung $(\overline{\kappa}_t)_{t\in\mathcal{T}_\mathcal{I}}$.

(ii) Für die neue Rangverteilung gilt $\#\overline{\kappa}_t \leq \#\kappa_t$ für alle $t \in \mathcal{T}_\mathcal{I}$.

(iii) Es gilt für alle $t \in \mathcal{T}_\mathcal{I}$, dass der Projektionsfehler

$$(I_{\mathbf{t}} - \Pi_{\acute{V}(\overline{V})_t})\widehat{V}(V,\overline{V})_t Z_t$$

in der vorgegebenen Fehlerart *opt* kleiner als ϵ_t ist.

(iv) Für den Basiswechsel gilt für $t \in \mathcal{T}_\mathcal{I}$

$$C(V,\overline{V})_t \;=\; \acute{V}(\overline{V})_t^T\widehat{V}(V,\overline{V})_t \;=\; \widehat{Q}_t^T\widehat{V}(V,\overline{V})_t.$$

Beweis: "(i)" Als Erstes zeigen wir die geschachtelte Darstellung unter Verwendung der Notationen $\overline{V}_t := \overline{V}(V, Z, opt, \epsilon)_t$ und $\overline{E}_t := \overline{E}(V, Z, opt, \epsilon)_t$ für alle $t \in \mathcal{T}_\mathcal{I}$. Für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ gilt

$$\overline{V}_t \;=\; \breve{V}(\overline{V})_t \widehat{Q}_t \;=\; \begin{pmatrix} \overline{V}_{t_1} & \cdots & \overline{V}_{t_\tau} \end{pmatrix} \begin{pmatrix} \overline{E}_{t_1} \\ \vdots \\ \overline{E}_{t_\tau} \end{pmatrix} \;=\; \sum_{\breve{t} \in \operatorname{chil}(t)} \overline{V}_{t_1} \overline{E}_{t_1}.$$

Also ist $((\overline{V}_t)_{t \in \mathcal{T}_\mathcal{I}}, (\overline{E}_t)_{t \in \mathcal{T}_\mathcal{I}})$ eine Clusterbasis.
Als Nächstes beweisen wir die Orthogonalität der Clusterbasis. Für alle $t \in \mathcal{L}_\mathcal{I}$ gilt, dass $\overline{V}_t = \breve{V}(\overline{V})_t \widehat{Q}_t = \widehat{Q}_t$ orthogonal ist.
Für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ ist für $\operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\}$

$$\begin{pmatrix} \overline{E}_{t_1} \\ \vdots \\ \overline{E}_{t_\tau} \end{pmatrix} \;=\; \widehat{Q}_t$$

orthogonal. Nach dem Kriterium aus Lemma 4.2.2 ist $\overline{V}$ somit eine orthogonale Clusterbasis. Insbesondere ist nach Bemerkung 4.2.5 $\widehat{Q}_t = \acute{V}(\overline{V})_t$ für alle $t \in \mathcal{T}_\mathcal{I}$.
"(ii)" Es sei $t \in \mathcal{T}_\mathcal{I}$. Die neue Rangverteilung $\overline{\kappa}_t$ ist die Spaltenindexmenge von $\widehat{Q}_t$. Weil diese durch den Aufruf von Algorithmus 2.4.3 für die Matrix $\widehat{V}(V, \overline{V}(V, Z))_t Z_t$ entsteht, gilt $\#\overline{\kappa}_t \leq \operatorname{Rang}(\widehat{V}(V, \overline{V}(V, Z))_t Z_t) \leq \#\kappa_t$.
"(iii)" Diese Eigenschaft folgt aus der Fehlerabschätzung für den Algorithmus 2.4.3 in Lemma 2.4.9 und $\widehat{Q}_t = \acute{V}(\overline{V})_t$.
"(iv)" Es sei $t \in \mathcal{T}_\mathcal{I}$. Nach Lemma 4.2.12 gilt $\overline{V}_t = \breve{V}(\overline{V})_t \acute{V}(\overline{V})_t$ und nach Lemma 4.2.4 gilt $\breve{V}(\overline{V})_t^T V_t = \widehat{V}(V, \overline{V})_t$. Zusammen mit $\acute{V}(\overline{V})_t = \widehat{Q}_r$ erhalten wir

$$C(V, \overline{V})_t \;=\; \overline{V}_t^T V_t \;=\; \acute{V}(\overline{V})_t^T \breve{V}(\overline{V})_t^T V_t \;=\; \acute{V}(\overline{V})_t^T \widehat{V}(V, \overline{V})_t \;=\; \widehat{Q}_t^T \widehat{V}(V, \overline{V})_t.$$

■

Nach Theorem 4.3.7 müssen wir für die Schranke des Projektionsfehlers die Fehlerterme $D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \overline{V})_t$ beschränken. Auf der anderen Seite ist durch Lemma 4.4.9 der Term $(I_\mathbf{t} - \Pi_{\acute{V}(\overline{V})_t}) \widehat{V}(V, \overline{V})_t Z_t$ in der Norm beschränkt, falls $\overline{V}$ durch Definition 4.4.7 gegeben ist. Deshalb schätzen wir die Terme gegeneinander ab.

Lemma 4.4.10
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix und $Z_t := Z(H, V)_t$, $t \in \mathcal{T}_\mathcal{I}$, reduzierte Gewichte. Weiter sei $\overline{V}$ eine orthogonale Clusterbasis. Dann gilt für alle $t \in \mathcal{T}_\mathcal{I}$

$$\|D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \overline{V})_t\| \;=\; \|(I_\mathbf{t} - \Pi_{\acute{V}(\overline{V})_t}) \widehat{V}(V, \overline{V})_t Z_t\|$$

sowohl für die Spektralnorm als auch für die Frobeniusnorm.

Beweis: Es sei $t \in \mathcal{T}_\mathcal{I}$. Mit (4.11), $\breve{V}_t := \breve{V}(\overline{V})_t$, $\acute{V}_t := \acute{V}(\overline{V})_t$ und der zu dem Gewicht Z_t gehörigen orthogonalen Matrix P_t gilt

$$D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \overline{V})_t \;=\; \breve{V}_t(I_\mathbf{t} - \Pi_{\acute{V}_t})\breve{V}_t^T X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t \;=\; \breve{V}_t(I_\mathbf{t} - \Pi_{\acute{V}_t})\breve{V}_t^T V_t Z_t P_t^T.$$

Weil die Matrizen $\breve{V}_t$ und P_t orthogonal sind, folgt nach Lemma 2.3.16

$$\|D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \overline{V})_t\| \;=\; \|\breve{V}_t(I_\mathbf{t} - \Pi_{\acute{V}_t})\breve{V}_t^T V_t Z_t P_t^T\| \;=\; \|(I_\mathbf{t} - \Pi_{\acute{V}_t})\breve{V}_t^T V_t Z_t\|.$$

Nach Lemma 4.2.4 gilt $\breve{V}_t^T V_t = \widehat{V}(V, \overline{V})_t =: \widehat{V}_t$ und somit

$$\|D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \overline{V})_t\| \;=\; \|(I_\mathbf{t} - \Pi_{\acute{V}_t})\breve{V}_t^T V_t Z_t\| \;=\; \|(I_\mathbf{t} - \Pi_{\acute{V}_t})\widehat{V}(V, \overline{V})_t Z_t\|.$$

■

Für reduzierte Gewichte Z_t liefert die Definition 4.4.7 somit eine adaptive Clusterbasis, für die der Fehler nach den Lemmata 4.4.10 und 4.4.9 (iii) kontrollierbar ist. Als Nächstes definieren wir den Algorithmus 4.4.1 zur Berechnung dieser adaptiven Clusterbasis mit Hilfe gegebener Gewichte. Den Aufwand schätzen wir in Lemma 4.4.11 ab.

Lemma 4.4.11 (Aufwand für die adaptive Clusterbasis)
Es seien $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix und Z Gewichte für V. Weiter seien κ und ζ die Rangverteilungen zu V bzw. Z. Es sei $k_{\max} := \max_{t\in\mathcal{T}_\mathcal{I}} \max\{\#\kappa_t, \#\zeta_t\}$. Dann ist der Aufwand für Algorithmus 4.4.1 höchstens

$$(c_{apr} + 4)k_{\max}^2 \#\mathcal{I} + (c_{apr} + 6)k_{\max}^3 \#\mathcal{T}_\mathcal{I}.$$

Beweis:[vergleiche [12, Theorem 6.27]]
Es sei $t \in \mathcal{L}_\mathcal{I}$. Dann wird $V_t Z_t$ in weniger als $2\#\mathbf{t}\#\kappa_t\#\zeta_t \leq 2k_{\max}^2\#\mathbf{t}$ Operationen berechnet. Die Berechnung von $\overline{V}_t$ mit Algorithmus 2.4.3 benötigt höchstens

$$c_{apr}\#\mathbf{t}\#\zeta_t \min\{\#\mathbf{t}, \#\zeta_t\} \;\leq\; c_{apr}k_{\max}^2\#\mathbf{t}$$

Operationen (siehe Lemma 2.4.9). Wegen $\#\overline{\kappa}_t \leq \mathrm{Rang}(\widehat{Y}_t) \leq \mathrm{Rang}(V_t) \leq \#\kappa_t \leq k_{\max}$ wird der Basiswechsel in höchstens $2\#\overline{\kappa}\#\mathbf{t}\#\kappa \leq 2k_{\max}^2\#\mathbf{t}$ Operationen berechnet. Also ist der Aufwand für ein Blattcluster durch

$$2k_{\max}^2\#\mathbf{t} + c_{apr}k_{\max}^2\#\mathbf{t} + 2k_{\max}^2\#\mathbf{t} \;=\; (c_{apr} + 4)k_{\max}^2\#\mathbf{t}$$

beschränkt.
Es sei $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$. Dann wird zum Aufstellen von $\widehat{V}_t$ für alle $\breve{t} \in \mathrm{chil}(t)$ das Produkt $C_{\breve{t}}E_{\breve{t}}$ berechnet. Der Aufwand dafür ist durch

$$\sum_{\breve{t}\in\mathrm{chil}(t)} 2\#\overline{\kappa}_{\breve{t}}\#\kappa_{\breve{t}}\#\kappa_t \;\leq\; 2\sum_{\breve{t}\in\mathrm{chil}(t)} k_{\max}^3$$

Algorithmus 4.4.1 [siehe [12, Algorithm 29]] Die Funktion berechnet die adaptive Clusterbasis $\overline{V}$ zur Clusterbasis V, den Gewichten Z, der Fehlerart *opt* und der Familie von Fehlerschranken ϵ (siehe Definition 4.4.7). Dabei wird der Basiswechsel $C = C(V, \overline{V})$ mitberechnet.

function ADAPTIVE_CLUSTERBASIS(t, V, κ, Z, ζ, $\overline{V}$, $\overline{\kappa}$, C, ϵ, *opt*)
  if $\mathrm{chil}(t) = \emptyset$ **then**
    $\widehat{Y}_t \in \mathbb{R}^{\mathbf{t} \times \zeta_t}$
    $\widehat{Y}_t \leftarrow V_t Z_t$
    ADAPTIVE_ORTHOGONALE_MATRIX($\widehat{Y}_t$, ϵ_t, *opt*, $\overline{V}_t$, $\overline{\kappa}_t$) ▷ Algorithmus 2.4.3
    $C_t \leftarrow \overline{V}_t^T V_t$
  else
    for $\breve{t} \in \mathrm{chil}(t)$ **do**
      ADAPTIVE_CLUSTERBASIS($\breve{t}$, V, κ, Z, ζ, $\overline{V}$, $\overline{\kappa}$, C, ϵ, *opt*) ▷ bottom-up
    end for
    $\ell \leftarrow \dot{\bigcup}_{\breve{t} \in \mathrm{chil}(t)} \overline{\kappa}_{\breve{t}}$
    $\widehat{V}_t \in \mathbb{R}^{\ell \times \kappa_t}$
    for $\breve{t} \in \mathrm{chil}(t)$ **do**
      $\widehat{V}_{t|\overline{\kappa}_{\breve{t}} \times \kappa_t} \leftarrow C_{\breve{t}} E_{\breve{t}}$
    end for
    $\widehat{Y}_t \in \mathbb{R}^{\ell \times \zeta_t}$
    $\widehat{Y}_t \leftarrow \widehat{V}_t Z_t$
    ADAPTIVE_ORTHOGONALE_MATRIX($\widehat{Y}_t$, ϵ_t, *opt*, $\widehat{Q}_t$, $\overline{\kappa}_t$) ▷ Algorithmus 2.4.3
    for $\breve{t} \in \mathrm{chil}(t)$ **do**
      $\overline{E}_{\breve{t}} \leftarrow \widehat{Q}_{t|\overline{\kappa}_{\breve{t}} \times \overline{\kappa}_t}$
    end for
    $C_t \leftarrow \widehat{Q}_t^T \widehat{V}_t$
  end if
end function

beschränkt. $\widehat{V}_t$ hat höchstens

$$\#\ell \;=\; \sum_{\check{t}\in\operatorname{chil}(t)} \#\overline{\kappa}_{\check{t}} \;\leq\; \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}$$

Zeilen. Also benötigt das Aufstellen von $\widehat{Y}_t$ weniger als

$$2\#\ell\#\kappa_t\#\zeta_t \;\leq\; 2\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}k_{\max}k_{\max} \;=\; 2\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

Operationen. Der Aufwand für Algorithmus 2.4.3 zur Berechnung von $\widehat{V}_t$ ist durch

$$c_{apr}\#\ell\#\zeta_t \min\{\#\ell,\#\zeta_t\} \;\leq\; c_{apr}\#\ell\#\zeta_t^2 \;\leq\; c_{apr}\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

beschränkt (siehe Lemma 2.4.9). Dann hat $\widehat{V}_t$ weniger als $\#\zeta_t$ Spalten und die Berechnung des Basiswechsels benötigt höchstens

$$2\#\zeta_t\#\ell\#\kappa_t \;\leq\; 2k_{\max}\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}k_{\max} \;=\; 2\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

Operationen. Somit ist der Aufwand für $t\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$ durch

$$(2+2+c_{apr}+2)\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3 = (c_{apr}+6)\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

beschränkt. Mit Lemma 3.2.8 folgt, dass der gesamte Algorithmus weniger als

$$\begin{aligned}
&\sum_{t\in\mathcal{L}_{\mathcal{I}}} (c_{apr}+4)k_{\max}^2\#\mathbf{t} + \sum_{t\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}} (c_{apr}+6)\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3\\
&\leq\; (c_{apr}+4)k_{\max}^2\#\mathcal{I} + \sum_{t\in\mathcal{T}_{\mathcal{I}}} (c_{apr}+6)k_{\max}^3\\
&=\; (c_{apr}+4)k_{\max}^2\#\mathcal{I} + (c_{apr}+6)k_{\max}^3\#\mathcal{T}_{\mathcal{I}}
\end{aligned}$$

Operationen benötigt. ∎

Wir können also eine adaptive Clusterbasis mit kontrollierbarem Fehler berechnen. Falls die Annahmen aus Bemerkung 3.4.7 gelten, erhalten wir einen Aufwand in $\mathcal{O}(nk^2)$. In Abschnitt 4.5 fassen wir die Ergebnisse im Rekompressions-Algorithmus zusammen. Zuvor untersuchen wir Möglichkeiten, den blockweisen Fehler zu beschränken.
Wie bei der clusterweisen Fehlerabschätzung bleiben also die Normen der Fehlerterme aus Definition 4.3.2 abzuschätzen, wobei es sich in diesem Fall um die blockweisen Fehler $D(H,\overline{V})_{\check{t},s}$ für $(t,s)\in\mathcal{L}_{\mathcal{I}}^+$ und $\check{t}\in\operatorname{desc}(t)$ handelt. Zuerst zeigen wir eine zu Lemma 4.4.10 vergleichbare Aussage.

Lemma 4.4.12
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix und $Z_t := Z(H,V)_t$, $t \in \mathcal{T}_{\mathcal{I}}$, reduzierte Gewichte. Weiter sei $\overline{V}$ eine orthogonale Clusterbasis. Dann gilt für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\| \;\leq\; \|\left(I_{\check{\mathbf{t}}} - \Pi_{\check{V}(\overline{V})_{\check{t}}}\right)\widehat{V}(V,\overline{V})_{\check{t}} Z_{\check{t}}\|$$

sowohl für die Spektralnorm als auch die Frobeniusnorm.

Beweis: Es sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\check{t} \in \operatorname{desc}(t)$. Weiter sei $\check{V}_{\check{t}} := \check{V}(\overline{V})_{\check{t}}$. Dann gilt $s \in \overline{\operatorname{row}}(\check{t})$ und somit nach Definition 4.3.2 der Fehlerterme und Definition 4.1.6 der totalen Clusterbasis

$$\begin{aligned} D(H,\overline{V})_{\check{t},s} \;&=\; \Pi_{D,\overline{V},\check{t}} H_{|\check{\mathbf{t}}\times\mathbf{s}} \;=\; \Pi_{D,\overline{V},\check{t}}\Big(\Pi_{\check{\mathbf{t}}} H \sum_{s'\in\overline{\operatorname{row}}(\check{t})} \Pi_{\mathbf{s}'}\Big)\Pi_{\mathbf{s}} \\ &=\; \Pi_{D,\overline{V},\check{t}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_{\check{t}}\Pi_{\mathbf{s}} \;=\; D(H,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_{\check{t}}\Pi_{\mathbf{s}}. \end{aligned} \tag{4.15}$$

Weil $\Pi_{\mathbf{s}}$ eine orthogonale Projektion bezüglich $\|.\|_2$ ist, folgt aus Lemma 2.3.20 zusammen mit Lemma 4.4.10 und $\widehat{V}_{\check{t}} := \widehat{V}(V,\overline{V})_{\check{t}}$

$$\|D(H,\overline{V})_{\check{t},s}\| \;=\; \|D(H,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_{\check{t}}\Pi_{\mathbf{s}}\| \;\leq\; \|D(H,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_{\check{t}}\| \;\leq\; \|\left(I_{\check{\mathbf{t}}} - \Pi_{\check{V}_{\check{t}}}\right)\widehat{V}_{\check{t}} Z_{\check{t}}\|.$$

■

Weil ein reduziertes Gewicht Z_t nicht nur einen Block (t,s), sondern die gesamte erweiterte Blockzeile $\operatorname{row}(t)$ beschreibt, können wir in Lemma 4.4.12 keine Gleichheit erwarten. Diese Eigenschaft nehmen wir in Kauf, um in Algorithmus 4.4.1 die adaptiven Clusterbasen in linearer Komplexität berechnen zu können.
Wir erhalten also eine blockweise Fehlerabschätzung für den Projektionsfehler, bei der die rechte Seite kontrollierbar ist. Nach Lemma 4.4.9 (iii) können wir die rechte Seite in Lemma 4.4.12 für jedes $t \in \mathcal{T}_{\mathcal{I}}$ kontrollieren. Allerdings ist diese Abschätzung nicht dazu geeignet den Fehler für jeden Block individuell zu steuern.
Unser Ziel ist, den relativen Fehler in jedem Block durch ϵ zu beschränken. Dafür verwenden wir die Techniken, die in [12, Abschnitt 6.8] beschrieben werden. Dies erreichen wir, indem wir jeden Block $H_{|\mathbf{t}\times\mathbf{s}}$, $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, auf 1 normieren. Dadurch gilt für den Fehler der ursprünglichen Matrix

$$\frac{\|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2}{\|H_{|\mathbf{b}}\|^2} \;=\; \left\|\frac{H_{|\mathbf{b}}}{\|H_{|\mathbf{b}}\|} - \Pi_{\overline{V}_t}\frac{H_{|\mathbf{b}}}{\|H_{|\mathbf{b}}\|}\right\|^2 \;\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|D(\tilde{H},\overline{V})_{\check{t},s}\|^2,$$

wobei $\tilde{H}$ die skalierte Matrix ist. Die Terme auf der rechten Seite können wir durch Lemma 4.4.12 und Lemma 4.4.9 (iii) gegen ϵ abschätzen.
Um diese Idee formal umzusetzen, führen wir die skalierten $\mathcal{H}^2$-Matrizen ein. Die zu diesen Matrizen berechneten projizierten Gewichte erfüllen dann eine skalierte Fehlerbschätzung, wobei die Skalierung für jeden Block individuell gewählt werden kann.

Definition 4.4.13 (skalierte Gewichte)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix und $\omega = (\omega_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ eine Skalierung. Dann ist die *skalierte $\mathcal{H}^2$-Matrix* $H_\omega = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S_\omega, N)$ zu H und ω durch $S_{\omega,b} := \frac{1}{\omega_b} S_b$, für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, definiert. Wir bezeichnen reduzierte Gewichte Z_ω von H_ω als *skalierte Gewichte* von H zur Skalierung ω.

Die skalierte $\mathcal{H}^2$-Matrix H_ω wird nicht explizit aufgestellt. Stattdessen können wir die skalierten Gewichte direkt mit Hilfe von H und ω berechnen. Mit Hilfe der skalierten Gewichte erhalten wir folgende Fehlerabschätzung.

Korollar 4.4.14
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ eine Skalierung und $Z_{\omega,t} := Z(H_\omega, V)_t$, $t \in \mathcal{T}_\mathcal{I}$, skalierte Gewichte von H zu ω. Weiter sei $\overline{V}$ eine orthogonale Clusterbasis. Dann gilt für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\| \;\leq\; \omega_{(t,s)} \| \left(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}} \|$$

sowohl für die Spektralnorm als auch die Frobeniusnorm.

Beweis: Es sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\check{t} \in \operatorname{desc}(t)$. Dann folgt aus der Definition der blockweisen Fehlerterme und der skalierten $\mathcal{H}^2$-Matrix zusammen mit Lemma 4.4.12

$$\|D(H,\overline{V})_{\check{t},s}\| \;=\; \omega_{(t,s)} \|D(H_\omega,\overline{V})_{\check{t},s}\| \;\leq\; \omega_{(t,s)} \| \left(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}} \|.$$

■

Indem wir die skalierten Gewichte berechnen, können wir also die Fehlerabschätzung in jedem zulässigen Blatt unabhängig voneinander manipulieren. Wir nehmen an, dass die Terme auf der rechten Seite in Korollar 4.4.14 durch $\|(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\check{t}}})\widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}}\| \leq \epsilon$ beschränkt sind. Mit der Skalierung $\omega^2_{(t,s)} = \|H_{\mathbf{t}\times\mathbf{s}}\|^2 / \#\operatorname{desc}(t)$ erhalten wir mit Lemma 4.3.11

$$\begin{aligned}
\|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 \;&\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H,\overline{V})_{\check{t},s}\|^2 \\
&\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \omega^2_{(t,s)} \| \left(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}} \|^2 \\
&\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} (\|H_{\mathbf{t}\times\mathbf{s}}\|^2 / \#\operatorname{desc}(t))\epsilon^2 \;=\; \|H_{\mathbf{t}\times\mathbf{s}}\|^2 \epsilon^2.
\end{aligned}$$

Wir können also die Gewichte so wählen, dass wir den relativen blockweisen Projektionsfehler durch ϵ beschränken. Bei dem oben beschriebenen Ansatz wird der Fehler gleichmäßig über alle Nachfahren verteilt.
Ein anderer Ansatz ist, die Summe mit Hilfe der geometrischen Reihe aufzufangen. Dabei

spalten wir die Summe nach Level auf und setzen dann die Fehlerschranke so, dass für höhere Leveln die Approximation immer genauer wird. Dieser Ansatz ist durch die typischen Fehlerabschätzungen motiviert, bei denen der Fehler umso kleiner wird, je kleiner das Gebiet der Approximation ist (siehe Lemma 2.5.2).
Um diesen Ansatz formal definieren zu können, führen wir die adaptiven totalen Clusterbasen ein.

Definition 4.4.15 (adaptive totale Clusterbasis)
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Weiter seien $\omega = (\omega_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta = (\theta_t)_{t\in\mathcal{T}_\mathcal{I}} \in \mathbb{R}_{>0}^{\mathcal{T}_\mathcal{I}}$ Skalierungen. Dann definieren wir die *adaptive totale Clusterbasis* $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X, \omega, \theta)_t$ zu X, ω und θ

$$X_t := \begin{cases} \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} X_{|\mathbf{t}\times\mathbf{s}} & \text{, falls } t = r_\mathcal{I} \\ \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} X_{|\mathbf{t}\times\mathbf{s}} + \frac{1}{\theta_{\acute{t}}} X_{\acute{t}|\mathbf{t}} & \text{, falls } t \neq r_\mathcal{I},\ \acute{t} := \operatorname{par}(t) \end{cases}$$

für alle $t \in \mathcal{T}_\mathcal{I}$ [vergleiche [12, Definition 6.34]].

Die adaptive totale Clusterbasis stellt die gleichen Teile der Matrix dar wie die totale Clusterbasis (siehe Abbildung 4.1). Allerdings werden die verschiedene Teile noch skaliert. Im Gegensatz zur totalen Clusterbasis der skalierten $\mathcal{H}^2$-Matrix kommt noch eine Skalierung über die Level hinzu. Im folgenden Lemma untersuchen wir die Gestalt der adaptiven totalen Clusterbasis näher.

Lemma 4.4.16
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum. Weiter seien $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_\mathcal{I}}$ Skalierungen. Wir bezeichnen die Level der Cluster mit $\ell_t := \operatorname{level}(t)$ für alle $t \in \mathcal{T}_\mathcal{I}$. Dann gilt für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\check{t} \in \operatorname{desc}(t)$

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\check{t}}\Pi_\mathbf{s} = \frac{1}{\omega_{(t,s)}} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t\le\ell_{t'}<\ell_{\check{t}}}} \frac{1}{\theta_{t'}} \right) X_{\check{\mathbf{t}}\times\mathbf{s}}.$$

Beweis: Es sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Wir beweisen die Aussage per Induktion über $\ell_{\check{t}} - \ell_t$.
Es sei $\check{t} \in \operatorname{desc}(t)$ mit $\ell_{\check{t}} - \ell_t = 0$. Dann gilt $\check{t} = t$ und, weil die Beschriftungen der Cluster in $\overline{\operatorname{row}}(t)$ paarweise disjunkt sind, folgt

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\check{t}}\Pi_\mathbf{s} &= X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t\Pi_\mathbf{s} \\ &= \frac{1}{\omega_{(t,s)}} X_{\mathbf{t}\times\mathbf{s}} = \frac{1}{\omega_{(t,s)}} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t\le\ell_{t'}<\ell_{\check{t}}}} \frac{1}{\theta_{t'}} \right) X_{\check{\mathbf{t}}\times\mathbf{s}}. \end{aligned}$$

Dabei ist das Produkt gleich 1, weil es über die leere Menge genommen wird.
Es sei $n \in \mathbb{N}$ derart, dass die Aussage für alle $\check{t} \in \operatorname{desc}(t)$ mit $\ell_{\check{t}} - \ell_t \le n$ gilt. Es sei

$\check{t} \in \operatorname{desc}(t)$ mit $\ell_{\check{t}} - \ell_t = n+1 > 0$. Dann existiert $\acute{t} := \operatorname{par}(\check{t}) \in \operatorname{desc}(t)$. Es gilt $\ell_{\acute{t}} - \ell_t = n$ und somit

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}}\Pi_{\mathbf{s}} = \frac{1}{\omega_{(t,s)}} \left(\prod_{\substack{t' \in \operatorname{desc}(t) \\ \ell_t \leq \ell_{t'} < \ell_{\acute{t}}}} \frac{1}{\theta_{t'}} \right) X_{\acute{\mathbf{t}}\times\mathbf{s}}.$$

Weil $t \neq \check{t}$ gilt, ist $s \in \overline{\operatorname{row}}(\check{t}) \setminus \operatorname{row}(\check{t})$. Damit folgt

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}}\Pi_{\mathbf{s}} &= \left(\sum_{s \in \operatorname{row}(\check{t})} \frac{1}{\omega_{(\check{t},s)}} V_{\check{t}} S_{(\check{t},s)} W_s^T + \frac{1}{\theta_{\acute{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\check{\mathbf{t}}} \right) \Pi_{\mathbf{s}} \\ &= \frac{1}{\theta_{\acute{t}}} \Pi_{\check{\mathbf{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}}\Pi_{\mathbf{s}} = \frac{1}{\theta_{\acute{t}}} \Pi_{\check{\mathbf{t}}} \frac{1}{\omega_{(t,s)}} \left(\prod_{\substack{t' \in \operatorname{desc}(t) \\ \ell_t \leq \ell_{t'} < \ell_{\acute{t}}}} \frac{1}{\theta_{t'}} \right) X_{\acute{\mathbf{t}}\times\mathbf{s}} \\ &= \frac{1}{\omega_{(t,s)}} \left(\prod_{\substack{t' \in \operatorname{desc}(t) \\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \frac{1}{\theta_{t'}} \right) X_{\check{\mathbf{t}}\times\mathbf{s}}. \end{aligned}$$

■

Analog zum reduzierten Gewicht für die totale Clusterbasis definieren wir das adaptive Gewicht zur adaptiven totalen Clusterbasis.

Definition 4.4.17 (adaptive Gewichte)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $Z = (Z_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ Gewichte zu V und Rangverteilungen $\kappa = (\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $\zeta = (\zeta_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ zu V bzw. Z. Weiter seien $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen. Dann heißt Z *adaptives Gewicht* von H zu ω und θ mit Rangverteilung ζ, falls für alle $t \in \mathcal{T}_{\mathcal{I}}$ eine orthogonale Matrix P_t mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X, \omega, \theta)_t = V_t Z_t P_t^T$ existiert.

Die adaptiven Gewichte sind eine Verallgemeinerung der reduzierten und der skalierten Gewichte.

Bemerkung 4.4.18
Falls $\theta_t = 1$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ gilt, ist $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta) = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega)$ und $Z(H, V, \omega, \theta)$ ein skaliertes Gewicht von H zur Skalierung ω. Falls zusätzlich $\omega_b = 1$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gilt, ist $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta) = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)$ die totale Clusterbasis von H und $Z(H, V, \omega, \theta)$ ein reduziertes Gewicht. Die Definition 4.4.15 ist also eine Verallgemeinerung der vorherigen Ansätze.

Für das adaptive Gewicht können wir analog zu Lemma 4.4.3 eine rekursive Formel angeben.

Lemma 4.4.19
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix und $(O_s)_{s\in\mathcal{T}_\mathcal{J}}$ orthogonale Gewichte zu W. Weiter seien $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_\mathcal{I}}$ Skalierungen. Die Matrix $Z_t := Z(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, V, \omega, \theta)_t \in \mathbb{R}^{\kappa_t\times\zeta_t}$ sei für alle $t \in \mathcal{T}_\mathcal{I}$ rekursiv von den Blättern aus durch die QR-Zerlegung $Z_t\widehat{P}_t^T = \widehat{Z}_t$ gegeben, wobei

$$\widehat{Z}_t := \begin{cases} \begin{pmatrix} \frac{1}{\omega_{(t,s_1)}} S_{(t,s_1)} O_{s_1}^T & \cdots & \frac{1}{\omega_{(t,s_\sigma)}} S_{(t,s_\sigma)} O_{s_\sigma}^T \end{pmatrix} & \text{, falls } t = r_\mathcal{I} \\ \begin{pmatrix} \frac{1}{\omega_{(t,s_1)}} S_{(t,s_1)} O_{s_1}^T & \cdots & \frac{1}{\omega_{(t,s_\sigma)}} S_{(t,s_\sigma)} O_{s_\sigma}^T & \frac{1}{\theta_{\acute{t}}} E_t Z_{\acute{t}} \end{pmatrix} & \text{, falls } t \neq r_\mathcal{I},\ \acute{t} := \operatorname{par}(t) \end{cases}$$

sei. Dann ist $Z(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, V, \omega, \theta)$ ein adaptives Gewicht von H mit Rangverteilung $\zeta = (\zeta_t)_{t\in\mathcal{T}_\mathcal{I}}$ zu den Skalierungen ω und θ.

Beweis: Für alle $t \in \mathcal{T}_\mathcal{I}$ sei $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t$ und für alle $s \in \mathcal{T}_\mathcal{J}$ sei P_s die zu dem orthogonalen Gewicht O_s gehörige orthogonale Matrix.
Für die Aussage müssen wir zeigen, dass für alle $t \in \mathcal{T}_\mathcal{I}$ eine orthogonale Matrix P_t mit $X_t = V_t Z P_t^T$ existiert. Dies zeigen wir per Induktion über $\#\operatorname{pred}(t)$.
Es sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{pred}(t) = 1$. Dann ist $t = r_\mathcal{I}$ und mit der Darstellung der adaptiven totalen Clusterbasis aus Definition 4.4.15 folgt

$$\begin{aligned} X_t &= \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} V_t S_{(t,s)} W_s^T = V_t \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} S_{(t,s)} O_s^T P_s^T \\ &= V_t \begin{pmatrix} \frac{1}{\omega_{(t,s_1)}} S_{(t,s_1)} O_{s_1}^T & \cdots & \frac{1}{\omega_{(t,s_\sigma)}} S_{(t,s_\sigma)} O_{s_\sigma}^T \end{pmatrix} \begin{pmatrix} P_{s_1} & \cdots & P_{s_\sigma} \end{pmatrix}^T \\ &=: V_t \widehat{Z}_t \widetilde{P}_t^T = V_t Z_t \widehat{P}_t^T \widetilde{P}_t^T =: V_t Z_t P_t^T. \end{aligned}$$

Die Matrix $\widetilde{P}_t$ ist nach Bemerkung 2.3.15 orthogonal. Somit ist auch $P_t = \widetilde{P}_t\widehat{P}_t$ orthogonal und die Behauptung gilt für t.
Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{pred}(t) \leq n$ gilt. Es sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{pred}(t) = n + 1 > 1$. Dann gilt $t \neq r_\mathcal{I}$ und es existiert $\acute{t} := \operatorname{par}(t)$. Es gilt $\#\operatorname{pred}(\acute{t}) = n$ und somit existiert für $Z_{\acute{t}}$ eine orthogonale Matrix $P_{\acute{t}}$ mit $X_{\acute{t}} = V_{\acute{t}} Z_{\acute{t}} P_{\acute{t}}$. Aus der Definition 4.4.15 der adaptiven totalen Clusterbasis folgt

$$\begin{aligned} X_t &= \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} V_t S_{(t,s)} W_s^T + \frac{1}{\theta_{\acute{t}}} X_{\acute{t}|\mathbf{t}} \\ &= V_t \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} S_{(t,s)} O_s^T P_s^T + \frac{1}{\theta_{\acute{t}}} V_{\acute{t}|\mathbf{t}} Z_{\acute{t}} P_{\acute{t}}^T \\ &= V_t \begin{pmatrix} \frac{1}{\omega_{(t,s_1)}} S_{(t,s_1)} O_{s_1}^T & \cdots & \frac{1}{\omega_{(t,s_\sigma)}} S_{(t,s_\sigma)} O_{s_\sigma}^T & \frac{1}{\theta_{\acute{t}}} E_t Z_{\acute{t}} \end{pmatrix} \begin{pmatrix} P_{s_1} & \cdots & P_{s_\sigma} & P_{\acute{t}} \end{pmatrix}^T \\ &=: V_t \widehat{Z}_t \widetilde{P}_t^T = V_t Z_t \widehat{P}_t^T \widetilde{P}_t^T =: V_t Z_t P_t^T. \end{aligned}$$

Wie oben ist die Matrix $\widetilde{P}_t$ nach Lemma 2.3.15 orthogonal. Also ist auch $P_t = \widetilde{P}_t\widehat{P}_t$ orthogonal und die Behauptung folgt. ∎

Bemerkung 4.4.20
Wie wir im Beweis gesehen haben, muss im Falle eines Nicht-Wurzel-Clusters das adaptive Gewicht des Vaters nicht aus der in Lemma 4.4.19 beschriebenen Konstruktion stammen. Dies werden wir später beim Niedrigrangupdate verwenden.

Mit Hilfe von Lemma 4.4.16 können wir die blockweisen Fehlerterme durch Terme der adaptiven Gewichte abschätzen.

Lemma 4.4.21
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen. Weiter seien $(Z_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ adaptive Gewichte von H zu ω und θ. Es sei $\overline{V}$ eine orthogonale Clusterbasis. Dann gilt für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\| \;\leq\; \omega_{(t,s)} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \right) \|(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}}) \widehat{V}_{\check{t}} Z_{\check{t}}\|$$

sowohl für die Spektralnorm als auch die Frobeniusnorm.

Beweis: Es sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\check{t} \in \operatorname{desc}(t)$. Aus der Definition der Fehlerterme 4.3.2 folgt mit $\breve{V}_{\check{t}} := \breve{V}(\overline{V})_{\check{t}}$ und Lemma 4.4.16

$$D(H,\overline{V})_{\check{t},s} \;=\; \Pi_{D,\overline{V},\check{t}} X_{\check{\mathbf{t}}\times\mathbf{s}} \;=\; \Pi_{D,\overline{V},\check{t}} \omega_{(t,s)} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \right) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\check{t}} \Pi_{\mathbf{s}} \tag{4.16}$$

Es sei $P_{\check{t}}$ die orthogonale Matrix zum adaptiven Gewicht $Z_{\check{t}}$. Dann gilt

$$D(H,\overline{V})_{\check{t},s} \;=\; \Pi_{D,\overline{V},\check{t}} \omega_{(t,s)} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \right) V_{\check{t}} Z_{\check{t}} P_{\check{t}}^T \Pi_{\mathbf{s}}$$

Nach (4.2.4) gilt $\breve{V}_{\check{t}}^T V_{\check{t}} = \widehat{V}(V,\overline{V})_{\check{t}} =: \widehat{V}_{\check{t}}$ und nach (4.2.12) gilt $\overline{V}_{\check{t}} = \breve{V}_{\check{t}} \acute{V}(\overline{V})_{\check{t}} =: \breve{V}_{\check{t}} \acute{V}_{\check{t}}$. Damit erhalten wir

$$(I_{\check{\mathbf{t}}} - \Pi_{\overline{V}_{\check{t}}}) \Pi_{\breve{V}_{\check{t}}} V_{\check{t}} \;=\; \breve{V}_{\check{t}} (I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}}) \breve{V}_{\check{t}}^T V_{\check{t}} \;=\; \breve{V}_{\check{t}} (I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}}) \widehat{V}_{\check{t}}. \tag{4.17}$$

Weil $\breve{V}_{\check{t}}$ und $P_{\check{t}}$ orthogonal sind und $\Pi_{\mathbf{s}}$ eine orthogonale Projektion bezüglich $\|.\|_2$ ist, folgt mit Lemma 2.3.16 und Lemma 2.3.20

$$\begin{aligned}
\|D(H,\overline{V})_{\check{t},s}\| \;&=\; \omega_{(t,s)} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \right) \|\breve{V}_{\check{t}} (I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}}) \widehat{V}_{\check{t}} Z_{\check{t}} P_{\check{t}}^T \Pi_{\mathbf{s}}\| \\
&\leq\; \omega_{(t,s)} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \right) \|(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}}) \widehat{V}_{\check{t}} Z_{\check{t}}\|.
\end{aligned}$$

■

Der Term in der Norm auf der rechten Seite ist bei der Konstruktion von $\overline{V}$ mit Algorithmus 4.4.1 durch Lemma 4.4.9 (iii) beschränkt. In Abschnitt 4.5 geben wir verschiedene Strategien zur Fehlerkontrolle an. Zuvor stellen wir noch den Algorithmus 4.4.2 zum Berechnen der adaptiven Gewichte vor. Nach Bemerkung 4.4.18 lassen sich damit auch reduzierte Gewichte berechnen, indem die Skalierungen gleich 1 gesetzt werden.

Algorithmus 4.4.2 [vergleiche [12, Algorithm 34]] Die Funktion berechnet die adaptiven Gewichte zu den Skalierungen ω und θ.

function ADAPTIVE_GEWICHTE(t, E, S, κ, O, ρ, ω, θ, Z, ζ)
  $\ell \leftarrow \dot{\bigcup}_{s \in \mathrm{row}(t)} \rho_s$
  if $t \neq r_{\mathcal{I}}$ **then**
    $\acute{t} \leftarrow \mathrm{par}(t)$
    $\ell \leftarrow \zeta_{\acute{t}}$
  end if
  $\widehat{Z} \in \mathbb{R}^{\kappa_t \times \ell}$
  for $s \in \mathrm{row}(t)$ **do**
    $\widehat{Z}_{t_{|\kappa_t \times \rho_s}} \leftarrow \frac{1}{\omega_{(t,s)}} S_{(t,s)} O_s^T$
  end for
  if $t \neq r_{\mathcal{I}}$ **then**
    $\widehat{Z}_{t_{|\kappa_t \times \zeta_{\acute{t}}}} \leftarrow \frac{1}{\theta_{\acute{t}}} E_t Z_{\acute{t}}$
  end if
  QR-ZERLEGUNG($\widehat{Z}^T$, ℓ, ℓ, κ_t, P_t, Z_t^T, ζ_t) ▷ Algorithmus 2.4.1
  for $\breve{t} \in \mathrm{chil}(t)$ **do**
    ADAPTIVE_GEWICHTE($\breve{t}$, E, S, κ, O, ρ, ω, θ, Z, ζ) ▷ top-down
  end for
end function

Lemma 4.4.22
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen κ und λ. Weiter seien $(O_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ orthogonale Gewichte für die Clusterbasis W mit Rangverteilung W. Wir definieren

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t,\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s,\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\rho_s\}.$$

Dann ist der Aufwand zur Berechnung der in Lemma 4.4.19 beschriebenen adaptiven Gewichte zu den Skalierungen ω und θ mit Algorithmus 4.4.2 höchstens

$$(c_{sp}+1)(c_{qr}+3)\#\mathcal{T}_{\mathcal{I}} k_{\max}^3.$$

Beweis: Für alle $t \in \mathcal{T}_{\mathcal{I}}$ ist die Anzahl der Spalten von Z_t beschränkt durch $\#\zeta_t \leq \#\kappa_t \leq k_{\max}$, da Z_t aus der orthogonalen Zerlegung von $\widehat{Z}_t$ mit $\#\kappa_t$ Zeilen entsteht.

Es sei $t = r_{\mathcal{I}}$. Dann werden für alle $s \in \operatorname{row}(t)$ die Produkte $\frac{1}{\omega_{(t,s)}} S_{(t,s)} O_s^T$ berechnet. Jedes dieser Produkte benötigt höchstens

$$3\#\kappa_t\#\lambda_s\#\rho_s \;\leq\; 3k_{\max}^3$$

Operationen. (Eigentlich könnten wir die Skalierung in $k_{\max}^2$ Operationen berechnen, aber wir wollen an dieser Stelle die Abschätzung einfach halten.) Die Anzahl der Zeilen der Matrix $\widehat{Z}_t$ ist durch

$$\# \dot{\bigcup_{s\in\operatorname{row}(t)}} \rho_s \;=\; \sum_{s\in\operatorname{row}(t)} \#\rho_s \;\leq\; \sum_{s\in\operatorname{row}(t)} k_{\max} \;\leq\; c_{sp}k_{\max}$$

und die Anzahl der Zeilen durch $\#\kappa_t \leq k_{\max}$ beschränkt. Somit lässt sich Z_t per orthogonaler Zerlegung aus $\widehat{Z}_t$ in nicht mehr als

$$c_{qr}k_{\max}c_{sp}k_{\max}\min\{k_{\max}, c_{sp}k_{\max}\} \;\leq\; c_{qr}c_{sp}k_{\max}^3$$

Operationen berechnen. Zusammen benötigt die Berechnung von Z_t höchstens $3c_{sp}k_{\max}^3 + c_{qr}c_{sp}k_{\max}^3$ Operationen.
Es sei $t \neq r_{\mathcal{I}}$. Dann existiert $\acute{t} := \operatorname{par}(t)$. Im Vergleich zu oben muss zum Aufstellen von $\widehat{Z}_t$ zusätzlich das Produkt $\frac{1}{\theta_{\acute{t}}} E_t Z_{\acute{t}}$ berechnet werden. Dies kostet höchstens $3\#\kappa_t\#\kappa_{\acute{t}}\#\zeta_{\acute{t}} \leq 3k_{\max}^3$ Operationen. Somit ist der Aufwand zum Aufstellen der Matrix $\widehat{Z}_t$ kleiner als $3(c_{sp}+1)k_{\max}^3$. Außerdem hat die Matrix $\widehat{Z}_t$ in diesem Fall

$$\begin{aligned}\#\left(\left(\dot{\bigcup_{s\in\operatorname{row}(t)}} \rho_s\right) \dot{\cup}\, \zeta_{\acute{t}}\right) \;&=\; \left(\sum_{s\in\operatorname{row}(t)} \#\rho_s\right) + \#\zeta_{\acute{t}} \leq \left(\sum_{s\in\operatorname{row}(t)} k_{\max}\right) + k_{\max} \\ &\leq\; (c_{sp}+1)k_{\max}\end{aligned}$$

Spalten. Dies führt zu einem Aufwand für die orthogonale Zerlegung von höchstens $c_{qr}(c_{sp}+1)k_{\max}^3$ Operationen. Der gesamte Aufwand für den Cluster t lässt sich somit durch $(c_{qr}+3)(c_{sp}+1)k_{\max}^3$ beschränken, welches auch eine obere Schranke für den Fall $t = r_{\mathcal{I}}$ ist. Durch Summieren über alle Cluster folgt die Behauptung. ∎

4.5 Rekompression

In diesem Abschnitt fassen wir die Ergebnisse des Kapitels im Rekompressions-Algorithmus zusammen. Wir schätzen den Aufwand und den entstandenen Fehler ab. Zusätzlich geben wir verschiedene Strategien an, um bestimmte Fehlerschranken zu erreichen. Beim Aufruf von Algorithmus 4.2.1 in Algorithmus 4.5.1 benötigen wir eigentlich das orthogonale Gewicht und nicht die orthogonale Clusterbasis. Wir verzichten darauf, dafür einen eigenen Algorithmus anzugeben, weil die Komplexität des Aufwands sich dadurch nicht verändert

Algorithmus 4.5.1 Die Funktion berechnet für die $\mathcal{H}^2$-Matrix $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ die rekomprimierte $\mathcal{H}^2$-Matrix $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ zu den Fehlerschranken $\epsilon_{\mathcal{I}}$ und $\epsilon_{\mathcal{J}}$, der Fehlerart *opt* und den Skalierungen ω und θ.

function REKOMPRESSION($r_{\mathcal{I}\times\mathcal{J}}$, V, E, κ, W, F, λ, N, S, $\overline{V}$, $\overline{E}$, $\overline{\kappa}$, $\overline{W}$, $\overline{F}$, $\overline{\lambda}$, $\overline{N}$, $\overline{S}$, $\epsilon_{\mathcal{I}}$, $\epsilon_{\mathcal{J}}$, *opt*, $\omega_{\mathcal{I}}$, $\omega_{\mathcal{J}}$, $\theta_{\mathcal{I}}$, $\theta_{\mathcal{J}}$)
  $(r_{\mathcal{I}}, r_{\mathcal{J}}) \leftarrow r_{\mathcal{I}\times\mathcal{J}}$
  ORTHOGONALE_CLUSTERBASIS($r_{\mathcal{I}}$, V, E, κ, $\tilde{V}$, $\tilde{E}$, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$) ▷ Algorithmus 4.2.1
  ORTHOGONALE_CLUSTERBASIS($r_{\mathcal{J}}$, W, F, λ, $\tilde{W}$, $\tilde{F}$, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$)
  ADAPTIVE_GEWICHTE($r_{\mathcal{I}}$, E, S, κ, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$, $\omega_{\mathcal{I}}$, θ, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$) ▷ Algorithmus 4.4.2
  ADAPTIVE_GEWICHTE($r_{\mathcal{J}}$, F, S, λ, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$, $\omega_{\mathcal{J}}$, θ, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$)
  ADAPTIVE_CLUSTERBASIS($r_{\mathcal{I}}$, V, κ, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$, $\overline{V}$, $\overline{\kappa}$, $C_{\mathcal{I}}$, ϵ, *opt*) ▷ Algorithmus 4.4.1
  ADAPTIVE_CLUSTERBASIS($r_{\mathcal{J}}$, W, λ, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$, $\overline{W}$, $\overline{\lambda}$, $C_{\mathcal{J}}$, ϵ, *opt*)
  H2-MATRIX-PROJEKTION($r_{\mathcal{I}\times\mathcal{J}}$, S, N, C, $\overline{\kappa}$, D, $\overline{\lambda}$, $\overline{S}$, $\overline{N}$) ▷ Algorithmus 4.2.2
end function

Lemma 4.5.1
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen κ und λ. Weiter seien $\epsilon_{\mathcal{I}} \in \mathbb{R}_{\geq 0}^{\mathcal{T}_{\mathcal{I}}}$ und $\epsilon_{\mathcal{J}} \in \mathbb{R}_{\geq 0}^{\mathcal{T}_{\mathcal{J}}}$ Fehlerschranken, *opt* eine Fehlerart und $\omega_{\mathcal{I}}, \omega_{\mathcal{J}} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$, $\theta_{\mathcal{I}} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ und $\theta_{\mathcal{J}} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{J}}}$ Skalierungen. Es sei $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_t\}$. Dann benötigt Algorithmus 4.5.1 zur Berechnung der rekomprimierten $\mathcal{H}^2$-Matrix höchstens

$$(c_{apr} + c_{qr} + 4)k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) \;+\; (c_{apr} + 2c_{qr} + c_{qr}c_{sp} + 5c_{sp} + 11)k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}})$$

Operationen.

BEWEIS: Der Aufwand für die Aufrufe des Algorithmus 4.2.1 zur Berechnung der orthogonalen Gewichte ist nach Lemma 4.2.10 kleiner als

$$c_{qr}k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) + (c_{qr} + 2)k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}}).$$

Die Aufrufe für den Algorithmus 4.4.2 zur Berechnung der adaptiven Gewichte benötigen nach Lemma 4.4.22 höchstens

$$(c_{sp} + 1)(c_{qr} + 3)\#k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}})$$

Operationen. Der Aufwand zur Berechnung der neuen Clusterbasen durch Algorithmus 4.4.1 ist nach Lemma 4.4.11 beschränkt durch

$$(c_{apr} + 4)k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) + (c_{apr} + 6)k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}}).$$

Für die neuen Ränge gilt $\#\overline{\kappa}_t \leq \#\kappa_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $\#\overline{\lambda}_s \leq \#\lambda_s$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ (siehe Lemma 4.4.9). Zuletzt werden die neuen Kopplungsmatrizen mit Algorithmus 4.2.2

nach Lemma 4.2.19 in weniger als $4c_{sp}k_{\max}^3 \min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\} \leq 2c_{sp}k_{\max}^3(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J})$ Operationen aufgestellt. Zusammen erhalten wir für den Aufwand die obere Schranke

$$\begin{aligned}
&(c_{qr} + (c_{apr} + 4))k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) \\
&\quad + \ ((c_{qr} + 2) + (c_{sp} + 1)(c_{qr} + 3) + (c_{apr} + 6) + 2c_{sp})k_{\max}^3(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}) \\
&= (c_{apr} + c_{qr} + 4)k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) \\
&\quad + \ (c_{apr} + 2c_{qr} + c_{qr}c_{sp} + 5c_{sp} + 11)k_{\max}^3(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}).
\end{aligned}$$

■

Der Fehler, der durch die Rekompression entsteht, kann durch die clusterweise Abschätzung aus Theorem 4.3.7 bzw. die blockweise Abschätzung aus Theorem 4.3.9 und Lemma 4.3.11, die Abschätzung der Fehlerterme in Lemma 4.4.10 bzw. Lemma 4.4.21 und die Fehlerschranke für die adaptive Clusterbasis in Lemma 4.4.9 (iii) beschränkt werden.
Weil wir den Fehler durch die Wahl der vier verschiedenen Größen ϵ, *opt*, ω und θ beeinflussen können, ist es schwer, eine allgemeine und übersichtliche Fehlerschranke zu formulieren. Deshalb beschränken wir uns auf drei Strategien zur Wahl der Optionen und beweisen für diese jeweils eine Fehlerschranke. In der ersten Strategie nutzen wir reduzierte Gewichte und den relativen Fehler.

Lemma 4.5.2 (Strategie 1)
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\omega_b := \theta_{\mathcal{I},t} := \theta_{\mathcal{J},s} := 1$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$, $\epsilon_t, \epsilon_s \in \mathbb{R}_{\geq 0}$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ und *opt* der relative Fehler bezüglich $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Weiter sei $\overline{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ die mit Algorithmus 4.5.1 zu den oben definierten Optionen berechnete neue $\mathcal{H}^2$-Matrix. Dann ist der Fehler der Rekompression für $\|.\| = \|.\|_S$ durch

$$\|H - \overline{H}\|_S^2 \ \leq \ 2\left(\sum_{t\in\mathcal{T}_\mathcal{I}} \epsilon_t^2 + \sum_{s\in\mathcal{T}_\mathcal{J}} \epsilon_s^2\right) \|H\|_S^2$$

und für $\|.\| = \|.\|_F$ durch

$$\|H - \overline{H}\|_F^2 \ \leq \ \left(\sum_{t\in\mathcal{T}_\mathcal{I}} \epsilon_t^2 + \sum_{s\in\mathcal{T}_\mathcal{J}} \epsilon_s^2\right) \|H\|_F^2$$

beschränkt.

Beweis: Nach Theorem 4.3.7 gilt

$$\|H - \overline{H}\|^2 \ \leq \ 2\left(\sum_{t\in\mathcal{T}_\mathcal{I}} \|D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|^2 + \sum_{s\in\mathcal{T}_\mathcal{J}} \|D(H^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|^2\right).$$

(Für die Frobeniusnorm erhalten wir die Abschätzung ohne den Faktor 2.) Weil die Clusterbasen $\overline{V}$ und $\overline{W}$ mit denselben Algorithmen konstruiert werden, schätzen wir zunächst $\|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|^2$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ ab. Dabei verwenden wir die Notation $Z_t := Z_{I,t}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$. Nach Bemerkung 4.4.18 sind die adaptiven Gewichte bei unserer Wahl von $\omega_{\mathcal{I}}$ und $\theta_{\mathcal{I}}$ gleich den reduzierten Gewichten.
Es sei $t \in \mathcal{T}_{\mathcal{I}}$. Dann folgt aus Lemma 4.4.10

$$\|D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\| \;=\; \|(I_{\mathbf{t}} - \Pi_{\acute{V}(\overline{V})_t})\widehat{V}(V, \overline{V})_t Z_t\|.$$

Nach Lemma 4.4.9 (iii) gilt

$$\|(I_{\mathbf{t}} - \Pi_{\acute{V}(\overline{V})_t})\widehat{V}(V, \overline{V})_t Z_t\| \;\leq\; \epsilon_t \|\widehat{V}(V, \overline{V})_t Z_t\|.$$

Es gilt nach Lemma 4.2.4 $\widehat{V}(V, \overline{V}) = \breve{V}(\overline{V})_t^T V_t$ und nach Lemma 4.2.12 ist $\breve{V}(\overline{V})_t$ orthogonal. Weil Z_t ein reduzierte Gewicht ist, existiert eine orthogonale Matrix P_t mit $V_t Z_t P_t^T = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t$. Nach Lemma 2.3.16 können wir orthogonale Matrizen hinzufügen bzw. entfernen, ohne die Norm zu verändern. Damit folgt

$$\begin{aligned}\|\widehat{V}(V, \overline{V})_t Z_t\| \;&=\; \|\breve{V}(\overline{V})_t^T V_t Z_t\| \;=\; \|\breve{V}(\overline{V})_t \breve{V}(\overline{V})_t^T V_t Z_t P_t^T\| \\ &=\; \|\Pi_{\breve{V}(\overline{V})_t} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t\| \;\leq\; \|X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t\|,\end{aligned}$$

wobei wir ausnutzen, dass $\Pi_{\breve{V}(\overline{V})_t}$ eine orthogonale Projektion bezüglich $\|.\|_2$ ist, die nach Lemma 2.3.20 die betrachteten Matrixnormen nicht vergrößert. Mit dem gleichen Argument folgt

$$\|\widehat{V}(V, \overline{V})_t Z_t\| \;\leq\; \|X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, V)_t\| \;=\; \|\Pi_{\mathbf{t}} H \sum_{s\in\overline{\mathrm{row}}(t)} \Pi_{\mathbf{s}}\| \;\leq\; \|H\|.$$

Damit erhalten wir für die Fehlerterme die Abschätzung

$$\|D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \overline{V})_t\| \;=\; \|(I_{\mathbf{t}} - \Pi_{\acute{V}(\overline{V})_t})\widehat{V}(V, \overline{V})_t Z_t\| \;\leq\; \epsilon_t \|\widehat{V}(V, \overline{V})_t Z_t\| \;\leq\; \epsilon_t \|H\|.$$

Analog folgt $\|D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T, H, \overline{W})_s\| \leq \epsilon_s \|H\|$ für alle $s \in \mathcal{T}_{\mathcal{J}}$. Durch Einsetzen dieser Abschätzungen in die Fehlerschranke aus Theorem 4.3.7 folgt die Aussage. ∎

Durch relative Fehlerkontrolle im Algorithmus 4.4.1 mit reduzierten Gewichten erhalten wir also global eine relative Fehlerschranke. Die Summe über alle Cluster kann z.B. durch gleichmäßige Fehlerverteilung (vergleiche Strategie 2) oder levelweise Fehlerverteilung und geometrischer Reihe (vergleiche Strategie 3) kontrolliert werden.

Bemerkung 4.5.3
Falls bei Strategie 1 der absolute Fehler statt dem relativen Fehler beim Berechnen der neuen Clusterbasen verwendet wird, erhalten wir die Fehlerabschätzungen in Lemma 4.5.2 ohne den Faktor $\|H\|^2$ auf der rechten Seite. Weil wir üblicherweise an relativen Fehlerschranken interessiert sind, formulieren wir diese Möglichkeit nicht als gesonderte Strategie.

Als Nächstes betrachten wir die Fehlerschranke für den blockweisen Fehler. Zuerst wollen wir diesen mit skalierten Gewichten kontrollieren.

Lemma 4.5.4 (Strategie 2)
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\theta_{\mathcal{I},t} := 1$ für alle $t \in \mathcal{T}_{\mathcal{I}}$, $\theta_{\mathcal{J},s} := 1$ für alle $s \in \mathcal{T}_{\mathcal{J}}$,

$$\omega_{\mathcal{I},(t,s)} \;:=\; \omega_{\mathcal{J},(t,s)} \;:=\; \begin{cases} 1 & \text{, falls } \|H_{|\mathbf{t}\times\mathbf{s}}\| = 0 \\ \frac{\|H_{|\mathbf{t}\times\mathbf{s}}\|}{\sqrt{\#\operatorname{desc}(t)+\#\operatorname{desc}(s)}} & \text{, falls } \|H_{|\mathbf{t}\times\mathbf{s}}\| \neq 0 \end{cases}$$

für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, $\epsilon_t := \epsilon_s := \epsilon \in \mathbb{R}_{\geq 0}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ und *opt* der absolute Fehler bezüglich $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Weiter sei $\overline{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ die mit Algorithmus 4.5.1 zu den oben definierten Optionen berechnete neue $\mathcal{H}^2$-Matrix. Dann ist der Fehler der Rekompression in jedem zulässigen Blatt $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ durch

$$\|H_{|\mathbf{t}\times\mathbf{s}} - \overline{H}_{|\mathbf{t}\times\mathbf{s}}\| \;\leq\; \epsilon\,\|H_{|\mathbf{t}\times\mathbf{s}}\|$$

beschränkt.

Beweis: Es sei $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Für den Fall $\|H_{|\mathbf{t}\times\mathbf{s}}\| = 0$ gilt

$$\overline{H}_{|\mathbf{t}\times\mathbf{s}} \;=\; \Pi_{\overline{V}_t} H_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s} \;=\; \Pi_{\overline{V}_t} 0 \Pi_{\overline{W}_s} \;=\; 0. \tag{4.18}$$

Damit ist in diesem Fall der blockweise Fehler gleich null und die Aussage gilt.
Es sei also im Folgenden $\|H_{|\mathbf{t}\times\mathbf{s}}\| \neq 0$. Wie im Beweis von Theorem 4.3.9 gilt

$$\|H_{|\mathbf{t}\times\mathbf{s}} - \overline{H}_{|\mathbf{t}\times\mathbf{s}}\|^2 \;=\; \|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}} \Pi_{\overline{W}_s}\|^2 \;\leq\; \|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 + \|H^T_{|\mathbf{b}} - \Pi_{\overline{W}_s} H^T_{|\mathbf{b}}\|^2.$$

Wir betrachten den Fehler bezüglich der Zeilenbasis. Für diesen gilt nach Lemma 4.3.11

$$\|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 \;\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H,\overline{V})_{\check{t},s}\|^2.$$

Weil wir θ überall gleich 1 wählen, ist das adaptive Gewicht nach Bemerkung 4.4.18 ein skaliertes Gewicht. Nach Korollar 4.4.14 gilt für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\| \;\leq\; \omega_{(t,s)} \| \left(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}}\|.$$

Weil wir als Fehlerart *opt* den absoluten Fehler gewählt haben, folgt aus Lemma 4.4.9 (iii) für alle $\check{t} \in \operatorname{desc}(t)$

$$\| \left(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}}\| \;\leq\; \epsilon_t \;=\; \epsilon.$$

Zusammen mit $\omega^2_{(t,s)} = \|H_{|\mathbf{t}\times\mathbf{s}}\|^2/(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))$ folgt für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\|^2 \;\leq\; \omega^2_{(t,s)} \| \left(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}}\|^2 \;\leq\; \frac{\|H_{|\mathbf{t}\times\mathbf{s}}\|^2 \epsilon^2}{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}.$$

Analog können wir zeigen, dass für alle $\check{s} \in \operatorname{desc}(s)$

$$\|D(H^T, \overline{W})_{\check{s},t}\|^2 \leq \frac{\|H_{|\mathbf{t}\times\mathbf{s}}\|^2\epsilon^2}{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}$$

gilt. Durch Einsetzen in die Abschätzung für den blockweisen Fehler erhalten wir

$$\begin{aligned}
\|H_{|\mathbf{t}\times\mathbf{s}} - \overline{H}_{|\mathbf{t}\times\mathbf{s}}\|^2 &\leq \|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 + \|H^T_{|\mathbf{b}} - \Pi_{\overline{W}_s} H^T_{|\mathbf{b}}\|^2 \\
&\leq \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H, \overline{V})_{\check{t},s}\|^2 + \sum_{\check{s}\in\operatorname{desc}(s)} \|D(H^T, \overline{W})_{\check{s},t}\|^2 \\
&\leq \sum_{\check{t}\in\operatorname{desc}(t)} \frac{\|H_{|\mathbf{t}\times\mathbf{s}}\|^2\epsilon^2}{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))} + \sum_{\check{s}\in\operatorname{desc}(s)} \frac{\|H_{|\mathbf{t}\times\mathbf{s}}\|^2\epsilon^2}{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))} \\
&\leq \epsilon^2\|H_{|\mathbf{t}\times\mathbf{s}}\|^2.
\end{aligned}$$

■

Mit der Strategie erhalten wir also eine relative Fehlerschranke für jeden Block. Mit Theorem 4.3.9 folgt daraus eine Abschätzung für den globalen Fehler. Dabei können wir die Normen in den Blöcken gegen die globale Norm abschätzen, sodass wir, wie bei der Strategie 1, auch eine relative globale Schranke erhalten.
Als Letztes stellen wir eine Strategie vor, die die adaptiven Gewichte nutzt. Dabei wird die Summe über die Nachfahren eines Clusters mit Hilfe der geometrische Reihe aufgefangen.

Lemma 4.5.5 (Strategie 3)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit $\operatorname{chil}(t), \operatorname{chil}(s) \leq c_{\text{chil}}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$. Weiter sei $p \in (0, \frac{1}{\sqrt{c_{\text{chil}}}})$, $\theta_{\mathcal{I},t} := \theta_t := p$ für alle $t \in \mathcal{T}_{\mathcal{I}}$, $\theta_{\mathcal{J},s} := \theta_s := p$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ und

$$\omega_{\mathcal{I},(t,s)} := \omega_{\mathcal{J},(t,s)} := \begin{cases} 1 & \text{, falls } \|H_{|\mathbf{t}\times\mathbf{s}}\| = 0 \\ \|H_{|\mathbf{t}\times\mathbf{s}}\| & \text{, falls } \|H_{|\mathbf{t}\times\mathbf{s}}\| \neq 0 \end{cases}$$

für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Es sei $\epsilon \in \mathbb{R}_{\geq 0}$, $\epsilon_t := \epsilon_s := \tilde{\epsilon} := \epsilon\sqrt{\frac{1}{2}(1 - p^2 c_{\text{chil}})}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ und *opt* der absolute Fehler bezüglich $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Weiter sei $\overline{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ die mit Algorithmus 4.5.1 zu den oben definierten Optionen berechnete neue $\mathcal{H}^2$-Matrix. Dann ist der Fehler der Rekompression in jedem zulässigen Blatt $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ durch

$$\|H_{|\mathbf{t}\times\mathbf{s}} - \overline{H}_{|\mathbf{t}\times\mathbf{s}}\| \leq \epsilon\|H_{|\mathbf{t}\times\mathbf{s}}\|$$

beschränkt.

Beweis:[vergleiche [12, Remark 6.35]]
Es sei $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Für den Fall $\|H_{|\mathbf{t}\times\mathbf{s}}\| = 0$ gilt mit (4.18), der blockweise Fehler

gleich null ist. Da in diesem Fall auch auf der rechten Seite null steht, folgt die Aussage. Im Folgenden sei $\|H_{|\mathbf{t}\times\mathbf{s}}\| \neq 0$. Wie im Beweis von Theorem 4.3.9 gilt

$$\begin{aligned}\|H_{|\mathbf{t}\times\mathbf{s}} - \overline{H}_{|\mathbf{t}\times\mathbf{s}}\|^2 &= \|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}} \Pi_{\overline{W}_s}\|^2 \\ &\leq \|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 + \|H_{|\mathbf{b}}^T - \Pi_{\overline{W}_s} H_{|\mathbf{b}}^T\|^2.\end{aligned} \tag{4.19}$$

Wir betrachten den Fehler bezüglich der Zeilenbasis. Für diesen gilt nach Lemma 4.3.11

$$\|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 \leq \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H,\overline{V})_{\check{t},s}\|^2.$$

Mit $\ell_{t'} := \operatorname{level}(t')$ für alle $t' \in \mathcal{T}_{\mathcal{I}}$ gilt nach Lemma 4.4.21 für alle $\check{t} \in \operatorname{desc}(t)$

$$\begin{aligned}\|D(H,\overline{V})_{\check{t},s}\| &\leq \omega_{(t,s)} \left(\prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \right) \|(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}}) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\check{t}}\| \\ &= \omega_{(t,s)} p^{\ell_{\check{t}} - \ell_t} \|(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}}) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\check{t}}\|.\end{aligned}$$

Weil wir als Fehlerart *opt* den absoluten Fehler gewählt haben, folgt aus Lemma 4.4.9 (iii) für alle $\check{t} \in \operatorname{desc}(t)$

$$\| \left(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}} \right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}}\| \leq \epsilon_t = \tilde{\epsilon}.$$

Zusammen mit $\omega_{(t,s)} = \|H_{|\mathbf{t}\times\mathbf{s}}\|$ folgt für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\| \leq \omega_{(t,s)} p^{\ell_{\check{t}} - \ell_t} \| \left(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}} \right) \widehat{V}(V,\overline{V})_{\check{t}} Z_{\omega,\check{t}}\| \leq \|H_{|\mathbf{t}\times\mathbf{s}}\| p^{\ell_{\check{t}} - \ell_t} \tilde{\epsilon}.$$

Durch Einsetzen in die Abschätzung für den blockweisen Fehler erhalten wir

$$\|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 \leq \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H,\overline{V})_{\check{t},s}\|^2 \leq \sum_{\check{t}\in\operatorname{desc}(t)} \|H_{|\mathbf{t}\times\mathbf{s}}\|^2 p^{2(\ell_{\check{t}} - \ell_t)} \tilde{\epsilon}^2.$$

Um die Summe weiter abzuschätzen, verwenden wir die geometrische Reihe. Weil die Anzahl der Knoten in $t' \in \operatorname{desc}(t) = \mathcal{T}_{\mathbf{t}}$ mit $\operatorname{level}_{\mathcal{T}_{\mathbf{t}}}(t') = \ell$ durch c_{chil}^{ℓ} beschränkt ist und nach Voraussetzung $p^2 c_{\mathrm{chil}} < 1$ gilt, folgt

$$\sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}} - \ell_t)} \leq \sum_{\ell=0}^{\operatorname{depth}(\mathcal{T}_{\mathbf{t}})} p^{2l} c_{\mathrm{chil}}^{\ell} \leq \sum_{\ell=0}^{\infty} (p^2 c_{\mathrm{chil}})^{\ell} = \frac{1}{1 - p^2 c_{\mathrm{chil}}}.$$

Daraus folgt

$$\begin{aligned}\|H_{|\mathbf{b}} - \Pi_{\overline{V}_t} H_{|\mathbf{b}}\|^2 &\leq \tilde{\epsilon}^2 \|H_{|\mathbf{t}\times\mathbf{s}}\|^2 \sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}} - \ell_t)} \leq \frac{1}{2}(1 - p^2 c_{\mathrm{chil}}) \epsilon^2 \|H_{|\mathbf{t}\times\mathbf{s}}\|^2 \frac{1}{1 - p^2 c_{\mathrm{chil}}} \\ &= \frac{\epsilon^2}{2} \|H_{|\mathbf{t}\times\mathbf{s}}\|^2.\end{aligned}$$

Analog erhalten wir für den Fehler bezüglich der neuen Spaltenbasis

$$\|H_{|\mathbf{b}}^T - \Pi_{\overline{W}_t} H_{|\mathbf{b}}^T\|^2 \;=\; \frac{\epsilon^2}{2}\|H_{|\mathbf{t}\times\mathbf{s}}\|^2.$$

Die Abschätzung für die Zeilenbasis und die Spaltenbasis setzen wir in die Fehlerschranke (4.19) ein und erhalten die Aussage. ∎

Bemerkung 4.5.6
Wie wir im Beweis von Lemma 4.5.5 gesehen, besteht der Schlüssel zur Beseitigung der Summe über $\operatorname{desc}(t)$ bei Strategie 3 daraus, dass die Summe über alle Nachfahren auf einem Level ℓ durch Konstante mal $\tilde{p}^\ell$ mit $\tilde{p} \in (0,1)$ abzuschätzen. Danach können wir die geometrische Reihe anwenden.
Eine alternative Möglichkeit ist, $\tilde{p} \in (0,1)$ zu wählen und $\theta_{\mathcal{I},t} := \sqrt{\tilde{p}/\#\operatorname{chil}(t)}$, $\theta_{\mathcal{J},s} := \sqrt{\tilde{p}/\#\operatorname{chil}(t)}$ und $\epsilon_t := \epsilon_s := \tilde{\epsilon} := (1-\tilde{p})\epsilon^2/2$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ zu setzen. Dann gilt für alle $\check{t} \in \operatorname{desc}(t)\backslash\mathcal{L}_\mathcal{I}$

$$\sum_{\operatorname{chil}(\check{t})} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t\le\ell_{t'}\le\ell_{\check{t}}}} \theta_{t'} \;=\; \theta_{\check{t}} \sum_{\operatorname{chil}(\check{t})} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t\le\ell_{t'}<\ell_{\check{t}}}} \theta_{t'} \;=\; \tilde{p} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t\le\ell_{t'}<\ell_{\check{t}}}} \theta_{t'}$$

und wir könne induktiv obige Bedingung zeigen. Mit diesem Ansatz folgt die gleiche Fehlerabschätzung wie in Lemma 4.5.5.

Bei Strategie 1 wird die relative Fehlerschranke durch die relativen Fehler bei der gekürzten Singulärwertzerlegung in Algorithmus 2.4.3 erreicht (siehe Lemma 2.4.7). Die Strategien 2 und 3 benutzen absolute Fehler bei der gekürzten Singulärwertzerlegung. Die relativen Fehlerschranken resultieren bei diesen Strategien aus der Wahl der Skalierungen $\omega_{(t,s)}$. Der Unterschied zwischen Strategie 2 und 3 ist die Art, wie die Summe über die Nachfahren $\operatorname{desc}(t)$ aufgehoben wird.
Um die Strategien 2 und 3 effizient anwenden zu können, benötigen wir einen Algorithmus zur Berechnung der Normen der Matrixblöcke $H_{|\mathbf{t}\times\mathbf{s}}$ für alle zulässigen Blätter $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Um die Berechnung effizient zu gestalten, gehen wir davon aus, dass orthogonale Gewichte $(O_{\mathcal{I},t})_{t\in\mathcal{T}_\mathcal{I}}$ und $(O_{\mathcal{J},s})_{s\in\mathcal{T}_\mathcal{J}}$ für die Clusterbasen V und W vorliegen. Weil die betrachteten Matrixnormen invariant unter orthogonaler Transformation sind, können wir für $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ die Gleichheit

$$\|H_{|\mathbf{t}\times\mathbf{s}}\| \;=\; \|V_t S_b W_s^T\| \;=\; \|P_{\mathcal{I},t} O_{\mathcal{I},t} S_{(t,s)} O_{\mathcal{J},s}^T P_{\mathcal{J},s}^T\| \;=\; \|O_{\mathcal{I},t} S_{(t,s)} O_{\mathcal{J},s}^T\|$$

nutzen, um den Aufwand zu reduzieren.
Wir berechnen also zuerst die Hilfsmatrix $Y := O_{\mathcal{I},t} S_{(t,s)} O_{\mathcal{J},s}^T$. Die Auswertung der Frobeniusnorm von Y kann direkt berechnet werden. Die Spektralnorm kann z.B. über die Singulärwertzerlegung berechnet werden. Wir verwenden stattdessen die näherungsweise Berechnung per Vektoriteration, wobei erfahrungsgemäß 10 bis 20 Iterationsschritte ausreichen. Deshalb gehen wir davon aus, dass der Aufwand zur Berechnung der Matrixnormen sich linear in der Anzahl der Matrixeinträge verhält.

Algorithmus 4.5.2 Die Funktion berechnet die Skalierung ω. Die Kopplungsmatrizen S die orthogonalen Gewichte $O_{\mathcal{I}}$ und $O_{\mathcal{J}}$ sowie die Strategie *str* werden übergeben. Für Strategie 2 wird angenommen, dass $\#\operatorname{desc}(t)$ und $\#\operatorname{desc}(s)$ direkt vorliegen.

```
function SKALIERUNG(b, S, κ, λ, O_I, ρ_I, O_J, ρ_J, opt, str, ω)
    if b ∈ L^+_{I×J} then
        (t, s) ← b
        if opt == Spektralnorm then
            ‖.‖ ← ‖.‖_S
        else if opt == Frobeniusnorm then
            ‖.‖ ← ‖.‖_F
        end if
        if str == Strategie 1 then
            ω_b ← 1
        else if str == Strategie 2 then
            Y_b ← O_{I,t} S_b O^T_{J,s} ∈ R^{ρ_{I,t} × ρ_{J,s}}
            ω_b ← ‖Y_b‖ / sqrt(#desc(t) + #desc(s))
            if ω_b == 0 then
                ω_b ← 1
            end if
        else if str == Strategie 3 then
            Y_b ← O_{I,t} S_b O^T_{J,s} ∈ R^{ρ_{I,t} × ρ_{J,s}}
            ω_b ← ‖Y_b‖
            if ω_b == 0 then
                ω_b ← 1
            end if
        end if
    else if chil(b) ≠ ∅ then
        for b̆ ∈ chil(b) do
            SKALIERUNG(b̆, S, κ, λ, O_I, ρ_I, O_J, ρ_J, opt, str, ω)
        end for
    end if
end function
```

Lemma 4.5.7
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen κ und λ. Weiter seien orthogonale Gewichte $O_{\mathcal{I}}$ und $O_{\mathcal{J}}$ für die Clusterbasen V und W mit Rangverteilungen ρ_I bzw. $\rho_{\mathcal{J}}$ gegeben. Wir definieren das Maximum der lokalen Ränge

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t,\ \max_{t\in\mathcal{T}_{\mathcal{I}}} \#\rho_{\mathcal{I},t},\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s,\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\rho_{\mathcal{J},s}\}$$

Es sei $c_{\|.\|} > 0$ derart, dass der Aufwand zur Berechnung beide Matrixnormen $\|.\| \in \{\|.\|_S, \|.\|_F\}$ für alle Indexmengen $\mathcal{K}_1, \mathcal{K}_2$ und alle Matrizen $X \in \mathbb{R}^{\mathcal{K}_1\times\mathcal{K}_2}$ kleiner als $c_{\|.\|}\#\mathcal{K}_1\#\mathcal{K}_2$ ist. Dann ist der Aufwand für die Berechnung der Skalierung $(\omega_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ für die Fehlerart *opt* und die Strategie *str* mit Algorithmus 4.5.2 beschränkt durch

$$(7 + c_{\|.\|})k_{\max}^3 \#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \ \leq\ (7 + c_{\|.\|})c_{sp}k_{\max}^3 \min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}\}.$$

Beweis: Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Das Aufstellen der Matrix Y_b benötigt höchstens

$$2\#\rho_{\mathcal{I},t}\#\kappa_t\#\lambda_s + 2\#\rho_{\mathcal{I},t}\#\lambda_s\#\rho_{\mathcal{J},t} \leq 4k_{\max}^3$$

Operationen. Der Aufwand für die Berechnung der Norm der $\rho_{\mathcal{I},t} \times \rho_{\mathcal{J},t}$-Matrix Y gilt nach Voraussetzung die obere Schranke

$$c_{\|.\|}\#\rho_{\mathcal{I},t}\#\rho_{\mathcal{J},t} \leq c_{\|.\|}k_{\max}^2 \leq c_{\|.\|}k_{\max}^3.$$

Die drei Operationen die im Falle von Strategie 2 zusätzlich benötigt werden, schätzen wir gegen $3k_{\max}^3$ ab, um eine einheitlich Abschätzung zu erhalten. Zusammen mit der Abschätzung für $\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ aus Lemma 3.3.6 folgt die Aussage. ∎

Bemerkung 4.5.8
Falls die Clusterbasen V und W orthogonal sind, können wir als orthogonale Gewichte die Identität wählen. In Algorithmus 4.5.2 erhalten wir dann $Y_b = S_b$. Also muss dass Produkt nicht berechnet werden und wir können direkt die Norm von S_b verwenden. Dies führt zu einem Aufwand von höchstens $(3 + c_{\|.\|})c_{sp}k_{\max}^2 \min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}\}$.

Bei den Strategien 2 und 3 zu den blockweisen Fehlerabschätzungen muss berücksichtigt werden, dass sie bei gleichen Rängen aufwendiger sind als die Strategie 1. Dies ist etwas, dass in Experimenten beobachtet werden kann. So sparen die blockweisen Strategien bei FEM-Problemen typischerweise Speicher im Vergleich zu Strategie 1 bei ähnlichen Genauigkeiten. Dies spiegelt geringere lokale Ränge wider. Allerdings sind die Rechenzeiten nicht in dem Maße kleiner, wie es die Verringerung der lokalen Ränge erhoffen ließe. Dies liegt am zusätzlichen Aufwand für die Berechnung der Skalierungen.

5 Niedrigrangupdates für $\mathcal{H}^2$-Matrizen

In diesem Kapitel beschreiben wir, wie die Summe einer $\mathcal{H}^2$-Matrix H und einer Niedrigrangmatrix R approximativ berechnet werden kann. Dabei nehmen wir an, dass die Niedrigrangmatrix nur in einem Block $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ Nicht-null-Einträge besitzt, wie es bei der Arithmetik in Kapitel 6 der Fall ist. Zuerst stellen wir die exakte Summe als neue $\mathcal{H}^2$-Matrix dar und komprimieren diese dann mit einer Adaption des in Kapitel 4 beschriebenen Rekompressions-Algorithmus. Diese grundlegende Idee wird auch in [17] und [4] verwendet, wobei wir an dieser Stelle das Niedrigrangupdate detaillierter untersuchen. Um den Aufwand zu reduzieren, benutzen wir im adaptierten Algorithmus die neue $\mathcal{H}^2$-Matrix nur implizit. Außerdem komprimieren wir nur den Block, der die Nicht-null-Einträge der Niedrigrangmatrix umfasst.
Bei der in diesem Kapitel vorgestellten Variante des Niedrigrangupdates können die Updates auf verschiedene Blöcke nacheinander angewendet werden, ohne zusätzliche Hilfsfunktionen für die Anpassung der Darstellung aufrufen zu müssen. In Kapitel 7 gehen wir davon aus, das der Blockbaum rekursiv durchlaufen wird. Deshalb nutzen wir bei diesen Alternativen Hilfsfunktionen für das Durchlaufen des Blockbaums.
Das Kapitel ist folgendermaßen strukturiert. In Abschnitt 5.1 stellen wir die $\mathcal{H}^2$-Matrix-Darstellung der Summe $H + R$ vor. Dabei untersuchen wir auch, welche Teile sich von der $\mathcal{H}^2$-Matrix-Darstellung von H unterscheiden. Danach führen wir in Abschnitt 5.2 die lokale Version der $\mathcal{H}^2$-Matrix-Projektion ein. Insbesondere betrachten wir, welche Teile der Matrix verändert werden und wie der lokale Projektionsfehler aussieht.
Die algorithmische Umsetzung beschreiben wir in Abschnitt 5.3. Dabei achten wir darauf, dass alle Änderungen lokal berechnet werden. Zusätzlich wollen wir die $\mathcal{H}^2$-Matrix-Darstellung von $H + R$ nicht explizit aufstellen. Deshalb werden die Algorithmen nur in Größen von H und R formuliert. Außerdem zeigen wir, dass die Voraussetzungen für die Niedrigrangupdates nach Durchführung eines Updates wieder erfüllt sind, und schätzen den Aufwand für das Niedrigrangupdate ab.
In Abschnitt 5.4 schätzen wir den Fehler für das Niedrigrangupdate ab, wobei wir die drei Strategien für die Rekompression aus Abschnitt 4.5 betrachten. Insbesondere gehen wir auf die Besonderheiten ein, die durch die lokale Durchführung entstehen.

5.1 $\mathcal{H}^2$-Darstellung des Niedrigrangupdates

Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix und $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ gegeben mit $A \in \mathcal{K}^{\mathcal{I}\times\mathcal{K}}$, $B \in \mathcal{K}^{\mathcal{J}\times\mathcal{K}}$. Als Erstes stellen wir $H + R$ als neue $\mathcal{H}^2$-Matrix dar. Für zulässige

Blätter $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ hat $H + R$ die Drei-Term-Darstellung

$$V_t S_b W_s^T + A_{|\mathbf{t}} B_{|\mathbf{s}}^T = \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} \begin{pmatrix} S_b & 0 \\ 0 & I_{\mathcal{K}} \end{pmatrix} \begin{pmatrix} W_s & B_{|\mathbf{s}} \end{pmatrix}^T. \tag{5.1}$$

Für unzulässige Blätter $b = (t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ können wir $H + R$ als vollbesetzte Matrix $N_b + A_{|\mathbf{t}} B_{|\mathbf{s}}^T$ speichern. Diese beiden Darstellungen liefern uns den Ansatz für eine $\mathcal{H}^2$-Matrix-Darstellung für $H + R$.
Um das Niedrigrangupdate möglichst lokal zu berechnen, nehmen wir an, dass wir einen Block (nicht notwendigerweise ein Blatt) $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ haben, der alle Nicht-null-Einträge von R umfasst, d.h. $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$. Mit Hilfe von Fallunterscheidungen verändern wir die Clusterbasen, Nahfeld- und Kopplungsmatrizen nur dort, wo eine Verbindung zu $\tilde{b}$ besteht. Dabei verwenden wir für die Nachfahren jeweils die Teilbaum-Schreibweise $\mathcal{T}_{\tilde{\mathbf{b}}} = \operatorname{desc}(\tilde{b})$, $\mathcal{T}_{\tilde{\mathbf{t}}} = \operatorname{desc}(\tilde{t})$ und $\mathcal{T}_{\tilde{\mathbf{s}}} = \operatorname{desc}(\tilde{s})$, um zu verdeutlichen, dass wir nur in diesen Teilbäumen arbeiten. Zuerst definieren wir eine geeignete Clusterbasis.

Definition 5.1.1 (Erweiterte Clusterbasis)
Es sei $V = ((V_t)_{t\in\mathcal{T}_\mathcal{I}}, (E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine Clusterbasis, $A \in \mathcal{K}^{\mathcal{I}\times\mathcal{K}}$ und $\tilde{t} \in \mathcal{T}_\mathcal{I}$. Dann definiert

$$\widetilde{V}(V, A, \tilde{t})_t := \begin{cases} V_t & , \ t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \\ \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} & , \ t \in \mathcal{T}_{\tilde{\mathbf{t}}} \end{cases} \tag{5.2}$$

für alle $t \in \mathcal{T}_\mathcal{I}$ die *erweiterte Clusterbasis*, welche aus V entsteht, indem V für alle Nachfahren $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ um $A_{|\mathbf{t}}$ erweitert wird. Die *erweiterten Transfermatrizen* seien für alle $t \in \mathcal{T}_\mathcal{I}\backslash r_\mathcal{I}$ und $\acute{t} := \operatorname{par}(t)$ durch

$$\widetilde{E}(E, \mathcal{K}, \tilde{t})_t := \begin{cases} E_t & , \ t, \acute{t} \notin \mathcal{T}_{\tilde{\mathbf{t}}} \\ \begin{pmatrix} E_t \\ 0 \end{pmatrix} & , \ t \in \mathcal{T}_{\tilde{\mathbf{t}}}, \ \acute{t} \notin \mathcal{T}_{\tilde{\mathbf{t}}} \\ \begin{pmatrix} E_t & 0 \\ 0 & I_{\mathcal{K}} \end{pmatrix} & , \ t, \acute{t} \in \mathcal{T}_{\tilde{\mathbf{t}}} \end{cases} \tag{5.3}$$

definiert.

Es sei darauf hingewiesen, dass für die Transfermatrizen die Matrix A keine Rolle spielt. Wie in Abschnitt 3.4 beschrieben, werden die Transfermatrizen benötigt, um die Clusterbasis effizient in geschachtelter Form zu speichern. In Lemma 5.1.2 zeigen wir, dass die erweiterte Clusterbasis wieder eine Clusterbasis ist.

Lemma 5.1.2 (Clusterbasis des Niedrigrangupdates)
Es sei $V = ((V_t)_{t\in\mathcal{T}_\mathcal{I}}(E_t)_{t\in\mathcal{T}_\mathcal{I}})$ eine Clusterbasis, $A \in \mathcal{K}^{\mathcal{I}\times\mathcal{K}}$ und $\tilde{t} \in \mathcal{T}_\mathcal{I}$. Dann definiert $\widetilde{V} := \left(\widetilde{V}(V, A, \tilde{t}), \widetilde{E}(E, \mathcal{K}, \tilde{t})\right)$ mit den in Definition 5.1.1 beschriebenen Matrizen eine Clusterbasis.

Beweis: Wir verwenden die Schreibweisen $\widetilde{V}_t := \widetilde{V}(V, A, \tilde{t})_t$ und $\widetilde{E}_t := \widetilde{E}(E, \mathcal{K}, \tilde{t})_t$ für alle $t \in \mathcal{T}_\mathcal{I}$. Es gilt $\Pi_\mathbf{t} \widetilde{V}_t = \widetilde{V}_t$ für alle $t \in \mathcal{T}_\mathcal{I}$. Es bleibt also für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ die geschachtelte Darstellung aus (3.25) zu zeigen. Wir verwenden die äquivalente Umformulierung (3.26). Es sei $t \in \mathcal{T}_\mathcal{I} \backslash \{r_\mathcal{I}\}$ und $\acute{t} := \text{par}(t)$. Es ist $\widetilde{V}_t \widetilde{E}_t = \widetilde{V}_{\acute{t}|\mathbf{t}}$ zu zeigen.
1. Fall: Es gelte $t, \acute{t} \notin \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt

$$\widetilde{V}_t \widetilde{E}_t \;=\; V_t E_t \;=\; V_{\acute{t}|\mathbf{t}} \;=\; \widetilde{V}_{\acute{t}|\mathbf{t}}.$$

2. Fall: Es gelte $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $\acute{t} \notin \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt

$$\widetilde{V}_t \widetilde{E}_t \;=\; \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} \begin{pmatrix} E_t \\ 0 \end{pmatrix} \;=\; V_t E_t \;=\; V_{\acute{t}|\mathbf{t}} \;=\; \widetilde{V}_{\acute{t}|\mathbf{t}}.$$

3. Fall: Es gelte $t, \acute{t} \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt

$$\widetilde{V}_t \widetilde{E}_t \;=\; \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} \begin{pmatrix} E_t & 0 \\ 0 & I_\mathcal{K} \end{pmatrix} \;=\; \begin{pmatrix} V_t E_t & A_{|\mathbf{t}} \end{pmatrix} \;=\; \begin{pmatrix} V_{\hat{t}} & A_{|\acute{\mathbf{t}}} \end{pmatrix}_{|\mathbf{t}} \;=\; \widetilde{V}_{\acute{t}|\mathbf{t}}.$$

■

Bemerkung 5.1.3
Formal müssten wir für die erweiterte Clusterbasis eine neue Rangverteilung $(\widetilde{\kappa}_t)_{t \in \mathcal{T}_\mathcal{I}}$ wählen, deren Elemente paarweise disjunkt sind. Um die Ausdrücke leserlich zu halten, verzichten wir darauf und identifizieren einfach $\widetilde{\kappa}_t$ mit $\kappa_t \cup \mathcal{K}$.

Nach der Definition der Clusterbasen benötigen wir für die Darstellung der Summe $H + R$ als $\mathcal{H}^2$-Matrix noch die Kopplungsmatrizen für die zulässigen Blätter. Diese geben wir in Definition 5.1.4 an.

Definition 5.1.4 (Erweiterte Kopplungsmatrizen)
Es sei $H = \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S\right)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Für alle Blöcke $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ definiere

$$\widetilde{S}(S, \mathcal{K}, \tilde{b})_b \;:=\; \begin{cases} S_b & ,\; t \notin \mathcal{T}_{\tilde{\mathbf{t}}}, s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \\ \begin{pmatrix} S_b \\ 0 \end{pmatrix} & ,\; t \in \mathcal{T}_{\tilde{\mathbf{t}}}, s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \\ \begin{pmatrix} S_b & 0 \end{pmatrix} & ,\; t \notin \mathcal{T}_{\tilde{\mathbf{t}}}, s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ \begin{pmatrix} S_b & 0 \\ 0 & I_\mathcal{K} \end{pmatrix} & ,\; t \in \mathcal{T}_{\tilde{\mathbf{t}}}, s \in \mathcal{T}_{\tilde{\mathbf{s}}} \end{cases} \tag{5.4}$$

die *erweiterten Kopplungsmatrizen*, wobei $I_\mathcal{K}$ die Identität bezeichnet.

Weil die Matrizen A und B in die Clusterbasen aufgenommen werden (siehe (5.2)), sind die neuen Kopplungsmatrizen nur von $\mathcal{K}$ und den alten Kopplungsmatrizen S_b abhängig. Als Nächstes betrachten wir, welche Matrix durch die erweiterten Clusterbasen und Kopplungsmatrizen dargestellt wird.

Lemma 5.1.5 (Kopplungsmatrizen des Niedrigrangupdates)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Weiter seien $\widetilde{V} := \widetilde{V}(V, A, \tilde{t})$ und $\widetilde{W} := \widetilde{V}(W, B, \tilde{s})$ die erweiterten Clusterbasen (siehe Definition 5.1.1) sowie $\widetilde{S} := \widetilde{S}(S, \mathcal{K}, \tilde{b})$ die erweiterten Kopplungsmatrizen (siehe Definition 5.1.4). Dann gilt für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$

$$\widetilde{V}_t \widetilde{S}_b \widetilde{W}_s^T = \begin{cases} (H+R)_{|\mathbf{b}} & , b \in \mathcal{T}_{\tilde{\mathbf{b}}} \\ H_{|\mathbf{b}} & , b \notin \mathcal{T}_{\tilde{\mathbf{b}}} \end{cases}.$$

Beweis: Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.
1. Fall: Es gelte $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt insbesondere $b \in \mathcal{T}_{\tilde{\mathbf{b}}}$ und es folgt

$$\begin{aligned} \widetilde{V}_t \widetilde{S}_b \widetilde{W}_s^T &= \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} \begin{pmatrix} S_b & 0 \\ 0 & I_{\mathcal{K}} \end{pmatrix} \begin{pmatrix} W_s & B_{|\mathbf{s}} \end{pmatrix}^T \\ &= \begin{pmatrix} V_s S_b & A_{|\mathbf{t}} \end{pmatrix} \begin{pmatrix} W_s & B_{|\mathbf{s}} \end{pmatrix}^T \\ &= V_t S_b W_s^T + A_{|\mathbf{t}} B_{|\mathbf{s}}^T \\ &= (H+R)_{|\mathbf{b}}. \end{aligned}$$

2. Fall: Es gelte $t \notin \mathcal{T}_{\tilde{\mathbf{b}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt insbesondere $b \notin \mathcal{T}_{\tilde{\mathbf{b}}}$ und es folgt

$$\widetilde{V}_t \widetilde{S}_b \widetilde{W}_s^T = V_t \begin{pmatrix} S_b & 0 \end{pmatrix} \begin{pmatrix} W_s & B_{|\mathbf{s}} \end{pmatrix}^T = V_t S_b W_s^T = H_{|\mathbf{b}}.$$

3. Fall: Es gelte $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt insbesondere $b \notin \mathcal{T}_{\tilde{\mathbf{b}}}$ und es folgt

$$\widetilde{V}_t \widetilde{S}_b \widetilde{W}_s^T = \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} \begin{pmatrix} S_b \\ 0 \end{pmatrix} W_s^T = V_t S_b W_s^T = H_{|\mathbf{b}}.$$

4. Fall: Es gelte $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt insbesondere $b \notin \mathcal{T}_{\tilde{\mathbf{b}}}$ und es folgt

$$\widetilde{V}_t \widetilde{S}_b \widetilde{W}_s^T = V_t S_b W_s^T = H_{|\mathbf{b}}.$$

■

Als Nächstes definieren wir die erweiterte $\mathcal{H}^2$-Matrix. Diese nutzt die erweiterten Clusterbasen und die erweiterten Kopplungsmatrizen.

Definition 5.1.6 (Erweiterte $\mathcal{H}^2$-Matrix)
Es sei $H = \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S\right)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block. Es bezeichnen

$$\widetilde{V} := \left(\widetilde{V}(V, A, \tilde{t}), \widetilde{E}(E, \mathcal{K}, \tilde{t})\right) \quad \text{und} \quad \widetilde{W} := \left(\widetilde{V}(W, B, \tilde{s}), \widetilde{E}(F, \mathcal{K}, \tilde{s})\right)$$

die zugehörigen erweiterten Clusterbasen (siehe Definition 5.1.1),

$$\widetilde{N} \;:=\; \left(N_b + R_{|\mathbf{b}}\right)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \quad \text{und} \quad \widetilde{S} := \widetilde{S}(S, \mathcal{K}, \tilde{b})$$

die erweiterten Nahfeldmatrizen bzw. Kopplungsmatrizen (siehe Definition 5.1.4) der Summe $H + R$. Dann definiert

$$\widetilde{H}(H, R, \tilde{b}) \;:=\; \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{V}, \widetilde{W}, \widetilde{N}, \widetilde{S}\right) \tag{5.5}$$

die erweiterte $\mathcal{H}^2$-Matrix zu H, R und $\tilde{b}$.

In den letzten Definitionen und Lemmata haben wir den Cluster $\tilde{t}$ bzw. den Block $\tilde{b}$ beliebig gewählt. Für die Zwecke des Niedrigrangupdates sollte ein Block $\tilde{b}$ gewählt werden, für den alle Nicht-null-Einträge von R im Block enthalten sind. Der Block $\tilde{b}$ sollte möglichst klein sein, um die Anzahl der veränderten Blöcke und Cluster und damit den Aufwand zu minimieren. In der Arithmetik wählen wir $\tilde{b}$ als den Block, in dem die Niedrigrangmatrix berechnet wird. Das folgende Theorem zeigt, dass die erweiterte $\mathcal{H}^2$-Matrix aus (5.5) für einen alle Nicht-null-Einträge umfassenden Block $\tilde{b}$ tatsächlich eine $\mathcal{H}^2$-Darstellung für $H + R$ liefert.

Theorem 5.1.7 ($\mathcal{H}^2$-Matrix des Niedrigrangupdates)
Es sei $H = \mathcal{H}^2\left(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S\right)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$, d.h. $R_{i,j} = 0$ für alle $(i, j) \in (\mathcal{I}\times\mathcal{J})\backslash\tilde{\mathbf{b}}$. Dann gilt für die Matrizen aus Definition 5.1.6

$$H + R \;=\; \widetilde{H}(H, R, \tilde{b}).$$

Beweis: Wir zeigen die Gleichheit blockweise. Es sei $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$.
1. Fall: Es sei $b \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$. Dann gilt

$$(H + R)_{|\mathbf{b}} \;=\; N_b + R_{|\mathbf{b}} \;=\; \widetilde{N}_b \;=\; \widetilde{H}(H, R, \tilde{b})_{|\mathbf{b}}.$$

2. Fall: Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Nach Definition 5.1.6 und Lemma 5.1.5 gilt

$$\widetilde{H}(H, R, \tilde{b})_{|\mathbf{b}} \;=\; \widetilde{V}_t \widetilde{S}_b \widetilde{W}_s^T \;=\; \begin{cases} (H + R)_{|\mathbf{b}} & ,\ b \in \mathcal{T}_{\tilde{\mathbf{b}}} \\ H_{|\mathbf{b}} & ,\ b \notin \mathcal{T}_{\tilde{\mathbf{b}}} \end{cases}.$$

Durch die Wahl von $\tilde{b}$ folgt

$$\widetilde{H}(H,R,\tilde{b})_{|\mathbf{b}} \;=\; \begin{cases} (H+R)_{|\mathbf{b}} & , \; b \in \mathcal{T}_{\tilde{\mathbf{b}}} \\ H_{|\mathbf{b}} & , \; b \notin \mathcal{T}_{\tilde{\mathbf{b}}} \end{cases} \;=\; (H+R)_{|\mathbf{b}}.$$

■

Somit haben wir eine $\mathcal{H}^2$-Matrix-Darstellung von $H+R$ gefunden. Diese könnten wir mit Algorithmus 4.5.1 rekomprimieren. Allerdings würde dieser Ansatz die gesamte Matrix behandeln und somit zu einem Aufwand in $O(\#\mathcal{I}+\#\mathcal{J})$ führen. Damit könnten wir keine effiziente Arithmetik konstruieren.
Deshalb werden wir nur in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$, in denen die Clusterbasen angepasst werden, adaptive Clusterbasen berechnen und diese dann in die alten integrieren (siehe Lemma 5.2.1). Dieses Vorgehen ist dadurch motiviert, dass wir davon ausgehen, dass die Darstellung der Matrix H effizient ist.
Zur Berechnung der adaptiven Clusterbasen benötigen wir Gewichte. Um das Aufstellen der neuen Clusterbasen lokal zu halten, wollen wir sicherstellen, dass außerhalb der Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ die Gewichte der Matrix H auch Gewichte für die erweiterte Matrix $\widetilde{H}$ sind. Weil die adaptiven totalen Clusterbasen zur Definition der adaptiven Gewichte verwendet werden, zeigen wir in Lemma 5.1.8, dass sich diese außerhalb von $\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{s}}}$ nicht verändern.

Lemma 5.1.8
Es sei $H=(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W,N,S)$ eine H^2-Matrix, $R=AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung, $\tilde{b}=(\tilde{t},\tilde{s})\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\widetilde{H}=\widetilde{H}(H,R,\tilde{b})$ die erweiterte $\mathcal{H}^2$-Matrix. Weiter seien $\omega,\tilde{\omega}\in\mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta,\tilde{\theta}\in\mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen, wobei $\tilde{\omega}_{(t,s)}=\omega_{(t,s)}$ und $\tilde{\theta}_t=\theta_t$ für alle $t\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\in\operatorname{row}(t)$ gelte. Dann gilt für alle $t\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},H,\omega,\theta)_t \;=\; X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\widetilde{H},\tilde{\omega},\tilde{\theta})_t.$$

Beweis: Für $\tilde{t}=r_{\mathcal{I}}$ ist die Aussage wegen $\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}=\emptyset$ trivial. Es sei also $\tilde{t}\neq r_{\mathcal{I}}$.
Für alle $t\in\mathcal{T}_{\mathcal{I}}$ bezeichne $\ell_t:=\operatorname{level}(t)$. Es sei $t\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$. Dann ist $\hat{t}\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ für alle $\hat{t}\in\operatorname{pred}(t)$ und somit $\theta_{\hat{t}}=\widetilde{\theta}_{\hat{t}}$. Es sei $s\in\overline{\operatorname{row}}(t)$. Dann existiert nach Definition 4.1.3 der erweiterten Blockzeile $\hat{t}\in\operatorname{pred}(t)$ mit $(\hat{t},s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Wegen $t,\hat{t}\in\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ ist $\tilde{\omega}_{(\hat{t},s)}=\omega_{(\hat{t},s)}$ sowie $\tilde{\theta}_{t'}=\theta_{t'}$ für alle $t\in\operatorname{desc}(\hat{t})$ mit $l_{t'}<l_t$. Also gilt

$$\frac{1}{\omega_{(\hat{t},s)}}\left(\prod_{\substack{t'\in\operatorname{desc}(\hat{t})\\ \ell_{\hat{t}}\le\ell_{t'}<\ell_t}}\theta_{t'}\right)V_{\hat{t}}S_{(\hat{t},s)}W_s^T \;=\; \frac{1}{\tilde{\omega}_{(\hat{t},s)}}\left(\prod_{\substack{t'\in\operatorname{desc}(\hat{t})\\ \ell_{\hat{t}}\le\ell_{t'}<\ell_t}}\tilde{\theta}_{t'}\right)\widetilde{V}_{\hat{t}}\widetilde{S}_{(\hat{t},s)}\widetilde{W}_s^T.$$

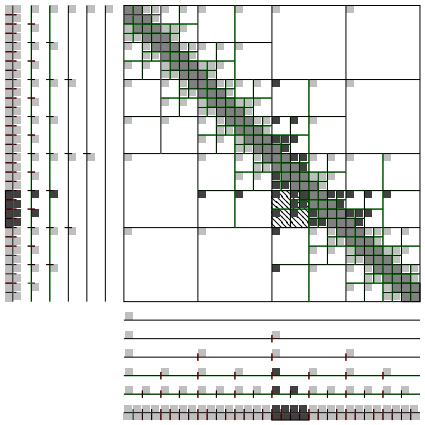


Abbildung 5.1: Erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H}$ in $\tilde{b}$: Der Block $\tilde{b}$ ist schraffiert und die angepassten Matrizen in den Clustern und Blöcken sind dunkelgrau markiert.

Mit Lemma 4.4.16 erhalten wir

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t \Pi_s &= \frac{1}{\omega_{(\hat{t},s)}} \left(\prod_{\substack{t' \in \operatorname{desc}(\hat{t}) \\ \ell_{\hat{t}} \le \ell_{t'} < \ell_t}} \theta_{t'} \right) \Pi_{\mathbf{t}} V_{\hat{t}} S_{(\hat{t},s)} W_s^T \\ &= \frac{1}{\widetilde{\omega}_{(\hat{t},s)}} \left(\prod_{\substack{t' \in \operatorname{desc}(\hat{t}) \\ \ell_{\hat{t}} \le \ell_{t'} < \ell_t}} \widetilde{\theta}_{t'} \right) \Pi_{\mathbf{t}} \widetilde{V}_{\hat{t}} \widetilde{S}_{(\hat{t},s)} \widetilde{W}_s^T = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \widetilde{\omega}, \widetilde{\theta})_t \Pi_s. \end{aligned}$$

Weil $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t, X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \widetilde{\omega}, \widetilde{\theta})_t \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\xi_t}$ für $\overline{\xi}_t = \dot{\bigcup}_{s\in\overline{\operatorname{row}}(t)} \mathbf{s}$ gilt, folgt die Aussage. ∎

Bemerkung 5.1.9
Die Voraussetzungen an die Skalierungen in Lemma 5.1.8 sind für die Strategien aus Abschnitt 4.5 erfüllt, weil sich die erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H}$ von der ursprünglichen Matrix H nur innerhalb des Blocks $\tilde{b}$ unterscheidet.

Die $\mathcal{H}^2$-Matrix-Darstellung der ursprünglichen Matrix H unterscheidet sich von der im Block $\tilde{b} = (\tilde{t}, \tilde{s})$ erweiterten Matrix $\tilde{H}$ nur in den Blöcken $b = (t, s)$, für die entweder der Zeilencluster t ein Nachfahre von $\tilde{t}$ ($t \in \mathcal{T}_{\tilde{\mathbf{t}}}$) oder der Spaltencluster s ein Nachfahre von $\tilde{s}$ ($s \in \mathcal{T}_{\tilde{\mathbf{s}}}$) ist (siehe Abbildung 5.1). Wenn wir annehmen, dass die Darstellung der Matrix H effizient ist, reicht es also aus, nur die zu $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ gehörenden Teile zu rekomprimieren. Deshalb führen wir im nächsten Abschnitt die lokale $\mathcal{H}^2$-Matrix-Projektion ein.

5.2 Lokale $\mathcal{H}^2$-Matrix-Projektion

Wir gehen davon aus, dass eine $\mathcal{H}^2$-Matrix $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ sowie orthogonale Clusterbasen $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ auf den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ gegeben sind. (Wir wollen nur in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ neue Clusterbasen berechnen, weil wir davon ausgehen, dass für die restlichen Cluster die Darstellung der Matrix effizient ist.) Damit wir für die neuen Clusterbasen einen entsprechenden $\mathcal{H}^2$-Matrix-Raum angeben können, müssen wir sie zu Clusterbasen auf $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$ erweitern. Dabei sollen die erweiterten Clusterbasen außerhalb der Teilbäume den alten Clusterbasen entsprechen, damit wir große Teile der $\mathcal{H}^2$-Matrix-Darstellung wiederverwenden können.

Lemma 5.2.1
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ eine orthogonale Clusterbasis auf dem Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$. Wir definieren für alle $t \in \mathcal{T}_\mathcal{I}$

$$\overline{V}_t := \begin{cases} \overline{V}_t & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}} \\ (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}})V_t & \text{, falls } t \in \operatorname{pred}(\tilde{t})\setminus\{\tilde{t}\} \\ V_t & \text{, falls } t \in \mathcal{T}_\mathcal{I}\setminus(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t})) \end{cases}$$

und

$$\overline{E}_t := \begin{cases} \overline{E}_t & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}}\setminus\{\tilde{t}\} \\ C(V,\overline{V})_t E_t & \text{, falls } t = \tilde{t} \\ E_t & \text{, falls } t \in \mathcal{T}_\mathcal{I}\setminus\mathcal{T}_{\tilde{\mathbf{t}}} \end{cases},$$

wobei $C_{\tilde{t}} = C(V,\overline{V})_t = \overline{V}_{\tilde{t}}^T V_{\tilde{t}}$ der Basiswechsel von V zu $\overline{V}$ ist. Dann ist $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis. Falls V_t orthogonal für alle $t \in \mathcal{T}_\mathcal{I}\setminus(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ ist, ist $\overline{V}_t$ orthogonal für alle $t \in \mathcal{T}_\mathcal{I}\setminus(\operatorname{pred}(\tilde{t})\setminus\{\tilde{t}\})$.

Beweis: Wir zeigen, dass $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis ist, und rechnen dazu die Eigenschaften aus Definition 3.4.14 nach. Dazu gehen wir die drei Fälle aus der Definition von $(\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ durch.
1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Weil $\overline{V}$ eine Clusterbasis auf $\mathcal{T}_{\tilde{\mathbf{t}}}$ ist, gilt $\Pi_\mathbf{t}\overline{V}_t = \overline{V}_t$. Falls $t \notin \mathcal{L}_\mathcal{I}$ ist, gilt $\operatorname{chil}(t) \subseteq \mathcal{T}_{\tilde{\mathbf{t}}}$ und es folgt $\overline{V}_t = \sum_{\check{t}\in\operatorname{chil}(t)} \overline{V}_{\check{t}}\overline{E}_{\check{t}}$.
2. Fall: Es sei $t \in \operatorname{pred}(\tilde{t})\setminus\{\tilde{t}\}$. Dann gilt $\Pi_\mathbf{t}\overline{V}_t = \Pi_\mathbf{t}(\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}})V_t = (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}})V_t = \overline{V}_t$. Die geschachtelte Eigenschaft erfordert eine Fallunterscheidung wegen der Fallunterscheidung in der Definition der Transfermatrizen.
2.1. Fall: Es sei $t = \operatorname{par}(\tilde{t})$. Dann ist $\tilde{t} \in \operatorname{chil}(t)$ und es gilt $\breve{t} \in \mathcal{T}_\mathcal{I}\setminus(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ für alle $\breve{t} \in \operatorname{chil}(t)\setminus\{\tilde{t}\}$. Damit gilt

$$\begin{aligned}\sum_{\breve{t}\in\operatorname{chil}(t)} \overline{V}_{\breve{t}}\overline{E}_{\breve{t}} &= \sum_{\breve{t}\in\operatorname{chil}(t)\setminus\{\tilde{t}\}} V_{\breve{t}}E_{\breve{t}} + \overline{V}_{\tilde{t}}C(V,\overline{V})_{\tilde{t}}E_{\tilde{t}} = \Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}} \sum_{\breve{t}\in\operatorname{chil}(t)\setminus\{\tilde{t}\}} V_{\breve{t}}E_{\breve{t}} + \Pi_{\overline{V}_{\tilde{t}}}V_{\tilde{t}}E_{\tilde{t}} \\ &= (\Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}}) \sum_{\breve{t}\in\operatorname{chil}(t)} V_{\breve{t}}E_{\breve{t}} = (\Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}})V_t = \overline{V}_t.\end{aligned}$$

2.2. Fall: Es sei $t \neq \operatorname{par}(\tilde{t})$. Weil die Pfade im Baum $\mathcal{T}_\mathcal{I}$ eindeutig sind, existiert genau ein $t' \in \operatorname{chil}(t)$ mit $t' \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und es gilt $\breve{t} \in \mathcal{T}_\mathcal{I}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ für alle $\breve{t} \in \operatorname{chil}(t)\backslash\{t'\}$. Somit gilt

$$\begin{aligned}\sum_{\breve{t}\in\operatorname{chil}(t)} \overline{V}_{\breve{t}}\overline{E}_{\breve{t}} &= \sum_{\breve{t}\in\operatorname{chil}(t)\backslash\{t'\}} V_{\breve{t}}E_{\breve{t}} + (\Pi_{\mathbf{t}'\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}})V_{t'}E_{t'} \\ &= \Pi_{\mathbf{t}\backslash\mathbf{t}'} \sum_{\breve{t}\in\operatorname{chil}(t)\backslash\{t'\}} V_{\breve{t}}E_{\breve{t}} + (\Pi_{\mathbf{t}'\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}})V_{t'}E_{t'} \\ &= (\Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}}) \sum_{\breve{t}\in\operatorname{chil}(t)} V_{\breve{t}}E_{\breve{t}} = (\Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}})V_t = \overline{V}_t.\end{aligned}$$

Also haben wir für den 2. Fall die geschachtelte Darstellung nachgewiesen.
3. Fall: $t \in \mathcal{T}_\mathcal{I}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$. Weil V eine Clusterbasis ist, gilt $\Pi_\mathbf{t}\overline{V}_t = \Pi_\mathbf{t}V_t = V_t = \overline{V}_t$. Es sei $t \notin \mathcal{L}_\mathcal{I}$. Dann gilt $\operatorname{chil}(t) \subseteq \mathcal{T}_\mathcal{I}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$. Daraus folgt

$$\sum_{\breve{t}\in\operatorname{chil}(t)} \overline{V}_{\breve{t}}\overline{E}_{\breve{t}} = \sum_{\breve{t}\in\operatorname{chil}(t)} V_{\breve{t}}E_{\breve{t}} = V_t = \overline{V}_t.$$

Somit gilt auch im 3. Fall die geschachtelte Darstellung und $\overline{V}$ ist eine Clusterbasis auf $\mathcal{T}_\mathcal{I}$. Die Orthogonalität von $\overline{V}_t$ für $t \in \mathcal{T}_\mathcal{I}\backslash(\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\})$ folgt aus der Definition der neuen Clusterbasis $\overline{V}$ und der Orthogonalität von $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und von V_t für alle $t \in \mathcal{T}_\mathcal{I}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$. ∎

In den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ sind die durch Lemma 5.2.1 definierten Clusterbasen $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ bzw. $\overline{W} = (\overline{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ orthogonal. Allerdings sind sie i. A. nicht in allen Clustern orthogonal und somit können wir für sie nicht die globale $\mathcal{H}^2$-Matrix-Projektion anwenden. Außerdem wollen wir nur lokal Berechnungen durchführen. Deshalb definieren wir die lokale $\mathcal{H}^2$-Matrix-Projektion.

Definition 5.2.2 (Lokale $\mathcal{H}^2$-Matrix-Projektion)
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $(\tilde{t}, \tilde{s}) = \tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen. Wir definieren zuerst für alle $t \in \mathcal{T}_\mathcal{I}$ die Abbildungen

$$\Pi_{\tilde{t},\overline{V},t} := \begin{cases} \Pi_{\overline{V}_t} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}} \\ (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) & \text{, falls } t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\} \\ \Pi_\mathbf{t} & \text{, falls } t \in \mathcal{T}_\mathcal{I}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t})) \end{cases} \tag{5.6}$$

und für alle $s \in \mathcal{T}_\mathcal{J}$ die Abbildung

$$\Pi_{\tilde{s},\overline{W},s} := \begin{cases} \Pi_{\overline{W}_s} & \text{, falls } s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ (\Pi_{\overline{W}_{\tilde{s}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{s}}}) & \text{, falls } s \in \operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\} \\ \Pi_\mathbf{s} & \text{, falls } s \in \mathcal{T}_\mathcal{J}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s})) \end{cases}. \tag{5.7}$$

Mit Hilfe dieser definieren wir $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}} : \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ für alle $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ durch

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X] \;=\; \sum_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} + \sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{\tilde{t},\overline{V},t} X \Pi_{\tilde{s},\overline{W},s}. \tag{5.8}$$

Wir bezeichnen $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}$ als die *lokale $\mathcal{H}^2$-Matrix-Projektion* bezüglich $\tilde{b}$, $\overline{V}$ und $\overline{W}$.

Bemerkung 5.2.3
Für die in Definition 5.2.2 definierten Abbildung gilt für alle $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$

$$\Big(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\Big)^T \;=\; \Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\tilde{b}^T,\overline{W},\overline{V}}[X]^T.$$

Wir können also mit Hilfe des transponierten Blockbaums Aussagen für die Zeilenbasis auf die Spaltenbasis übertragen.

In Abbildung 5.2 sind die Teile der Blockmatrix eingezeichnet, die durch die lokale Projektion verändert werden. Dabei ist zu beachten, dass die unzulässigen Blätter von den Anpassungen ausgenommen sind. Ansonsten wird die Matrix über das gesamte Kreuz verändert.
Im nächsten Schritt zeigen wir, dass $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}$ eine Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ ist. Weil $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}$ Blöcke $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ mit $t \notin (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ und $s \notin (\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s}))$ unverändert lässt, können wir nicht erwarten, dass es sich um eine Projektion auf $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W})$ handelt. Wir zeigen aber, dass die alte $\mathcal{H}^2$-Matrix $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ in den $\mathcal{H}^2$-Matrix-Raum $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W})$ zu den neuen Clusterbasen abgebildet wird, falls $\overline{V}$ und $\overline{W}$ wie in Lemma 5.2.1 definiert werden. Dabei geben wir auch die neue $\mathcal{H}^2$-Matrix-Darstellung an.

Lemma 5.2.4
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $V = (V_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $W = (W_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ Clusterbasen. Weiter seien $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen. Es seien $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $\overline{W} = (\overline{W}_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ die nach Lemma 5.2.1 definierten Clusterbasen. Dann ist $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ mit

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)] = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$$

für alle $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S) \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$, wobei $\overline{N}_b := N_b$ für alle $b \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ und

$$\overline{S}_b \;:=\; \begin{cases} C(V,\overline{V})_t S_b C(W,\overline{W})_s^T & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ S_b C(W,\overline{W})_s^T & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ C(V,\overline{V})_t S_b & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \\ S_b & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}},\ s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \end{cases} \quad \text{für alle } b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \text{ sei.}$$

(Dabei sind $C(V,\overline{V})$ und $C(W,\overline{W})$ die Basiswechsel aus Definition 4.2.3.)

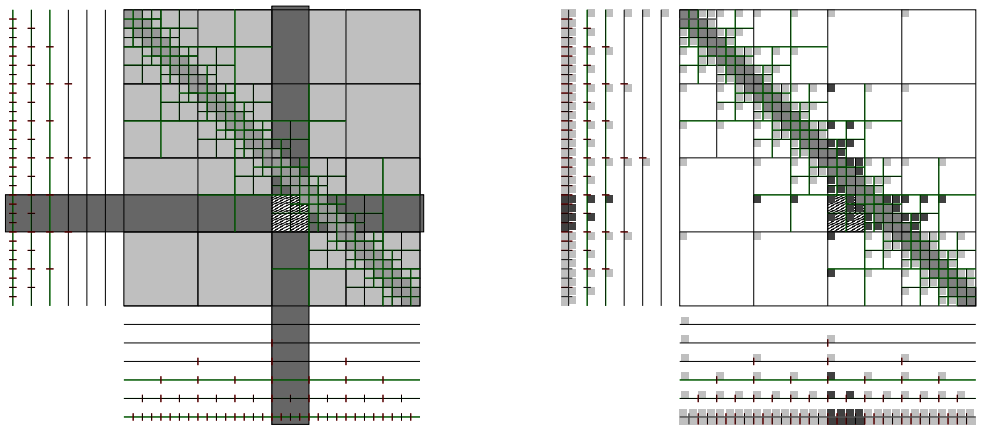


Abbildung 5.2: Veränderungen durch lokale Projektion: In der linken Abbildung ist der Bereich der Blockmatrix markiert, der durch die Projektion verändert wird, und in der rechten Abbildung sind die Matrizen der $\mathcal{H}^2$-Matrix-Darstellung markiert, die für die Projektion angepasst werden müssen.

Beweis: Für alle $t \in \mathcal{T}_\mathcal{I}$ ist $\Pi_\mathbf{t}$ nach Beispiel 2.3.17 eine orthogonale Projektion mit $\text{Bild}(\Pi_\mathbf{t}) = \mathbb{R}^\mathcal{I}_\mathbf{t}$. Das Gleiche gilt nach Beispiel 2.3.18 für $\Pi_{\overline{V}_t}$ für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$, weil $\overline{V}_t$ für diese Cluster orthogonal ist. Nach Lemma 2.1.6 folgt, dass $(\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}})$ für $t \in \text{pred}(\tilde{t})\setminus\{\tilde{t}\}$ eine orthogonale Projektion mit Bild in $\mathbb{R}^\mathcal{I}_\mathbf{t}$ ist, da $\text{Bild}(\Pi_{\overline{V}_{\tilde{t}}}) \perp \text{Bild}(\Pi_{\mathbf{t}\setminus\tilde{\mathbf{t}}})$ gilt. Also ist $\Pi_{\tilde{t},\overline{V},t}$ für alle $t \in \mathcal{T}_\mathcal{I}$ eine orthogonale Projektion mit $\text{Bild}(\Pi_{\tilde{t},\overline{V},t}) \subseteq \mathbb{R}^\mathcal{I}_\mathbf{t}$. Analog folgt, dass $\Pi_\mathbf{s}$ und $\Pi_{\tilde{s},\overline{W},s}$ für alle $s \in \mathcal{T}_\mathcal{J}$ orthogonale Projektionen mit Bild in $\mathbb{R}^\mathcal{J}_\mathbf{s}$ sind. Nach Beispiel 2.3.19 ist somit für alle $b = (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ die Abbildung

$$\Pi_{\tilde{b},\overline{V},\overline{W},b} : \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}},\ X \mapsto \begin{cases} \Pi_\mathbf{t} X \Pi_\mathbf{s} & \text{, falls } (t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}} \\ \Pi_{\tilde{t},\overline{V},t} X \Pi_{\tilde{s},\overline{W},s} & \text{, falls } (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \end{cases}$$

eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ mit $\text{Bild}(\Pi_{\tilde{b},\overline{V},\overline{W},b}) \subseteq \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}}$. Für $\mathcal{I}\times\mathcal{J}$ bildet $\{\mathbf{t}\times\mathbf{s} \,|\, (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}\}$ eine Partition und somit ist $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}} = \sum_{b\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \Pi_{\tilde{b},\overline{V},\overline{W},b}$ nach Lemma 4.2.16 eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$.
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S) \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$. Zuerst betrachten wir die Wirkung der Abbildungen aus (5.6) und (5.7) auf die Clusterbasen. Mit Lemma 5.2.1 erhalten wir

$$\Pi_{\tilde{t},\overline{V},t} V_t \ = \ \begin{cases} \overline{V}_t C(V,\overline{V})_t & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}} \\ \overline{V}_t & \text{, falls } t \in \mathcal{T}_\mathcal{I}\setminus\mathcal{T}_{\tilde{\mathbf{t}}} \end{cases}$$

und

$$\Pi_{\tilde{s},\overline{W},s} W_s \ = \ \begin{cases} \overline{W}_s C(W,\overline{W})_s & \text{, falls } s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ \overline{W}_s & \text{, falls } s \in \mathcal{T}_\mathcal{J}\setminus\mathcal{T}_{\tilde{\mathbf{s}}} \end{cases}.$$

Damit gilt für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$

$$(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[H])_{|\mathbf{t}\times\mathbf{s}} = \Pi_{\tilde{b},\overline{V},\overline{W},b}[V_t S_b W_s^T] = \Pi_{\tilde{t},\overline{V},t} V_t S_b W_s^T \Pi_{\tilde{s},\overline{V},s}$$
$$= \begin{cases} \overline{V}_t C(V,\overline{V})_t S_b C(W,\overline{W})_s^T \overline{W}_s^T & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ \overline{V}_t S_b C(W,\overline{W})_s^T \overline{W}_s^T & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ \overline{V}_t C(V,\overline{V})_t S_b \overline{W}_s^T & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \\ \overline{V}_t S_b \overline{W}_s^T & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}},\ s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \end{cases}.$$

Für $b = (t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ gilt

$$(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W},\tilde{b}}[H])_{|\mathbf{t}\times\mathbf{s}} = H_{|\mathbf{t}\times\mathbf{s}} = N_b = \overline{N}_b.$$

Damit folgt die Aussage. ∎

In Abbildung 5.2 sind die Anpassung eingezeichnet, die nach den Lemmata 5.2.1 und 5.2.4 durch die lokale Projektion an der $\mathcal{H}^2$-Darstellung vorgenommen werden müssen. Dabei ist zu beachten, das die Anpassungen nur zu Clustern in $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ vorgenommen werden. Die Clusterbasen zu Vorfahren und die entsprechenden Kopplungsmatrizen werden indirekt über die modifizierten Transfermatrizen angepasst.
Im Vergleich zu Abbildung 5.1 fällt auf, dass mit Ausnahme der Nahfeldmatrizen in den unzulässigen Blättern für die lokale Projektion die gleichen Matrizen wie für die erweiterte $\mathcal{H}^2$-Matrix angepasst werden. Deshalb können wir die Algorithmen in Abschnitt 5.3 derart formulieren, dass sie Berechnungen nur in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ mit zugehörigen Blockzeilen/-spalten sowie $\mathcal{T}_{\tilde{\mathbf{b}}}$ vornehmen.
Bevor wir den Projektionsfehler untersuchen, zeigen wir eine mit Lemma 5.1.8 vergleichbare Aussage für die adaptive totale Clusterbasis. Weil die adaptiven totalen Clusterbasen zur Definition der adaptiven Gewichte verwendet werden, ermöglicht uns das Lemma 5.2.5 später die Wiederverwendung der alten adaptiven Gewichte für die Cluster in $\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{s}}}$ nach der lokalen Projektion.

Lemma 5.2.5
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ sowie $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen. Weiter seien $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $\overline{W} = (\overline{W}_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ die nach Lemma 5.2.1 definierten Clusterbasen. Zusätzlich seien $\omega, \overline{\omega} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta, \overline{\theta} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen, wobei $\overline{\omega}_b\omega_b$ und $\overline{\theta}_t = \theta_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$ gelte. Dann gilt für die adaptive totale Clusterbasis der neuen $\mathcal{H}^2$-Matrix $\overline{H} := \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\mathbf{t},\overline{V},\overline{W}}[H]$, dass für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ eine orthogonale Projektion $\Pi_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\overline{\xi}_t\times\overline{\xi}_t}$, $\overline{\xi}_t = \dot{\bigcup}_{s\in\overline{\operatorname{row}}(t)} \mathbf{s}$, in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$ mit

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_t = \begin{cases} (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t \Pi_t & \text{, falls } t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\} \\ X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t \Pi_t & \text{, falls } t \in \mathcal{T}_{\mathcal{I}}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t})) \end{cases}$$

existiert. Für $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ ist $\Pi_t := \Pi_{\xi_t}$ und kann somit weggelassen werden.

Beweis: Weil $\mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}} = \emptyset$ für $\tilde{t} = r_{\mathcal{I}}$ gilt, sei o. E. d. A. $\tilde{t} \neq r_{\mathcal{I}}$.
Für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$ gilt

$$\overline{H}_{\mathbf{t}\times\mathbf{s}} = \begin{cases} (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) H \Pi_{\mathbf{s}} & \text{, falls } t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\} \\ \Pi_{\mathbf{t}} H \Pi_{\overline{W}_s} & \text{, falls } s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ \Pi_{\mathbf{t}} H (\Pi_{\overline{W}_{\tilde{s}}} + \Pi_{\mathbf{s}\backslash\tilde{\mathbf{s}}}) & \text{, falls } s \in \operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\} \\ \Pi_{\mathbf{t}} H \Pi_{\mathbf{s}} & \text{sonst} \end{cases}. \tag{5.9}$$

Für $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ suchen wir eine passende Projektion $\Pi_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_t\times\xi_t}$. Zuerst suchen wir eine Projektion $\widehat{\Pi}_t$, die die Blockzeile $\operatorname{row}(t)$ zusammenfasst. Für $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ reicht es nach (5.9), die Cluster aus $\operatorname{row}(t)$ aufzusammeln. Deshalb definieren wir für alle $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$

$$\widehat{\Pi}_t := \Pi_{\xi_t} = \sum_{s\in\operatorname{row}(t)} \Pi_{\mathbf{s}} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_t\times\xi_t} \quad \text{mit } \xi_t = \dot{\bigcup_{s\in\operatorname{row}(t)}} \mathbf{s}.$$

Für $t \in \mathcal{T}_{\mathcal{I}}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ müssen wir nach (5.9) drei Fälle unterscheiden. Für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ definieren wir

$$\widehat{\Pi}_t := \sum_{\substack{s\in\operatorname{row}(t)\\ s\in\mathcal{T}_{\tilde{\mathbf{s}}}}} \Pi_{\overline{W}_s} + \sum_{\substack{s\in\operatorname{row}(t)\\ s\in\operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}}} (\Pi_{\overline{W}_{\tilde{s}}} + \Pi_{\mathbf{s}\backslash\tilde{\mathbf{s}}}) + \sum_{\substack{s\in\operatorname{row}(t)\\ s\in\mathcal{T}_{\mathcal{I}}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}}\cup\operatorname{pred}(\tilde{s}))}} \Pi_{\mathbf{s}} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_t\times\xi_t}.$$

Aus den obigen Definitionen und (5.9) folgt für alle $t \in \mathcal{T}_{\mathcal{I}}$ und für alle $s \in \operatorname{row}(t)$

$$\overline{H}_{\mathbf{t}\times\mathbf{s}} = \begin{cases} (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) H_{\mathbf{t}\times\mathbf{s}} \widehat{\Pi}_t & \text{, falls } t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\} \\ \Pi_{\mathbf{t}} H_{\mathbf{t}\times\mathbf{s}} \widehat{\Pi}_t & \text{, falls } t \in \mathcal{T}_{\mathcal{I}}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t})) \end{cases}. \tag{5.10}$$

Damit haben wir die Blockzeile in eine Form gebracht, die eine ähnliche Form hat, wie die Aussage des Lemmas. Weil die adaptive totale Clusterbasis auch die Blockzeilen der Vorfahren umfasst, definieren wir für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ die Matrix

$$\Pi_t := \sum_{\hat{t}\in\operatorname{pred}(t)} \widehat{\Pi}_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_t\times\xi_t}.$$

Weil die Mengen $\mathbf{s}$ für $s \in \overline{\operatorname{row}}(t)$ paarweise disjunkt und $\Pi_{\overline{W}_s}$, $(\Pi_{\overline{W}_{\tilde{s}}} + \Pi_{\mathbf{s}\backslash\tilde{\mathbf{s}}})$ und $\Pi_{\mathbf{s}}$ orthogonale Projektionen sind, ist Π_t nach Lemma 2.1.6 eine orthogonale Projektion. Also ist Π_t ein Kandidat für unsere Aussage.
Wir zeigen die Aussage für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ per Induktion über $\operatorname{level}(t) \in \mathbb{N}$. Es sei $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ mit $\operatorname{level}(t) = 0$. Dann gilt $t = r_{\mathcal{I}} \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $\overline{\omega}_{(t,s)} = \omega_{(t,s)}$ für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.
Für die adaptive totale Clusterbasis von t folgt nach Definition 4.4.15 und (5.10)

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_t &= \sum_{s\in\operatorname{row}(t)} \frac{1}{\overline{\omega}_{(t,s)}} \overline{H}_{\mathbf{t}\times\mathbf{s}} = \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) H_{\mathbf{t}\times\mathbf{s}} \Pi_t \\ &= (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t \Pi_t. \end{aligned}$$

Es sei $n \in \mathbb{N}$ derart, dass die Aussage für alle $t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}$ mit $\operatorname{level}(t) \leq n$ gilt. Weiter sei $t \in \operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\}$ mit $\operatorname{level}(t) = n+1$. Dann existiert $\acute{t} := \operatorname{par}(t)$ und es gilt $\operatorname{level}(\acute{t}) = n$. Also gilt die Aussage für $\acute{t}$. Wir unterscheiden die beiden Fälle der Aussage.
1. Fall: Es sei $t \in \operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\}$. Dann ist auch $\acute{t} := \operatorname{par}(t) \in \operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\}$. Mit der Induktionsannahme für $\acute{t}$ folgt

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_{\acute{t}|\mathbf{t}} &= \left((\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\acute{\mathbf{t}}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}} \Pi_{\acute{t}} \right)_{|\mathbf{t}} \\ &= (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\mathbf{t}} \Pi_{\acute{t}}. \end{aligned}$$

Zusammen mit (5.10), $\overline{\theta}_{\acute{t}} = \theta_{\acute{t}}$ und $\overline{\omega}_{(t,s)} = \omega_{(t,s)}$ für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ erhalten wir

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_t &= \sum_{s\in\operatorname{row}(t)} \frac{1}{\overline{\omega}_{(t,s)}} \overline{H}_{\mathbf{t}\times\mathbf{s}} + \frac{1}{\overline{\theta}_{\acute{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_{\acute{t}|\mathbf{t}} \\ &= (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} H_{\mathbf{t}\times\mathbf{s}} \widehat{\Pi}_t + \frac{1}{\theta_{\acute{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\mathbf{t}} \Pi_{\acute{t}} \\ &= (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t \Pi_t. \end{aligned}$$

2. Fall: Es sei $t \in \mathcal{T}_\mathcal{I} \backslash (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$. Wir haben dann zwei Fälle für das Elter $\acute{t}$ zu unterscheiden.
2.1. Fall: Es sei $\acute{t} \in \operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\}$. In diesem Fall gilt nach Induktionsvoraussetzung

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_{\acute{t}|\mathbf{t}} = \Pi_\mathbf{t} (\Pi_{V_{\tilde{t}}} + \Pi_{\acute{\mathbf{t}}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}} \Pi_{\acute{t}} = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\mathbf{t}} \Pi_{\acute{t}}.$$

2.2. Fall: Es sei $\acute{t} \in \mathcal{T}_\mathcal{I} \backslash (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ Aus der Induktionsvoraussetzung folgt

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_{\acute{t}|\mathbf{t}} = \Pi_\mathbf{t} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}} \Pi_{\acute{t}} = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\mathbf{t}} \Pi_{\acute{t}}.$$

Zusammen mit (5.10), $\overline{\theta}_{\acute{t}} = \theta_{\acute{t}}$ und $\overline{\omega}_{(t,s)} = \omega_{(t,s)}$ für alle $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gilt in beiden Fällen

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_t &= \sum_{s\in\operatorname{row}(t)} \frac{1}{\overline{\omega}_{(t,s)}} \overline{H}_{\mathbf{t}\times\mathbf{s}} + \frac{1}{\overline{\theta}_{\acute{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta})_{\acute{t}|\mathbf{t}} \\ &= \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} H_{\mathbf{t}\times\mathbf{s}} \widehat{\Pi}_t + \frac{1}{\theta_{\acute{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\mathbf{t}} \Pi_{\acute{t}} \\ &= X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t \Pi_t. \end{aligned}$$

Also gilt die Aussage des Lemma. ∎

Bemerkung 5.2.6
Die Bedingung an die Skalierung $\overline{\omega}$ ist im Falle der Strategien 2 und 3 aus Abschnitt 4.5 problematisch, weil die Normen der Teilmatrizen $H_{\mathbf{t}\times\mathbf{s}}$ und $\overline{H}_{\mathbf{t}\times\mathbf{s}}$ für $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{b}}}$ i.A. nicht gleich sind. Auf dieses Problem gehen wir bei der Untersuchung des Fehlers des Niedrigrangupdates näher ein.

Lemma 5.2.5 ermöglicht uns die Weiterverwendung der alten Gewichte außerhalb der Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ nach der lokalen Projektion, falls wir zusätzlich Projektionen auf der rechten Seite zulassen (siehe Definition 5.3.4 der sogenannten projizierten Gewichte). Damit haben wir die Aspekte der lokalen Projektion untersucht, die wir für die algorithmische Umsetzung in Abschnitt 5.3 benötigen.
Bevor wir die lokalen Algorithmen vorstellen, schätzen wir den Fehler der lokalen Projektion ab. Um den Fehler clusterweise abzuschätzen, teilen wir den Fehler wie für die Projektion in Kapitel 4 in einen Teil für die Zeilenbasis und einen Teil für die Spaltenbasis auf. Dazu definieren wir die einseitigen lokalen $\mathcal{H}^2$-Matrix-Projektionen.

Definition 5.2.7 (Lokale einseitige Projektion)
Es seien ein zulässiger Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, ein Block $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ orthogonale Clusterbasen $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ gegeben. Dann definieren wir die einseitigen lokalen Projektionen durch

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] \;=\; \sum_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} + \sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{\tilde{t},\overline{V},t} X \Pi_{\mathbf{s}} \tag{5.11}$$

und

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[X] \;=\; \sum_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} + \sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \Pi_{\mathbf{t}} X \Pi_{\tilde{s},\overline{W},s}. \tag{5.12}$$

Im folgenden Lemma zeigen wir, dass die lokalen einseitigen Projektionen auch orthogonale Projektionen sind und wir mit diesen die lokale $\mathcal{H}^2$-Matrix-Projektion ausdrücken können. Dies ermöglicht uns wie das Lemma 4.2.16 bei der globalen $\mathcal{H}^2$-Matrix-Projektion später das Aufteilen des Projektionsfehlers in Teile für die Zeilenbasis und die Spaltenbasis.

Lemma 5.2.8
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(V_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(W_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen.

(i) Die einseitigen lokalen Projektionen $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}$ und $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}$ sind orthogonale Projektionen in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$.

(ii) Es gilt $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}} = \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}} = \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}$.

(iii) Für alle $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ gilt $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[X] = (\Pi_{\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\tilde{b}^T,\overline{W},*}[X^T])^T$.

Beweis: (i): Statt der Abbildung $\Pi_{\tilde{b},\overline{V},\overline{W},b}$ definieren wir für alle $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$

$$\Pi_{\tilde{b},\overline{V},*,b} : \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}},\; X \mapsto \begin{cases} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} & \text{, falls } (t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}} \\ \Pi_{\tilde{t},\overline{V},t} X \Pi_{\mathbf{s}} & \text{, falls } (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \end{cases}.$$

Wie in dem Beweis des Lemmas 5.2.4 ist dies für alle $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ mit $\text{Bild}(\Pi_{\tilde{b},\overline{V},*,b}) \subseteq \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}}$. Nach Lemma 2.1.6 ist dann

$$\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*} \;=\; \sum_{b\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \Pi_{\tilde{b},\overline{V},*,b}$$

eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$. Die Aussage folgt für $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}$ analog, indem wir für alle $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ die blockweisen Projektionen

$$\Pi_{\tilde{b},*,\overline{W},b} : \mathbb{R}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}^{\mathcal{I}\times\mathcal{J}},\; X \mapsto \begin{cases} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} & \text{, falls } (t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}} \\ \Pi_{\mathbf{t}} X \Pi_{\tilde{s},\overline{W},s} & \text{, falls } (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \end{cases} \quad \text{definieren.}$$

(ii): Die zweite Aussage folgt, weil $\Pi_{\tilde{b},\overline{V},\overline{W},b} = \Pi_{\tilde{b},\overline{V},*,b}\Pi_{\tilde{b},*,\overline{W},b} = \Pi_{\tilde{b},*,\overline{W},b}\Pi_{\tilde{b},\overline{V},*,b}$ für alle $b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ gilt.
(iii): Für $b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ gilt

$$\Pi_{\tilde{b},*,\overline{W},b} X \;=\; \begin{cases} \Pi_{\mathbf{t}} X \Pi_{\mathbf{s}} = (\Pi_{\mathbf{s}} X \Pi_{\mathbf{t}})^T & \text{, falls } (t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}} \\ \Pi_{\mathbf{t}} X \Pi_{\tilde{s},\overline{W},s} = (\Pi_{\tilde{s},\overline{W},s} X^T \Pi_{\mathbf{t}})^T & \text{, falls } (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \end{cases} \;=\; (\Pi_{\tilde{b},*,\overline{W},b} X^T)^T.$$

Damit folgt die dritte Aussage. ∎

Mit Hilfe der einseitigen Projektionen können wir den Projektionsfehler näher untersuchen. Wie in Abschnitt 4.3 betrachten wir als Erstes den clusterweisen Projektionsfehler. Dazu stellen wir den lokalen Projektionsfehler durch die clusterweisen Fehlerterme aus Definition 4.3.2 dar (vergleiche Theorem 4.3.5).

Lemma 5.2.9
Es seien $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ eine orthogonale Clusterbasis. Dann gilt für den Fehler der lokalen $\mathcal{H}^2$-Matrix-Projektion aus Definition 5.2.2

$$X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] \;=\; \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t.$$

und $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_{t'}^T D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t = 0$ für alle $t, t' \in \mathcal{T}_{\mathcal{I}}$ mit $t \neq t'$.

Beweis: Für den Fehler der einseitigen Projektion bezüglich der Zeilenbasis gilt

$$X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] \;=\; \sum_{\substack{t\in\mathcal{T}_{\tilde{\mathbf{t}}} \\ s\in\text{row}(t)}} X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} + \sum_{\substack{t\in\text{pred}(\tilde{t})\setminus\{\tilde{t}\} \\ s\in\text{row}(t)}} X_{|\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{\overline{V}_{\tilde{t}}} X_{|\tilde{\mathbf{t}}\times\mathbf{s}}.$$

Mit der Darstellung des blockweisen Fehlers durch die Nachfahren aus Lemma 4.3.10 folgt

$$\begin{aligned} X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] &= \sum_{\substack{t\in\mathcal{T}_{\tilde{\mathbf{t}}}\\ s\in\operatorname{row}(t)}} X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} + \sum_{\substack{t\in\operatorname{pred}(\tilde{t})\setminus\{\tilde{t}\}\\ s\in\operatorname{row}(t)}} X_{|\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{\overline{V}_{\tilde{t}}} X_{|\tilde{\mathbf{t}}\times\mathbf{s}} \\ &= \sum_{\substack{t\in\mathcal{T}_{\tilde{\mathbf{t}}}\\ s\in\operatorname{row}(t)}} \sum_{\check{t}\in\operatorname{desc}(t)} D(X,\overline{V})_{\check{t},s} + \sum_{\substack{t\in\operatorname{pred}(\tilde{t})\setminus\{\tilde{t}\}\\ s\in\operatorname{row}(t)}} \sum_{\check{t}\in\operatorname{desc}(\tilde{t})} D(X,\overline{V})_{\check{t},s}. \end{aligned}$$

Für $t \in \mathcal{T}_{\mathcal{I}}\setminus(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ gilt $\operatorname{desc}(t) \cap \mathcal{T}_{\tilde{\mathbf{t}}} = \emptyset$. Zusammen mit einer Umsortierung der Summanden erhalten wir

$$\begin{aligned} X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] &= \sum_{t\in\mathcal{T}_{\mathcal{I}}} \sum_{\check{t}\in(\operatorname{desc}(t)\cap\mathcal{T}_{\tilde{\mathbf{t}}})} \sum_{s\in\operatorname{row}(t)} D(X,\overline{V})_{\check{t},s} \\ &= \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \sum_{\hat{t}\in\operatorname{pred}(t)} \sum_{s\in\operatorname{row}(\hat{t})} D(X,\overline{V})_{t,s}. \end{aligned}$$

Durch Einsetzen der Definition 4.3.2 der blockweisen Fehlerterme folgt

$$X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] = \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \sum_{\hat{t}\in\operatorname{pred}(t)} \sum_{s\in\operatorname{row}(\hat{t})} \Pi_{D,\overline{V},t} X_{|\mathbf{t}\times\mathbf{s}} = \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \Pi_{D,\overline{V},t}\Pi_{\mathbf{t}} X \sum_{s\in\overline{\operatorname{row}}(t)} \Pi_{\mathbf{s}}.$$

Der hintere Teil ist die totale Clusterbasis in t (siehe Definition 4.1.6). Mit der Definition 4.3.2 der clusterweisen Fehlerterme gilt

$$X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] = \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \Pi_{D,\overline{V},t} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, X)_t = \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t.$$

Für $t, t' \in \mathcal{T}_{\tilde{\mathbf{t}}}$ mit $t \neq t'$ folgt aus Lemma 4.3.4 (4.7)

$$D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_{t'}^T D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},X)_{t'}\Pi_{D,\overline{V},t'}^T \Pi_{D,\overline{V},t} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},X)_t = 0.$$

■

Wir spalten den Fehler später wie bei der globalen $\mathcal{H}^2$-Matrix-Projektion in einen Anteil bezüglich der Zeilenbasis und einen bezüglich der Spaltenbasis auf. Für den Fehler der Projektion bezüglich der Spaltenbasis erhalten wir dabei

$$\begin{aligned} &\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]] \\ &\quad= \left((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T,\tilde{b}^T,\overline{W},*}[(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T]\right)^T \\ &\quad= \left(\sum_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}} D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T, \overline{W})_s\right)^T. \end{aligned}$$

Um auch für die Spaltenbasis Terme bezüglich der ursprünglichen Matrix X zu erhalten, zeigen wir, dass das Weglassen der Projektion bezüglich der Zeilen die Norm nur vergrößert (vergleiche Lemma 4.3.6).

Lemma 5.2.10
Es seien $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(W_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ orthogonale Clusterbasen. Dann gilt für alle $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$

$$\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S \;\leq\; \|D(X^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S.$$

Beweis: Für die totalen Clusterbasen führen wir die beiden Schreibweisen $X_{\Pi,s} := X(s, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, (\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T)$ und $X_{T,s} := X(s, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, X^T)$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ ein (siehe Definition 4.1.6 der totalen Clusterbasis). Es sei $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Mit $\operatorname{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, s') = \operatorname{col}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, s)$ (siehe Bemerkung 4.1.2) gilt nach Definition 4.1.3 der erweiterten Blockzeile wie in (4.9)

$$X_{\Pi,s} \;=\; \Pi_{\mathbf{s}} \sum_{\hat{s}\in\operatorname{pred}(s)} \sum_{t\in\operatorname{col}(\hat{s})} (\Pi_{\mathbf{t}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])\Pi_{\hat{\mathbf{s}}})^T.$$

Im Gegensatz zum Beweis von Lemma 4.3.6 müssen wir beim Term in der Summe verschiedene Fälle unterscheiden.
1. Fall: Es sei $\hat{s} \in \operatorname{pred}(s)$ und $t \in \operatorname{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \hat{s})$ mit $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt

$$\Pi_{\mathbf{s}}(\Pi_{\mathbf{t}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])\Pi_{\hat{\mathbf{s}}})^T \;=\; \Pi_{\mathbf{s}}(\Pi_{\overline{V}_t} X \Pi_{\hat{\mathbf{s}}})^T \;=\; \Pi_{\mathbf{s}} X^T \Pi_{\mathbf{t}} \Pi_{\overline{V}_t}.$$

2. Fall: Es sei $\hat{s} \in \operatorname{pred}(s)$ und $t \in \operatorname{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \hat{s})$ mit $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$. Dann gilt

$$\Pi_{\mathbf{s}}(\Pi_{\mathbf{t}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])\Pi_{\hat{\mathbf{s}}})^T \;=\; \Pi_{\mathbf{s}}((\Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_t}) X \Pi_{\hat{\mathbf{s}}})^T \;=\; \Pi_{\mathbf{s}} X^T \Pi_{\mathbf{t}} (\Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_t}).$$

3. Fall: Es sei $\hat{s} \in \operatorname{pred}(s)$ und $t \in \operatorname{row}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \hat{s})$ mit $t \in \mathcal{T}_{\mathcal{I}}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$. Dann gilt

$$\Pi_{\mathbf{s}}(\Pi_{\mathbf{t}}(\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])\Pi_{\hat{\mathbf{s}}})^T \;=\; \Pi_{\mathbf{s}}(\Pi_{\mathbf{t}} X \Pi_{\hat{\mathbf{s}}})^T \;=\; \Pi_{\mathbf{s}} X^T \Pi_{\mathbf{t}} \Pi_{\mathbf{t}}.$$

Wir definieren folgende Matrix (vergleiche Beweis von Lemma 5.2.5)

$$\Pi_s \;:=\; \sum_{\substack{t\in\overline{\operatorname{col}}(s)\\ t\in\mathcal{T}_{\tilde{\mathbf{t}}}}} \Pi_{\overline{V}_t} + \sum_{\substack{t\in\overline{\operatorname{col}}(s)\\ t\in\operatorname{pred} t\backslash\{\tilde{t}\}}} (\Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}} + \Pi_{\overline{V}_{\tilde{t}}}) + \sum_{\substack{t\in\overline{\operatorname{col}}(s)\\ t\notin\mathcal{T}_{\tilde{\mathbf{t}}}\cup\operatorname{pred}\tilde{t}}} \Pi_{\mathbf{t}} \;\in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\bar{\xi}_s\times\bar{\xi}_s}.$$

Weil die Mengen $\mathbf{t}$ für alle $t \in \overline{\operatorname{col}}(s)$ paarweise disjunkt sind, ist Π_s eine orthogonale Projektion (siehe Lemma 2.1.6). Für diese gilt

$$X_{\Pi,s} \;=\; \sum_{t\in\overline{\operatorname{row}}(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},s)} \Pi_{\mathbf{s}} X^T \Pi_{\mathbf{t}} \Pi_s \;=\; X_{T,s}\Pi_s.$$

Wir nutzen für alle $s' \in \mathcal{T}_{\mathcal{J}}$ folgende Notation $\breve{W}_{s'} := \breve{V}(\overline{W})_{s'}$ (siehe Definition 4.2.3). Mit der Definition 4.3.2 für die Fehlerterme und der Invarianz der Spektralnorm unter Adjungieren (siehe Lemma 2.3.13) folgt

$$\begin{aligned}\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \overline{W})_s\|_S \;&=\; \|\Pi_{D,\overline{W},s} X_{\Pi,s}\|_S \;=\; \|X^T_{\Pi,s}\Pi_{D,\overline{W},s}\|_S\\ &=\; \|\Pi_s X^T_{T,s}\Pi_{D,\overline{W},s}\|_S.\end{aligned}$$

Weil Π_s eine orthogonale Projektion ist, folgt mit Lemma 2.3.20

$$\begin{aligned}\|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_S &= \|\Pi_{\overline{\mathrm{row}}(s)}X^T_{T,s}\Pi_{D,\overline{W},s}\|_S \leq \|X^T_{T,s}\Pi_{D,\overline{W},s}\|_S\\ &= \|\Pi_{D,\overline{W},s}X_{T,s}\|_S = \|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_S.\end{aligned}$$

■

Mit diesen beiden Lemmata können wir eine Fehlerabschätzung der lokalen Projektion angeben, bei der die clusterweisen Fehlerterme aus Definition 4.3.2 verwendet werden. Dabei kommen im Gegensatz zur Abschätzung für die globale $\mathcal{H}^2$-Matrix-Projektion in Theorem 4.3.7 nur die Fehlerterme vor, die zu den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ gehören.

Theorem 5.2.11
Es sei $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\tilde{b} = (\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen. Dann gilt

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\|_S^2 \leq 2\left(\sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}\|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_S^2 + \sum_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}\|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_S^2\right)$$

und

$$\begin{aligned}&\max\left\{\sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}\|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_S^2, \sum_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}\|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_F^2\right\}\\ &\leq \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\|_F^2 \leq \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}\|D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|_F^2 + \sum_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}\|D(X^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|_F^2.\end{aligned}$$

Beweis: Den Fehler der lokalen $\mathcal{H}^2$-Matrix-Projektion können wir mit Lemma 5.2.8 durch die Fehler der einseitigen lokalen $\mathcal{H}^2$-Matrix-Projektionen ausdrücken. Es gilt

$$\begin{aligned}X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X] &= X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]\\ &\quad + \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]].\end{aligned}$$

Zuerst betrachten wir die Spektralnorm. Wir trennen den Fehler bezüglich der Zeilenbasis und der Spaltenbasis.

$$\begin{aligned}&\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\|_S^2\\ &\leq \left(\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]\|_S + \|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]]\|_S\right)^2\\ &\leq 2\left(\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]\|_S^2 + \|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]]\|_S^2\right).\end{aligned}$$

Für den Fehler bezüglich der Spaltennorm gilt nach Lemma 5.2.9

$$X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] = \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} D(X,\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t$$

und $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_{t_1}^T D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_{t_2} = 0$ für alle $t_1, t_2 \in \mathcal{T}_{\tilde{\mathfrak{t}}}$ mit $t_1 \neq t_2$. Nach Lemma 2.3.21 folgt für alle $x \in \mathbb{R}^{\mathcal{J}}$, dass die Terme $D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t x$ paarweise orthogonal sind. Mit Lemma 2.1.2 erhalten wir

$$\|(X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])x\|_2^2 \;=\; \sum_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t x\|_2^2. \tag{5.13}$$

Dies können wir mit Lemma 2.3.9 als Ungleichung auf die Spektralnorm übertragen und erhalten

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]\|_S^2 \;\leq\; \sum_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|_S^2.$$

Analog erhalten wir für den Fehler bezüglich der Spaltenbasis

$$\begin{aligned}\|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]]\|_S^2 \\ \leq \sum_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}} \|D((\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X])^T, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T, \overline{W})_t\|_S^2.\end{aligned}$$

Mit Lemma 5.2.10 können wir die Zeilenprojektion in den Termen auf der rechten Seite entfernen. Damit gilt

$$\|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X] - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]]\|_2^S \;\leq\; \sum_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}} \|D(X^T, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T, \overline{W})_t x\|_S^2.$$

Indem wir die beiden Fehlerabschätzungen oben einsetzen, erhalten wir die Fehlerschranke bezüglich der Spektralnorm.
Weil $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}\times\mathcal{J}}, \|.\|_F)$ ist, gilt mit Lemma 2.1.7, Korollar 2.1.8 und den Aussagen zu den einseitigen Projektionen in Lemma 5.2.8

$$\begin{aligned}&\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\|_F^2 \\ &\quad= \|(I - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*})[X]\|_F^2 + \|\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},*,\overline{W}}[X]]\|_F^2 \\ &\quad\leq \|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]\|_F^2 + \|X^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T,\tilde{b}^T,\overline{W},*}[X^T]\|_F^2.\end{aligned} \tag{5.14}$$

Dabei nutzen wir im zweiten Schritt die Norminvarianz unter Adjungieren (siehe Lemma 2.3.13) und den Zusammenhang zwischen den einseitigen lokalen Projektionen (siehe Lemma 5.2.8) aus. Mit Lemma 2.3.9 folgt aus (5.13)

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},*}[X]\|_F^2 \;=\; \sum_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}} \|D(X, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\|_F^2,$$

wobei sich die Gleichheit für die Frobeniusnorm überträgt. Analog erhalten wir für den Fehler bezüglich der Spaltenbasis

$$\|X^T - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T,\tilde{b}^T,\overline{W},*}[X^T]\|_F^2 \;=\; \sum_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}} \|D(X^T, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}^T, \overline{W})_t\|_F^2.$$

Durch Einsetzen erhalten wir die Abschätzung für die Frobeniusnorm. Die untere Schranke folgt wie für Lemma 4.3.1 aus (5.14). ∎

Dieses Ergebnis entspricht der Erwartung, weil wir nur bezüglich der in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ definierten orthogonalen Clusterbasen projizieren. Außerdem lässt die Fehlerabschätzung eine weitere Deutung des Ansatzes zu. Es handelt sich um den gleichen Fehler, den wir erwarten, wenn wir die Rekompression für die gesamte Matrix so durchführen, dass die Fehlerschranken für alle $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{s}}}$ gleich null gesetzt werden. Dieses Vorgehen erscheint sinnvoll, weil wir davon ausgehen, dass die Darstellung der ursprünglichen $\mathcal{H}^2$-Matrix H effizient ist.
Für die Betrachtung des blockweisen Fehlers zeigen wir eine Theorem 4.3.9 entsprechende Aussage für die lokale Projektion.

Lemma 5.2.12
Es seien $X \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen. Dann gilt für den Fehler der lokalen $\mathcal{H}^2$-Matrix-Projektion aus Definition 5.2.2

$$\begin{aligned}\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\|^2 \leq &\sum_{\substack{t\in\mathcal{T}_{\tilde{\mathbf{t}}}\\ s\in\operatorname{row}(t)}} \|X_{\mathbf{t}\times\mathbf{s}} - \Pi_{V_t} X_{\mathbf{t}\times\mathbf{s}}\|^2 + \sum_{\substack{t\in\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}\\ s\in\operatorname{row}(t)}} \|X_{\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{V_{\tilde{t}}} X_{\tilde{\mathbf{t}}\times\mathbf{s}}\|^2\\ &+ \sum_{\substack{s\in\mathcal{T}_{\tilde{\mathbf{s}}}\\ t\in\operatorname{col}(s)}} \|X^T_{\mathbf{t}\times\mathbf{s}} - \Pi_{W_s} X^T_{\mathbf{t}\times\mathbf{s}}\|^2 + \sum_{\substack{s\in\operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}\\ t\in\operatorname{row}(s)}} \|X^T_{\mathbf{t}\times\tilde{\mathbf{s}}} - \Pi_{W_{\tilde{s}}} X^T_{\mathbf{t}\times\tilde{\mathbf{s}}}\|^2\end{aligned}$$

für beide Normen $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Für die Frobeniusnorm gilt zusätzlich die untere Schranke

$$\begin{aligned}\max\Bigg\{ &\sum_{\substack{t\in\mathcal{T}_{\tilde{\mathbf{t}}}\\ s\in\operatorname{row}(t)}} \|X_{\mathbf{t}\times\mathbf{s}} - \Pi_{V_t} X_{\mathbf{t}\times\mathbf{s}}\|_F^2 + \sum_{\substack{t\in\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}\\ s\in\operatorname{row}(t)}} \|X_{\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{V_{\tilde{t}}} X_{\tilde{\mathbf{t}}\times\mathbf{s}}\|_F^2,\\ &\sum_{\substack{s\in\mathcal{T}_{\tilde{\mathbf{s}}}\\ t\in\operatorname{col}(s)}} \|X^T_{\mathbf{t}\times\mathbf{s}} - \Pi_{W_s} X^T_{\mathbf{t}\times\mathbf{s}}\|_F^2 + \sum_{\substack{s\in\operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}\\ t\in\operatorname{row}(s)}} \|X^T_{\mathbf{t}\times\tilde{\mathbf{s}}} - \Pi_{W_{\tilde{s}}} X^T_{\mathbf{t}\times\tilde{\mathbf{s}}}\|_F^2 \Bigg\}.\end{aligned}$$

Beweis: Da $\{\mathbf{t}\times\mathbf{s} \,|\, (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}\}$ nach Lemma 3.3.4 eine Blockpartition von $\mathcal{I}\times\mathcal{J}$ bildet und in den unzulässigen Blättern kein Fehler entsteht, gilt nach Lemma 2.3.10

$$\|X - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X]\|^2 \leq \sum_{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|^2. \tag{5.15}$$

Deshalb betrachten wir den Fehler blockweise. Dazu sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Wir untersuchen die verschiedenen Fälle aus der Definition 5.2.2 der lokalen $\mathcal{H}^2$-Matrix-Projektion.

1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt

$$\begin{aligned} X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}] &= X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s} \\ &= X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} + \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s}. \end{aligned}$$

Weil Π_{V_t} nach Beispiel 2.3.18 eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{I}}, \|.\|_2)$ ist, gilt mit Lemma 2.1.7, Lemma 2.1.2 und Korollar 2.1.8 für alle $x \in \mathbb{R}^{\mathcal{J}}$

$$\begin{aligned} &\|(X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}])x\|_2^2 \\ &\quad= \|(X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}})x\|_2^2 + \|(\Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s})x\|_2^2 \qquad (5.16) \\ &\quad\leq \|(X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}})x\|_2^2 + \|(X_{|\mathbf{t}\times\mathbf{s}} - X_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s})x\|_2^2. \end{aligned}$$

Mit Lemma 2.3.9 und Lemma 2.3.13 folgt

$$\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|^2 \leq \|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}}\|^2 + \|X^T_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{W}_s} X^T_{|\mathbf{t}\times\mathbf{s}}\|^2.$$

2. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Nach Lemma 3.3.14 ist $s \notin \operatorname{pred}(\tilde{s})$ und es gilt

$$\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|^2 = \|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}}\|^2.$$

3. Fall: Es sei $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Wieder ist nach Lemma 3.3.14 $s \notin \operatorname{pred}(\tilde{s})$ und es folgt

$$\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|^2 = \|X_{|\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{\overline{V}_{\tilde{t}}} X_{|\tilde{\mathbf{t}}\times\mathbf{s}}\|^2.$$

4. Fall: Es sei $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Nach Lemma 3.3.14 ist $t \notin \operatorname{pred}(\tilde{t})$ und es gilt

$$\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|^2 = \|X_{|\mathbf{t}\times\mathbf{s}} - X_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s}\|^2.$$

5. Fall: Es sei $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}$. Wieder ist nach Lemma 3.3.14 $t \notin \operatorname{pred}(\tilde{t})$ und es folgt

$$\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|^2 = \|X_{|\mathbf{t}\times\tilde{\mathbf{s}}} - X_{|\mathbf{t}\times\tilde{\mathbf{s}}} \Pi_{\overline{W}_{\tilde{s}}}\|^2.$$

6. Fall: Es sei $t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t})$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s})$. Dann gilt nach Definition

$$X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}] = 0.$$

Die Ergebnisse setzen wir in (5.15) ein. Die Summe über $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$ ergibt sich aus den ersten beiden Fällen. Weil nach Lemma 3.3.14 $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ für $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $s \in \operatorname{row}(t)$ ist, folgt die Summe über $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $s \in \operatorname{row}(t)$ aus dem dritten

Fall. Analog folgen die Summen für die Spaltencluster aus dem ersten und vierten bzw. dem fünften Fall.
Es bleibt noch die untere Schranke für die Frobeniusnorm zu beweisen. In Lemma 2.3.10 und somit in (5.15) gilt für die Frobeniusnorm Gleichheit. Ansonsten schätzen wir nur im 1. Fall weiter nach oben ab. Mit der ersten Gleichheit in (5.16) und Lemma 2.3.9 folgt für $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ mit $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$

$$\begin{aligned}
&\|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[X_{|\mathbf{t}\times\mathbf{s}}]\|_F^2 \\
&\quad = \|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}}\|_F^2 + \|\Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}} \Pi_{\overline{W}_s}\|_F^2 \\
&\quad \geq \|X_{|\mathbf{t}\times\mathbf{s}} - \Pi_{\overline{V}_t} X_{|\mathbf{t}\times\mathbf{s}}\|_F^2.
\end{aligned}$$

Damit gilt die untere Schranke

$$\sum_{\substack{t\in\mathcal{T}_{\tilde{\mathbf{t}}} \\ s\in\mathrm{row}(t)}} \|X_{\mathbf{t}\times\mathbf{s}} - \Pi_{V_t} X_{\mathbf{t}\times\mathbf{s}}\|_F^2 + \sum_{\substack{t\in\mathrm{pred}(\tilde{t})\setminus\{\tilde{t}\} \\ s\in\mathrm{row}(t)}} \|X_{\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{V_{\tilde{t}}} X_{\tilde{\mathbf{t}}\times\mathbf{s}}\|_F^2.$$

Die untere Schranke bezüglich der Spaltenbasis erhalten wir durch Vertauschen der Rollen von $\Pi_{\overline{V}_t}$ und $\Pi_{\overline{W}_s}$ in (5.16). Beide Abschätzungen zusammen ergeben die untere Schranke aus der Behauptung. ∎

Bei der blockweisen Fehlerabschätzung fällt auf, dass nicht in allen zulässigen Blättern ein Fehler entsteht und bei manchen nur ein einseitiger Fehler. Außerdem sind die Fehler, die in Blöcken mit $t \in \mathrm{pred}(\tilde{t})$ oder $s \in \mathrm{pred}(\tilde{s})$ entstehen, auf den zu $\tilde{t}$ bzw. $\tilde{s}$ gehörigen Teil beschränkt.
Die blockweisen Fehler aus Lemma 5.2.12 lassen sich wie bei der globalen Projektion mit Lemma 4.3.11 gegen die blockweisen Fehlerterme abschätzen. Diese können wir in der Norm mit Hilfe von Lemma 5.4.1 beschränken. Durch die unteren Fehlerschranken für die Frobeniusnorm stellen wir wie bei der Rekompression sicher, dass die oberen Schranken aus Theorem 5.2.11 und Lemma 5.2.12 den Fehler maximal um den Faktor 2 überschätzen. Am Ende von Abschnitt 5.4 untersuchen wir, wie sich die Fehler für die in Kapitel 4 beschriebenen Strategien beschränken lassen.
Im nächsten Abschnitt untersuchen wir die algorithmische Umsetzung dieser lokalen Rekompression.

5.3 Algorithmische Umsetzung des Niedrigrangupdates

Es sei weiterhin $H = (\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine H^2-Matrix, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{t}\times\tilde{s}}$. Des Weiteren sei $\widetilde{H} = \widetilde{H}(H,R,\tilde{b}) = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{V}, \widetilde{W}, \widetilde{N}, \widetilde{S})$ die erweiterte $\mathcal{H}^2$-Matrix aus Abschnitt 5.1.
Wir stellen in diesem Abschnitt eine Variante des Niedrigrangupdates vor, die erlaubt, mehrere Updates in verschiedenen Blöcken in beliebiger Reihenfolge nacheinander durchzuführen. In Kapitel 7 beschreiben wir dann zwei Varianten, die das Durchlaufen der

Matrix in gewisser Reihenfolge erfordern.
Das Niedrigrangupdate lässt sich analog zu Algorithmus 4.5.1 in fünf Schritte teilen. Dabei achten wir darauf, möglichst nur in den Teilbäumen $\mathcal{T}_{\widetilde{\mathbf{b}}}$, $\mathcal{T}_{\widetilde{\mathbf{t}}}$ und $\mathcal{T}_{\widetilde{\mathbf{s}}}$ zu arbeiten. Anstatt der adaptiven Gewichte berechnen wir für die Niedrigrangupdates die sogenannten projizierten Gewichte, welche wir später einführen. Diese stellen eine kleine Abwandlung der adaptiven Gewichte dar und sind notwendig, um die Berechnungen lokal zu halten.

(i) Die Funktion ORTHOGONALE_GEWICHTE_UPDATE berechnet die orthogonalen Gewichte (siehe Definition 4.2.6) für die erweiterten Clusterbasen $\widetilde{V}$ und $\widetilde{W}$ in den Teilbäumen $\mathcal{T}_{\widetilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\widetilde{\mathbf{s}}}$ (vergleiche Algorithmus 4.2.1).

(ii) Die Funktion PROJIZIERTE_GEWICHTE_UPDATE berechnet mit Hilfe der projizierten Gewichte von V und W (vergleiche Definition 4.4.1) die projizierten Gewichte für die erweiterten Clusterbasen $\widetilde{V}$ und $\widetilde{W}$ in den Teilbäumen $\mathcal{T}_{\widetilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\widetilde{\mathbf{s}}}$ (vergleiche Algorithmus 4.4.2).

(iii) Die Funktion ADAPTIVE_CLUSTERBASIS_UPDATE berechnet die neuen adaptiven Clusterbasen $\overline{V}$ und $\overline{W}$ in den Teilbäumen $\mathcal{T}_{\widetilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\widetilde{\mathbf{s}}}$ (vergleiche Algorithmus 4.4.1).

(iv) Die Funktion H2-MATRIX-PROJEKTION_UPDATE berechnet die neuen Nahfeld- und Kopplungsmatrizen $\overline{N}$ bzw. $\overline{S}$ zur Projektion von $\widetilde{H}$ auf die neuen Clusterbasen $\overline{V}$ und $\overline{W}$ in allen Blöcken $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ mit $t \in \mathcal{T}_{\widetilde{\mathbf{t}}}$ oder $s \in \mathcal{T}_{\widetilde{\mathbf{s}}}$ (vergleiche Algorithmus 4.2.2).

(v) In einem letzten Schritt werden die projizierten Gewichte für die neue $\mathcal{H}^2$-Matrix $\overline{H}$ in den Teilbäumen $\mathcal{T}_{\widetilde{\mathbf{t}}}$ und $\mathcal{T}_{\widetilde{\mathbf{s}}}$ berechnet (vergleiche Algorithmus 4.4.2).

Die Bezeichnung mit *Update* weist darauf hin, dass die Berechnungen für das Niedrigrangupdate angepasst sind. Der letzte Schritt kommt im Vergleich zur Rekompression hinzu, um die Größen für das nächste Niedrigrangupdate bereitzuhalten.
Im Folgenden betrachten wir die Algorithmen für die lokale Rekompression bezüglich der Zeilenclusterbasis $\widetilde{V}$. Alle Aussagen können wir direkt auf die Spaltenclusterbasis $\widetilde{W}$ übertragen, indem wir zur transponierten Matrix übergehen.

5.3.1 Voraussetzungen des Niedrigrangupdates

Bevor wir die lokalen Versionen für die Funktionen der Rekompression definieren, wollen wir uns Voraussetzungen überlegen, die

(i) vor dem Niedrigrangupdate erfüllt sein sollen, damit die neue $\mathcal{H}^2$-Matrix effizient berechnet werden kann und gewisse Fehlerabschätzungen erfüllt, und

(ii) nach dem Niedrigrangupdate wieder erfüllt werden können, ohne einen hohen zusätzlichen Aufwand zu erfordern.

Hierbei sollen an dieser Stelle vor allem die Voraussetzungen motiviert werden. Dass diese ausreichen, um das Niedrigrangupdate durchzuführen, zeigen wir später.

Nach Lemma 5.2.1 sind die Clusterbasen, die durch Integrieren der neuen Clusterbasen auf $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ in die alten Clusterbasen entstehen, für die Vorfahren in $\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $\operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}$ nicht mehr orthogonal. Für ein $\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ haben wir die Darstellung $\overline{V}_t = (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}})V_t$. Also wird die alte Clusterbasis projiziert. Deshalb führen wir die sogenannten projizierten Clusterbasen ein.

Definition 5.3.1 (Projizierte Clusterbasis)
Es sei $V = (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis. Dann bezeichnen wir V_t für $t \in \mathcal{T}_\mathcal{I}$ als *projizierte Clusterbasis*, falls eine Matrix $\Pi_t \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\mathbf{t}\times\mathbf{t}}$, die Produkt von orthogonalen Projektionen $\Pi_i \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\mathbf{t}\times\mathbf{t}}$, $i \in \{0,\dots,m\}$, in $(\mathbb{R}^I, \|.\|_2)$ ist, und eine orthogonale Matrix $Q_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t}$ mit $V_t = \Pi_t Q_t$ existieren. Falls V_t für alle $t \in \mathcal{T}_\mathcal{I}$ eine projizierte Clusterbasis ist, bezeichnen wir V als *projizierte Clusterbasis*.

Bemerkung 5.3.2
Weil wir $\Pi_t := \Pi_{\mathbf{t}} \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\mathbf{t}\times\mathbf{t}}$ wählen können, ist jede orthogonale Clusterbasis auch eine projizierte Clusterbasis.

Wir zeigen im nächsten Lemma, dass durch das Integrieren einer orthogonalen Clusterbasis $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ mit Lemma 5.2.1 die Eigenschaft der projizierten Clusterbasis erhalten bleibt.

Lemma 5.3.3
Es sei $V = (V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine projizierte Clusterbasis, $\tilde{t} \in \mathcal{T}_\mathcal{I}$ ein Cluster und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ eine orthogonale Clusterbasis im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$. Dann ist die durch Lemma 5.2.1 definierte Clusterbasis $\overline{V} := (\overline{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine projizierte Clusterbasis.

Beweis: Wir gehen die drei Fälle der Definition von $\overline{V}$ in Lemma 5.2.1 durch.
1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann ist $\overline{V}_t$ nach Voraussetzung orthogonal und nach Bemerkung 5.3.2 eine projizierte Clusterbasis.
2. Fall: Es sei $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$. Dann ist $\overline{V}_t = (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}})V_t$. Weil V_t eine projizierte Clusterbasis ist, existiert ein Produkt von Projektionen Π_t und eine orthogonale Matrix $Q_t \in \mathbb{R}^{\mathcal{I}\times\kappa_t}_{\mathbf{t}\times\kappa_t}$ mit $V_t = \Pi_t Q_t$. Die Matrix $(\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}_{\mathbf{t}\times\mathbf{t}}$ ist eine Projektion in $(\mathbb{R}^\mathcal{I}, \|.\|_2)$. Damit ist $(\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}})\Pi_t$ ein Produkt von Projektionen wie in Definition 5.3.1 und wegen $\overline{V}_t = (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}})\Pi_t Q_t$ ist $\overline{V}_t$ eine projizierte Clusterbasis.
3. Fall: Es sei $t \in \mathcal{T}_\mathcal{I}\backslash(\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$. Dann ist $\overline{V}_t = V_t$ nach Voraussetzung eine projizierte Clusterbasis. ∎

Also erhalten wir nach dem Niedrigrangupdate wieder projizierte Clusterbasen, falls wir zuvor sichergestellt haben, dass die ursprüngliche $\mathcal{H}^2$-Matrix durch projizierte Clusterbasen dargestellt wird. Indem wir die Clusterbasen der $\mathcal{H}^2$-Matrix-Darstellung zu Beginn der Arithmetik mit Algorithmus 4.2.3 orthogonalisieren, können wir dies nach Bemerkung 5.3.2 sicherstellen.
Das zweite Problem betrifft die adaptiven Gewichte. Einerseits benötigen wir die adaptiven Gewichte von H, um die neuen Gewichte für $\widetilde{H}$ effizient in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$

zu berechnen. Andererseits werden Blöcke $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ mit $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ durch die lokale Projektion bezüglich der neuen Clusterbasis $\overline{W}_s$ verändert. Für die adaptiven Gewichte in $\mathcal{T}_{\tilde{\mathbf{t}}}$ können wir diese Veränderung abfangen, indem wir die adaptiven Gewichte in $\mathcal{T}_{\tilde{\mathbf{t}}}$ neu berechnen. Wie wir später sehen werden, lässt sich dies durch eine lokale Berechnung effizient durchführen.
Dies ist keine befriedigende Lösung für die Gewichte von $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$, weil der Aufwand zu groß würde. Allerdings liefert uns Lemma 5.2.5 eine Darstellung für die adaptive totale Clusterbasis der projizierten Matrix, die im Wesentlichen aus der alten adaptiven totalen Clusterbasis und einer Projektion besteht. Deshalb führen wir die projizierten Gewichte ein, die den adaptiven Gewichten der Matrix vor gewissen Projektionen entsprechen. Bei der Fehlerbetrachtung in Abschnitt 5.4 zeigen wir, dass die projizierten Gewichte geeignet sind, den Fehler zu kontrollieren.

Definition 5.3.4 (projizierte Gewichte)
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix. Dann heißen $(Z(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, V, \omega, \theta)_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ *projizierte Gewichte* zu $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, H, V, $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$, falls für alle $t \in \mathcal{T}_{\mathcal{I}}$ eine orthogonale Matrix $P_t \in \mathbb{R}^{\mathcal{J}\times\zeta_t}_{\tilde{\xi}_t\times\zeta_t}$ und eine Matrix $\Pi_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\tilde{\xi}_t\times\tilde{\xi}_t}$, die Produkt von orthogonalen Projektionen $\Pi_i \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\tilde{\xi}_t\times\tilde{\xi}_t}$, $i \in \{0,\dots,m\}$, in $(\mathbb{R}^J, \|.\|_2)$ ist, mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t = V_t Z_t P_t^T \Pi_t^T$ existieren.

Bemerkung 5.3.5
Durch die Wahl von $\Pi_t = \Pi_{\xi_t}$ in Definition 5.3.4 folgt, dass adaptive Gewichte (siehe Definition 4.4.17) stets auch projizierte Gewichte sind.

Die Matrizen Π_t sollten nicht mit den Einschränkungen $\Pi_{\mathbf{t}}$ verwechselt werden. Die Bezeichnung *projizierte Gewichte* haben wir gewählt, weil Π_t Produkt von orthogonalen Projektionen ist. Da Produkte von Projektionen im Allgemeinen selbst keine Projektionen sind, ist diese Bezeichnung etwas ungenau. Außerdem sind nicht die Gewichte an sich projiziert, sondern die Matrix, die diese beschreiben sollen. Im folgenden Lemma zeigen wir, dass die alten projizierten Gewichte für Cluster $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ auch projizierte Gewichte für die neue Matrix nach dem Niedrigrangupdate sind.

Lemma 5.3.6
Es sei $H = \mathcal{H}^2(V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}}$. Weiter seien $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen und $\overline{V} := (\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $\overline{W} := (\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ die Clusterbasen aus Lemma 5.2.1. Zusätzlich seien $\omega, \overline{\omega} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta, \overline{\theta} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen, wobei $\overline{\omega}_{(t,s)} = \omega_{(t,s)}$ und $\overline{\theta}_t = \theta_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$ gelte. Es seien projizierte Gewichte $(Z_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ von H zu ω und θ gegeben. Dann sind für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ die Matrizen Z_t projizierte Gewichte von $\overline{H} := \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[\widetilde{H}(H, R, \tilde{b})]$ zu $\overline{\omega}$ und $\overline{\theta}$.

Beweis: Es sei $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann folgt aus Lemma 5.1.8

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta) \;=\; X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \overline{\omega}, \overline{\theta}).$$

Wir wählen zu Z_t die Matrizen $P_t \in \mathbb{R}^{\zeta_t \times \mathcal{J}}_{\zeta_t \times \overline{\xi}_t}$ orthogonal und $\Pi_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_t \times \xi_t}$ Produkt von orthogonalen Projektionen in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$ aus der Definition 5.3.4 der projizierten Gewichte. Wir unterscheiden die beiden Fälle aus Lemma 5.2.5.
1. Fall: Es sei $t \in \operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\}$. Dann ist insbesondere $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und nach Lemma 5.2.5 existiert eine Projektion $\widehat{\Pi}_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\overline{\xi}_t \times \overline{\xi}_t}$ in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$ mit

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta}) \;&=\; (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \overline{\omega}, \overline{\theta}) \widehat{\Pi}_t \\ &=\; (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta) \widehat{\Pi}_t. \end{aligned}$$

Durch Einsetzen des projizierten Gewichts und der zugehörigen Matrizen erhalten wir

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta}) \;=\; (\Pi_{\overline{V}_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}}) V_t Z_t P_t^T \Pi_t^T \widehat{\Pi}_t \;=\; \overline{V}_t Z_t P_t^T \Pi_t^T \widehat{\Pi}_t^T \;=:\; \overline{V}_t Z_t P_t^T \overline{\Pi}_t^T.$$

Weil $\overline{\Pi}_t = \widehat{\Pi}_t \Pi_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\overline{\xi}_t \times \overline{\xi}_t}$ Produkt von Projektionen mit den geforderten Eigenschaften ist, ist Z_t ein projiziertes Gewicht von $\overline{H}$ zu $\overline{\omega}$ und $\overline{\theta}$.
2. Fall: Es sei $t \in \mathcal{T}_{\mathcal{I}} \backslash (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$.
Dann existiert nach Lemma 5.2.5 eine Projektion $\widehat{\Pi}_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\overline{\xi}_t \times \overline{\xi}_t}$ in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$ mit

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \overline{\omega}, \overline{\theta}) \;&=\; X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \overline{\omega}, \overline{\theta}) \widehat{\Pi}_t \;=\; X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta) \widehat{\Pi}_t \\ &=\; V_t Z_t P_t^T \Pi_t^T \widehat{\Pi}_t \;=\; \overline{V}_t Z_t P_t^T \Pi_t^T \widehat{\Pi}_t^T \;=:\; \overline{V}_t Z_t P_t^T \overline{\Pi}_t^T. \end{aligned}$$

Da $\overline{\Pi}_t = \widehat{\Pi}_t \Pi_t \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\overline{\xi}_t \times \overline{\xi}_t}$ wie oben Produkt von Projektionen mit den geforderten Eigenschaften ist, ist Z_t ein projiziertes Gewicht von $\overline{H}$ zu $\overline{\omega}$ und $\overline{\theta}$. ∎

Also können wir außerhalb der Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ die alten Gewichte wieder verwenden. (Die Aussage für die Gewichte der Spaltenbasis folgt über den transponierten Blockbaum.) Wir werden sehen, dass damit die Berechnung der projizierten Gewichte innerhalb der Teilbäume effizient gestaltet werden kann, sodass nach dem Niedrigrangupdate wieder projizierte Gewichte bekannt sind. Wie in Bemerkung 5.2.6 bereits erwähnt, sind wir allerdings bei der Wahl der Skalierungen eingeschränkt.

5.3.2 Orthogonale Gewichte des Niedrigrangupdates

Für die Berechnung der adaptiven Gewichte für eine $\mathcal{H}^2$-Matrix mit Algorithmus 4.4.2 werden die orthogonalen Gewichte benötigt, die dazugehörigen orthogonalen Clusterbasen allerdings nicht. Also ändern wir Algorithmus 4.2.1 so ab, dass nur die orthogonalen Gewichte gespeichert werden.

Nach Definition 5.1.1 stimmen die erweiterten Clusterbasen $\widetilde{V}$ und $\widetilde{W}$ mit den ursprünglichen Clusterbasen V und W für alle Cluster $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $s \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{s}}}$ überein. Wenn wir annehmen, dass V und W projizierte Clusterbasen sind, überträgt sich diese Eigenschaft auf diesen Teilen der Clusterbäume auf $\widetilde{V}$ und $\widetilde{W}$. Weil die projizierten Gewichte im Vergleich zu adaptiven Gewichten zusätzliche Projektionen erlauben, können wir somit für alle $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $s \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{s}}}$ die Identität als orthogonales Gewicht verwenden. (Einen ausführlichen Beweis führen wir im nächsten Teilabschnitt.)
Deshalb formulieren wir den neuen Algorithmus so, dass die orthogonalen Gewichte nur für die Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ berechnet werden. Weil der Algorithmus grundsätzlich rekursiv von den Blättern zur Wurzel arbeitet, entstehen durch die lokale Berechnung keine weiteren Probleme. Dies ermöglicht uns, lineare Komplexität in $(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$ zu erreichen.
Nach Lemma 4.2.8 und Bemerkung 4.2.11 liefern die Matrizen $(O_t)_{t \in \mathcal{T}_{\mathcal{I}}}$ aus Definition 4.2.6 orthogonale Gewichte. In Lemma 5.3.7 stellen wir eine Darstellung für die relevanten Matrizen zur Berechnung der orthogonalen Gewichte von $\widetilde{H}$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ in Termen der alten $\mathcal{H}^2$-Matrix H und der Niedrigrangmatrix $R = AB^T$ vor.

Lemma 5.3.7
Es sei $V = (V_t)_{t \in \mathcal{T}_{\mathcal{I}}}$ eine Clusterbasis, $A \in \mathbb{R}^{\mathcal{I} \times \mathcal{K}}$, $\tilde{t} \in \mathcal{T}_{\mathcal{I}}$ und $\widetilde{V} = \widetilde{V}(V, A, \tilde{t})$ die erweiterte Clusterbasis. Für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ werden orthogonale Gewichte O_t von $\widetilde{V}$ rekursiv von den Blättern aus durch QR-Zerlegungen $\widehat{Q}_t O_t = \widehat{V}(\widetilde{V}, \overline{V}(\widetilde{V}))_t$ definiert, wobei die Matrix aus Definition 4.2.6 die Gestalt

$$\widehat{V}(\widetilde{V}, \overline{V}(\widetilde{V}))_t = \begin{cases} \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} & , \text{falls } t \in \mathcal{L}_{\mathcal{I}} \\ \begin{pmatrix} O_{t_1|\rho_t \times \kappa_{t_1}} E_t & O_{t_1|\rho_t \times \mathcal{K}} \\ \vdots & \\ O_{t_\tau|\rho_t \times \kappa_{t_\tau}} E_t & O_{t_\tau|\rho_t \times \mathcal{K}} \end{pmatrix} & , \text{falls } \operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\} \neq \emptyset \end{cases}$$

besitzt.

Beweis: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Das orthogonale Gewicht O_t wird nach Definition 4.2.6 und Bemerkung 4.2.11 durch die QR-Zerlegung $\widehat{Q}_t O_t = \widehat{V}(\widetilde{V}, \overline{V}(\widetilde{V}))_t$ definiert, wobei $\overline{V} := \overline{V}(\widetilde{V})$ die Clusterbasis aus Definition 4.2.6 ist, die aus der Orthogonalisierung von $\widetilde{V}$ entsteht. Für die Matrix auf der rechten Seite gilt nach Definition 4.2.3

$$\widehat{V}(\widetilde{V}, \overline{V})_t := \begin{cases} \widetilde{V}_t & , \text{falls } t \in \mathcal{L}_{\mathcal{I}} \\ \begin{pmatrix} C(\widetilde{V}, \overline{V})_{t_1} \widetilde{E}_{t_1} \\ \vdots \\ C(\widetilde{V}, \overline{V})_{t_\tau} \widetilde{E}_{t_\tau} \end{pmatrix} & , \text{falls } \operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\} \neq \emptyset \end{cases}.$$

Für den Fall $t \in \mathcal{L}_{\tilde{\mathbf{t}}}$ folgt die Gestalt von $\widehat{V}(\widetilde{V}, \overline{V})_t$ aus $\widetilde{V}_t = \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix}$.
Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}} \backslash \mathcal{L}_{\tilde{\mathbf{t}}}$ und $\check{t} \in \operatorname{chil}(t)$. Nach Lemma 4.2.8 gilt $C(\widetilde{V}, \overline{V})_{\check{t}} = O_{\check{t}}$. Es gilt $\operatorname{par}(\check{t}) = t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Mit der Definition 5.1.1 der erweiterten Clusterbasen folgt

$$C(\widetilde{V}, \overline{V})_{\check{t}} \widetilde{E}_{\check{t}} = O_{\check{t}} \begin{pmatrix} E_{\check{t}} & 0 \\ 0 & I_{\mathcal{K}} \end{pmatrix} = \begin{pmatrix} O_{\check{t}|\rho_t \times \kappa_{\check{t}}} E_{\check{t}} & O_{\check{t}|\rho_t \times \mathcal{K}} \end{pmatrix}.$$

■

Mit Lemma 5.3.7 können wir den Algorithmus 5.3.1 zur Berechnung der orthogonalen Gewichte definieren. In Lemma 5.3.8 schätzen wir dann den Aufwand für die Berechnung der orthogonalen Gewichte im Teilbaum ab.

Algorithmus 5.3.1 Die Funktion aufgerufen mit $t = \tilde{t}$ berechnet zur erweiterten Clusterbasis $\widetilde{V}(V, A, \tilde{t})$ die orthogonalen Gewichte $(O_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und deren Zeilenindexmengen $(\rho_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$.

---

function ORTHOGONALE_GEWICHTE_UPDATE(t, V, κ, A, $\mathcal{K}$, O, ρ)
  if chil(t) $= \emptyset$ **then**
    $\widehat{V}_t \in \mathbb{R}^{\mathcal{I}\times(\kappa_t \dot\cup \mathcal{K})}_{\mathbf{t}\times(\kappa_t \dot\cup \mathcal{K})}$
    $\widehat{V}_t \leftarrow \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix}$
    QR-ZERLEGUNG($\widehat{V}_t$, $\mathcal{I}$, $\mathbf{t}$, $(\kappa_t \dot\cup \mathcal{K})$, $\overline{V}_t$, O_t, ρ_t) ▷ Algorithmus 2.4.1
  else
    for $\check{t} \in$ chil(t) **do**
      ORTHOGONALE_GEWICHTE_UPDATE($\check{t}$, V, κ, A, $\mathcal{K}$, O, ρ) ▷ bottom-up
    end for
    $\ell \leftarrow \dot\bigcup_{\check{t}\in\text{chil}(t)} \rho_{\check{t}}$
    $\widehat{V}_t \in \mathbb{R}^{\ell\times(\kappa_t \dot\cup \mathcal{K})}$
    for $\check{t} \in$ chil(t) **do**
      $\widehat{V}_{t|\rho_{\check{t}}\times(\kappa_t \dot\cup \mathcal{K})} \leftarrow \begin{pmatrix} O_{\check{t}|\rho_{\check{t}}\times\kappa_{\check{t}}} E_{\check{t}} & O_{\check{t}|\rho_{\check{t}}\times\mathcal{K}} \end{pmatrix}$
    end for
    QR-ZERLEGUNG($\widehat{V}_t$, ℓ, ℓ, $(\kappa_t \dot\cup \mathcal{K})$, $\widehat{Q}_t$, O_t, ρ_t) ▷ Algorithmus 2.4.1
  end if
end function

Lemma 5.3.8
Es sei $(V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis mit Rangverteilung $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$, $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$ und $\tilde{t} \in \mathcal{T}_\mathcal{I}$. Weiter sei $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \#\kappa_t, \#\mathcal{K}\}$. Dann ist der Aufwand für das Berechnen der orthogonalen Gewichte von $\widetilde{V}(V, A, \tilde{t})$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ mit Algorithmus 5.3.1 aufgerufen mit $t = \tilde{t}$ beschränkt durch

$$4c_{qr}\,\#\tilde{\mathbf{t}}\,k_{\max}^2 + (4 + 8c_{qr})\,\#\mathcal{T}_{\tilde{\mathbf{t}}}\,k_{\max}^3.$$

BEWEIS: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$.
Es gelte $t \in \mathcal{L}_\mathcal{I}$. Dann entsteht kein Aufwand zur Berechnung der Matrix $\widehat{V}_t \in \mathbb{R}^{\mathcal{I}\times(\kappa_t \dot\cup \mathcal{K})}_{\mathbf{t}\times(\kappa_t \dot\cup \mathcal{K})}$.
Die QR-Zerlegung braucht nach Lemma 2.4.3 höchstens

$$c_{qr}\#\mathbf{t}(\#\kappa_t + \#\mathcal{K})\min\{\#\mathbf{t}, (\#\kappa_t + \#\mathcal{K})\} \;\le\; c_{qr}\#\mathbf{t}(\#\kappa_t + \#\mathcal{K})^2 \;\le\; 4c_{qr}\#\mathbf{t}k_{\max}^2$$

Operationen.
Es gelte $t \notin \mathcal{L}_\mathcal{I}$. Dann muss zum Aufstellen der Matrix $\widehat{V}_t \in \mathbb{R}^{\ell \times (\kappa_t \dot\cup \mathcal{K})}$ für alle $\check{t} \in \operatorname{chil}(t)$ das Produkt $O_{\check{t}|_{\rho_{\check{t}} \times \kappa_{\check{t}}}} E_{\check{t}}$ berechnet werden. Nach Lemma 2.4.3 gilt $\#\rho_{\check{t}} \leq (\#\kappa_{\check{t}} + \#\mathcal{K})$. Somit benötigt die Multiplikation höchstens

$$2\#\rho_{\check{t}}\#\kappa_{\check{t}}\#\kappa_t \;\leq\; 2(\#\kappa_{\check{t}} + \#\mathcal{K})\#\kappa_{\check{t}}\#\kappa_t \;\leq\; 4k_{\max}^3$$

Operationen. Für die Berechnung der QR-Zerlegung werden nach Bemerkung 2.4.2 wegen $\ell = \dot\bigcup_{\check{t} \in \operatorname{chil}(t)} \rho_{\check{t}}$ höchstens

$$\begin{aligned} c_{qr}\#\ell(\#\kappa_t + \#\mathcal{K}) \min\{\#\ell, (\#\kappa_t + \#\mathcal{K})\} \;&\leq\; c_{qr}\#\ell 2k_{\max} 2k_{\max} \\ &= \sum_{\check{t} \in \operatorname{chil}(t)} \#\rho_{\check{t}}\, c_{qr}\, 4k_{\max}^2 \;\leq\; \sum_{\check{t} \in \operatorname{chil}(t)} 8c_{qr}\, k_{\max}^3 \end{aligned}$$

Operationen benötigt.
Der Aufruf von Algorithmus 5.3.1 mit $\tilde{t}$ berechnet nach Lemma 5.3.7 die orthogonalen Gewichte für den Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$. Der Aufwand lässt sich mit den Vorbetrachtungen und mit Lemma 3.2.8 durch

$$\begin{aligned} &\sum_{t \in \mathcal{L}_{\tilde{\mathbf{t}}}} 4c_{qr}\#\mathbf{t}k_{\max}^2 + \sum_{t \in \mathcal{T}_{\tilde{\mathbf{t}}} \setminus \mathcal{L}_{\tilde{\mathbf{t}}}} \sum_{\check{t} \in \operatorname{chil}(t)} 4k_{\max}^3 + 8c_{qr}\, k_{\max}^3 \\ &\quad\leq 4c_{qr}k_{\max}^2 \sum_{t \in \mathcal{L}_{\tilde{\mathbf{t}}}} \#\mathbf{t} + \sum_{t \in \mathcal{T}_{\tilde{\mathbf{t}}}} (4 + 8c_{qr})\, k_{\max}^3 \\ &\quad= \; 4c_{qr}\, \#\tilde{\mathbf{t}}\, k_{\max}^2 + (4 + 8c_{qr})\, \#\mathcal{T}_{\tilde{\mathbf{t}}}\, k_{\max}^3 \end{aligned}$$

beschränken. ∎

Also können wir die orthogonalen Gewichte effizient in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ berechnen. Dies ermöglicht uns die Berechnung der projizierten Gewichte, wie wir sie im nächsten Abschnitt vorstellen.

5.3.3 Projizierte Gewichte des Niedrigrangupdates

Im nächsten Schritt wollen wir die projizierten Gewichte für $\widetilde{H}$ berechnen. Dabei betrachten wir die Zeilenclusterbasis $\widetilde{V}$. Wie wir in Algorithmus 4.4.2 gesehen haben, werden die adaptiven Gewichte rekursiv von der Wurzel aus berechnet. Dieses Vorgehen ergibt sich aus der rekursiven Definition der Matrizen $\widehat{Z}_t$ in Lemma 4.4.19, die für $t \neq r_\mathcal{I}$ das Gewicht des Vaters benötigt. Aus den Matrizen $\widehat{Z}_t$ wird dann mit Hilfe einer orthogonalen Zerlegung das adaptive Gewicht Z_t berechnet.
Als Erstes zeigen wir, dass die Matrizen aus Lemma 4.4.19 eine rekursive Darstellung für die projizierten Gewichte für einen Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ liefern, falls (im Falle $\tilde{t} \neq r_\mathcal{I}$) ein projiziertes Gewicht im Elter $\operatorname{par}(\tilde{t})$ gegeben ist.

Lemma 5.3.9
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block, $(O_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Gewichte für W im Teilbaum $\mathcal{T}_{\tilde{\mathbf{s}}}$ und für alle $s \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{s}}}$ sei W_s eine projizierte Clusterbasis. Weiter seien $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen. Falls $\tilde{t} \neq r_{\mathcal{I}}$ ist, sei für $\acute{t} := \operatorname{par}(\tilde{t})$ ein projiziertes Gewicht $Z_{\acute{t}}$ von H zu ω und θ gegeben. Dann werden rekursiv von $\tilde{t}$ aus für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ durch die QR-Zerlegung $Z_t\widehat{P}_t^T = \widehat{Z}_t$ der Matrix

$$\widehat{Z}_t := \begin{cases} \begin{pmatrix} \frac{1}{\omega_{(t,s_1)}} S_{(t,s_1)} O_{s_1}^T & \cdots & \frac{1}{\omega_{(t,s_\sigma)}} S_{(t,s_\sigma)} O_{s_\sigma}^T \end{pmatrix} & \text{, falls } t = r_{\mathcal{I}} \\ \begin{pmatrix} \frac{1}{\omega_{(t,s_1)}} S_{(t,s_1)} O_{s_1}^T & \cdots & \frac{1}{\omega_{(t,s_\sigma)}} S_{(t,s_\sigma)} O_{s_\sigma}^T & \frac{1}{\theta_{\acute{t}}} E_t Z_{\acute{t}} \end{pmatrix} & \text{, falls } t \neq r_{\mathcal{I}},\ \acute{t} := \operatorname{par}(t) \end{cases}$$

projizierte Gewichte $Z_t \in \mathbb{R}^{\kappa_t\times\zeta_t}$ für H zu ω und θ definiert, wobei $\{s_1, \ldots, s_\sigma\} := \operatorname{row}(t)$ und $O_s := I_{\lambda_s}$ für alle $s \in \mathcal{T}_{\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{s}}}$ sei (vergleiche Lemma 4.4.19).

Beweis:
Wir zeigen die Aussage für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ per Induktion über $\operatorname{level}(t) - \operatorname{level}(\tilde{t}) \in \mathbb{N}$. Für $s \in \mathcal{T}_{\mathcal{J}}$ bezeichne P_s die zum orthogonalen Gewicht O_s gehörige orthogonale Matrix. Als Erstes betrachten wir $t = \tilde{t}$.
1.Fall: Es sei $\tilde{t} = r_{\mathcal{I}}$.
Dann gilt

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} &= \sum_{s\in\operatorname{row}(t)} V_{\tilde{t}} S_{(t,s)} W_s^T \\ &= V_{\tilde{t}} \sum_{s\in(\operatorname{row}(\tilde{t})\cap\mathcal{T}_{\tilde{\mathbf{s}}})} S_{(\tilde{t},s)} O_s^T P_s^T + V_{\tilde{t}} \sum_{s\in(\operatorname{row}(\tilde{t})\backslash\mathcal{T}_{\tilde{\mathbf{s}}})} S_{(\tilde{t},s)} P_s^T \Pi_s^T. \end{aligned}$$

Wir definieren $\Pi_s := \Pi_{\mathbf{s}}$ für alle $s \in (\operatorname{row}(\tilde{t}) \cap \mathcal{T}_{\tilde{\mathbf{s}}})$ und nach Voraussetzung ist $O_s = I_{\lambda_s}$ für alle $s \in (\operatorname{row}(\tilde{t})\backslash\mathcal{T}_{\tilde{\mathbf{s}}})$. Damit gilt

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} &= V_{\tilde{t}} \sum_{s\in\operatorname{row}(\tilde{t})} S_{(\tilde{t},s)} O_s^T P_s^T \Pi_s^T \\ &= V_{\tilde{t}} \begin{pmatrix} S_{(\tilde{t},s_1)} O_{s_1}^T & \cdots & S_{(\tilde{t},s_\sigma)} O_{s_\sigma}^T \end{pmatrix} \begin{pmatrix} \Pi_{s_1} P_{s_1} & \cdots & \Pi_{s_1} P_{s_\sigma} \end{pmatrix}^T. \end{aligned}$$

Die zweite Matrix ist gleich $\widehat{Z}_t$. Für die hintere Matrix gilt

$$\begin{pmatrix} \Pi_{s_1} P_{s_1} & \cdots & \Pi_{s_1} P_{s_\sigma} \end{pmatrix} = \left(\sum_{i=1}^{\sigma} \Pi_{s_1} \right) \begin{pmatrix} P_{s_1} & \ldots & P_{s_\sigma} \end{pmatrix} =: \Pi_{\tilde{t}} \widetilde{P}_{\tilde{t}}.$$

Dabei nutzen wir aus, dass $\Pi_{s_i} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\mathbf{s}_i\times\mathbf{s}_i}$ und $P_{s_i} \in \mathbb{R}^{\mathcal{J}\times\rho_i}_{\mathbf{s}_i\times\rho_i}$ für alle $i \in \{1, \ldots, \sigma\}$ gilt und die Mengen $\mathbf{s}_i$ paarweise disjunkt sind. Die Matrix $\widetilde{P}_{\tilde{t}} \in \mathbb{R}^{\mathcal{J}\times\ell_{\tilde{t}}}_{\xi_{\tilde{t}}\times\ell_{\tilde{t}}}$, $\ell_{\tilde{t}} := \dot{\bigcup}_{i=1}^{\sigma} \rho_i$, ist nach Bemerkung 2.3.15 orthogonal. Die Summanden der Matrix $\Pi_{\tilde{t}}$ sind Produkte von Projektionen. Für alle $i \in \{1, \ldots, \sigma\}$ finden wir eine Darstellung $\Pi_{s_i} = \prod_{j=1}^{m} \Pi_{s_i,j}$ mit orthogonalen Projektionen $\Pi_{s_i,j} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\mathbf{s}_i\times\mathbf{s}_i}$ in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$. (Dazu wählen wir m maximal

über alle $i \in \{1, \dots, \sigma\}$ und ergänzen die Produkte ggf. um $\Pi_{s_i,j} := \Pi_{\mathbf{s}_i}$.) Mit dieser Darstellung erhalten wir

$$\Pi_{\tilde{t}} \;=\; \sum_{i=1}^{\sigma} \Pi_{s_1} \;=\; \sum_{i=1}^{\sigma} \prod_{j=1}^{m} \Pi_{s_i,j} \;=\; \prod_{j=1}^{m} \sum_{i=1}^{\sigma} \Pi_{s_i,j} \;=:\; \prod_{j=1}^{m} \Pi_{\tilde{t},j}.$$

Weil die Mengen $\mathbf{s}_i$ paarweise disjunkt sind und $\Pi_{s_i,j} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\mathbf{s}_i\times\mathbf{s}_i}$ für alle $i \in \{1, \dots, \sigma\}$ und alle $j \in \{1, \dots, m\}$ gilt, ist $\Pi_{\tilde{t},j} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_i\times\xi_i}$ nach Lemma 2.1.6 für alle $j \in \{1, \dots, m\}$ eine orthogonale Projektion in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$.
Mit der orthogonalen Zerlegung $\widehat{P}_{\tilde{t}} Z_{\tilde{t}}^T = \widehat{Z}_{\tilde{t}}^T$ erhalten wir

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} \;=\; V_{\tilde{t}} \widehat{Z}_{\tilde{t}} \widetilde{P}_{\tilde{t}}^T \Pi_{\tilde{t}}^T \;=\; V_{\tilde{t}} Z_{\tilde{t}} \widehat{P}_{\tilde{t}}^T \widetilde{P}_{\tilde{t}}^T \Pi_{\tilde{t}}^T \;=:\; V_{\tilde{t}} Z_{\tilde{t}} P_{\tilde{t}}^T \Pi_{\tilde{t}}^T.$$

Weil die Matrizen $\widehat{P}_{\tilde{t}} \in \mathbb{R}^{\ell_{\tilde{t}}\times\zeta_{\tilde{t}}}$ und $\widetilde{P}_{\tilde{t}} \in \mathbb{R}^{\mathcal{J}\times\ell_{\tilde{t}}}_{\xi_{\tilde{t}}\times\ell_{\tilde{t}}}$ orthogonal sind, ist auch $P_{\tilde{t}} \in \mathbb{R}^{\mathcal{J}\times\zeta_{\tilde{t}}}_{\xi_{\tilde{t}}\times\ell_{\tilde{t}}}$ orthogonal. Außerdem ist $\Pi_{\tilde{t}}$ ein Produkt von Projektionen wie in der Definition 5.3.4 der projizierten Gewichte. Also ist $Z_{\tilde{t}}$ ein projiziertes Gewicht.
2.Fall: Es sei $\tilde{t} \neq r_{\mathcal{I}}$.
Es bezeichne $\acute{t} := \mathrm{par}(\tilde{t})$ den Elter von $\tilde{t}$. Dann ist nach Voraussetzung ein projiziertes Gewicht $Z_{\acute{t}}$ zu H gegeben. Somit existieren $P_{\acute{t}} \in \mathbb{R}^{\mathcal{J}\times\zeta_{\acute{t}}}_{\xi_{\acute{t}}\times\zeta_{\acute{t}}}$ orthogonal und ein $\Pi_{\acute{t}} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_{\acute{t}}\times\xi_{\acute{t}}}$, das Produkt von orthogonalen Projektionen $\Pi_{\acute{t},j} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_{\acute{t}}\times\xi_{\acute{t}}}$, $j \in \{1, \dots, m\}$, in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$ ist, mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}} = V_{\acute{t}} Z_{\acute{t}} P_{\acute{t}}^T \Pi_{\acute{t}}^T$. Wie im 1. Fall sei $\Pi_s := \Pi_{\mathbf{s}}$ für alle $s \in (\mathrm{row}(\tilde{t}) \cap \mathcal{T}_{\tilde{\mathbf{s}}})$ und $O_s := I_{\lambda_s}$ für alle $s \in (\mathrm{row}(\tilde{t}) \backslash \mathcal{T}_{\tilde{\mathbf{s}}})$. Mit der rekursiven Darstellung der adaptiven totalen Clusterbasis aus Lemma 4.4.16 und den zu den orthogonalen Gewichten O_s gehörigen orthogonalen Matrizen P_s folgt

$$\begin{aligned}
X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} \;&=\; \sum_{s\in\mathrm{row}(t)} V_t S_{(t,s)} W_s^T + X(\acute{t}, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{|\tilde{\mathbf{t}}} \\
&=\; V_{\tilde{t}} \sum_{s\in\mathrm{row}(\tilde{t})} S_{(\tilde{t},s)} O_s^T P_s^T \Pi_s^T + V_{\acute{t}} Z_{\acute{t}} P_{\acute{t}}^T \Pi_{\acute{t}}^T \\
&=\; V_{\tilde{t}} \begin{pmatrix} S_{(\tilde{t},s_1)} O_{s_1}^T & \cdots & S_{(\tilde{t},s_\sigma)} O_{s_\sigma}^T & E_{\tilde{t}} Z_{\acute{t}} \end{pmatrix} \\
&\quad \begin{pmatrix} \Pi_{s_1} P_{s_1} & \cdots & \Pi_{s_\sigma} P_{s_\sigma} & \Pi_{\acute{t}} P_{\acute{t}} \end{pmatrix}^T.
\end{aligned}$$

Die zweite Matrix ist gleich $\widehat{Z}_{\tilde{t}}$. Die dritte Matrix untersuchen wir auf ähnliche Weise wie im 1. Fall. Weil $\Pi_{\acute{t}} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\xi_{\acute{t}}\times\xi_{\acute{t}}}$, $P_{\acute{t}} \in \mathbb{R}^{\mathcal{J}\times\zeta_{\acute{t}}}_{\xi_{\acute{t}}\times\zeta_{\acute{t}}}$, $\Pi_{s_i} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\mathbf{s}_i\times\mathbf{s}_i}$ und $P_{s_i} \in \mathbb{R}^{\mathcal{J}\times\rho_i}_{\mathbf{s}_i\times\rho_i}$, $i \in \{1, \dots, \sigma\}$, gilt und die Mengen $\mathbf{s}_1, \dots, \mathbf{s}_\sigma, \xi_{\acute{t}}$ paarweise disjunkt sind, folgt

$$\begin{pmatrix} \Pi_{s_1} P_{s_1} & \cdots & \Pi_{s_1} P_{s_\sigma} & \Pi_{\acute{t}} P_{\acute{t}} \end{pmatrix} \;=\; \left(\Pi_{\acute{t}} + \sum_{i=1}^{\sigma} \Pi_{s_1} \right) \begin{pmatrix} P_{s_1} & \dots & P_{s_\sigma} & P_{\acute{t}} \end{pmatrix} \;=:\; \Pi_{\tilde{t}} \widetilde{P}_{\tilde{t}}.$$

Die Matrix $\widetilde{P}_{\tilde{t}} \in \mathbb{R}^{\mathcal{J}\times\ell_{\tilde{t}}}_{\xi_{\tilde{t}}\times\ell_{\tilde{t}}}$, $\ell_{\tilde{t}} := \left(\dot{\bigcup}_{i=1}^{\sigma} \rho_i \right) \dot{\cup}\, \zeta_{\acute{t}}$ ist nach Bemerkung 2.3.15 orthogonal. Die Summanden der Matrix $\Pi_{\tilde{t}}$ sind Produkte von Projektionen. Für alle $i \in \{1, \dots, \sigma\}$ finden

wir eine Darstellung $\Pi_{s_i} = \prod_{j=1}^{m} \Pi_{s_i,j}$ mit orthogonalen Projektionen $\Pi_{s_i,j} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\mathbf{s}_i\times\mathbf{s}_i}$ in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$. (Dazu wählen wir m wie im 1. Fall maximal und ergänzen die Produkte ggf. um Einschränkungen.) Damit erhalten wir

$$\Pi_{\tilde{t}} \;=\; \Pi_{\acute{t}} + \sum_{i=1}^{\sigma} \Pi_{s_1} \;=\; \prod_{j=1}^{m} \Pi_{\acute{t},j} + \sum_{i=1}^{\sigma} \prod_{j=1}^{m} \Pi_{s_i,j} \;=\; \prod_{j=1}^{m} \left(\Pi_{\acute{t},j} + \sum_{i=1}^{\sigma} \Pi_{s_i,j} \right) \;=:\; \prod_{j=1}^{m} \Pi_{\tilde{t},j}.$$

Dann sind $\Pi_{\tilde{t},j} \in \mathbb{R}^{\mathcal{J}\times\mathcal{J}}_{\bar{\xi}_{\tilde{t}}\times\bar{\xi}_{\tilde{t}}}$ nach Lemma 2.1.6 für alle $j \in \{1,\dots,m\}$ orthogonale Projektion in $(\mathbb{R}^{\mathcal{J}}, \|.\|_2)$. Mit der orthogonalen Zerlegung $\widehat{P}_{\tilde{t}} Z_{\tilde{t}}^T = \widehat{Z}_{\tilde{t}}^T$ erhalten wir

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} \;=\; V_{\tilde{t}} \widehat{Z}_{\tilde{t}} \widetilde{P}_{\tilde{t}}^T \Pi_{\tilde{t}}^T \;=\; V_{\tilde{t}} Z_{\tilde{t}} \widehat{P}_{\tilde{t}}^T \widetilde{P}_{\tilde{t}}^T \Pi_{\tilde{t}}^T \;=:\; V_{\tilde{t}} Z_{\tilde{t}} P_{\tilde{t}}^T \Pi_{\tilde{t}}^T.$$

Weil die Matrizen $\widehat{P}_{\tilde{t}} \in \mathbb{R}^{\ell_{\tilde{t}}\times\bar{\xi}_{\tilde{t}}}$ und $\widetilde{P}_{\tilde{t}} \in \mathbb{R}^{\mathcal{J}\times\ell_{\tilde{t}}}_{\bar{\xi}_{\tilde{t}}\times\ell_{\tilde{t}}}$ orthogonal sind, ist auch $P_{\tilde{t}}$ orthogonal. Außerdem ist $\Pi_{\tilde{t}}$ ein Produkt von Projektionen wie in der Definition 5.3.4 der projizierten Gewichte. Also ist $Z_{\tilde{t}}$ ein projiziertes Gewicht.
Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Aussage für alle $t \in \mathcal{T}_{\mathcal{I}}$ mit $\operatorname{level}(t) - \operatorname{level}(\tilde{t}) \leq n$ gilt. Es sei $t \in \mathcal{T}_{\mathcal{I}}$ mit $\operatorname{level}(t) - \operatorname{level}(\tilde{t}) = n+1 > 1$. Dann gilt $t \neq r_{\mathcal{I}}$ und es existiert $\acute{t} := \operatorname{par}(t)$. Es gilt $\operatorname{level}(\acute{t}) - \operatorname{level}(\tilde{t}) = \operatorname{level}(t) - 1 - \operatorname{level}(\tilde{t}) = n$ und somit ist $Z_{\acute{t}}$ ein projiziertes Gewicht, für das eine orthogonale Matrix $P_{\acute{t}}$ und ein Produkt von orthogonalen Projektionen $\Pi_{\acute{t}}$ mit $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}} = V_{\acute{t}} Z_{\acute{t}} P_{\acute{t}}^T \Pi_t$ existiert. Damit können wir den Beweis analog zu $t = \tilde{t} \neq r_{\mathcal{I}}$ führen und die Aussage folgt. ∎

Bemerkung 5.3.10
Falls W eine projizierte Clusterbasis ist, können wir vollständig auf orthogonale Gewichte verzichten und $O_s := I_{\lambda_s}$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ setzen.

Bemerkung 5.3.11
Falls für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$ die Clusterbasis W_s orthogonal oder ein orthogonales Gewicht O_s gegeben ist und wir für das Elter $\operatorname{par}(\tilde{t})$ ein adaptives Gewicht $Z_{\operatorname{par}(\tilde{t})}$ haben, erhalten wir in Lemma 5.3.9 adaptive Gewichte.

Als Nächstes betrachten wir die Situation unseres Niedrigrangupdates. Insbesondere wollen wir die Matrizen $\widehat{Z}$ aus Lemma 5.3.9 für unsere erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H}$ nur in Termen der alten Matrix H und der orthogonalen Gewichte von $\widetilde{W}$ in $\mathcal{T}_{\tilde{\mathbf{s}}}$ ausdrücken.

Lemma 5.3.12
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit projizierter Clusterbasis W, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{t}\times\tilde{s}}$ und $\widetilde{H} := \widetilde{H}(H, R, \tilde{b}) =: \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{V}, \widetilde{W}, \widetilde{N}, \widetilde{S})$. Weiter seien Skalierungen $\omega, \tilde{\omega} \in \mathbb{R}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}_{>0}$ und $\theta, \tilde{\theta} \in \mathbb{R}^{\mathcal{T}_{\mathcal{I}}}_{>0}$ mit $\omega_b = \tilde{\omega}_b$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{b}}}$ und $\theta_t = \tilde{\theta}_t$ für alle $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}$ gegeben. Es seien $(\widetilde{O}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Gewichte für die Clusterbasis $\widetilde{W}$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{s}}}$ und, falls $\tilde{t} \neq r_{\mathcal{I}}$ ist, für $\acute{t} := \operatorname{par}(t)$ ein projiziertes Gewicht $Z_{\acute{t}}$ von H zu ω

und θ gegeben.
Für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ definieren wir rekursiv von $\tilde{t}$ zu den Blättern $\mathcal{L}_{\tilde{\mathbf{t}}}$ die QR-Zerlegung $\widehat{P}_t \widehat{Z}_t^T = \widehat{Z}_t^T$ der Matrix

$$\widehat{Z}_t := \begin{cases} \begin{pmatrix} \frac{1}{\tilde{\omega}_{(t,s_1)}} \check{S}_{(t,s_1)} & \cdots & \frac{1}{\tilde{\omega}_{(t,s_\sigma)}} \check{S}_{(t,s_\sigma)} \end{pmatrix} & \text{, falls } t = r_{\mathcal{I}} \\ \begin{pmatrix} \frac{1}{\tilde{\omega}_{(t,s_1)}} \check{S}_{(t,s_1)} & \cdots & \frac{1}{\tilde{\omega}_{(t,s_\sigma)}} \check{S}_{(t,s_\sigma)} & \frac{1}{\tilde{\theta}_{\acute{t}}} \check{Z}_{\acute{t}} \end{pmatrix} & \text{, falls } t \neq r_{\mathcal{I}},\ \acute{t} := \operatorname{par}(t) \end{cases},$$

wobei wir die Hilfsmatrizen

$$\check{S}_b := \begin{cases} \begin{pmatrix} S_b \\ 0 \end{pmatrix} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{s}}} \\ \begin{pmatrix} S_b \widetilde{O}^T_{s|\rho_s \times \lambda_s} \\ \widetilde{O}^T_{s|\rho_s \times \mathcal{K}} \end{pmatrix} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} \end{cases}$$

und

$$\check{Z}_t := \begin{cases} \begin{pmatrix} E_t Z_{\acute{t}} \\ 0 \end{pmatrix} & \text{, falls } t = \tilde{t}, \text{ also } \acute{t} \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}} \\ \begin{pmatrix} E_t \widetilde{Z}_{\acute{t}|\kappa_{\acute{t}} \times \zeta_{\acute{t}}} \\ \widetilde{Z}_{\acute{t}|k \times \zeta_{\acute{t}}} \end{pmatrix} & \text{, falls } t, \acute{t} \in \mathcal{T}_{\tilde{\mathbf{t}}} \end{cases}.$$

verwenden. Dann ist für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ die Matrix $\widetilde{Z}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \widetilde{V}, \tilde{\omega}, \tilde{\theta})_t := \widetilde{Z}_t \in \mathbb{R}^{\widetilde{\kappa}_t \times \widetilde{\zeta}_t}$ ein projiziertes Gewichte von $\widetilde{H}$ zu $\tilde{\omega}$ und $\tilde{\theta}$ mit Rangverteilung $\widetilde{\zeta}$.

Beweis: Für den Fall, dass $\tilde{t} \neq r_{\mathcal{I}}$ ist, ist ein projiziertes Gewicht $Z_{\acute{t}}$ für $\acute{t} := \operatorname{par}(\tilde{t})$ zu H, ω und θ gegeben. Weil die Skalierungen außerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{t}}}$ gleich sind, folgt nach Lemma 5.1.8 $X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}} = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \tilde{\omega}, \tilde{\theta})_{\acute{t}}$. Also ist $\widetilde{Z}_{\acute{t}} := Z_{\acute{t}}$ auch ein projiziertes Gewicht zu $\widetilde{H}$ und den Skalierungen $\tilde{\omega}$ und $\tilde{\theta}$ (siehe Definition 5.3.4).
Weil $\widetilde{W}_s = W_s$ eine projizierte Clusterbasis für alle $s \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{s}}}$ ist, gibt uns Lemma 5.3.9 eine Darstellung für projizierte Gewichte für $\widetilde{H}$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$. Es bleibt noch zu zeigen, dass diese Matrizen die obige Gestalt besitzen.
Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$. Dann ist $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Zum Aufstellen der in Lemma 4.4.19 beschriebenen Matrix $\widehat{Z}$ wird das Matrixprodukt $\widetilde{S}_{(t,s)} O_s^T$ benötigt. Dieses wollen wir nun näher beschreiben.
1.Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann ist $\widetilde{W}_s = W_s$ projiziert und wir können $\widetilde{O}_s$ gleich der Identität setzen. Mit Definition 5.1.4 der erweiterten Kopplungsmatrizen folgt

$$\widetilde{S}_b \widetilde{O}_s^T = \widetilde{S}_b = \begin{pmatrix} S_b \\ 0 \end{pmatrix}.$$

2.Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt nach Definition 5.1.4

$$\widetilde{S}_b O_s^T = \begin{pmatrix} S_b & 0 \\ 0 & I_k \end{pmatrix} O_s^T = \begin{pmatrix} S_b O^T_{s|\rho_s \times \lambda_s} \\ O^T_{s|\rho_s \times k} \end{pmatrix}.$$

Als Nächstes betrachten wir $\widetilde{E}_t\widetilde{Z}_{\mathrm{par}(t)}$ für $t \neq r_{\mathcal{I}}$. Es sei also $t \in \mathcal{T}_{\mathcal{I}}\backslash\{r_{\mathcal{I}}\}$ und $\acute{t} = \mathrm{par}(t)$.
1.Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $\acute{t} \notin \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt $t = \tilde{t}$ und nach Definition 5.1.1 der erweiterten Clusterbasis

$$\widetilde{E}_t\widetilde{Z}_{\acute{t}} \;=\; \begin{pmatrix} E_t \\ 0 \end{pmatrix} Z_{\acute{t}} \;=\; \begin{pmatrix} E_t Z_{\acute{t}} \\ 0 \end{pmatrix}.$$

2.Fall: Es sei $t, \acute{t} \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt nach Definition 5.1.1

$$\widetilde{E}_t\widetilde{Z}_{\acute{t}} \;=\; \begin{pmatrix} E_t & 0 \\ 0 & I_k \end{pmatrix} \widetilde{Z}_{\acute{t}} \;=\; \begin{pmatrix} E_t \widetilde{Z}_{\acute{t}|\kappa_{\acute{t}}\times\zeta_t} \\ \widetilde{Z}_{\acute{t}|k\times\zeta_t} \end{pmatrix}.$$

Damit ist $\widehat{Z}_t$ wie in Lemma 4.4.19 definiert. ■

Mit der Beschreibung der projizierten Gewichte aus Lemma 5.3.12 können wir den Algorithmus zur Berechnung der projizierten Gewichte der erweiterten $\mathcal{H}^2$-Matrix $\widetilde{H}$ definieren. Dabei gehen wir davon aus, dass der Algorithmus mit $t = \tilde{t}$ aufgerufen wird und dass das reduzierte Gewicht $Z_{\mathrm{par}(\tilde{t})}$ für das Elter $\mathrm{par}(\tilde{t})$ von $\tilde{t}$ bereits berechnet ist. Außerdem verwenden wir im Algorithmus O und Z statt $\widetilde{O}$ und $\widetilde{Z}$, weil wir beim Niedrigrangupdate jeweils die alten Größen mit den neuen Ergebnissen überschreiben wollen.
Bei der Beschreibung des Algorithmus 5.3.2 nutzen wir, dass $\rho_s = \lambda_s$ für alle $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ gilt. Dies erlaubt es uns, auf weitere Fallunterscheidungen zu verzichten.

Lemma 5.3.13
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen κ und λ. Weiter sei ein Block $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, eine Niedrigrangmatrix $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\mathbf{t}\times\mathbf{s}}$ mit Rang-$\mathcal{K}$-Darstellung und die erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H} = \mathcal{H}^2(H, R, \tilde{b}) = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{V}, \widetilde{W}, \widetilde{S}, \widetilde{N})$ gegeben.
Die Clusterbasis W_s sei für alle $s \in \mathcal{T}_{\mathcal{J}}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}} \cup \mathrm{pred}(\tilde{s}))$ projiziert und $O_s \in \mathbb{R}^{(\lambda_s \dot\cup \mathcal{K})\times\rho_s}$ sei für alle $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ ein orthogonales Gewicht von $\widetilde{W}_s$ mit $\#\rho_s \leq \#\lambda_s + \#\mathcal{K}$. Es seien $\omega, \tilde{\omega} \in \mathbb{R}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}_{>0}$ und $\theta, \tilde{\theta} \in \mathbb{R}^{\mathcal{T}_{\mathcal{I}}}_{>0}$ Skalierungen mit $\omega_b = \tilde{\omega}_b$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{b}}}$ und $\theta_t = \tilde{\theta}_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$. Falls $\tilde{t} \neq r_{\mathcal{I}}$ ist, sei $Z_{\mathrm{par}(\tilde{t})} \in \mathbb{R}^{\kappa_t\times\zeta_t}$ ein projiziertes Gewicht von H zu Skalierungen ω und θ mit $\#\zeta_t \leq \#\kappa_t$.
Dann berechnet der mit $\tilde{t}$ aufgerufene Algorithmus 5.3.2 für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ die projizierten Gewichte $Z_t \in \mathbb{R}^{(\kappa_t \dot\cup \mathcal{K})\times\zeta_t}$ von $\widetilde{H}$ zu $\widetilde{V}$, $\tilde{\omega}$ und $\tilde{\theta}$ mit $\#\zeta_t \leq \#\kappa_t + \#\mathcal{K}$ in weniger als

$$(6c_{sp} + 8c_{sp}c_{qr} + 8c_{qr} + 8)k_{\max}^3 \#\mathcal{T}_{\tilde{\mathbf{t}}}$$

Operationen, wobei

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \#\kappa_t,\ \#\mathcal{K},\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s\}$$

sei.

Algorithmus 5.3.2 Die Funktion berechnet die projizierten Gewichte für die erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H}$ aus den alten Transfer- und Kopplungsmatrizen E bzw. S, den orthogonalen Gewichten O für die erweiterte Clusterbasis $\widetilde{W}$ und ggf. dem Gewicht $Z_{\acute{t}}$ für das Elter $\acute{t} := \operatorname{par}(\tilde{t})$.

function PROJIZIERTE_GEWICHTE_UPDATE(t, $\tilde{t}$, E, S, κ, $\mathcal{K}$, O, ρ, Z, ζ, $\tilde{\omega}$, $\tilde{\theta}$)
  $\ell \leftarrow \emptyset$
  for $s \in \operatorname{row}(t)$ **do**
    if $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ **then**
      $\ell \leftarrow \ell \dot{\cup} \rho_s$
    else
      $\ell \leftarrow \ell \dot{\cup} \lambda_s$
    end if
  end for
  if $t \neq r_{\mathcal{I}}$ **then**
    $\acute{t} \leftarrow \operatorname{par}(t)$
    $\ell \leftarrow \ell \dot{\cup} \zeta_{\acute{t}}$
  end if
  $\widehat{Z}_t \in \mathbb{R}^{(\kappa_t \dot{\cup} \mathcal{K}) \times \ell}$
  for $s \in \operatorname{row}(t)$ **do**
    if $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ **then**
      $\widehat{Z}_{t|\kappa_t \times \rho_s} \leftarrow \frac{1}{\tilde{\omega}_{t,s}} S_{(t,s)} O^T_{s|\rho_s \times \lambda_s}$
      $\widehat{Z}_{t|\mathcal{K} \times \rho_s} \leftarrow \frac{1}{\tilde{\omega}_{t,s}} O^T_{s|\rho_s \times \mathcal{K}}$
    else
      $\widehat{Z}_{t|\kappa_t \times \rho_s} \leftarrow \frac{1}{\tilde{\omega}_{t,s}} S_{(t,s)}$
      $\widehat{Z}_{t|\mathcal{K} \times \rho_s} \leftarrow 0$
    end if
  end for
  if $t \neq r_{\mathcal{I}}$ **then**
    $\widehat{Z}_{t|\kappa_t \times \zeta_{\acute{t}}} \leftarrow \frac{1}{\tilde{\theta}_t} E_t Z_{\acute{t}|\kappa_{\acute{t}} \times \zeta_{\acute{t}}}$
    if $t = \tilde{t}$ **then**
      $\widehat{Z}_{t|\mathcal{K} \times \zeta_{\acute{t}}} \leftarrow 0$
    else
      $\widehat{Z}_{t|\mathcal{K} \times \zeta_{\acute{t}}} \leftarrow \frac{1}{\tilde{\theta}_t} Z_{\acute{t}|\mathcal{K} \times \zeta_{\acute{t}}}$
    end if
  end if
  QR-ZERLEGUNG($\widehat{Z}_t^T$, ℓ, ℓ, $(\kappa_t \dot{\cup} \mathcal{K})$, $\widehat{P}_t$, Z_t, ζ_t) ▷ Algorithmus 2.4.1
  for $\breve{t} \in \operatorname{chil}(t)$ **do**
    PROJIZIERTE_GEWICHTE_UPDATE($\breve{t}$, $\tilde{t}$, E, S, κ, $\mathcal{K}$, O, ρ, Z, ζ, $\tilde{\omega}$, $\tilde{\theta}$)
  end for
end function

Beweis: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Zuerst betrachten wir den Aufwand für das Aufstellen von $\widehat{Z}_t$. Für alle $s \in \operatorname{row}(t) \cap \mathcal{T}_{\tilde{\mathbf{s}}}$ muss $\frac{1}{\tilde{\omega}_{t,s}} S_{(t,s)} O^T_{s|\rho_s \times \lambda_s}$ und $\frac{1}{\tilde{\omega}_{t,s}} O^T_{s|\rho_s \times \mathcal{K}}$ berechnet werden. Dies benötigt wegen $\#\rho_s \leq \#\lambda_s + \#\mathcal{K} \leq 2k_{\max}$ nicht mehr als

$$2\#\kappa_t \#\lambda_s \#\rho_s + \#\mathcal{K}\#\rho_s \;\leq\; 4k^3_{\max} + 2k^2_{\max} \;\leq\; 6k^3_{\max}$$

Operationen. Für alle $s \in \operatorname{row}(t) \backslash \mathcal{T}_{\tilde{\mathbf{s}}}$ wird $S_{(t,s)}$ mit $\frac{1}{\tilde{\omega}_{t,s}}$ skaliert. Der Aufwand dafür lässt sich auch durch $\#\kappa_t \#\lambda_s \leq k^2_{\max} \leq 6k^3_{\max}$ beschränken.
Außer in der Wurzel wird zusätzlich das Produkts $\frac{1}{\tilde{\theta}_t} E_t Z_{\acute{t}|\kappa_{\acute{t}} \times \zeta_{\acute{t}}}$ mit $\acute{t} := \operatorname{par}(t)$ benötigt. Für den Fall $t = \tilde{t}$ ist $\#\zeta_{\acute{t}} \leq \#\kappa_t \leq k_{\max}$. Ansonsten entsteht $Z_{\acute{t}}$ aus der QR-Zerlegung der Matrix $\widehat{Z}_{\acute{t}}$ mit $\#\kappa_t + \#\mathcal{K}$ Zeilen. Also gilt in diesem Fall $\#\zeta_{\acute{t}} \leq \#\kappa_t + \#\mathcal{K} \leq 2k_{\max}$. Somit kann das Produkt mit weniger als

$$3\#\kappa_t \#\kappa_{\acute{t}} \#\zeta_{\acute{t}} \;\leq\; 6k^3_{\max}$$

Operationen berechnet werden. Für $t \neq \tilde{t}$ skalieren wir zusätzlich $Z_{\acute{t}|\mathcal{K} \times \zeta_{\acute{t}}}$ mit $\frac{1}{\tilde{\theta}_t}$ in weniger als $\#\mathcal{K}\#\zeta_{\acute{t}} \leq 2k^2_{\max} \leq 2k^3_{\max}$ Operationen.
Wegen $\#\operatorname{row}(t) \leq c_{sp}$ ist der Aufwand für das Aufstellen von $\widehat{Z}_t$ somit kleiner als

$$8k^3_{\max} + \sum_{\substack{s \in \operatorname{row}(t) \\ s \in \mathcal{T}_{\tilde{\mathbf{s}}}}} 6k^3_{\max} \;\leq\; (6c_{sp} + 8)k^3_{\max}.$$

Als Nächstes schätzen wir den Aufwand der QR-Zerlegung von $\widehat{Z}_t$ ab. Aus der Definition von $k_{\max}$ und $\#\operatorname{row}(t) \leq c_{sp}$ folgt, dass die Matrix $\widehat{Z}$ maximal $\#\kappa_t + \#\mathcal{K} \leq 2k_{\max}$ Zeilen und

$$\#\ell \;\leq\; \#\zeta_{\acute{t}} + \sum_{\substack{s \in \operatorname{row}(t) \\ s \in \mathcal{T}_{\tilde{\mathbf{s}}}}} \#\rho_s + \sum_{\substack{s \in \operatorname{row}(t) \\ s \notin \mathcal{T}_{\tilde{\mathbf{s}}}}} \#\lambda_s \;\leq\; (c_{sp} + 1)2k_{\max}$$

Spalten hat. (Für $t = r_{\mathcal{I}}$ fällt der Term $\#\zeta_{\acute{t}}$ weg.) Nach Lemma 2.4.3 folgt für den Aufwand der QR-Zerlegung von $\widehat{Z}$ die obere Schranke

$$\begin{aligned} & c_{qr} 2k_{\max}(c_{sp} + 1)2k_{\max} \min\{2k_{\max}, (c_{sp} + 1)2k_{\max}\} \\ &\leq\; c_{qr} 2k_{\max}(c_{sp} + 1)2k_{\max} 2k_{\max} \\ &\leq\; 8c_{qr}(c_{sp} + 1)k^3_{\max}. \end{aligned}$$

Da diese Abschätzung unabhängig von der Wahl von $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ gilt, folgt für den Aufwand des Algorithmus 5.3.2 die obere Schranke

$$\sum_{t \in \mathcal{T}_{\tilde{\mathbf{t}}}} (6c_{sp} + 8)k^3_{\max} + 8c_{qr}(c_{sp} + 1)k^3_{\max} \;\leq\; (6c_{sp} + 8c_{sp}c_{qr} + 8c_{qr} + 8)k^3_{\max} \#\mathcal{T}_{\tilde{\mathbf{t}}}.$$

■

Falls wir mit projizierten Clusterbasen und projizierten Gewichten für H beginnen, können wir mit Algorithmus 5.3.2 projizierte Gewichte für die erweiterte Matrix $\widetilde{H}$ in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ berechnen. Der Aufwand dafür ist linear in der Größe der Teilbäume. Als Nächstes wollen wir adaptive Clusterbasen in den Teilbäumen berechnen.

5.3.4 Adaptive Clusterbasen des Niedrigrangupdates

In Abschnitt 4.4 haben wir gezeigt, dass sich mit Hilfe von Gewichten adaptive Clusterbasen für $\mathcal{H}^2$-Matrizen berechnen lassen. Diese Berechnung ist effizient (siehe Lemma 4.4.11) und der Approximationsfehler bezüglich der neuen Basis ist kontrollierbar (siehe die Lemmata 4.5.2, 4.5.4 und 4.5.5). Bevor wir den Algorithmus beschreiben, untersuchen wir die Gestalt der Matrizen aus Lemma 4.4.7. Um diese aufzustellen, benötigen wir einerseits Gewichte, wie wir sie im letzten Teilabschnitt beschrieben haben, und andererseits die Hilfsmatrizen $\widehat{V}(\widetilde{V}, \overline{V})_t$. Für Letztere erhalten wir folgende Gestalt.

Lemma 5.3.14
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$, $\tilde{t} \in \mathcal{T}_{\mathcal{I}}$ und $\widetilde{V} := \widetilde{V}(V, A, \tilde{t})$. Außerdem sei $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ eine orthogonale Clusterbasis. Wir definieren für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}\backslash\mathcal{L}_{\tilde{\mathbf{t}}}$ und $\breve{t} \in \operatorname{chil}(t)$

$$\check{C}_{\breve{t}} := \begin{pmatrix} C_{\breve{t}|\overline{\kappa}_{\breve{t}}\times\kappa_{\breve{t}}} E_{\breve{t}} & C_{\breve{t}|\overline{\kappa}_{\breve{t}}\times\mathcal{K}} \end{pmatrix},$$

wobei $C := C(\widetilde{V}, \overline{V})$ der Basiswechsel von $\widetilde{V}$ zu $\overline{V}$ in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $(\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $(\overline{\kappa}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ die Rangverteilung von V bzw. $\overline{V}$ seien. Dann gilt für die Matrizen $\widehat{V}(\widetilde{V}, \overline{V})_t$ aus Definition 4.2.3 für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$

$$\widehat{V}(\widetilde{V}, \overline{V})_t = \begin{cases} \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} & \text{, falls } t \in \mathcal{L}_{\tilde{\mathbf{t}}} \\ \begin{pmatrix} \check{C}_{t_1} \\ \vdots \\ \check{C}_{t_\tau} \end{pmatrix} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}}\backslash\mathcal{L}_{\tilde{\mathbf{t}}},\ \{t_1, \ldots, t_\tau\} = \operatorname{chil}(t). \end{cases}$$

Beweis: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$.
1.Fall: Es sei $t \in \mathcal{L}_{\tilde{\mathbf{t}}}$. Dann folgt die Behauptung aus der Definition 4.2.3 von $\widehat{V}$ und der Definition 5.1.1 der erweiterten Clusterbasis $\widetilde{V}$.
2.Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}\backslash\mathcal{L}_{\tilde{\mathbf{t}}}$. Zuerst beobachten wir, dass nach Definition der erweiterten Clusterbasis für alle $\breve{t} \in \operatorname{chil}(t)$

$$C_{\breve{t}}\widetilde{E}_{\breve{t}} = C_{\breve{t}} \begin{pmatrix} E_{\breve{t}} & 0 \\ 0 & I_{\mathcal{K}} \end{pmatrix} = \begin{pmatrix} C_{\breve{t}|\overline{\kappa}_{\breve{t}}\times\kappa_{\breve{t}}} E_{\breve{t}} & C_{\breve{t}|\overline{\kappa}_{\breve{t}}\times\mathcal{K}} \end{pmatrix}$$

gilt. Durch Einsetzen in die Formel der Definition 4.2.3 von $\widehat{V}$ folgt die Behauptung. ∎

Das Lemma 5.3.14 gibt uns also die notwendigen Darstellungen, um die adaptive Clusterbasis nach Definition 4.4.7 in $\mathcal{T}_{\tilde{\mathbf{t}}}$ zu berechnen. Indem wir die Clusterbasis $(\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ für den Teilbaum mit Lemma 5.2.1 in die alte Clusterbasis V integrieren, erhalten wir nach Lemma 5.3.3 eine projizierte Clusterbasis $\overline{V} := (\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$, falls wir mit einer projizierten Clusterbasis begonnen haben. Also können wir mit Lemma 5.3.14 und Lemma 5.2.1 den

Algorithmus 5.3.3 Berechnung der adaptiven Clusterbasis für die erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H}$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ mit Gewichten Z, Fehlerart *opt* und Fehlerschranke $\epsilon := (\epsilon_t)_{t\in\mathcal{T}_\mathcal{I}}$. Dabei wird der Basiswechsel C mit berechnet.

function ADAPTIVE_CLUSTERBASIS_UPDATE(t, $\tilde{t}$, V, E, κ, A, $\mathcal{K}$, Z, ζ, $\overline{\kappa}$, C, *opt*, ϵ)
  if chil$(t) = \emptyset$ **then**
    $\widehat{Y}_t \leftarrow V_t Z_{t|\kappa_t\times\zeta_t} + A_{|\mathbf{t}\times\mathcal{K}} Z_{t|\mathcal{K}\times\zeta_t} \in \mathbb{R}^{\mathcal{I}\times\zeta_t}_{\mathbf{t}\times\zeta_t}$
    ADAPTIVE_ORTHOGONALE_MATRIX($\widehat{Y}_t$, $\mathcal{I}$, $\mathbf{t}$, ζ_t, $\overline{V}_t$, $\overline{\kappa}_t$, *opt*, ϵ)
      ▷ Algorithmus 2.4.3
    $C_t \leftarrow \overline{V}_t^T \begin{pmatrix} V_t & A_{|\mathbf{t}} \end{pmatrix} \in \mathbb{R}^{\overline{\kappa}_t\times(\kappa_t\cup\mathcal{K})}$
    $V_t \leftarrow \overline{V}_t \in \mathbb{R}^{\mathcal{I}\times\overline{\kappa}_t}_{\mathbf{t}\times\overline{\kappa}_t}$
  else
    $\ell \leftarrow \emptyset$
    for $\breve{t} \in$ chil(t) **do** ▷ bottom-up
      ADAPTIVE_CLUSTERBASIS_UPDATE($\breve{t}$, $\tilde{t}$, V, E, κ, A, $\mathcal{K}$, Z, ζ, $\overline{\kappa}$, C, *opt*, ϵ)
      $\ell \leftarrow \ell \dot\cup \overline{\kappa}_{\breve{t}}$
    end for
    $\widehat{V}_t \in \mathbb{R}^{\ell\times(\kappa_t\,\dot\cup\,\mathcal{K})}$
    for $\breve{t} \in$ chil(t) **do**
      $\widehat{V}_{t|\overline{\kappa}_{\breve{t}}\times\kappa_t} = C_{\breve{t}|\overline{\kappa}_{\breve{t}}\times\kappa_{\breve{t}}} E_{\breve{t}}$
      $\widehat{V}_{t|\overline{\kappa}_{\breve{t}}\times\mathcal{K}} = C_{\breve{t}|\overline{\kappa}_{\breve{t}}\times\mathcal{K}}$
    end for
    $\widehat{Y}_t \in \mathbb{R}^{\ell\times\zeta_t}$
    $\widehat{Y}_t \leftarrow \widehat{V}_t Z_t$
    ADAPTIVE_ORTHOGONALE_MATRIX($\widehat{Y}_t, \widehat{Q}_t, \overline{\kappa}_t$, *opt*, ϵ) ▷ Algorithmus 2.4.3
    for $\breve{t} \in$ chil(t) **do**
      $E_{\breve{t}} \leftarrow \widehat{Q}_{t|\overline{\kappa}_{\breve{t}}\times\overline{\kappa}_t}$
    end for
    $C_t \leftarrow \widehat{Q}_t^T \widehat{V}_t$
  end if
  if $t = \tilde{t}$ **then**
    $E_t \leftarrow C_t E_t$
  end if
end function

Algorithmus 5.3.3 zur Berechnung der adaptiven Clusterbasis für das Niedrigrangupdate formulieren. Dabei überschreiben wir die alte Clusterbasis V mit der neuen $\overline{V}$, um außerhalb des Teilbaums $\mathcal{T}_{\tilde{\mathbf{t}}}$ keine weiteren Anpassungen vornehmen zu müssen.

Lemma 5.3.15
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $(\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ die Rangverteilung zu V, $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$ und $t \in \tilde{t}$. Weiter seien $Z_t \in \mathbb{R}^{\widetilde{\kappa}_t\times\zeta_t}$, $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$, Gewichte für die erweiterte Clusterbasis $\widetilde{V}(V, A, \tilde{t})$ mit $\#\zeta_t \leq \#\kappa_t + \#\mathcal{K}$. Dann erfordert das Berechnen der adaptiven Clusterbasis $\overline{V}$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ mit Algorithmus 5.3.3 höchstens

$$(4c_{apr} + 12)k_{\max}^2\#\tilde{\mathbf{t}} + (4c_{apr} + 18)k_{\max}^3\#\mathcal{T}_{\tilde{\mathbf{t}}}$$

Operationen, wobei $k_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \kappa_t,\ \max_{t\in\mathcal{T}_\mathcal{I}} \overline{\kappa}_t,\ \#\mathcal{K}\}$ ist.

Beweis: Es sei $t \in \mathcal{L}_\mathcal{I}$. Als Erstes stellen wir $\widehat{Y}_t$ auf. Dazu wird das Produkt $V_t Z_{t|\kappa_t\times\zeta_t}$ in höchstens $2\#\mathbf{t}\#\kappa_t\#\zeta_t \leq 4k_{\max}^2\#\mathbf{t}$ Operationen berechnet ($\#\zeta_t \leq \#\kappa_t + \#\mathcal{K} \leq 2k_{\max}$) und das Produkt $A_{\mathbf{t}\times k} Z_{t|k\times\zeta_t}$ in höchstens $2\#\mathbf{t}\#\mathcal{K}\#\zeta_t \leq 4k_{\max}^2\#\mathbf{t}$ Operationen hinzuaddiert. Der Aufwand für die Berechnung von $\overline{V}_t$ aus $\widehat{Y}_t$ mit Algorithmus 2.4.3 ist durch

$$c_{apr}\#\mathbf{t}\#\zeta_t \min\{\#\mathbf{t}, \#\zeta_t\} \ \leq\ c_{apr}\#\mathbf{t}\#\zeta_t\#\zeta_t \ \leq\ 4c_{apr}k_{\max}^2\#\mathbf{t}$$

beschränkt (siehe Lemma 2.4.9). Der Basiswechsel wird in höchstens $2\#\overline{\kappa}\#\mathbf{t}(\#\kappa + \#\mathcal{K}) \leq 4k_{\max}^2\#\mathbf{t}$ Operationen berechnet. Also ist der Aufwand für ein Blattcluster durch

$$8k_{\max}^2\#\mathbf{t} + 4c_{apr}k_{\max}^2\#\mathbf{t} + 4k_{\max}^2\#\mathbf{t} \ =\ (4c_{apr} + 12)k_{\max}^2\#\mathbf{t}$$

beschränkt.
Es sei $t \notin \mathcal{L}_\mathcal{I}$. Dann wird zum Aufstellen von $\widehat{V}_t$ für alle $\check{t} \in \operatorname{chil}(t)$ das Produkt $C_{\check{t}|\overline{\kappa}_{\check{t}}\times\kappa_{\check{t}}} E_{\check{t}}$ berechnet. Der Aufwand dafür ist durch

$$\sum_{\check{t}\in\operatorname{chil}(t)} 2\#\overline{\kappa}_{\check{t}}\#\kappa_{\check{t}}\#\kappa_t \ \leq\ 2\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

beschränkt. $\widehat{V}_t$ hat höchstens

$$\#\ell \ =\ \sum_{\check{t}\in\operatorname{chil}(t)} \#\overline{\kappa}_{\check{t}} \ \leq\ \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}$$

Zeilen. Also benötigt das Aufstellen von $\widehat{Y}_t$ weniger als

$$2\#\ell(\#\kappa_t + \#\mathcal{K})\#\zeta_t \ \leq\ 2\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}2k_{\max}2k_{\max} \ \leq\ 8\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

Operationen. Der Aufwand für die Berechnung von $\widehat{Q}_t$ mit Algorithmus 2.4.3 ist durch

$$c_{apr}\#\ell\#\zeta_t \min\{\#\ell, \#\zeta_t\} \ \leq\ c_{apr}\#\ell\#\zeta_t^2 \ \leq\ 4c_{apr}\sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

beschränkt (siehe Lemma 2.4.9). Dann hat $\widehat{Q}_t$ weniger als $\#\zeta_t$ Spalten und die Berechnung des Basiswechsels benötigt höchstens

$$2\#\zeta_t\#\ell(\#\kappa_t + \#\mathcal{K}) \;\leq\; 4k_{\max} \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max} 2k_{\max} \;\leq\; 8 \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

Operationen. Somit ist der Aufwand für $t \notin \mathcal{L}_{\mathcal{I}}$ durch

$$(2 + 8 + 4c_{apr} + 8) \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3 = (4c_{apr} + 18) \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3$$

beschränkt. Mit Lemma 3.2.8 folgt, dass der gesamte Algorithmus weniger als

$$\begin{aligned}
&\sum_{t\in\mathcal{L}_{\tilde{\mathbf{t}}}} (4c_{apr} + 12)k_{\max}^2 \#\mathbf{t} + \sum_{t\notin\mathcal{T}_{\tilde{\mathbf{t}}}\setminus\mathcal{L}_{\tilde{\mathbf{t}}}} (4c_{apr} + 18) \sum_{\check{t}\in\operatorname{chil}(t)} k_{\max}^3 \\
&\leq\; (4c_{apr} + 12)k_{\max}^2 \sum_{t\in\mathcal{L}_{\tilde{\mathbf{t}}}} \#\mathbf{t} + \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} (4c_{apr} + 18)k_{\max}^3 \\
&=\; (4c_{apr} + 12)k_{\max}^2 \#\tilde{\mathbf{t}} + (4c_{apr} + 18)k_{\max}^3 \#\mathcal{T}_{\tilde{\mathbf{t}}}
\end{aligned}$$

Operationen benötigt. ∎

Damit haben wir einen Algorithmus zur Berechnung einer adaptiven Clusterbasis im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ mit linearem Aufwand in den Größen $\#\tilde{\mathbf{t}}$ und $\#\mathcal{T}_{\tilde{\mathbf{t}}}$. Mit Hilfe von Lemma 5.2.1 erhalten wir die neue Clusterbasis $\overline{V}$, welche im Algorithmus direkt die alte Clusterbasis V überschreibt. Als Nächstes wollen wir einen Repräsentanten für die lokal projizierte $\mathcal{H}^2$-Matrix aufstellen.

5.3.5 Lokale $\mathcal{H}^2$-Matrix-Projektion

In diesem Abschnitt beschreiben wir die algorithmische Umsetzung der lokalen $\mathcal{H}^2$-Matrix-Projektion aus Abschnitt 5.2. Nach Lemma 5.2.4 haben wir mit der lokale Projektion $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}$ eine Abbildung, die uns mit Hilfe von lokalen Anpassungen der Kopplungsmatrizen die Bestapproximation $\overline{H}$ bezüglich der Frobeniusnorm von der erweiterten $\mathcal{H}^2$-Matrix $\widetilde{H}$ im $\mathcal{H}^2$-Matrix-Raum zu den neuen Clusterbasen $\overline{V}$ und $\overline{W}$ liefert. Im nächsten Lemma drücken wir die neuen Kopplungsmatrizen nur durch die alten Kopplungsmatrizen und den Basiswechseln aus.

Lemma 5.3.16
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ und $\widetilde{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{V}, \widetilde{W}, \widetilde{S}, \widetilde{N}) = \mathcal{H}^2(H, R, \tilde{b})$ die erweiterte $\mathcal{H}^2$-Matrix. Außerdem seien $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen. Es seien $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ die

nach Lemma 5.2.1 definierten Clusterbasen. Dann gilt für die Kopplungsmatrizen von $\Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W},\tilde{b}}[\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\widetilde{V},\widetilde{W},\widetilde{N},\widetilde{S})] =: \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W},\overline{N},\overline{S})$ aus Lemma 5.2.4

$$\overline{S}_b := \begin{cases} C_{t|\overline{\kappa}_t\times\kappa_t} S_b (D_{s|\overline{\lambda}_s\times\lambda_s})^T + C_{t|\overline{\kappa}_t\times\mathcal{K}} (D_{s|\overline{\lambda}_s\times\mathcal{K}})^T & \text{, falls } t\in\mathcal{T}_{\tilde{\mathbf{t}}},\ s\in\mathcal{T}_{\tilde{\mathbf{s}}} \\ S_b (D_{s|\overline{\lambda}_s\times\lambda_s})^T & \text{, falls } t\notin\mathcal{T}_{\tilde{\mathbf{t}}},\ s\in\mathcal{T}_{\tilde{\mathbf{s}}} \\ C_{t|\overline{\kappa}_t\times\kappa_t} S_b & \text{, falls } t\in\mathcal{T}_{\tilde{\mathbf{t}}},\ s\notin\mathcal{T}_{\tilde{\mathbf{s}}} \\ S_b & \text{, falls } t\notin\mathcal{T}_{\tilde{\mathbf{t}}},\ s\notin\mathcal{T}_{\tilde{\mathbf{s}}} \end{cases}$$

für alle $(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, wobei wir die Notation $C_t := C(\widetilde{V},\overline{V})_t$ für alle $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ und $D_s := C(\widetilde{W},\overline{W})_s$ für alle $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$ verwenden.

Beweis: Es sei $b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.
1. Fall: Es sei $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt nach Definition 5.1.4 der erweiterten Kopplungsmatrizen

$$\begin{aligned}\overline{S}_b &= C(\widetilde{V},\overline{V})_t \widetilde{S}_b C(\widetilde{W},\overline{W})_s^T = C_t \begin{pmatrix} S_b & 0 \\ 0 & I_k \end{pmatrix} D_s^T \\ &= C_{t|\overline{\kappa}_t\times\kappa_t} S_b (D_{s|\overline{\lambda}_s\times\lambda_s})^T + C_{t|\overline{\kappa}_t\times\mathcal{K}} (D_{s|\overline{\lambda}_s\times\mathcal{K}})^T.\end{aligned}$$

2. Fall: Es sei $t\notin\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$. Wir setzen die Definition der erweiterten Kopplungsmatrizen ein und erhalten

$$\overline{S}_b = \widetilde{S}_b C(\widetilde{W},\overline{W})_s^T = \begin{pmatrix} S_b & 0 \end{pmatrix} D_s^T = S_b (D_{s|\overline{\lambda}_s\times\lambda_s})^T.$$

3. Fall: Es sei $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\notin\mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt

$$\overline{S}_b = C(\widetilde{V},\overline{V})_t \widetilde{S}_b = C_t \begin{pmatrix} S_b \\ 0 \end{pmatrix} = C_{t|\overline{\kappa}_t\times\kappa_t} S_b.$$

4. Fall: Es sei $t\notin\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\notin\mathcal{T}_{\tilde{\mathbf{s}}}$. In diesem Fall gilt $\overline{S}_b = \widetilde{S}_b = S_b$. ∎

Der letzte Fall in Lemma 5.3.16 zeigt, dass wir eine Kopplungsmatrix $S_{(t,s)}$ nicht ändern müssen, die weder $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ als Zeilencluster noch $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$ als Spaltencluster haben. Wir teilen die Berechnungen der neuen Kopplungsmatrizen in zwei Schritte auf. Als Erstes werden für die Blockzeilen/-spalten von Clustern $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$ außerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$ die alten Kopplungsmatrizen S_b mit den Basiswechseln $C_{t|\overline{\kappa}_t\times\kappa_t}$ bzw. $D^T_{s|\overline{\lambda}_s\times\lambda_s}$ multipliziert (siehe Algorithmus 5.3.4). Danach werden mit Algorithmus 5.3.5 die Kopplungs- und Nahfeldmatrizen innerhalb des Teilbaums $\mathcal{T}_{\tilde{\mathbf{b}}}$ aktualisiert. Die gesamte Projektion fassen wir dann in Algorithmus 5.3.6 zusammen.

Die Funktionen überschreiben die alten Nahfeld- und Kopplungsmatrizen, weil wir beim lokalen Niedrigrangupdate nur lokale Änderungen vornehmen wollen. Den Aufwand der Funktionen schätzen wir in Lemma 5.3.17 ab.

Algorithmus 5.3.4 Die Funktion berechnet die neuen Kopplungsmatrizen $\overline{S}$ aus den alten S in den Blockzeilen von $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ außerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$. Dabei werden die Basiswechsel C verwendet.

function H2-Matrix-Projektion_ausserhalb(t, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\tilde{s}$, S, κ, λ, C, $\overline{\kappa}$)
  for $s \in \operatorname{row}(t)\backslash\mathcal{T}_{\tilde{\mathbf{s}}}$ **do**
    $\overline{S}_{(t,s)} \in \mathbb{R}^{\overline{\kappa}\times\lambda_s}$
    $\overline{S}_{(t,s)} \leftarrow C_{t|\overline{\kappa}_t\times\kappa_t} S_{(t,s)}$
    $S_{(t,s)} \leftarrow \overline{S}_{(t,s)}$
  end for
  for $\breve{t} \in \operatorname{chil}(t)$ **do**
    H2-Matrix-Projektion_ausserhalb($\breve{t}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\tilde{s}$, S, κ, λ, C, $\overline{\kappa}$)
  end for
end function

Algorithmus 5.3.5 Die Funktion berechnet die neuen Kopplungsmatrizen $\overline{S}$ innerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$. Dabei werden die Basiswechsel C_t und D_s verwendet. Außerdem werden die neuen Nahfeldmatrizen $\overline{N}_b$ aus den alten N_b und den Matrizen A und B der Niedrigrangmatrix berechnet.

function H2-Matrix-Projektion_innerhalb(b, N, S, κ, λ, A, B, $\mathcal{K}$, C, $\overline{\kappa}$, D, $\overline{\lambda}$)
  $(t,s) \leftarrow b$
  if $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ **then**
    $\widehat{S} \leftarrow C_{t|\overline{\kappa}_t\times\kappa_t} S_{(t,s)} (D_{s|\overline{\lambda}_s\times\lambda_s})^T \in \mathbb{R}^{\overline{\kappa}_t\times\overline{\lambda}_s}$
    $S_{(t,s)} \leftarrow \widehat{S} + C_{t|\overline{\kappa}_t\times\mathcal{K}} D^T_{s|\overline{\lambda}_s\times\mathcal{K}} \in \mathbb{R}^{\overline{\kappa}_t\times\overline{\lambda}_s}$
  else
    if $b \in \mathcal{L}_{\mathcal{I}}$ **then**
      $N_b \leftarrow N_b + A_{|\mathbf{t}\times\mathcal{K}} B^T_{|\mathbf{s}\times\mathcal{K}}$
    else ▷ $b \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$
      for $\breve{b} \in \operatorname{chil}(b)$ **do**
        H2-Matrix-Projektion_innerhalb($\breve{b}, N, S, \kappa, \lambda, A, B, \mathcal{K}, C, \overline{\kappa}, D, \overline{\lambda}$)
      end for
    end if
  end if
end function

Algorithmus 5.3.6 Die Funktion überschreibt die alten Kopplungsmatrizen S und Nahfeldmatrizen N mit den neuen Kopplungsmatrizen $\overline{S}$ und Nahfeldmatrizen $\overline{N}$ des rekomprimierten Niedrigrangupdates.

function H2-Matrix-Projektion_Update($\tilde{b}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, N, S, κ, λ, A, B, $\mathcal{K}$, C, $\overline{\kappa}$, D, $\overline{\lambda}$)

$(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$

H2-Matrix-Projektion_ausserhalb($\tilde{t}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\tilde{s}$, S, κ, λ, C, $\overline{\kappa}$)

H2-Matrix-Projektion_ausserhalb($\tilde{s}$, $\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}$, $\tilde{t}$, S^T, λ, κ, D, $\overline{\lambda}$)

▷ Algorithmus 5.3.4

H2-Matrix-Projektion_innerhalb($\tilde{b}$, N, S, κ, λ, A, B, $\mathcal{K}$, C, $\overline{\kappa}$, D, $\overline{\lambda}$)

▷ Algorithmus 5.3.5

end function

Lemma 5.3.17
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix mit strikt zulässigem, c_{sp}-schwachbesetztem Blockbaum, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $R = AB \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung. Zusätzlich seien $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen mit Rangverteilungen $\overline{\kappa}$ bzw. $\overline{\lambda}$. Wir definieren den maximalen lokalen Rang

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s, \#\mathcal{K}, \max_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \#\overline{\kappa}_t, \max_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}} \#\overline{\lambda}_s\}$$

und die maximale Blattgröße $n_{\max} := \max\{\max_{t\in\mathcal{L}_{\mathcal{I}}} \#\mathbf{t}, \max_{s\in\mathcal{L}_{\mathcal{J}}} \#\mathbf{s}\}$. Dann ist der Aufwand für das Aktualisieren der Nahfeld- und Kopplungsmatrizen durch den Algorithmus 5.3.6 beschränkt durch

$$c_{sp}(2k^3_{\max} + \max\{k^2_{\max}, n^2_{\max}\}k_{\max})(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).$$

Beweis: Der Algorithmus 5.3.4 aufgerufen in $\tilde{t}$ benötigt in jedem Cluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ höchstens

$$\sum_{\substack{s\in\operatorname{row}(t)\\ s\notin\mathcal{T}_{\tilde{\mathbf{s}}}}} 2\#\overline{\kappa}_t\#\kappa_t\#\lambda_s \;\le\; \sum_{\substack{s\in\operatorname{row}(t)\\ s\notin\mathcal{T}_{\tilde{\mathbf{s}}}}} 2k^3_{\max}$$

Operationen. Analog ist der Aufwand für den Aufruf von Algorithmus 5.3.4 aufgerufen in $\tilde{s}$ in jedem Cluster $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ beschränkt durch

$$\sum_{\substack{t\in\operatorname{col}(s)\\ t\notin\mathcal{T}_{\tilde{\mathbf{t}}}}} 2\#\overline{\lambda}_s\#\lambda_s\#\kappa_t \;\le\; \sum_{\substack{t\in\operatorname{col}(s)\\ t\notin\mathcal{T}_{\tilde{\mathbf{t}}}}} 2k^3_{\max}.$$

Für den Algorithmus 5.3.5 aufgerufen mit $\tilde{b}$ ergibt sich in einem zulässigen Blatt $(t, s) = b \in \mathcal{L}^+_{\mathbf{b}}$ ein Aufwand von höchstens

$$2\#\overline{\kappa}_t\#\kappa_t\#\lambda_s \;+\; 2\#\overline{\kappa}_t\#\lambda_s\#\overline{\lambda}_s \;+\; 2\#\overline{\kappa}_t\#\mathcal{K}\#\overline{\lambda}_s \le\; (2+2+2)k^3_{\max} \;=\; 6k^3_{\max}.$$

Für ein unzulässiges Blatt $b = (t, s) \in \mathcal{L}_{\mathbf{b}}^-$ ist $t \in \mathcal{L}_{\mathcal{I}}$ und $s \in \mathcal{L}_{\mathcal{J}}$. Also gilt $\#\mathbf{t} \leq n_{\max}$ und $\#\mathbf{s} \leq n_{\max}$ und es folgt die obere Schranke

$$2\#\mathbf{t}\#\mathcal{K}\#\mathbf{s} \;\leq\; 2k_{\max}n_{\max}^2.$$

Bevor wir den Aufwand von Algorithmus 5.3.6 abschätzen, formen wir den Aufwand in den zulässigen Blättern um und erhalten

$$\sum_{b\in\mathcal{L}_{\mathbf{b}}^+} 6k_{\max}^3 \;=\; \sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \sum_{\substack{s\in\operatorname{row}(t)\\ s\in\mathcal{T}_{\tilde{\mathbf{s}}}}} 2k_{\max}^3 + \sum_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}} \sum_{\substack{t\in\operatorname{row}(s)\\ t\in\mathcal{T}_{\tilde{\mathbf{t}}}}} 2k_{\max}^3 + \sum_{b\in\mathcal{L}_{\mathbf{b}}^+} 2k_{\max}^3.$$

Zusammen mit Lemma 3.3.6 ergibt sich ein Aufwand für Algorithmus 5.3.6 von höchstens

$$\begin{aligned}
&\sum_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}} \sum_{s\in\operatorname{row}(t)} 2k_{\max}^3 + \sum_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}} \sum_{t\in\operatorname{row}(s)} 2k_{\max}^3 + \sum_{b\in\mathcal{L}_{\tilde{\mathbf{b}}}^+} 2k_{\max}^3 + \sum_{b\in\mathcal{L}_{\tilde{\mathbf{b}}}^-} 2k_{\max}n_{\max}^2 \\
&\qquad\leq\; 2c_{sp}k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}) + 2\max\{k_{\max}^2, n_{\max}^2\}k_{\max}\#\mathcal{L}_{\tilde{\mathbf{b}}} \\
&\qquad\leq\; c_{sp}(2k_{\max}^3 + \max\{k_{\max}^2, n_{\max}^2\}k_{\max})(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).
\end{aligned}$$

■

Damit haben wir einen Weg gefunden, die neue Matrix in linearer Komplexität bezüglich der Mächtigkeit der Teilbäume $(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$ zu berechnen. Den entstehenden Fehler schätzen wir in Abschnitt 5.4 ab.

5.3.6 Neue projizierte Gewichte

Nach Lemma 5.3.6 können wir für $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ die alten projizierten Gewichte für die neue Matrix $\overline{H}$ verwenden. Falls im Falle von $\tilde{t} \neq r_{\mathcal{I}}$ ein projiziertes Gewicht für das Elter $\operatorname{par}(\tilde{t})$ gegeben ist, müssen wir die projizierten Gewichte also lediglich im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ neu berechnen. In Lemma 5.3.18 beweisen wir für diesen Fall eine Darstellung für neue projizierte Gewichte in $\mathcal{T}_{\tilde{\mathbf{t}}}$, die ohne orthogonale Gewichte auskommt.

Lemma 5.3.18
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit projizierter Clusterbasis W und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block. Weiter seien $(\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ orthogonale Clusterbasen. Es sei $\overline{H} := \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S}) := \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V},\overline{W},\tilde{b}}[H]$ die lokale $\mathcal{H}^2$-Projektion, wobei $\overline{V} = (\overline{V}_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $\overline{W} = (\overline{W}_s)_{s\in\mathcal{T}_{\mathcal{J}}}$ die in Lemma 5.2.1 definierten Clusterbasen seien. Zusätzlich seien $\omega, \overline{\omega} \in \mathbb{R}_{>0}^{\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+}$ und $\theta, \overline{\theta} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ Skalierungen mit $\overline{\omega}_{(t,s)} = \omega_{(t,s)}$ und $\overline{\theta}_t = \theta_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$. Falls $\tilde{t} \neq r_{\mathcal{I}}$ ist, sei für $\acute{t} := \operatorname{par}(t)$ ein projiziertes Gewicht $Z_{\acute{t}}$ von H zu ω und θ gegeben.
Für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ definieren wir rekursiv von $\tilde{t}$ zu den Blättern $\mathcal{L}_{\tilde{\mathbf{t}}}$

$$\widehat{Z}_t \;:=\; \begin{cases} \begin{pmatrix} \frac{1}{\overline{\omega}_{(t,s_1)}}\overline{S}_{(t,s_1)} & \cdots & \frac{1}{\overline{\omega}_{(t,s_\sigma)}}\overline{S}_{(t,s_\sigma)} \end{pmatrix} & \text{, falls } t = r_{\mathcal{I}} \\ \begin{pmatrix} \frac{1}{\overline{\omega}_{(t,s_1)}}\overline{S}_{(t,s_1)} & \cdots & \frac{1}{\overline{\omega}_{(t,s_\sigma)}}\overline{S}_{(t,s_\sigma)} & \frac{1}{\overline{\theta}_t}\overline{E}_t\overline{Z}_{\acute{t}} \end{pmatrix} & \text{, falls } t \neq r_{\mathcal{I}},\ \acute{t} := \operatorname{par}(t) \end{cases},$$

wobei $\overline{Z}_t$ jeweils durch die QR-Zerlegung $\widehat{P}_t \overline{Z}_t^T := \widehat{Z}_t^T$ gegeben sei. Dann ist für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ die Matrix $\overline{Z}_t \in \mathbb{R}^{\overline{\kappa}_t \times \overline{\zeta}_t}$ ein projiziertes Gewicht von $\overline{H}$ zu $\overline{\omega}$ und $\overline{\theta}$ mit Rangverteilung $\overline{\zeta}$ mit $\#\zeta_t \leq \#\overline{\kappa}_t$ für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$.

Beweis: Die neue Clusterbasis $\overline{W}$ ist nach Lemma 5.3.3 wieder eine projizierte Clusterbasis. Für den Fall $\tilde{t} \neq r_{\mathcal{I}}$ ist nach Lemma 5.2.5 $Z_{\acute{t}}$ ein projiziertes Gewicht von $\overline{H}$ zu $\overline{\omega}$ und $\overline{\theta}$ für das Elter von $\tilde{t}$ (vergleiche Beweis von Lemma 5.3.6). Somit liefert uns Lemma 5.3.9 eine Darstellung für projizierte Gewichte in $\mathcal{T}_{\tilde{\mathbf{t}}}$, bei der nach Bemerkung 5.3.10 keine orthogonalen Gewichte notwendig sind. Also gilt die Aussage des Lemmas. ∎

Die Matrizen in Lemma 5.3.18 sind gleich denen in Lemma 4.4.19 mit Identitäten als orthogonalen Gewichten. Aus der Darstellung in Lemma 5.3.18 ergibt sich Algorithmus 5.3.7 zur Berechnung der neuen projizierten Gewichte in $\mathcal{T}_{\tilde{\mathbf{t}}}$. Den Aufwand für Algorithmus 5.3.7 schätzen wir in Lemma 5.3.19 ab.

Algorithmus 5.3.7 Die Funktion berechnet die projizierten Gewichte zu den Skalierungen ω und θ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$. Dabei wird davon ausgegangen, dass die Spaltenbasis eine projizierte Clusterbasis ist.

```
function PROJIZIERTE_GEWICHTE(t, E, S, κ, λ, ω, θ, Z, ζ)
    ℓ ← ⋃̇_{s∈row(t)} λ_s
    if t ≠ r_I then
        ť := par(t)
        ℓ ← ℓ ∪̇ ζ_ť
    end if
    Ẑ ∈ R^{κ_t × ℓ}
    if t ≠ r_I then
        Ẑ_{t_{|κ_t × ζ_ť}} ← (1/θ_ť) E_t Z_ť
    end if
    for s ∈ row(t) do
        Ẑ_{t_{|κ_t × λ_s}} ← (1/ω_(t,s)) S_(t,s)
    end for
    QR-ZERLEGUNG(Ẑ^T, ℓ, ℓ, κ_t, P_t, Z_t^T, ζ_t)        ▷ Algorithmus 2.4.1
    for ť ∈ chil(t) do
        PROJIZIERTE_GEWICHTE(ť, E, S, κ, λ, ω, θ, Z, ζ)        ▷ top-down
    end for
end function
```

Lemma 5.3.19
Es sei $H \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen κ und λ. Weiter sei $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t,\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s\}$ und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block. Dann ist der Aufwand für den Aufruf von Algorithmus 5.3.7

im Cluster $t = \tilde{t}$ mit den Skalierungen ω und θ höchstens

$$(c_{sp} + c_{qr}c_{sp} + c_{qr} + 3)k_{\max}^3 \#\mathcal{T}_{\tilde{\mathbf{t}}}.$$

Beweis: Im Vergleich zu Algorithmus 4.4.2 wird das Produkt $\frac{1}{\omega_{(t,s)}} S_{(t,s)} O_s$ in Algorithmus 5.3.7 durch die Skalierung $\frac{1}{\omega_{(t,s)}} S_{(t,s)} O_s$ ersetzt. Weil die Gesamtkomplexität sich dadurch nicht ändert können wir den Algorithmus wie in Lemma 4.4.22 abschätzen und erhalten die Aufwandsschranke. ∎

Damit können wir die projizierten Gewichte in $\mathcal{T}_{\tilde{\mathbf{t}}}$ in linearer Komplexität bezüglich $\#\mathcal{T}_{\tilde{\mathbf{t}}}$ berechnen. Als Nächstes fassen wir die einzelnen Teile zum lokalen Niedrigrangupdate zusammen.

5.3.7 Lokales Niedrigrangupdate

Mit Hilfe der vorherigen Funktionen kann nun das Niedrigrangupdate für eine $\mathcal{H}^2$-Matrix beschrieben werden. Es seien also eine $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$, eine Matrix in Rang-$\mathcal{K}$-Darstellung $R = AB^T$ und $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ gegeben. Dann berechnet der Algorithmus 5.3.8 eine Approximation der Summe $H + R$. In Theorem 5.3.20 geben wir eine Aufwandsabschätzung für den Algorithmus an.

Theorem 5.3.20 (Aufwand des Niedrigrangupdate)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit projizierten Clusterbasen V und W, Rangverteilungen κ und λ und strikt zulässigem, c_{sp}-schwachbesetztem Blockbaum. Weiter sei $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung. Es seien projizierte Gewichte Z_I und Z_J zu H bzw. H^T und den Skalierungen $\omega_{\mathcal{I}} \in \mathbb{R}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}_{>0}$ und $\theta_{\mathcal{I}} \in \mathbb{R}^{\mathcal{T}_{\mathcal{I}}}_{>0}$ bzw. $\omega_{\mathcal{J}} \in \mathbb{R}^{(\mathcal{L}^T_{\mathcal{I}\times\mathcal{J}})^+}_{>0}$ und $\theta_{\mathcal{J}} \in \mathbb{R}^{\mathcal{T}_{\mathcal{J}}}_{>0}$ gegeben. Wir definieren den maximalen Rang

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t,\ \#\mathcal{K},\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s,\ \max_{t\in\mathcal{T}_{\mathcal{I}}} \#\overline{\kappa}_t,\ \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\overline{\lambda}_s\},$$

wobei $\overline{\kappa}$ und $\overline{\lambda}$ die Rangverteilungen der von Algorithmus 5.3.8 berechneten $\mathcal{H}^2$-Matrix $\overline{H}$ sei, und die maximale Blattgröße $n_{\max} := \max\{\max_{t\in\mathcal{L}_{\mathcal{I}}} \#\mathbf{t}, \max_{s\in\mathcal{L}_{\mathcal{J}}} \#\mathbf{s}\}$. Dann benötigt der Algorithmus 5.3.8 für die Berechnung von $\overline{H} \approx H + X$ höchstens

$$\begin{aligned}
&(12 + 4c_{qr} + 4c_{apr})k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) \\
+\ &(33 + 17c_{qr} + 9c_{qr}c_{sp} + 9c_{\|.\|}c_{sp} + 72c_{sp} + 4c_{apr})k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}) \\
+\ &c_{sp}\max\{k_{\max}^2, n_{\max}^2\}k_{\max}(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).
\end{aligned}$$

Operationen. Die $\mathcal{H}^2$-Matrix-Darstellung von H wird mit der Darstellung von $\overline{H}$ mit den projizierten Clusterbasen $\overline{V}$ und $\overline{W}$ überschrieben. In den alten Gewichten $Z_{\mathcal{I}}$ und $Z_{\mathcal{J}}$ werden projizierte Gewichte für $\overline{H}$ zu den Skalierungen $\overline{\omega}_{\mathcal{I}}$, $\overline{\omega}_{\mathcal{J}}$, $\overline{\theta}_{\mathcal{I}}$ und $\overline{\theta}_{\mathcal{J}}$, wobei diese außerhalb der Bäume $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ jeweils gleich den alten Skalierungen gesetzt werden.

Algorithmus 5.3.8 Die Funktion berechnet das Niedrigrangupdate einer $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ für eine Rang-$\mathcal{K}$-Matrix $R = AB^T$.

function NIEDRIGRANGUPDATE($\tilde{b}$, H, $Z_\mathcal{I}$, $\zeta_\mathcal{I}$, $Z_\mathcal{J}$, $\zeta_\mathcal{J}$, R, $\mathcal{K}$, *opt*, ϵ_I, $\epsilon_\mathcal{J}$, *str*)
ORTHOGONALE_GEWICHTE_UPDATE($\tilde{t}$, V, κ, A, $\mathcal{K}$, $O_\mathcal{I}$, $\rho_\mathcal{I}$)
ORTHOGONALE_GEWICHTE_UPDATE($\tilde{s}$, W, λ, B, $\mathcal{K}$, $O_\mathcal{J}$, $\rho_\mathcal{J}$)
▷ Algorithmus 5.3.1
SKALIERUNG($\tilde{b}$, S, κ, λ, $O_\mathcal{I}$, $\rho_\mathcal{I}$, $O_\mathcal{J}$, $\rho_\mathcal{J}$, *opt*, *str*, $\omega_\mathcal{I}$)
SKALIERUNG($\tilde{b}^T$, S^T, λ, κ, $O_\mathcal{J}$, $\rho_\mathcal{J}$, $O_\mathcal{I}$, $\rho_\mathcal{I}$, *opt*, *str*, $\omega_\mathcal{J}$)
▷ Algorithmus 4.5.2
PROJIZIERTE_GEWICHTE_UPDATE($\tilde{t}, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \tilde{t}, E, S, \kappa, \mathcal{K}, O_\mathcal{I}, \rho_\mathcal{I}, Z_\mathcal{I}, \zeta_\mathcal{I}, \tilde{\omega}_\mathcal{I}, \tilde{\theta}_\mathcal{I}$)
PROJIZIERTE_GEWICHTE_UPDATE($\tilde{s}, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \tilde{s}, F, S^T, \lambda, \mathcal{K}, O_\mathcal{J}, \rho_\mathcal{J}, Z_\mathcal{J}, \zeta_\mathcal{J},$
$\tilde{\omega}_\mathcal{J}, \tilde{\theta}_\mathcal{J}$) ▷ Algorithmus 5.3.2
ADAPTIVE_CLUSTERBASIS_UPDATE($\tilde{t}$, $\tilde{t}$, V, κ, A, $\mathcal{K}$, $Z_\mathcal{I}$, $\zeta_\mathcal{I}$, $\overline{\kappa}$, $C_\mathcal{I}$, *opt*, $\epsilon_\mathcal{I}$)
ADAPTIVE_CLUSTERBASIS_UPDATE($\tilde{s}$, $\tilde{s}$, W, λ, B, $\mathcal{K}$, $Z_\mathcal{J}$, $\zeta_\mathcal{J}$, $\overline{\lambda}$, $C_\mathcal{J}$, *opt*, $\epsilon_\mathcal{J}$)
▷ Algorithmus 5.3.3
H2-MATRIX-PROJEKTION_UPDATE($\tilde{b}$, N, S, κ, λ, A, B, $\mathcal{K}$, $C_\mathcal{I}$, $\overline{\kappa}$, $C_\mathcal{J}$, $\overline{\lambda}$)
▷ Algorithmus 5.3.6
SKALIERUNG($\tilde{b}$, S, $\overline{\kappa}$, $\overline{\lambda}$, $(I_{\overline{\kappa}_t})_{t\in\mathcal{T}_\mathcal{I}}$, $\overline{\kappa}$, $(I_{\overline{\lambda}_s})_{s\in\mathcal{T}_\mathcal{J}}$, $\overline{\lambda}$, *opt*, *str*, $\omega_\mathcal{I}$)
SKALIERUNG($\tilde{b}^T$, S^T, $\overline{\lambda}$, $\overline{\kappa}$, $(I_{\overline{\lambda}_s})_{s\in\mathcal{T}_\mathcal{J}}$, $\overline{\lambda}$, $(I_{\overline{\kappa}_t})_{t\in\mathcal{T}_\mathcal{I}}$, $\overline{\kappa}$, *opt*, *str*, $\omega_\mathcal{J}$)
▷ Algorithmus 4.5.2
PROJIZIERTE_GEWICHTE($\tilde{t}$, E, S, $\overline{\kappa}$, $\overline{\lambda}$, $\overline{\omega}_\mathcal{I}$, $\overline{\theta}_\mathcal{I}$, $Z_\mathcal{I}$, $\zeta_\mathcal{I}$)
PROJIZIERTE_GEWICHTE($\tilde{s}$, F, S^T, $\overline{\lambda}$, $\overline{\kappa}$, $\overline{\omega}_\mathcal{J}$, $\overline{\theta}_\mathcal{J}$ $Z_\mathcal{J}$, $\zeta_\mathcal{J}$)
▷ Algorithmus 5.3.7
end function

Beweis: Um $\overline{H}$ zu berechnen, wird Algorithmus 5.3.8 im Block $\tilde{b}$ aufgerufen. Die Berechnung der orthogonalen Gewichte für $\widetilde{H} := \widetilde{H}(H, R, \tilde{b})$ in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ benötigt nach Lemma 5.3.8 höchstens

$$4c_{qr}\, k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (4 + 8c_{qr})\, k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen. Die Skalierungen werden nach Lemma 4.5.7 mit Algorithmus 4.5.2 mit nicht mehr als $8(7 + c_{\|.\|})c_{sp}k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$ Operationen berechnet. Nach Lemma 5.3.9 berechnet Algorithmus 5.3.7 projizierte Gewichte von $\widetilde{H}$ zu den Skalierungen $\tilde{\omega}_{\mathcal{I}}$, $\tilde{\omega}_{\mathcal{J}}$, $\tilde{\theta}_{\mathcal{I}}$ und $\tilde{\theta}_{\mathcal{J}}$, wobei diese außerhalb der Teilbäume $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ gleich den alten Skalierungen gesetzt werden. Für den Aufwand gilt nach Lemma 5.3.13 die obere Schranke

$$(6c_{sp} + 8c_{qr}c_{sp} + 8c_{qr} + 8)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).$$

Danach werden die neuen Clusterbasen $\overline{V}$ und $\overline{W}$ in $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ in höchstens

$$(4c_{apr} + 12)k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (4c_{apr} + 18)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen aufgestellt (siehe Lemma 5.3.15). Dabei werden die Ergebnisse in V bzw. W gespeichert und gleichzeitig die Basiswechsel C und D berechnet. Also sind die neuen Clusterbasen $\overline{V}$ und $\overline{W}$ in V und W gespeichert und nach Lemma 5.3.3 sind $\overline{V}$ und $\overline{W}$ projizierte Clusterbasen. Mit den Basiswechseln werden die Nahfeld- und Kopplungsmatrizen der bezüglich $\overline{V}$ und $\overline{W}$ lokal projizierten Matrix $\overline{H}$ mit Algorithmus 5.3.6 nach Lemma 5.3.17 in weniger als

$$c_{sp}(2k_{\max}^3 + \max\{k_{\max}^2, n_{\max}^2\}k_{\max})(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen berechnet. Dabei werden die alten Kopplungs- und Nahfeldmatrizen mit den neuen überschrieben. Somit können die Skalierungen in diesem Fall nach Lemma 4.5.7 mit Algorithmus 4.5.2 in höchstens $(7 + c_{\|.\|})c_{sp}k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$ Operationen berechnet werden. (Tatsächlich können wir durch die Verwendung der Identität als orthogonale Gewichte den Aufwand deutlich reduzieren.) Für den Aufwand der Berechnung der neuen projizierten Gewichte gilt nach Lemma 5.3.19 die obere Schranke

$$(c_{sp} + c_{qr}c_{sp} + c_{qr} + 3)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).$$

Weil die neuen Skalierungen $\overline{\omega}_{\mathcal{I}}$, $\overline{\omega}_{\mathcal{J}}$, $\overline{\theta}_{\mathcal{I}}$ und $\overline{\theta}_{\mathcal{J}}$ außerhalb der Bäume $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ jeweils gleich den alten Skalierungen gesetzt werden, sind in $Z_{\mathcal{I}}$ und $Z_{\mathcal{J}}$ nach Lemma 5.3.6 projizierte Gewichte für $\overline{H}$ gespeichert. Für den Gesamtaufwand folgt aus den Vorbetrachtungen für Algorithmus 5.3.8 aufgerufen im Block $\tilde{b}$ die obere Schranke

$$\begin{aligned}
&(12 + 4c_{qr} + 4c_{apr})k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) \\
+\ &(33 + 17c_{qr} + 9c_{qr}c_{sp} + 9c_{\|.\|}c_{sp} + 72c_{sp} + 4c_{apr})k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}) \\
+\ &c_{sp}\max\{k_{\max}^2, n_{\max}^2\}k_{\max}(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).
\end{aligned}$$

■

Bemerkung 5.3.21
Falls die Annahmen aus Bemerkung 3.4.7 sowie $\mathcal{T}_{\tilde{\mathbf{t}}} \in \mathcal{O}(\#\mathbf{t}/k)$ und $\mathcal{T}_{\tilde{\mathbf{s}}} \in \mathcal{O}(\#\mathbf{s}/k)$ gelten, ist der Aufwand für das lokale Niedrigrangupdate mit Algorithmus 5.3.8 in $O(k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}))$. Dieser verhält sich also linear in der Größe der Cluster $\tilde{t}$ und $\tilde{s}$ und quadratisch im lokalen Rang.

Das Theorem 5.3.20 gibt nicht nur eine Aufwandsschranke für die Berechnung des Niedrigrangupdates mit Algorithmus 5.3.8, sondern zeigt auch, dass die Voraussetzungen für das Niedrigrangupdate nach den Berechnungen wieder erfüllt sind. Insbesondere können also verschiedene Niedrigrangupdates in unterschiedlichen Blöcken nacheinander durchgeführt werden. Um diesen Ansatz für die Arithmetik anzuwenden, müssen wir sicherstellen, dass wir mit projizierten Clusterbasen und Gewichten anfangen. Dies erreichen wir, indem wir anfangs die Clusterbasen orthogonalisieren und adaptive Gewichte berechnen (siehe Bemerkung 5.3.2 und Bemerkung 5.3.5).

Algorithmus 5.3.9 Die Funktion bereitet die $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ für die Durchführung von Niedrigrangupdates vor.

---

function VORBEREITUNG_NIEDRIGRANGUPDATE($H, Z_{\mathcal{I}}, \zeta_{\mathcal{I}}, Z_{\mathcal{J}}, \zeta_{\mathcal{J}}, opt, \epsilon_{\mathcal{I}}, \epsilon_{\mathcal{J}}, str$)
  $(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, \kappa, W, \lambda, N, S) \leftarrow H$
  ORTHOGONALE_H2-MATRIX($\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, \kappa, W, \lambda, N, S, \overline{V}, \overline{\kappa}, \overline{W}, \overline{\lambda}, \overline{N}, \overline{S}$)
  ▷ Algorithmus 4.2.3
  $H \leftarrow (\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{\kappa}, \overline{W}, \overline{\lambda}, \overline{N}, \overline{S})$
  SKALIERUNG($r_{\mathcal{I}\times\mathcal{J}}, \overline{S}, \overline{\kappa}, \overline{\lambda}, (I_{\overline{\kappa}_t})_{t\in\mathcal{T}_{\mathcal{I}}}, \overline{\kappa}, (I_{\overline{\lambda}_s})_{s\in\mathcal{T}_{\mathcal{J}}}, \overline{\lambda}, opt, str, \omega_{\mathcal{I}}$)
  SKALIERUNG($r^T_{\mathcal{I}\times\mathcal{J}}, \overline{S}^T, \overline{\kappa}, \overline{\lambda}, (I_{\overline{\lambda}_s})_{s\in\mathcal{T}_{\mathcal{J}}}, \overline{\lambda}, (I_{\overline{\kappa}_t})_{t\in\mathcal{T}_{\mathcal{I}}}, \overline{\kappa}, opt, str, \omega_{\mathcal{J}}$)
  ▷ Algorithmus 4.5.2
  PROJIZIERTE_GEWICHTE($\tilde{t}, E, S, \overline{\kappa}, \overline{\lambda}, \overline{\omega}_{\mathcal{I}}, \overline{\theta}_{\mathcal{I}}, Z_{\mathcal{I}}, \zeta_{\mathcal{I}}$)
  PROJIZIERTE_GEWICHTE($\tilde{s}, F, S^T, \overline{\lambda}, \overline{\kappa}, \overline{\omega}_{\mathcal{J}}, \overline{\theta}_{\mathcal{J}}\, Z_{\mathcal{J}}, \zeta_{\mathcal{J}}$)
  ▷ Algorithmus 5.3.7
end function

Lemma 5.3.22
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit Rangverteilungen $\kappa = (\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $\lambda = (\lambda_s)_{s\in\mathcal{T}_{\mathcal{J}}}$. Dann überschreibt Algorithmus 5.3.9 die $\mathcal{H}^2$-Matrix-Darstellung H mit einer Darstellung von H mit orthogonalen Clusterbasen. Zusätzlich werden adaptive Gewichte für die Strategie *str* aufgestellt. Der Aufwand dafür ist durch

$$c_{qr}k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) \;+\; (10c_{sp} + c_{sp}c_{\|.\|} + c_{sp}c_{qr} + 2c_{qr} + 5)k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}})$$

beschränkt, wobei $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s\}$ ist.

Beweis: Der Algorithmus 4.2.3 berechnet nach Lemma 4.2.20 eine Darstellung der Matrix H mit orthogonalen Clusterbasen in weniger als

$$c_{qr}k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) \;+\; (2c_{sp} + c_{qr} + 2)k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}})$$

Operationen. Der Aufwand zur Berechnung der Skalierungen durch Algorithmus 4.5.2 ist nach Lemma 4.5.7 durch

$$(7 + c_{\|.\|})c_{sp}k_{\max}^3(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}})$$

beschränkt, wobei wir ausnutzen, dass $\#\overline{\kappa}_t \leq \#\kappa_t$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $\#\overline{\lambda}_s \leq \#\lambda_s$ für alle $s \in \mathcal{T}_{\mathcal{J}}$ gilt. Die Berechnung der adaptiven Gewichte mit 5.3.7 erfordert nach Lemma 5.3.19 höchstens

$$(c_{sp} + c_{qr}c_{sp} + c_{qr} + 3)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathfrak{t}}} + \#\mathcal{T}_{\tilde{\mathfrak{s}}})$$

Operationen, wobei wir wieder ausnutzen, dass die neuen Rangverteilungen nicht größer als die alten sind. (Dass in diesem Fall tatsächlich adaptive Gewichte berechnet werden, liegt an den orthogonalen Clusterbasen und der Berechnung der Gewichte im gesamten Baum (siehe Bemerkung 5.3.11). Wir verzichten an dieser Stelle auf einen detaillierten Beweis.) Insgesamt erhalten wir die Aufwandsabschätzung aus der Behauptung. ∎

Wir haben also einen Algorithmus zur Berechnung der einzelnen Niedrigrangupdates und einen Algorithmus zur Vorbereitung der Updates. Durch die Vorberechnungen in Algorithmus 5.3.9 muss die Reihenfolge, in der die Niedrigrangupdates mit Algorithmen 5.3.8 berechnet werden, keine speziellen Voraussetzungen erfüllen. In Kapitel 7 stellen wir zwei Varianten des Niedrigrangupdates vor, die ausnutzen, dass wir bei der Arithmetik nicht beliebig durch den Blockbaum springen, sondern grundsätzlich im Baum auf- und abwärts laufen. Zuvor wollen wir den Fehler des Niedrigrangupdates mit Hilfe der Ergebnisse aus dem Abschnitt 5.2 über die lokale $\mathcal{H}^2$-Matrix-Projektion abschätzen.

5.4 Fehlerkontrolle

In diesem Abschnitt untersuchen wir den lokalen Projektionsfehler während des Niedrigrangupdates. Dabei untersuchen wir die drei Strategien, die wir für die Rekompression in Abschnitt 4.5 vorgestellt haben. Weil wir nur projizierte anstatt adaptiver Gewichte haben, beweisen wir zuerst eine Lemma 4.4.21 entsprechende Aussage für projizierte Gewichte.

Lemma 5.4.1
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix und $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}}$ eine orthogonale Clusterbasis. Weiter seien $(Z_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ die projizierten Gewichte zu H, V, $\omega \in \mathbb{R}_{>0}^{\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+}$ und $\theta \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ mit zugehörigen Matrizen $(P_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $(\Pi_t)_{t\in\mathcal{T}_{\mathcal{I}}}$. Es sei $\|.\| \in \{\|.\|_S, \|.\|_F\}$ und $\ell_t := \operatorname{level}(t)$ für alle $t \in \mathcal{T}_{\mathcal{I}}$. Dann gilt für alle $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+$ und für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H,\overline{V})_{\check{t},s}\| \;\leq\; \omega_{(t,s)} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t\leq\ell_{t'}<\ell_{\check{t}}}} \theta_{t'} \|\left(I_{\tilde{\mathfrak{t}}} - \Pi_{\acute{V}(\overline{V})_{\check{t}}}\right)\widehat{V}(V,\overline{V})_{\check{t}}Z_{\check{t}}\|.$$

Falls $\omega_b = 1$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\theta_t = 1$ für alle $t \in \mathcal{T}_\mathcal{I}$ ist, gilt für alle $t \in \mathcal{T}_\mathcal{I}$

$$\|D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\| \;\leq\; \| \left(I_{\mathbf{t}} - \Pi_{\acute{V}(\overline{V})_t}\right) \widehat{V}(V, \overline{V})_t Z_t\|.$$

Beweis: Für die Hilfsmatrizen aus Definition 4.2.3 verwenden wir für alle $t \in \mathcal{T}_\mathcal{I}$ die Notationen $\breve{V}_t := \breve{V}(\overline{V})_t$, $\acute{V}_t := \acute{V}(\overline{V})_t$ und $\widehat{V}_t := \widehat{V}(V, \overline{V})_t$. Es sei $(t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $\check{t} \in \operatorname{desc}(t)$. Nach (4.17) gilt

$$(I_{\check{\mathbf{t}}} - \Pi_{\overline{V}_{\check{t}}})\Pi_{\breve{V}_{\check{t}}} V_{\check{t}} \;=\; \breve{V}_{\check{t}}(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}})\widehat{V}_{\check{t}}.$$

Dies setzen wir in (4.16) ein und erhalten

$$\begin{aligned} D(H, \overline{V})_{\check{t},s} \;&=\; (I_{\check{\mathbf{t}}} - \Pi_{\overline{V}_{\check{t}}})\Pi_{\breve{V}_{\check{t}}}\omega_{t,s} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\check{t}} \Pi_{\mathbf{s}} \\ &=\; \omega_{t,s} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \breve{V}_{\check{t}}(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}})\widehat{V}_{\check{t}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\check{t}} \Pi_{\mathbf{s}}. \end{aligned}$$

Es seien $P_{\check{t}}$ die orthogonale Matrix und $\Pi_{\check{t}}$ das Produkt von Projektionen zum projizierten Gewicht $Z_{\check{t}}$. Weil $\breve{V}_{\check{t}}$ und $P_{\check{t}}$ orthogonal sowie $\Pi_{\mathbf{s}}$ eine orthogonale Projektion bzw. Π_t ein Produkt von orthogonalen Projektionen bezüglich $\|.\|_2$ sind, folgt mit Lemma 2.3.16 und Lemma 2.3.20

$$\begin{aligned} \|D(H, \overline{V})_{\check{t},s}\| \;&=\; \omega_{t,s} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \|\breve{V}_{\check{t}}(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}})\widehat{V}_{\check{t}} Z_{\check{t}} P_{\check{t}}^T \Pi_t^T \Pi_{\mathbf{s}}\| \\ &\leq\; \omega_{t,s} \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \|(I_{\check{\mathbf{t}}} - \Pi_{\acute{V}_{\check{t}}})\widehat{V}_{\check{t}} Z_{\check{t}}\|. \end{aligned}$$

Somit gilt die erste Abschätzung für die blockweisen Fehlerterme.
Für die Betrachtung der zweiten Abschätzung seien $\omega_b = 1 = \theta_t$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und alle $t \in \mathcal{T}_\mathcal{I}$. Es sei $t \in \mathcal{T}_\mathcal{I}$. Dann gilt mit (4.11) und $X_t := X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_t$

$$D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t \;=\; \breve{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \breve{V}_t^T X_t \;=\; \breve{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \breve{V}_t^T V_t Z_t P_t \Pi_t.$$

Weil Π_t Produkt orthogonaler Projektionen bezüglich $\|.\|_2$ ist, folgt mit Lemma 2.3.20

$$\|D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\| \;= \|\breve{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \breve{V}_t^T V_t Z_t P_t^T \Pi_t^T\| \;\leq\; \|\breve{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \breve{V}_t^T V_t Z_t P_t^T\|.$$

Da sowohl $\breve{V}_t$ nach Lemma 4.2.12 als auch P_t nach Definition 5.3.4 orthogonal sind, folgt mit Lemma 2.3.16

$$\|D(H, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V})_t\| \;\leq\; \|\breve{V}_t \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \breve{V}_t^T V_t Z_t P_t^T\| \;= \| \left(I_{\mathbf{t}} - \Pi_{\acute{V}_t}\right) \breve{V}_t^T V_t Z_t\|.$$

Nach Lemma 4.2.4 gilt $\widehat{V}_t = \breve{V}_t^T V_t$ und somit die clusterweise Abschätzung. ■

Damit können wir uns der Strategie 1 zuwenden. Dabei nutzen wir den relativen Fehler, sodass auf der rechten Seite die Norm von $\|V_t Z_t\|$ auftaucht (vergleiche den Beweis von Lemma 4.5.2). Weil orthogonale Transformationen die Norm invariant lassen, erhalten wir $\|V_t Z_t P_t^T\|$, welches für den Fall von adaptiven Gewichten und Strategie 1 der Norm der totalen Clusterbasis entspricht. Diese können wir wiederum gegen die Norm der gesamten Matrix abschätzen und erhalten die relative Fehlerabschätzung.
Für projizierte Gewichte und die Strategie 1 haben wir $V_t Z_t P_t^T \Pi_t^T = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t$. Damit erhalten wir in der Norm nur

$$\|V_t Z_t P_t^T\| \;\geq\; \|V_t Z_t P_t^T \Pi_t^T\| \;=\; \|X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H)_t\| \;\leq\; \|H\|.$$

Wir können also nicht auf die gleiche Weise wie in Lemma 4.5.2 die relative Fehlerschranke folgern. Da die Projektionen typischerweise aus den vorherigen Niedrigrangupdates stammen, können wir davon ausgehen, dass der Fehler zwischen $V_t Z_t P_t^T$ und $V_t Z_t P_t^T \Pi_t^T$ kontrollierbar ist. Die Bedingung (5.17) in Lemma 5.4.2 beschreibt eine solche Fehlerschranke und ermöglicht uns eine relative Fehlerschranke.

Lemma 5.4.2 (Strategie 1 für das Niedrigrangupdate)
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$. Alle Skalierungen seien gleich 1, $\epsilon_t, \epsilon_s \in \mathbb{R}_{\geq 0}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ und *opt* der relative Fehler bezüglich $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Weiter sei $\overline{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ die mit Algorithmus 5.3.8 im Block $\tilde{b}$ zu den oben definierten Optionen berechnete neue $\mathcal{H}^2$-Matrix. Weiter existiere $\hat{\epsilon} \in [0, 1)$ derart, dass für die projizierten Gewichte für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$

$$\begin{aligned} \|V_t Z_{\mathcal{I},t} P_{\mathcal{I},t}^T - V_t Z_{\mathcal{I},t} P_{\mathcal{I},t}^T \Pi_{\mathcal{I},t}^T\| &\leq \hat{\epsilon} \|V_t Z_{\mathcal{I},t} P_{\mathcal{I},t}^T\| \quad \text{und} \\ \|W_s Z_{\mathcal{J},s} P_{\mathcal{J},s}^T - W_s Z_{\mathcal{J},s} P_{\mathcal{J},s}^T \Pi_{\mathcal{J},s}^T\| &\leq \hat{\epsilon} \|W_s Z_{\mathcal{J},s} P_{\mathcal{I},s}^T\| \end{aligned} \tag{5.17}$$

gilt. Dann ist der Fehler des Niedrigrangupdates für $\|.\| = \|.\|_S$ durch

$$\|(H + R) - \overline{H}\|_S^2 \;\leq\; \frac{2}{(1-\hat{\epsilon})^2} \left(\sum_{t \in \mathcal{T}_{\tilde{\mathbf{t}}}} \epsilon_t^2 + \sum_{s \in \mathcal{T}_{\tilde{\mathbf{s}}}} \epsilon_s^2 \right) \|H + R\|_S^2$$

und für $\|.\| = \|.\|_F$ durch

$$\|(H + R) - \overline{H}\|_F^2 \;\leq\; \frac{1}{(1-\hat{\epsilon})^2} \left(\sum_{t \in \mathcal{T}_{\tilde{\mathbf{t}}}} \epsilon_t^2 + \sum_{s \in \mathcal{T}_{\tilde{\mathbf{s}}}} \epsilon_s^2 \right) \|H + R\|_F^2$$

beschränkt.

Beweis: Nach Theorem 5.2.11 gilt

$$\|(H+R)-\overline{H}\|^2 \leq 2\Bigg(\sum_{t\in\mathcal{T}_{\hat{\mathbf{t}}}} \|D((H+R),\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|^2 + \sum_{s\in\mathcal{T}_{\hat{\mathbf{s}}}} \|D((H+R)^T,\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},\overline{W})_s\|^2\Bigg).$$

(Für die Frobeniusnorm erhalten wir die Abschätzung ohne den Faktor 2.) Weil die Clusterbasen $\overline{V}$ und $\overline{W}$ mit denselben Algorithmen konstruiert werden, schätzen wir zunächst $\|D((H+R),\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\|^2$ für alle $t\in\mathcal{T}_{\hat{\mathbf{t}}}$ ab. Dabei verwenden wir die Notation $Z_t := Z_{I,t}$ für alle $t\in\mathcal{T}_{\mathcal{I}}$. Nach Lemma 5.4.1 gilt für unsere Wahl von ω und θ für alle $t\in\mathcal{T}_{\mathcal{I}}$

$$\|D((H+R),\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\overline{V})_t\| = \|(I_{\mathbf{t}}-\Pi_{\check{V}(\overline{V})_t})\widehat{V}(V,\overline{V})_t Z_t\|.$$

Nach Lemma 4.4.9 (iii) gilt aufgrund der Konstruktion der adaptiven Clusterbasen

$$\|(I_{\mathbf{t}}-\Pi_{\check{V}(\overline{V})_t})\widehat{V}(V,\overline{V})_t Z_t\| \leq \epsilon_t\|\widehat{V}(V,\overline{V})_t Z_t\|.$$

Es gilt nach Lemma 4.2.4 $\widehat{V}(V,\overline{V}) = \check{V}(\overline{V})_t^T V_t$ und nach Lemma 4.2.12 ist $\check{V}(\overline{V})_t$ orthogonal. Weil die Skalierungen alle gleich 1 gewählt sind, existiert eine orthogonale Matrix P_t und ein Produkt von Projektion Π_t mit $V_t Z_t P_t^T \Pi_t^T = X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},(H+R))_t$. Nach Lemma 2.3.16 können wir orthogonale Matrizen hinzufügen bzw. entfernen, ohne die Norm zu verändern. Damit folgt

$$\begin{aligned}\|\widehat{V}(V,\overline{V})_t Z_t\| &= \|\check{V}(\overline{V})_t^T V_t Z_t\| = \|\check{V}(\overline{V})_t\check{V}(\overline{V})_t^T V_t Z_t P_t^T\| \\ &= \|\Pi_{\check{V}(\overline{V})_t} V_t Z_t P_t^T\| \leq \|V_t Z_t P_t^T\|,\end{aligned}$$

wobei wir ausnutzen, dass $\Pi_{\check{V}(\overline{V})_t}$ eine orthogonale Projektion bezüglich $\|.\|_2$ ist, die nach Lemma 2.3.20 die betrachteten Matrixnormen nicht vergrößert. Aus der Voraussetzung (5.17) folgt

$$\begin{aligned}\|V_t Z_t P_t^T\| &= \|V_t Z_t P_t^T - V_t Z_t P_t^T\Pi_t^T + V_t Z_t P_t^T\Pi_t^T\| \\ &\leq \|V_t Z_t P_t^T - V_t Z_t P_t^T\Pi_t^T\| + \|V_t Z_t P_t^T\Pi_t^T\| \\ &\leq \hat{\epsilon}\|V_t Z_t P_t^T\| + \|V_t Z_t P_t^T\Pi_t^T\|.\end{aligned}$$

Zusammen mit $\hat{\epsilon}\in[0,1)$ und $\|X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},(H+R))_t\| \leq \|(H+R)\|$ (siehe den Beweis von Lemma 4.5.2) folgt

$$\begin{aligned}\|\widehat{V}(V,\overline{V})_t Z_t\| &\leq \frac{1}{(1-\hat{\epsilon})}\|V_t Z_t P_t^T\Pi_t^T\| = \frac{1}{(1-\hat{\epsilon})}\|X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},(H+R))_t\| \\ &\leq \frac{1}{(1-\hat{\epsilon})}\|(H+R)\|.\end{aligned}$$

Damit erhalten wir für die Fehlerterme die Abschätzung

$$\begin{aligned}\|D(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},(H+R),\overline{V})_t\| &= \|(I_{\mathbf{t}}-\Pi_{\hat{V}(\overline{V})_t})\widehat{V}(V,\overline{V})_t Z_t\| \leq \epsilon_t\|\widehat{V}(V,\overline{V})_t Z_t\| \\ &\leq \frac{\epsilon_t}{(1-\hat{\epsilon})}\|H+R\|.\end{aligned}$$

Analog folgt $\|D(\mathcal{T}^T_{\mathcal{I}\times\mathcal{J}},(H+R),\overline{W})_s\| \leq (\epsilon_s/(1-\hat{\epsilon}))\|H+R\|$ für alle $s\in\mathcal{T}_{\mathcal{J}}$. Durch Einsetzen dieser Abschätzungen in die Fehlerschranke aus Theorem 5.2.11 folgt die Aussage. ∎

Bemerkung 5.4.3
Falls wir adaptive Gewichte für $H+R$ berechnen, gilt (5.17) mit $\hat{\epsilon}=0$ und wir erhalten Fehlerabschätzungen ohne den Faktor $1/(1+\hat{\epsilon})^2$.

Bemerkung 5.4.4
Für den Fall, dass *opt* den absolute Fehler beschreibt, folgt eine absolute Fehlerschranke. Dabei wird die Bedingung (5.17) nicht benötigt und dementsprechend fällt auf der rechten Seite der Faktor $1/(1-\hat{\epsilon})$ weg.

Für den relativen Fehler erhalten wir also eine Abschätzung, die von den bisherigen Fehlern abhängt. Weil für den Block des Updates die Gewichte neu berechnet werden, sammeln sich für einen Block nur die Projektionen seit dem letzten Update an, bei dem zum Block etwas hinzu addiert wurde.
Für den absoluten Fehler fällt dieses Problem weg, wie in Bemerkung 5.4.4 erwähnt. Bei den Strategien für den blockweisen Fehler werden absolute Fehler bei der Konstruktion der Clusterbasis verwendet, weil die relative Fehlerschranke durch die Skalierungen erreicht wird. Dadurch fällt auch bei diesen Strategien das Problem weg, das in Lemma 5.4.2 durch die Verwendung der projizierten Gewichte entsteht.
Allerdings entstehen hier Probleme durch die eingeschränkte Wahl der Skalierungen. Für die Wahl der θ ist dies typischerweise kein Problem, weil wir diese nur abhängig von der Baumstruktur wählen, welche sich während der Updates nicht ändert. Weil die Skalierungen ω_b von der Norm von $H_{|\mathbf{b}}$ abhängen und sich diese während der Updates ändern, die Skalierungen aber nur für Blöcke $b=(t,s)\in\mathcal{T}_{\tilde{\mathbf{b}}}$ angepasst werden können, wird teilweise mit veralteten Normen gerechnet. Weil die Matrizen typischerweise durch Projektionen verändert wurden, deren Fehler kontrollierbar sind, gehen wir davon aus, dass die alten Gewichte fast den richtigen Gewichten entsprechen.

Lemma 5.4.5 (Strategie 2 für das Niedrigrangupdate)
Es seien $H=\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}},V,W,N,S)$ eine $\mathcal{H}^2$-Matrix, $R=AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b}=(\tilde{t},\tilde{s})\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $R\in\mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$. Es sei $\epsilon_t:=\epsilon_s:=\epsilon\in\mathbb{R}_{\geq 0}$ für alle $t\in\mathcal{T}_{\mathcal{I}}$ und $s\in\mathcal{T}_{\mathcal{J}}$ und *opt* der totale Fehler bezüglich $\|.\|\in\{\|.\|_S,\|.\|_F\}$. Für die Skalierungen in den Blöcken existiere $\hat{\epsilon}\in[0,1)$ derart, dass für $\omega_{\mathcal{I},b}=\omega_{\mathcal{J},b}=:\omega_b$

$$\begin{cases}\omega_b=1 & \text{, falls } \|H_{|\mathbf{b}}\|=0\\ |\omega_b-\frac{\|H_{|\mathbf{b}}\|}{\sqrt{(\#\operatorname{desc}(t)+\#\operatorname{desc}(s))}}|\leq\hat{\epsilon}\omega_b & \text{, falls } \|H_{|\mathbf{b}}\|\neq 0\end{cases} \tag{5.18}$$

für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gilt. Für die Skalierungen in den Clustern gelte $\theta_{\mathcal{I},t} := \theta_{\mathcal{J},s} := 1$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Weiter sei $\overline{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ die mit Algorithmus 4.5.1 neu berechnete $\mathcal{H}^2$-Matrix, wobei die Skalierungen durch Strategie 2 aus Abschnitt 4.5 gewählt seien. Dann ist der Fehler der Rekompression in jedem zulässigen Blatt $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ durch

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\| \quad \leq \quad \begin{cases} \epsilon\,\|(H+R)_{|\mathbf{b}}\| & \text{, falls } b \in \mathcal{T}_{\tilde{\mathbf{b}}} \\ \frac{\epsilon}{1-\tilde{\epsilon}}\|(H+R)_{|\mathbf{b}}\| & \text{, falls } (t \in (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))) \wedge (s \in \mathcal{T}_\mathcal{J} \backslash \mathcal{T}_{\tilde{\mathbf{s}}}) \\ \frac{\epsilon}{1-\tilde{\epsilon}}\|(H+R)_{|\mathbf{b}}\| & \text{, falls } (t \in \mathcal{T}_\mathcal{I} \backslash \mathcal{T}_{\tilde{\mathbf{t}}}) \wedge (s \in (\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s}))) \\ 0 & \text{sonst} \end{cases}$$

beschränkt.

Beweis: Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Der Fall $\|(H+R)_{|\mathbf{b}}\| = 0$ folgt wie in Lemma 4.5.4. Es sei also o. E. d. A. $\|(H+R)_{|\mathbf{b}}\| \neq 0$
1. Fall: Es sei $b \in \mathcal{L}_{\tilde{\mathbf{b}}}$, also $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Nach Lemma 5.2.12 gilt dann

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 \;=\; \|(I_\mathbf{t} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 \;+\; \|(I_\mathbf{s} - \Pi_{\overline{W}_s})((H+R)_{|\mathbf{b}})^T\|^2.$$

Wir betrachten den Fehler bezüglich der Zeilenbasis für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Für diesen gilt nach Lemma 4.3.11

$$\|(I_\mathbf{t} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 \;\leq \sum_{\check{t}\in\operatorname{desc}(t)} \|D((H+R), \overline{V})_{\check{t},s}\|^2.$$

Weil wir θ_I überall gleich 1 wählen, ist das adaptive Gewicht nach Bemerkung 4.4.18 ein skaliertes Gewicht. Nach Korollar 4.4.14 gilt für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D((H+R), \overline{V})_{\check{t},s}\| \;\leq\; \tilde{\omega}_{\mathcal{I},b} \| \left(I_{\check{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V, \overline{V})_{\check{t}} Z_{\tilde{\omega}_\mathcal{I},\check{t}}\|.$$

Weil wir als Fehlerart *opt* den absoluten Fehler gewählt haben, folgt aus Lemma 4.4.9 (iii) für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D((H+R), \overline{V})_{\check{t},s}\| \;\leq\; \tilde{\omega}_{\mathcal{I},b} \| \left(I_{\check{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\check{t}}}\right) \widehat{V}(V, \overline{V})_{\check{t}} Z_{\tilde{\omega}_\mathcal{I},\check{t}}\| \;\leq\; \tilde{\omega}_{\mathcal{I},b}\epsilon_t \;=\; \tilde{\omega}_{\mathcal{I},b}\epsilon.$$

Analog folgt für den Fehler bezüglich der Spaltenbasis in $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ für alle $\check{s} \in \operatorname{desc}(s)$

$$\|D((H+R)^T, \overline{W})_{\check{s},t}\| \;\leq\; \tilde{\omega}_{\mathcal{J},b}\epsilon.$$

Indem wir die Fehlerschranken einsetzen, erhalten wir für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ den blockweisen Fehler

$$\begin{aligned} \|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 \;&\leq\; \|(I_\mathbf{t} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 + \|(I_\mathbf{s} - \Pi_{\overline{W}_s})((H+R)_{|\mathbf{b}})^T\|^2 \\ &\leq \sum_{\check{t}\in\operatorname{desc}(t)} \tilde{\omega}^2_{\mathcal{I},b}\epsilon^2 + \sum_{\check{s}\in\operatorname{desc}(s)} \tilde{\omega}^2_{\mathcal{J},b}\epsilon^2. \end{aligned}$$

Wegen $b \in \mathcal{L}^+_{\tilde{\mathbf{b}}}$ sind die Skalierungen für $H+R$ im Block b neu berechnet und es gilt $\tilde{\omega}^2_{\mathcal{I},b} = \tilde{\omega}^2_{\mathcal{J},b} = \|(H+R)_{\mathbf{b}}\|^2/(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))$. Damit folgt

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 \;\leq\; \|(H+R)_{\mathbf{b}}\|^2\epsilon^2.$$

2. Fall: Es sei $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}^+_{\tilde{\mathbf{b}}}$.
Dann werden die Skalierungen für die Blöcke übernommen und es gilt nach Voraussetzung (5.18) wegen $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ für die Skalierung des Blocks $\tilde{\omega}_{\mathcal{I},b} = \tilde{\omega}_{\mathcal{J},b} =: \tilde{\omega}_b$

$$\begin{aligned}\tilde{\omega}_b \;=\; \omega_b \;&\leq\; \left|\omega_b - \frac{\|H_{|\mathbf{b}}\|}{\sqrt{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}}\right| + \left|\frac{\|H_{|\mathbf{b}}\|}{\sqrt{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}}\right| \\ &\leq\; \hat{\epsilon}\omega_b + \frac{\|H_{|\mathbf{b}}\|}{\sqrt{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}} \;=\; \hat{\epsilon}\tilde{\omega}_b + \frac{\|(H+R)_{|\mathbf{b}}\|}{\sqrt{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}}.\end{aligned}$$

Durch Umstellen erhalten wir

$$\tilde{\omega}_b \;\leq\; \frac{\|(H+R)_{|\mathbf{b}}\|}{(1-\hat{\epsilon})\sqrt{(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))}}.$$

2. 1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ oder $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$.
Für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ gilt

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 \;=\; \|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2$$

und für $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ gilt

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 \;=\; \|(I_{\mathbf{s}} - \Pi_{\overline{W}_s})(H+R)^T_{|\mathbf{b}}\|^2.$$

Beide Terme tauchen auch im 1. Fall auf und können analog abgeschätzt werden. Für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ erhalten wir

$$\begin{aligned}\|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 \;&\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|D((H+R),\overline{V})_{\check{t},s}\|^2 \;\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \tilde{\omega}^2_b\epsilon^2 \\ &\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \frac{\epsilon^2\|(H+R)_{|\mathbf{b}}\|^2}{(1-\hat{\epsilon})(\#\operatorname{desc}(t) + \#\operatorname{desc}(s))} \\ &\leq\; \frac{\epsilon^2}{(1-\hat{\epsilon})}\|(H+R)_{|\mathbf{b}}\|.\end{aligned}$$

Für $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ folgt analog

$$\|(I_{\mathbf{s}} - \Pi_{\overline{W}_s})((H+R)_{|\mathbf{b}})^T\|^2 \;\leq\; \frac{\epsilon^2}{(1-\hat{\epsilon})}\|(H+R)_{|\mathbf{b}}\|.$$

2. 2. Fall: Es sei $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ oder $s \in \operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}$. Weil beide Fälle analog verlaufen,

sei o. E. d. A. $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$. Insbesondere ist dann $s \in \mathcal{T}_{\mathcal{J}}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{t}))$.
In diesem Fall gilt

$$\begin{aligned}
\|((H+R)_{\mathbf{t}\times\mathbf{s}} - \overline{H}_{\mathbf{t}\times\mathbf{s}}\|^2 &= \|((H+R)_{\tilde{\mathbf{t}}\times\mathbf{s}} - \Pi_{\overline{V}_{\check{t}}}((H+R)_{\tilde{\mathbf{t}}\times\mathbf{s}}\|^2 \\
&= \sum_{\check{t}\in\operatorname{desc}(\tilde{t})} \|D(H+R,\overline{V})_{\check{t},s}\|^2 \;\leq\; \sum_{\check{t}\in\operatorname{desc}(\tilde{t})} \tilde{\omega}^2_{(t,s)}\;\epsilon^2 \\
&\leq \sum_{\check{t}\in\operatorname{desc}(\tilde{t})} \frac{\epsilon^2\|(H+R)_{|\mathbf{b}}\|^2}{(1-\hat{\epsilon})(\#\operatorname{desc}(t)+\#\operatorname{desc}(s))} \\
&= \frac{\epsilon^2}{(1-\hat{\epsilon})^2}\|(H+R)_{|\mathbf{b}}\|^2.
\end{aligned}$$

Damit gilt die Behauptung. ∎

Bemerkung 5.4.6
Wie wir im Beweis von Lemma 5.4.5 gesehen haben, werden die Fehler teilweise durch Hinzufügen von Fehlertermen zusätzlich überschätzt. So können wir die Gewichte für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ auch als $\|H_{|\mathbf{t}\times\mathbf{s}}\|^2/(\#\operatorname{desc}(t))$ setzen, um die gleiche obere Schranke zu erhalten. Analoges gilt für $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$.
Für die Fehler von Blöcken (t,s) mit $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ würde es ausreichen, bei den vorberechneten Gewichten die Skalierung $\omega_{(t,s)} := \|H_{|\mathbf{t}\times\mathbf{s}}\|$ zu verwenden und bei der Berechnung des Gewichts das Gewicht des Elter $\acute{t} := \operatorname{par}(\tilde{t})$ mit dem Faktor $1/\sqrt{\#\operatorname{desc}(\tilde{t})}$ zu skalieren. Genauso können wir im Fall $s \in \operatorname{pred}(s)\backslash\{\tilde{s}\}$ vorgehen.

Die Bedingung (5.18) ist Resultat der Projektion für die Vorfahren $t \in \operatorname{pred}(\tilde{t})$ und $s \in \operatorname{pred}(\tilde{s})$ zusammen mit der Bedingung an die Skalierungen, die für diese Cluster bzw. Blöcke mit diesen Clustern nicht angepasst werden können. Allerdings können wir den relativen Fehler in jedem Block mit Hilfe von Strategie 2 (und auch mit Strategie 3) kontrollieren. Damit lässt sich die Voraussetzung (5.18) erfüllen. Für Strategie 3 erhalten wir mit einer ähnlichen Voraussetzung entsprechende Fehlerabschätzungen.

Lemma 5.4.7 (Strategie 3 für das Niedrigrangupdate)
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\#\operatorname{chil}(t), \#\operatorname{chil}(s) \leq c_{\text{chil}}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$. Weiter seien $\epsilon \in \mathbb{R}_{\geq 0}$, $p \in (0, \frac{1}{\sqrt{c_{\text{chil}}}})$, $\epsilon_t^2 := \epsilon_s^2 := \tilde{\epsilon} := (1-p^2 c_{\text{chil}})\epsilon^2$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ und *opt* der totale Fehler bezüglich $\|.\| \in \{\|.\|_S, \|.\|_F\}$. Für die Skalierungen in den Blöcken existiere $\hat{\epsilon} \in [0,1)$ derart, dass für $\omega_{\mathcal{I},b} = \omega_{\mathcal{J},b} =: \omega_b$

$$\begin{cases} \omega_b = 1 & \text{, falls } \|H_{|\mathbf{b}}\| = 0 \\ |\omega_b - \|H_{|\mathbf{b}}\|| \leq \hat{\epsilon}\omega_b & \text{, falls } \|H_{|\mathbf{b}}\| \neq 0 \end{cases} \tag{5.19}$$

für alle $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gilt. Für die Skalierungen in den Clustern gelte $\theta_{\mathcal{I},t} := \theta_{\mathcal{J},s} := p$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Weiter sei $\overline{H} = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{V}, \overline{W}, \overline{N}, \overline{S})$ die mit Algorithmus

4.5.1 berechnete $\mathcal{H}^2$-Matrix, wobei die Skalierungen durch Strategie 3 mit dem oben gewählten p aus Abschnitt 4.5 gewählt seien. Dann ist der Fehler der Rekompression in jedem zulässigen Blatt $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ durch

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\| \;\leq\; \begin{cases} \epsilon\,\|(H+R)_{|\mathbf{b}}\| & \text{, falls } b \in \mathcal{T}_{\tilde{\mathbf{b}}} \\ \frac{\epsilon}{2(1-\hat{\epsilon})}\|(H+R)_{|\mathbf{b}}\| & \text{, falls } t \in (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t})) \wedge s \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{s}}} \\ \frac{\epsilon}{2(1-\hat{\epsilon})}\|(H+R)_{|\mathbf{b}}\| & \text{, falls } t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{T}_{\tilde{\mathbf{t}}} \wedge s \in (\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s})) \\ 0 & \text{sonst} \end{cases}$$

beschränkt.

Beweis: Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Der Fall $\|(H+R)_{|\mathbf{b}}\| = 0$ folgt wie in Lemma 4.5.5. Es sei also o. E. d. A. $\|(H+R)_{|\mathbf{b}}\| \neq 0$
1. Fall: Es sei $b \in \mathcal{T}_{\tilde{\mathbf{b}}}$, also $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$. Nach Lemma 5.2.12 gilt dann

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 \;\leq\; \|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 + \|(I_{\mathbf{s}} - \Pi_{\overline{W}_s})(H+R)^T_{|\mathbf{b}}\|^2.$$

Wir betrachten den Fehler bezüglich der Zeilenbasis für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Für diesen gilt nach Lemma 4.3.11

$$\|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 \;\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H+R,\overline{V})_{\check{t},s}\|^2.$$

Mit $\ell_{t'} := \operatorname{level}(t')$ für alle $t' \in \mathcal{T}_{\mathcal{I}}$ gilt nach Lemma 4.4.21 für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H+R,\overline{V})_{\check{t},s}\| \;\leq\; \tilde{\omega}_b \prod_{\substack{t'\in\operatorname{desc}(t)\\ \ell_t \leq \ell_{t'} < \ell_{\check{t}}}} \theta_{t'} \|(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}_{\hat{t}}})\widehat{V}_{\hat{t}} Z_{\tilde{\omega},\check{t}}\| \;\leq\; \tilde{\omega}_b p^{\ell_{\check{t}}-\ell_t} \|(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}_{\hat{t}}})\widehat{V}_{\hat{t}} Z_{\tilde{\omega},\check{t}}\|.$$

Weil wir als Fehlerart *opt* den absoluten Fehler gewählt haben, folgt aus Lemma 4.4.9 (iii) für alle $\check{t} \in \operatorname{desc}(t)$

$$\|(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\hat{t}}})\widehat{V}_{\hat{t}} Z_{\tilde{\omega},\check{t}}\| \;\leq\; \epsilon_t \;=\; \tilde{\epsilon}.$$

Weil $b \in \mathcal{T}_{\tilde{\mathbf{b}}}$ gilt, wird die Skalierung für den Block neu berechnet. Also gilt $\tilde{\omega}_b^2 = \|H_{|\mathbf{b}}\|^2$ und $\tilde{\theta}_{t'} = p$ für alle $t' \in \operatorname{desc}(t)$ und mit den vorherigen Betrachtungen folgt für alle $\check{t} \in \operatorname{desc}(t)$

$$\|D(H+R,\overline{V})_{\check{t},s}\|^2 \;\leq\; \tilde{\omega}_b^2 p^{\ell_{\check{t}}-\ell_t} \|(I_{\hat{\mathbf{t}}} - \Pi_{\hat{V}(\overline{V})_{\hat{t}}})\widehat{V}_{\hat{t}} Z_{\tilde{\omega},\check{t}}\|^2 \;\leq\; \|(H+R)_{|\mathbf{b}}\|^2 p^{\ell_{\check{t}}-\ell_t} \tilde{\epsilon}^2.$$

Durch Einsetzen in die Abschätzung für den blockweisen Fehler erhalten wir

$$\begin{aligned} \|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 \;&\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H+R,\overline{V})_{\check{t},s}\|^2 \\ &\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} \|(H+R)_{|\mathbf{b}}\|^2 p^{2(\ell_{\check{t}}-\ell_t)} \tilde{\epsilon}^2. \end{aligned}$$

Die Summe schätzen wir wie in Lemma 4.5.5 mit der geometrische Reihe ab und erhalten

$$\begin{aligned}\|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 &\leq \|(H+R)_{|\mathbf{b}}\|^2 \sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}}-\ell_t)}\tilde{\epsilon}^2 \\ &\leq \|(H+R)_{|\mathbf{b}}\|^2 \frac{1}{(1-p^2c_{\text{chil}})}\frac{(1-p^2c_{\text{chil}})}{2}\epsilon^2 \\ &= \frac{\epsilon^2}{2}\|(H+R)_{|\mathbf{b}}\|^2.\end{aligned}$$

Analog erhalten wir für den Fehler bezüglich der neuen Spaltenbasis

$$\|(I_{\mathbf{s}} - \Pi_{\overline{W}_s})(H+R)^T_{|\mathbf{b}}\|^2 \leq \frac{\epsilon^2}{2}\|(H+R)_{|\mathbf{b}}\|^2.$$

Die Abschätzung für die Zeilenbasis und die Spaltenbasis setzen wir in die Fehlerschranke (4.19) ein und erhalten die Aussage.
2. Fall: Es sei $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}^+_{\tilde{\mathbf{b}}}$.
Dann gilt nach Voraussetzung (5.19) für die Skalierung im Block

$$\tilde{\omega}_b = \omega_b \leq |\omega_b - \|H_{|\mathbf{b}}\|| + \|H_{|\mathbf{b}}\| \leq \hat{\epsilon}\omega_b + \|H_{|\mathbf{b}}\| = \hat{\epsilon}\tilde{\omega}_b + \|(H+R)_{|\mathbf{b}}\|.$$

Durch Umformen und Einsetzen in die vorherige Ungleichung erhalten wir

$$\tilde{\omega}_b \leq \frac{1}{(1-\hat{\epsilon})}\|(H+R)_{|\mathbf{b}}\|. \tag{5.20}$$

2. 1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ oder $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$.
Wegen $b = (t,s) \notin \mathcal{L}_{\tilde{\mathbf{b}}}$ ist entweder $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s})$ oder $t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{s})$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Weil beide Fälle analog bewiesen werden, sei o. E. d. A. $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt

$$\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 = \|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2.$$

Wie im 1. Fall folgt

$$\|(I_{\mathbf{t}} - \Pi_{\overline{V}_t})(H+R)_{|\mathbf{b}}\|^2 \leq \sum_{\check{t}\in\operatorname{desc}(t)} \|D(H+R,\mathcal{T}_{\mathcal{I}\times\mathcal{J}})_{\check{t},s}\|^2 \leq \tilde{\omega}_b^2 \sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}}-\ell_t)}\tilde{\epsilon}^2.$$

Mit (5.20), der Definition von $\tilde{\epsilon}$ und der Anwendung der geometrischen Reihe wie im 1. Fall folgt

$$\begin{aligned}\|(H+R)_{|\mathbf{b}} - \overline{H}_{|\mathbf{b}}\|^2 &\leq \tilde{\omega}_b^2 \sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}}-\ell_t)}\tilde{\epsilon}^2 \\ &\leq \frac{\|(H+R)_{|\mathbf{b}}\|^2}{(1-\hat{\epsilon})^2}\frac{1}{(1-p^2c_{\text{chil}})}\frac{(1-p^2c_{\text{chil}})}{2}\epsilon^2 \\ &\leq \frac{\epsilon^2}{2(1-\hat{\epsilon})^2}\|(H+R)_{|\mathbf{b}}\|^2.\end{aligned}$$

2. 2. Fall: Es sei $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ oder $s \in \operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}$.
Dann ist entweder $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s})$ oder $t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{s})$ und $s \in \operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\}$. Weil beide Fälle analog verlaufen, sei o. E. d. A. $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$. Insbesondere ist dann $s \in \mathcal{T}_{\mathcal{J}}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{t}))$. In diesem Fall gilt

$$\begin{aligned}\|(H+R)_{\mathbf{b}} - \overline{H}_{\mathbf{b}}\|^2 &= \|(I_{\tilde{\mathbf{t}}} - \Pi_{\overline{V}_{\tilde{t}}})(H+R)_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}\|^2 \\ &= \sum_{\check{t}\in\operatorname{desc}(\tilde{t})} \|D(H+R,\overline{V})_{\check{t},s}\|^2 \\ &\leq \sum_{\check{t}\in\operatorname{desc}(\tilde{t})} p^{2(\ell_{\check{t}}-\ell_t)}\tilde{\epsilon}^2\tilde{\omega}_b \;\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}}-\ell_t)}\tilde{\epsilon}^2\tilde{\omega}_b.\end{aligned}$$

Wie im 2. 2. Fall folgt mit (5.20), der Definition von $\tilde{\epsilon}$ und der geometrischen Reihe

$$\|(H+R)_{\mathbf{b}} - \overline{H}_{\mathbf{b}}\|^2 \;\leq\; \sum_{\check{t}\in\operatorname{desc}(t)} p^{2(\ell_{\check{t}}-\ell_t)}\tilde{\epsilon}^2\tilde{\omega}_b \;\leq\; \frac{\epsilon^2}{2(1-\hat{\epsilon})^2}\|(H+R)_{|\mathbf{b}}\|^2.$$

Damit folgt die Behauptung. ∎

Bemerkung 5.4.8
Bei der Fehlerabschätzung für die Vorfahren fällt auf, dass wir einen zusätzlichen Faktor $\frac{1}{2(1-\hat{\epsilon})}$ erhalten, wobei $\hat{\epsilon}$ die bisherigen block-relativen Projektionsfehler beschreibt. Falls $\hat{\epsilon} \leq \frac{1}{2}$ ist, erhalten wir also in allen Fällen den gewünschten relativen Fehler.
Außerdem wird für $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ die Summe der blockweisen Fehlerterme nur über die Cluster in $\mathcal{T}_{\tilde{\mathbf{t}}}$ genommen. Deshalb können die Skalierungen θ_t für $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ auch gleich 1 gesetzt werden und die Fehlerabschätzung würde trotzdem genauso gelten.

Bemerkung 5.4.9
Auch für die in Bemerkung 4.5.6 modifizierte Strategie 3 gelten die Fehlerabschätzungen aus Lemma 5.4.7.

Es gelten also für alle drei Strategien aus der Rekompression Fehlerabschätzungen, die sehr ähnlich zu denen aus Abschnitt 4.5 sind, wobei allerdings die Fehler nicht in der gesamten Matrix auftauchen und die bisherigen Fehler einen Einfluss haben.

Bemerkung 5.4.10 (Implementierung)
Bei der Implementierung in [39] speichern wir die Skalierungen ω_b nicht ab, sondern berechnen sie immer dann für einen Block, wenn dieser zu einem Gewicht hinzugefügt wird. Dies führt zu einem etwas größeren Aufwand, weil wir die Skalierungen während des Updates für die gesamte Blockzeile und Blockspalte berechnen. Allerdings wird die Komplexität für das gesamte Niedrigrangupdate dadurch nicht erhöht.
Bei den Berechnungen verwenden wir nur innerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$ orthogonale Gewichte. Dies führt zu einem Fehler bei der Berechnung der Skalierungen außerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$, weil wir nur mit projizierten Clusterbasen arbeiten. Dieser Fehler entspricht dem der Verwendung der alten Skalierungen, weil bei unserer Vorgehensweise die zwischenzeitlichen Projektionen auch ignoriert werden.

6 Arithmetik

In diesem Kapitel stellen wir die Arithmetik unter der Verwendung des Niedrigrangupdates aus Kapitel 5 vor. Wir gehen dabei davon aus, dass keine passenden Clusterbasen für die Darstellung der Ergebnisse bekannt sind, sodass der in [8] beschriebene Ansatz für eine Arithmetik in linearer Komplexität nicht anwendbar ist. Der in [12, Chapter 8] beschriebene Algorithmus zur Matrix-Matrix-Multiplikation ist derart gestaltet, dass bei einem rekursiven Aufruf für jedes Kind eine $\mathcal{H}^2$-Matrix aufgestellt und dann die Darstellungen im aktuellen Block vereinheitlicht werden. Dies führt bei der Inversion und Dreieckszerlegungen zu zusätzlichem Aufwand zur Vereinheitlichung der Darstellung der $\mathcal{H}^2$-Matrix-Darstellung.
Der hier beschriebene rekursive Ansatz ähnelt dem für $\mathcal{H}$-Matrizen aus [26]. Für die Matrix-Matrix-Multiplikation $C + AB$ berechnen wir, sobald einer der zugehörigen Blöcke ein Blatt ist, das Produkt AB als Niedrigrangmatrix und addieren das Ergebnis zur Zielmatrix C per Niedrigrangupdate aus Kapitel 5.
Wir beginnen in Abschnitt 6.1 mit einfachen arithmetischen Operationen wie der Multiplikation eines Vektors mit einer $\mathcal{H}^2$-Matrix. In Abschnitt 6.2 erläutern wir das Matrix-Produkt von $\mathcal{H}^2$-Matrizen in der Form $C + AB$. Dieses verwenden wir danach, um Algorithmen für die Inversion und die Cholesky-Zerlegung von $\mathcal{H}^2$-Matrizen zu formulieren.
Bei den Formulierungen der Algorithmen verzichten wir auf Funktionen für transponierte Matrizen. Dies ist durch die Tatsache motiviert, dass sich für die verwendeten Matrix-Formate (wie z. B. $\mathcal{H}^2$-Matrizen) die transponierte Matrix einfach in dem gleichen Format darstellen lässt (siehe z. B. Lemma 3.4.26).
Das Ziel des von uns vorgestellten Pseudocodes ist nicht, die optimale Umsetzung zu erläutern (soweit so etwas überhaupt existiert). Stattdessen konzentrieren wir uns darauf, den Ansatz möglichst einfach zu halten, ohne den asymptotischen Aufwand zu vergrößern. Die beschriebenen Algorithmen können somit teilweise von den tatsächlich implementierten Programmen leicht abweichen.

6.1 Einfache arithmetische Operationen für $\mathcal{H}^2$-Matrizen

In diesem Abschnitt stellen wir die Multiplikation eines Vektors mit einer $\mathcal{H}^2$-Matrix und weitere Hilfsfunktionen vor. Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $x \in \mathbb{R}^{\mathcal{J}}$ und $y \in \mathbb{R}^{\mathcal{I}}$.
Bei den Funktionen, die eine Multiplikation berechnen, stellen wir diese als Addition

eines Produkts zu einer bestehenden Größe dar. Wir wählen diesen Ansatz aufgrund der vorliegenden hierarchischen Blockstruktur. In einem unterteilten Block führt diese für die Matrix-Vektor-Multiplikation zu den Teilproblemen

$$\begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} H_{11}x_1 + H_{12}x_2 \\ H_{21}x_1 + H_{22}x_2 \end{pmatrix}. \tag{6.1}$$

Im ersten Schritt könnten wir $H_{11}x_1$ und $H_{21}x_1$ durch die Matrix-Vektor-Multiplikation ohne Hinzuaddieren berechnen. Anschließend müssen wir dann aber $H_{12}x_2$ bzw. $H_{22}x_2$ zum Ergebnis der ersten beiden Berechnungen hinzuaddieren. Deshalb bietet es sich an, gleich die Multiplikation in der Form $y + Hx$ zu berechnen. Für den Fall, dass wir Hx benötigen, rufen wir die Funktion mit $y = 0$ auf.
Für die Matrix-Vektor-Multiplikation der Gestalt $y+Hx$ können wir wie in (6.1) vorgehen, bis wir für H ein Blatt erreichen. Für $b = (t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ können wir $y_{|\mathbf{t}} + N_b x_{|\mathbf{s}}$ effizient berechnen, weil in diesem Fall $t \in \mathcal{L}_\mathcal{I}$ und $s \in \mathcal{L}_\mathcal{J}$ gilt.
Für zulässige Blöcke $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ müssen wir $V_t S_b W_s^T x_{|\mathbf{s}}$ zu $y_{|\mathbf{t}}$ addieren. Dies ist komplizierter, weil die Clusterbasen V_t und W_s außer für Blätter $t \in \mathcal{L}_\mathcal{I}$ bzw. $s \in \mathcal{L}_\mathcal{J}$ nur indirekt über die geschachtelte Darstellung mit den Transfermatrizen vorliegen.
Deshalb untersuchen wir zuerst, wie sich das Produkt $W_s^T x_{|\mathbf{s}}$ berechnen lässt. Für ein Blatt $s \in \mathcal{L}_\mathcal{J}$ liegt W_s direkt vor und wir können das Produkt berechnen. Für Nicht-Blatt-Cluster $s \in \mathcal{T}_\mathcal{J}\backslash\mathcal{L}_\mathcal{J}$ gilt

$$W_s^T x_{|\mathbf{s}} = \sum_{\check{s}\in\mathrm{chil}(s)} F_{\check{s}}^T W_{\check{s}}^T x_{|\check{\mathbf{s}}}.$$

Falls die Produkte für die Kinder berechnet sind, können wir also $x_s := W_s^T x_{|\mathbf{s}}$ berechnen. Mit diesem Ansatz erhalten wir den Bottom-up-Algorithmus 6.1.1, dessen Aufwand wir wir in Lemma 6.1.1 abschätzen.

Algorithmus 6.1.1 Die Funktion berechnet das Produkt $x_s = W_s^T x_{|\mathbf{s}}$ für die Clusterbasis (W, F) zum Clusterbaum $\mathcal{T}_\mathcal{J}$ und der Rangverteilung λ.

```
function MULTTRANS_CLUSTERBASIS_VEKTOR(s, W, x, x_s)
    x_s ← 0 ∈ ℝ^{λ_s}
    if s ∈ L_J then
        x_s ← W_s^T x_|s
    else                                              ▷ s ∈ T_J \ L_J
        for š ∈ chil(s) do
            MULTTRANS_CLUSTERBASIS_VEKTOR(š, W, x, x_š)   ▷ bottom-up
            x_s ← x_s + F_š^T x_š
        end for
    end if
end function
```

Lemma 6.1.1
Es sei $(W_s)_{s\in\mathcal{T}_\mathcal{J}}$ eine Clusterbasis mit Rangverteilung λ und $k_{\max} := \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s$. Dann

benötigt der Algorithmus 6.1.1 für jedes $x \in \mathbb{R}^{\mathcal{J}}$ höchstens

$$2k_{\max}\#\mathcal{J} + 2k_{\max}^2\#\mathcal{T}_{\mathcal{J}}$$

Operationen.

Beweis: Für den Aufwand des Algorithmus 6.1.1 gilt für einen Blattcluster $s \in \mathcal{L}_{\mathcal{J}}$ die obere Schranke $2\#\lambda_s\#\mathbf{s} \leq 2k_{\max}\#\mathbf{s}$. Für einen Nicht-Blattcluster $s \in \mathcal{T}_{\mathcal{J}}\backslash\mathcal{L}_{\mathcal{J}}$ benötigt die Berechnung von $x_s + F_{\check{s}}^T x_{\check{s}}$ für jedes $\check{s} \in \operatorname{chil}(s)$ maximal $2\#\lambda_s\#\lambda_{\check{s}} \leq 2k_{\max}^2$ Operationen. Mit Korollar 3.2.10 folgt die Aufwandsabschätzung für Algorithmus 6.1.1

$$\begin{aligned}\sum_{s\in\mathcal{L}_{\mathcal{J}}} 2k_{\max}\#\mathbf{s} + \sum_{s\in\mathcal{T}_{\mathcal{J}}\backslash\mathcal{L}_{\mathcal{J}}} \sum_{\check{s}\in\operatorname{chil}(s)} 2k_{\max}^2 &\leq 2k_{\max}\#\mathcal{J} + \sum_{s\in\mathcal{T}_{\mathcal{J}}} 2k_{\max}^2 \\ &= 2k_{\max}\#\mathcal{J} + 2k_{\max}^2\#\mathcal{T}_{\mathcal{J}}.\end{aligned}$$

■

Die Multiplikation mit der Kopplungsmatrix ist direkt möglich und kostet höchstens $2k_{\max}^2$ Rechenoperationen. Die Multiplikation mit der Clusterbasis V_t berechnen wir umgekehrt zu Algorithmus 6.1.1 top-down. Hierbei verwenden wir wieder die hierarchische Struktur der Clusterbasis. Es gilt für ein Cluster $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$ und einen Vektor $y_t \in \mathbb{R}^{\kappa_t}$

$$V_t y_t = \sum_{\check{t}\in\operatorname{chil}(t)} V_{\check{t}} E_{\check{t}} y_t =: \sum_{\check{t}\in\operatorname{chil}(t)} V_{\check{t}} y_{\check{t}}$$

mit Hilfsvektoren $y_{\check{t}} := E_{\check{t}} y_t \in \mathbb{R}^{\kappa_{\check{t}}}$. Dies ergibt den rekursiven Algorithmus 6.1.2. Den Aufwand des Algorithmus schätzen wir in Lemma 6.1.2 ab.

Algorithmus 6.1.2 Die Funktion berechnet das Produkt $y_{|\mathbf{t}} = y_{|\mathbf{t}} + V_t y_t$ für die Clusterbasis (V, E) zum Clusterbaum $\mathcal{T}_{\mathcal{I}}$ mit Rangverteilung κ.

```
function MULT_CLUSTERBASIS_VEKTOR(t, V, y_t, y)
    if t ∈ L_I then
        y_|t ← y_|t + V_t y_t
    else                                              ▷ t ∈ T_I \ L_I
        for ť ∈ chil(t) do
            y_ť ← E_ť y_t ∈ R^{κ_ť}
            MULT_CLUSTERBASIS_VEKTOR(ť, V, y_ť, y)    ▷ top-down
        end for
    end if
end function
```

Lemma 6.1.2
Es sei $(V_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ eine Clusterbasis mit Rangverteilung κ und $k_{\max} := \max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t$. Dann benötigt der Algorithmus 6.1.2 für jedes $y_t \in \mathbb{R}^{\kappa_{r_{\mathcal{I}}}}$ höchstens

$$2k_{\max}\#\mathcal{I} + 2k_{\max}^2\#\mathcal{T}_{\mathcal{I}}$$

Operationen.

Beweis: Der Aufwand des Algorithmus 6.1.2 in einem Blattcluster $t \in \mathcal{L}_\mathcal{I}$ ist durch $2\#\kappa_t\#\mathbf{t} \leq 2k_{\max}\#\mathbf{t}$ beschränkt. In einem nicht-Blattcluster $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ benötigt die Berechnung von $E_{\check{t}}y_t$ für jedes $\check{t} \in \operatorname{chil}(t)$ höchstens $2\#\kappa_t\#\kappa_{\check{t}} \leq 2k_{\max}^2$ Operationen. Mit Korollar 3.2.10 folgt wie in Lemma 6.1.1, dass der Algorithmus 6.1.2 höchstens

$$\sum_{t\in\mathcal{L}_\mathcal{I}} 2k_{\max}\#\mathbf{t} + \sum_{t\in\mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}} \sum_{\check{t}\in\operatorname{chil}(t)} 2k_{\max}^2 \leq 2k_{\max}\#\mathcal{I} + 2k_{\max}^2\#\mathcal{T}_\mathcal{I}$$

Operationen benötigt. ∎

Falls wir die $\mathcal{H}^2$-Matrix-Vektor-Multiplikation mit Hilfe der Algorithmen 6.1.1 und 6.1.2 realisieren, wäre der Aufwand in jedem Block linear und wir würden vergleichbar mit dem Speicheraufwand für $\mathcal{H}$-Matrizen einen Rechenaufwand in $\mathcal{O}(n\log(n))$ erhalten (siehe Lemma 3.3.6).
Deshalb beschreiben wir den Algorithmus für die Matrix-Vektor-Multiplikation aus [12], der eine Berechnung in linearer Komplexität ermöglicht. Eine schnellere Auswertung ist typischerweise nicht zu erreichen, weil dies der Komplexität des Speicheraufwands entspricht. Dazu teilen wir die Berechnung in drei Schritte. Erstens berechnen wir die Hilfsgrößen $x_s := W_s^T x_\mathbf{s}$ für alle $s \in \mathcal{T}_\mathcal{J}$. Durch die Berechnungen dieser Größen für alle s durch eine Funktion vermeiden wir die mehrfache Berechnungen der x_s. Danach multiplizieren wir diese für alle $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+$ mit den entsprechenden Kopplungsmatrizen und addieren die Ergebnisse jeweils zu den Hilfsvektoren y_t. Damit erhalten für alle $t \in \mathcal{T}_\mathcal{I}$ Hilfsvektoren

$$y_t := \sum_{s\in\operatorname{row}(t)} S_{(t,s)}W_s^T x_{|\mathbf{s}} = \sum_{s\in\operatorname{row}(t)} S_{(t,s)}x_s,$$

wobei wir im Falle von $\operatorname{row}(t) = \emptyset$ wie üblich $y_t = 0$ setzen. Für diese gilt

$$\begin{aligned} Hx &= \sum_{(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} H_{|\mathbf{t}\times\mathbf{s}}x = \sum_{(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^-} H_{|\mathbf{t}\times\mathbf{s}}x + \sum_{(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+} V_t S_{(t,s)}W_s^T x \\ &= \sum_{(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^-} H_{|\mathbf{t}\times\mathbf{s}}x + \sum_{t\in\mathcal{T}_\mathcal{I}} V_t y_t. \end{aligned}$$

Also berechnen wir im zweiten Schritt die Hilfsvektoren y_t und verarbeiten die Nahfeldmatrizen. Abschließend müssen wir dann noch die Hilfsgrößen mit den entsprechenden Clusterbasen multiplizieren.
Die Hilfsgrößen $x_s = W_s^T x$ berechnen wir wie in Algorithmus 6.1.1, wobei wir diesmal alle Hilfsvektoren speichern. Dies führt zu Algorithmus 6.1.3, dessen Speicher- und Rechenaufwand wir in Lemma 6.1.3 abschätzen.

Lemma 6.1.3
Es sei $(W_s)_{s\in\mathcal{T}_\mathcal{J}}$ eine Clusterbasis mit Rangverteilung λ, $k_{\max} := \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s$ und $x \in \mathbb{R}^\mathcal{J}$. Dann benötigt Algorithmus 6.1.3 höchstens

$$2k_{\max}\#\mathcal{J} + 2k_{\max}^2\#\mathcal{T}_\mathcal{J}$$

Algorithmus 6.1.3 [siehe [12, Algorithm 6]] Die Funktion berechnet die Familie $x_{s'} := W_{s'}^T x_{|\mathbf{s}'} \in \mathbb{R}^{\lambda_{s'}}$, $s' \in \operatorname{desc}(s)$, für die Clusterbasis (W, F) mit der Rangverteilung λ.

```
function Vorwärtstransformation(s, W, x, (x_{s'})_{s'∈T_J})
    if s ∈ L_J then
        x_s ← W_s^T x_{|s} ∈ R^{λ_s}
    else                                                  ▷ s ∈ T_J \ L_J
        x_s ← 0 ∈ R^{λ_s}
        for š ∈ chil(s) do
            Vorwärtstransformation(š, W, λ, x, (x_s)_{s∈T_J})   ▷ bottom-up
            x_s ← x_s + F_š^T x_š
        end for
    end if
end function
```

Operationen zur Berechnung von $(x_s)_{s\in\mathcal{T}_\mathcal{J}}$ und der Speicheraufwand für diese Familie von Vektoren ist kleiner als $k_{\max}\#\mathcal{T}_\mathcal{J}$.

Beweis: [vergleiche [12, Lemma 3.41]] Der Algorithmus 6.1.3 benötigt dieselben Rechenoperationen wie Algorithmus 6.1.1. Mit Lemma 6.1.1 folgt die Aufwandsabschätzung.
In jedem Cluster $s \in \mathcal{T}_\mathcal{J}$ benötigt die Familie $\#\lambda_s \leq k_{\max}$ Speicher. Daraus folgt die Speicherabschätzung. ∎

Nach der Berechnung der Hilfsvektoren multiplizieren wir die Teil- bzw. Hilfsvektoren mit den Nahfeld- und Kopplungsmatrizen. Dies beschreiben wir in Algorithmus 6.1.4. Dabei berechnen wir die Multiplikation für die Nahfeldmatrizen in den unzulässigen Blättern und für die Kopplungsmatrizen in den zulässigen Blättern in einer Funktion, weil dies nur einen Durchlauf des Blockbaums benötigt. Der Aufwand für Algorithmus 6.1.4 schätzen wir in Lemma 6.1.4 ab.

Lemma 6.1.4
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit strikt zulässigem Blockbaum und Rangverteilungen κ und λ. Weiter seien

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\} \quad \text{und} \quad n_{\max} := \max\{\max_{t\in\mathcal{L}_\mathcal{I}} \#\mathbf{t}, \max_{s\in\mathcal{L}_\mathcal{J}} \#\mathbf{s}\}.$$

Dann benötigt der Algorithmus 6.1.4 aufgerufen mit $b = r_{\mathcal{I}\times\mathcal{J}}$ maximal

$$2\max\{n_{\max}^2, k_{\max}^2\}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$$

Rechenoperationen.

Beweis: [vergleiche [12, Theorem 3.42]] Es sei $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Falls $b \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ gilt, benötigt der Algorithmus 6.1.4 keine arithmetischen Operationen. Für $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ wird $y_t + S_b x_s$ mit höchstens $2\#\kappa_t\#\lambda_s \leq 2k_{\max}^2$ Operationen berechnet. Für den Aufwand der

Algorithmus 6.1.4 [vergleiche [12, Algorithm 8]] Die Funktion berechnet die Multiplikation mit Nahfeld- und Kopplungsmatrizen der $\mathcal{H}^2$-Matrix $\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$, wobei die Hilfsvektoren y_t mit 0 initialisiert seien.

```
function MULT_BLOCK(b, N, S, x, (x_t)_{t∈T_I}, y, (y_t)_{t∈T_I})
    (t, s) ← b
    if b ∈ L⁺_{I×J} then
        y_t = y_t + S_b x_s
    else if b ∈ L⁻_{I×J} then
        y_{|t} = y_{|t} + N_b x_{|s}
    else                                              ▷ b ∈ T_{I×J} \ L_{I×J}
        for b̆ ∈ chil(b) do
            MULT_BLOCK(b̆, N, S, x, (x_t)_{t∈T_I}, y, (y_t)_{t∈T_I})
        end for
    end if
end function
```

Berechnung von $y_{|\mathbf{t}} + N_b x_{|\mathbf{s}}$ im Falle von $b \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ gilt wegen $t \in \mathcal{L}_\mathcal{I}$ und $s \in \mathcal{L}_\mathcal{J}$ die obere Schranke $2\#\mathbf{t}\#\mathbf{s} \leq 2n^2_{\max}$. Damit folgt die Aufwandsabschätzung

$$\sum_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} 2k^2_{\max} \;+\; \sum_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} 2n^2_{\max} \;\leq\; 2\max\{n^2_{\max}, k^2_{\max}\}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}.$$

■

Damit sind die Hilfsvektoren y_t berechnet und die Produkte $H_{\mathbf{t}\times\mathbf{s}}x_{|\mathbf{s}}$ für alle $(t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ auf $y_{|\mathbf{t}}$ addiert. Es bleibt $V_t y_t$ effizient auszuwerten und dabei Mehrfachauswertungen zu vermeiden. Es gilt für den Ergebnisvektor ohne Nahfeldanteil in einem Blattcluster $t \in \mathcal{L}_\mathcal{I}$

$$\sum_{\hat{t}\in\operatorname{pred}(t)} V_{\hat{t}|\mathbf{t}} y_{\hat{t}} \;=\; \sum_{\hat{t}\in\operatorname{pred}(t)} V_t E_{t,\hat{t}} y_{\hat{t}} \;=\; V_t(y_t + E_t(y_{t_{m-1}} + E_{t_{m-1}}(\cdots(y_{t_1} + E_{t_1} y_{t_0})))),$$

wobei $m = \operatorname{level}(t)$ sei und t_i jeweils den Vorgänger von t mit $\operatorname{level}(t_i) = i$ bezeichne. Also ergibt sich eine Art Horner-Schema, auf dessen Basis die Rücktransformation der Hilfsvektoren y_t zum Ergebnis y berechnet werden kann. Mit Hilfe dieses Ansatzes formulieren wir Algorithmus 6.1.5, der die Berechnungen von der Wurzel ausgehend für alle y_t, $t \in \mathcal{T}_\mathcal{I}$, zusammen ausführt. Den Aufwand beschränken wir in Lemma 6.1.5.

Lemma 6.1.5
Es sei $(V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis mit Rangverteilung κ, $k_{\max} := \max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t$ und $y_t \in \mathbb{R}^{\kappa_t}$, für alle $t \in \mathcal{T}_\mathcal{I}$. Dann benötigt Algorithmus 6.1.5 höchstens

$$2k_{\max}\#\mathcal{I} + 2k^2_{\max}\#\mathcal{T}_\mathcal{I}$$

Operationen zur Berechnung von $y_{|\mathbf{t}} \;=\; y_{|\mathbf{t}} + \sum_{\hat{t}\in\operatorname{pred}(t)} V_{\hat{t}|\mathbf{t}} y_{\hat{t}}$ und der Speicheraufwand für $(y_t)_{t\in\mathcal{T}_\mathcal{I}}$ ist kleiner als $k_{\max}\#\mathcal{T}_\mathcal{I}$.

Algorithmus 6.1.5 [siehe [12, Algorithm 7]] Die Funktion berechnet y mit $y_{|\mathbf{t}'} = y_{|\mathbf{t}'} + \sum_{\hat{t}\in\operatorname{pred}(t')} V_{\hat{t}|\mathbf{t}'} y_{\hat{t}}$ für alle $t' \in \mathcal{L}_\mathcal{I}$ für die Familie $(y_{t'})_{t'\in\mathcal{T}_\mathcal{I}}$ und die Clusterbasis (V, E) im Clusterbaum $\mathcal{T}_\mathcal{I}$ mit Rangverteilung κ.

```
function RÜCKWÄRTSTRANSFORMATION(t, V, (y_t)_{t∈T_I}, y)
    if t ∈ L_I then
        y_|t = y_|t + V_t y_t
    else                                              ▷ t ∈ T_I \ L_I
        for ť ∈ chil(t) do
            y_ť = y_ť + E_ť y_t
            RÜCKWÄRTSTRANSFORMATION(ť, V, (y_t)_{t∈T_I}, y)   ▷ top-down
        end for
    end if
end function
```

Beweis: [vergleiche [12, Lemma 3.41]] Weil wir in Lemma 6.1.2 die Matrix-Vektor-Multiplikationen mit Überschreiben in Algorithmus 6.1.2 gegen $2k_{\max}$ abschätzen und für die Matrix-Vektor-Multiplikationen mit Addition in Algorithmus 6.1.5 dieselbe Schranke gilt, folgt die Aufwandsabschätzung wie in Lemma 6.1.2.
Für die Hilfsvektoren werden für alle $t \in \mathcal{T}_\mathcal{I}$ maximal $\kappa_t \leq k_{\max}$ Einträge gespeichert. Durch Summieren ergibt sich die Abschätzung. ■

Mit Hilfe dieser Funktionen können wir die Matrix-Vektor-Multiplikation in Algorithmus 6.1.6 realisieren. Den Rechen- und Speicheraufwand schätzen wir in 6.1.6 ab.

Algorithmus 6.1.6 [vergleiche [12, Algorithm 8]] Die Funktion berechnet die $\mathcal{H}^2$-Matrix-Vektor-Multiplikation $y = y + Hx$ für eine $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ mit Rangverteilungen κ und λ.

```
function H2MATRIX_VEKTOR_MULTIPLIKATION(H, x, y)
    for s ∈ T_J do                          ▷ Initialisierung der Hilfsgrößen
        x_s ← 0 ∈ R^{λ_s}
    end for
    for t ∈ T_I do
        y_t ← 0 ∈ R^{κ_t}
    end for
    VORWÄRTSTRANSFORMATION(r_J, W, x, (x_s)_{s∈T_J})          ▷ Algorithmus 6.1.3
    MULT_BLOCK(r_{I×J}, N, S, x, (x_t)_{t∈T_I}, y, (y_t)_{t∈T_I})   ▷ Algorithmus 6.1.4
    RÜCKWÄRTSTRANSFORMATION(r_I, V, (y_t)_{t∈T_I}, y)         ▷ Algorithmus 6.1.5
end function
```

Lemma 6.1.6
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit strikt zulässigem Blockbaum und

Rangverteilungen κ und λ. Weiter seien

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\} \quad \text{und} \quad n_{\max} := \max\{\max_{t\in\mathcal{L}_\mathcal{I}} \#\mathbf{t}, \max_{s\in\mathcal{L}_\mathcal{J}} \#\mathbf{s}\}.$$

Dann benötigt der Algorithmus 6.1.6 maximal

$$2k_{\max}(\#\mathcal{I} + \#\mathcal{J}) \ + \ 2k_{\max}^2(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}) \ + \ 2\max\{n_{\max}^2, k_{\max}^2\}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$$

Rechenoperationen. Für die Hilfsvektoren werden höchstens $k_{\max}(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J})$ Einträge gespeichert.

Beweis: [vergleiche [12, Theorem 3.42]] Für das Anlegen der Hilfsvektoren sind keine Rechenoperationen notwendig. Die Abschätzung für Rechen- und Speicheraufwand folgt durch das Summieren der Schranken der Lemmata 6.1.3, 6.1.4 und 6.1.5. ■

Der Rechenaufwand für die Matrix-Vektor-Multiplikation lässt sich also durch den zweifachen Speicheraufwand für eine $\mathcal{H}^2$-Matrix beschränken (siehe Lemma 3.4.25). Dies ist das gleiche Ergebnis wie für vollbesetzte Matrizen und $\mathcal{H}$-Matrizen.
Nachdem wir die Multiplikation von $\mathcal{H}^2$-Matrizen mit Vektoren untersucht haben, wenden wir uns jetzt dem Produkt von $\mathcal{H}^2$-Matrizen mit nicht unterteilten Matrizen zu. Dabei betrachten wir Matrizen im vollbesetzten Format und in der Drei-Term-Darstellung VSW^T. Dies sind die Matrixformate, die wir später in den Spezialfällen für das Produkt von $\mathcal{H}^2$-Matrizen benötigen.
Das Produkt einer $\mathcal{H}^2$-Matrix mit einer voll gespeicherten Matrix realisieren wir spaltenweise mit Hilfe der $\mathcal{H}^2$-Matrix-Vektor-Multiplikation. Dazu bezeichnen wir den i-ten Spaltenvektor einer Matrix X mit x^i. Mit dieser Notation formulieren wir Algorithmus 6.1.7, dessen Aufwand wir in Lemma 6.1.7 untersuchen.

Algorithmus 6.1.7 Die Funktion berechnet die Multiplikation $Y = Y + HX$ für eine $\mathcal{H}^2$-Matrix H und zwei Matrizen X und Y.

```
function H2Matrix_Matrix_Multiplikation(H, X, Y)
    for k ∈ K do
        H2Matrix_Vektor_Multiplikation(H, x^k, y^k)        ▷ Algorithmus 6.1.6
    end for
end function
```

Lemma 6.1.7
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit strikt zulässigem Blockbaum und Rangverteilungen κ und λ. Weiter seien $X \in \mathbb{R}^{\mathcal{J}\times\mathcal{K}}$, $Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$,

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\} \quad \text{und} \quad n_{\max} := \max\{\max_{t\in\mathcal{L}_\mathcal{I}} \#\mathbf{t}, \max_{s\in\mathcal{L}_\mathcal{J}} \#\mathbf{s}\}.$$

Dann benötigt der Algorithmus 6.1.7 maximal

$$\big(2k_{\max}(\#\mathcal{I} + \#\mathcal{J}) \ + \ 2k_{\max}^2(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}) \ + \ 2\max\{n_{\max}^2, k_{\max}^2\}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}\big) \ \#\mathcal{K}$$

Rechenoperationen. Für die Hilfsvektoren werden höchstens $k_{\max}(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J})$ Einträge gespeichert.

Beweis: Der Algorithmus 6.1.7 ruft $\#\mathcal{K}$-mal den Algorithmus 6.1.6 auf. Mit Lemma 6.1.6 folgt die Abschätzung für Rechen- und Speicheraufwand. ∎

Die Matrizen in der Drei-Term-Darstellung $U = V_t S_b W_s^T$ tauchen in den zulässigen Blättern der $\mathcal{H}^2$-Matrizen auf. Diese uniformen Matrizen wollen wir dann mit einer $\mathcal{H}^2$-Matrix H multiplizieren, wobei wir das Ergebnis als Rang-$\mathcal{K}$-Darstellung berechnen, um später das Niedrigrangupdate aus Kapitel 5 verwenden zu können. Um die Berechnung einfach zu halten, transformieren wir die uniforme Matrix zuerst in eine Rang-$\mathcal{K}$-Darstellung $U = \widetilde{R} = \widetilde{A}B^T$. Danach multiplizieren wir H und $\widetilde{A}$ mit Algorithmus 6.1.7 und erhalten die Niedrigrangmatrix $R := AB^T := (H\widetilde{A})B^T = HU$.
Mit Hilfe des Algorithmus 6.1.2 können wir die Multiplikation einer Clusterbasis mit einer Matrix berechnen. Indem wir wie bei der Multiplikation einer $\mathcal{H}^2$-Matrix mit einer Matrix spaltenweise vorgehen, erhalten wir Algorithmus 6.1.8. Dabei bezeichnen wir die Spalten der Matrizen X und Y mit x^i bzw. y^i. Den Aufwand für diese Berechnung schätzen wir in Lemma 6.1.8 ab.

Algorithmus 6.1.8 Die Funktion berechnet die Multiplikation $Y = Y + VX$ für eine Clusterbasis V mit Rangverteilung κ und Matrizen X und Y.

```
function MULT_CLUSTERBASIS_MATRIX(t, V, X, Y)
    for k ∈ K do
        MULT_CLUSTERBASIS_VEKTOR(t, V, x^i, y^i)        ▷ Algorithmus 6.1.2
    end for
end function
```

Lemma 6.1.8
Es sei $(V_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis mit Rangverteilung κ, $X \in \mathbb{R}^{\kappa_{r_\mathcal{I}}\times\mathcal{K}}$, $Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{K}}$ und $k_{\max} := \max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t$. Dann benötigt der Algorithmus 6.1.8 höchstens

$$\left(2k_{\max}\#\mathcal{I} + 2k_{\max}^2\#\mathcal{T}_\mathcal{I}\right)\#\mathcal{K}$$

Operationen.

Beweis: Die Abschätzung folgt aus Lemma 6.1.2. ∎

Mit Hilfe dieser Funktion kann eine uniforme Matrix einfach in eine Niedrigrangmatrix konvertieren, indem wir für $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$

$$V_t S_b W_s^T \;=\; (V_t S_b)(W_s I_{\lambda_s})^T \;=:\; A_b B_b^T$$

setzen. Diesen Ansatz verwenden wir in Algorithmus 6.1.9 und schätzen den Aufwand in Lemma 6.1.9 ab.

Algorithmus 6.1.9 Die Funktion konvertiert eine uniforme Matrix $U = V_t S_{(t,s)} W_s^T$ mit Clusterbasen V und W zu Clusterbäumen $\mathcal{T}_\mathcal{I}$ bzw. $\mathcal{T}_\mathcal{J}$ mit Rangverteilungen κ bzw. λ in eine Rang-λ_s-Matrix $R = AB^T$.

function UNIFORM_RKMATRIX_KONVERTIEREN(t, s, U, R)
  $A = 0 \in \mathbb{R}^{\mathcal{I} \times \lambda_s}$
  $B = 0 \in \mathbb{R}^{\mathcal{J} \times \lambda_s}$
  MULT_CLUSTERBASIS_MATRIX(t, V, S, A) ▷ Algorithmus 6.1.8
  MULT_CLUSTERBASIS_MATRIX(s, W, I, B) ▷ Algorithmus 6.1.8
  $R \leftarrow (A, B)$
end function

Lemma 6.1.9
Es seien $(V_t)_{t \in \mathcal{T}_\mathcal{I}}$ und $(W_s)_{s \in \mathcal{T}_\mathcal{J}}$ Clusterbasen mit Rangverteilungen κ bzw. λ, $S_b \in \mathbb{R}^{\kappa_{r_\mathcal{I}} \times \lambda_{r_\mathcal{J}}}$ und $U = V_{r_\mathcal{I}} S_U W_{r_\mathcal{J}}^T$. Dann gilt mit $k_{\max} := \max\{\max_{t \in \mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s \in \mathcal{T}_\mathcal{J}} \#\lambda_s\}$ für den Aufwand des Algorithmus 6.1.9 die obere Schranke

$$2k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) + 2k_{\max}^3(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}).$$

BEWEIS: In Algorithmus 6.1.9 wird der Algorithmus 6.1.8 zweimal aufgerufen. Die Aufwandsabschätzung folgt mit Lemma 6.1.8 und $\#\lambda_{r_\mathcal{I}} \leq k_{\max}$. ∎

Um den Rang des Ergebnisses zu minimieren, ist es in der Praxis sinnvoll, für $\#\kappa_t < \#\lambda_s$ die Matrizen $A := V_t I_{\kappa_t}$ und $B := W_s S_{(t,s)}^T$ zu setzen. Wir können also eine uniforme Matrix mit linearem Aufwand in eine Niedrigrangmatrix konvertieren und damit den oben beschriebenen Ansatz verwenden, um die Multiplikation einer $\mathcal{H}^2$-Matrix mit einer uniformen Matrix zu berechnen. Dieses Vorgehen beschreiben wir in Algorithmus 6.1.10, dessen Aufwand wir in Lemma 6.1.10 abschätzen.

Algorithmus 6.1.10 Die Funktion berechnet das Produkt $R = HU$ einer uniformen Matrix $U = V_U S_U W_U^T$ mit Clusterbäumen $\mathcal{T}_\mathcal{J}$ und $\mathcal{T}_\mathcal{K}$ und einer $\mathcal{H}^2$-Matrix H, wobei das Ergebnis als Niedrigrangmatrix $R = AB^T$ gespeichert wird.

function H2MATRIX_UNIFORM_MULTIPLIKATION(H, U, R)
  UNIFORM_RKMATRIX_KONVERTIEREN($r_\mathcal{J}$, $r_\mathcal{K}$, U, $(\widetilde{A}, B)$) ▷ Algorithmus 6.1.9
  H2MATRIX_MATRIX_MULTIPLIKATION(H, $\widetilde{A}$, A) ▷ Algorithmus 6.1.7
  $R \leftarrow (A, B)$
end function

Lemma 6.1.10
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_H, W_H, S_H, N_H)$ eine $\mathcal{H}^2$-Matrix mit streng zulässigem Blockbaum und Rangverteilungen κ_H und λ_H. Weiter seien $(V_{U,s})_{s \in \mathcal{T}_\mathcal{J}}$ und $(W_{U,r})_{r \in \mathcal{T}_\mathcal{K}}$ Clusterbasen mit Rangverteilungen κ_U und λ_U, $S_U \in \mathbb{R}^{\kappa_{U,r_\mathcal{J}} \times \lambda_{U,r_\mathcal{K}}}$ und $U = V_{U,r_\mathcal{J}} S_{U,\mathcal{J}\times\mathcal{K}} W_{U,r_\mathcal{K}}^T$. Dann gilt für den maximalen lokalen Rang

$$k_{\max} := \max\{\max_{t \in \mathcal{T}_\mathcal{I}} \#\kappa_{H,t}, \max_{s \in \mathcal{T}_\mathcal{J}} \#\lambda_{H,s}, \max_{t \in \mathcal{T}_\mathcal{J}} \#\kappa_{U,t}, \max_{s \in \mathcal{T}_\mathcal{K}} \#\lambda_{U,s}\}$$

und die maximale Blattgröße $n_{\max} := \max\{\max_{t\in\mathcal{L}_\mathcal{I}} \#\mathbf{t}, \max_{s\in\mathcal{L}_\mathcal{J}} \#\mathbf{s}, \max_{r\in\mathcal{L}_\mathcal{K}} \#\mathbf{r}\}$, dass der Algorithmus 6.1.10 höchstens

$$\begin{aligned} 2k_{\max}^2(\#\mathcal{I} + 2\#\mathcal{J} + \#\mathcal{K}) \;+\; & 2k_{\max}^3(\#\mathcal{T}_\mathcal{I} + 2\#\mathcal{T}_\mathcal{J} + \#\mathcal{T}_\mathcal{K}) \\ & +\; 2\max\{n_{\max}^2, k_{\max}^2\}k_{\max}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \end{aligned}$$

Rechenoperationen benötigt.

Beweis: Nach Lemma 6.1.9 benötigt der Aufruf von Algorithmus 6.1.9 maximal

$$2k_{\max}^2(\#\mathcal{J} + \#\mathcal{K}) + 2k_{\max}^3(\#\mathcal{T}_\mathcal{J} + \#\mathcal{T}_\mathcal{K})$$

Operationen. Weil der Rang der resultierenden Niedrigrangmatrix kleiner als $k_{\max}$ ist, gilt für den Aufwand von Algorithmus 6.1.7 nach Lemma 6.1.7 die obere Schranke

$$2k_{\max}^2(\#\mathcal{I} + \#\mathcal{J}) \;+\; 2k_{\max}^3(\#\mathcal{T}_\mathcal{I} + \#\mathcal{T}_\mathcal{J}) \;+\; 2\max\{n_{\max}^2, k_{\max}^2\}k_{\max}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}.$$

Daraus folgt die Behauptung. ∎

Damit können wir die Multiplikation einer $\mathcal{H}^2$-Matrix mit einer uniformen Matrix in linearem Aufwand berechnen. Im nächsten Abschnitt untersuchen wir die Multiplikation zweier $\mathcal{H}^2$-Matrizen.

6.2 Produkt von $\mathcal{H}^2$-Matrizen

Nachdem wir einige Spezialfälle untersucht haben, können wir die Multiplikation zweier $\mathcal{H}^2$-Matrizen beschreiben. Die Funktion für die Multiplikation modellieren wir in der Form $C + AB$. Dies ist, wie am Anfang dieses Kapitels beschrieben, der hierarchischen Blockstruktur geschuldet.
In [12, Theorem 7.29] ist ein $\mathcal{H}^2$-Matrix-Raum angegeben, in dem das Produkt $C + AB$ enthalten ist. Allerdings haben die für die Darstellung verwendeten Clusterbasen deutlich höheren Rang als die ursprünglichen (siehe [12, Lemma 7.23]) und der verwendete Blockbaum ist deutlich größer (die Konstante der Schwachbesetztheit entspricht ca. dem Quadrat der Konstanten der zugrunde liegenden Bäume, siehe [12, Lemma 7.28]).
Stattdessen wollen wir das Ergebnis in einem $\mathcal{H}^2$-Matrix-Raum zum Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ der Matrix C mit adaptiven Clusterbasen approximieren. Als Erstes betrachten wir den Fall, dass alle drei Matrizen weiter unterteilt sind. Dabei betrachten wir vorerst den Fall, dass alle drei Matrizen in 2×2 Kinder unterteilt sind. Damit erhalten wir für die Multiplikation der Form $C + AB$ die Darstellung

$$\begin{aligned} \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} \;+\; & \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \\ = \; & \begin{pmatrix} C_{11} + A_{11}B_{11} + A_{12}B_{21} & C_{12} + A_{11}B_{12} + A_{12}B_{22} \\ C_{21} + A_{21}B_{11} + A_{22}B_{21} & C_{22} + A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}. \end{aligned}$$

(Im allgemeinen Fall müssen wir alle $\check{t} \in \operatorname{chil}^*(t)$, $\check{s} \in \operatorname{chil}^*(s)$ und $\check{r} \in \operatorname{chil}^*(r)$ durchlaufen und jeweils das Produkt $A_{|\check{\mathbf{t}} \times \check{\mathbf{s}}} B_{|\check{\mathbf{s}} \times \check{\mathbf{r}}}$ zur Teilmatrix $C_{|\check{\mathbf{t}} \times \check{\mathbf{r}}}$ addieren.) Dieser Ansatz führt zu einer Rekursion, die wir für die Fälle, in denen einer der Blöcke ein Blatt ist, auflösen müssen. Wie schon in Kapitel 5 erwähnt, wollen wir dann das Produkt von A und B als Niedrigrangmatrix XY^T darstellen, um diese anschließend per Niedrigrangupdate zu C zu addieren. Die Aufgabe besteht also darin, AB in den verschiedenen Fällen als Niedrigrangmatrix darzustellen.
Es sei A ein zulässiges Blatt. Dann ist A durch eine uniforme Matrix dargestellt und wir können AB effizient als Produkt einer $\mathcal{H}^2$-Matrix mit einer uniformen Matrix berechnen (siehe Algorithmus 6.1.10). Das Resultat ist, wie gewünscht, eine Niedrigrangmatrix.
Es sei A ein unzulässiges Blatt. Dann ist A eine "kleine" vollbesetzte Matrix. Somit können wir AB effizient berechnen (siehe Algorithmus 6.1.7). Da das Resultat "wenige" Zeilen besitzt, können wir es als Niedrigrangmatrix darstellen, indem wir die Identität als linken Faktor nutzen.
Falls B ein Blatt ist, können wir analog zu den Fällen von A vorgehen, um die Niedrigrangmatrix zu konstruieren.
Es bleibt der Fall, dass sowohl A als auch B keine Blätter sind, aber C ein Blatt ist. Für diesen Fall definieren wir eine Rekursion, die das Produkt AB approximativ als Niedrigrangmatrix berechnet. Falls A oder B ein Blatt ist, gehen wir wie oben beschrieben vor. Ansonsten rufen wir die Funktion rekursiv für die Teilblöcke von A und B auf und fassen die resultierenden Niedrigrangmatrizen approximativ zu einer neuen großen Niedrigrangmatrix zusammen.
Falls wir für die Blöcke b' und b'' die Funktion rekursiv aufgerufen haben, addieren wir das Ergebnis $\widetilde{R}$ zur Niedrigrangmatrix R. Dazu erweitern wir $\widetilde{R}$ durch Nullblöcke auf die Größe von R und verwenden dann Algorithmus 2.4.5. Durch diesen Ansatz erhalten wir Algorithmus 6.2.1, der das Produkt zweier $\mathcal{H}^2$-Matrizen durch eine Niedrigrangmatrix approximiert. Diesen verwenden wir dann in dem rekursiven Multiplikations-Algorithmus 6.2.2 zur Berechnung des Produkts von $\mathcal{H}^2$-Matrizen.

Bemerkung 6.2.1
Um die approximativen Algorithmen im Folgenden übersichtlich zu halten, fassen wir die Optionen zum Kürzen *opt*, die dabei verwendeten Genauigkeiten $\epsilon_{\mathcal{I}}$ und $\epsilon_{\mathcal{J}}$, die Strategien zur Berechnung der Skalierungen *str* sowie die Skalierungen $\omega_{\mathcal{I}}$, $\omega_{\mathcal{J}}$, $\theta_{\mathcal{I}}$ und $\theta_{\mathcal{J}}$ in dem Parameter *opt* zusammen. Ebenso werden wir die Gewichte von $\mathcal{H}^2$-Matrizen bezüglich der Zeilenbasis $Z_{\mathcal{I}}$ und der Spaltenbasis $Z_{\mathcal{J}}$ in einer Größe Z zusammenfassen.

Mit dieser Funktion können wir den Algorithmus zur Berechnung des Produkts zweier $\mathcal{H}^2$-Matrizen definieren, wobei das Ergebnis zu einer weiteren $\mathcal{H}^2$-Matrix addiert wird. Diese besteht im Wesentlichen aus dem rekursiven Durchlaufen der jeweiligen Kinder und im Falle, dass ein Block ein Blatt ist, dem Aufruf von Algorithmus 6.2.1. Weil die Ergebnisse zu der Matrix C addiert werden und wir dabei das Niedrigrangupdate aus Kapitel 5 verwenden, müssen wir projizierte Gewichte für die Matrix mitführen. Die Gewichte können wir nach den Bemerkungen 5.3.2 und 5.3.5 mit Hilfe der Algorithmen 4.2.3 und 4.4.2 initialisieren.

Algorithmus 6.2.1 Die Funktion approximiert das Produkt AB zweier $\mathcal{H}^2$-Matrizen $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_A, W_A, N_A, S_A)$ und $B = \mathcal{H}^2(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}, V_B, W_B, N_B, S_B)$ mit Rangverteilungen κ_A, λ_A, κ_B und λ_B. Das Ergebnis wird als Rang-κ_R-Matrix R dargestellt.

function H2Matrix_Produkt_RkMatrizen(t, s, r, A, B, R, *opt*)
  if $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ **then**
    H2Matrix_Uniform_Produkt(B^T, (W_A, S_A^T, V_A), R) ▷ Algorithmus 6.1.10
  else if $(t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ **then**
    H2Matrix_Matrix_Produkt(B^T, N_A^T, Y) ▷ Algorithmus 6.1.7
    $R \leftarrow (I_{\mathbf{t}}, Y)$
  else if $(s,r) \in \mathcal{L}^+_{\mathcal{J}\times\mathcal{K}}$ **then**
    H2Matrix_Uniform_Produkt(A, (V_B, S_B, W_B), R) ▷ Algorithmus 6.1.10
  else if $(s,r) \in \mathcal{L}^-_{\mathcal{J}\times\mathcal{K}}$ **then**
    H2Matrix_Matrix_Produkt(A, N_B, X) ▷ Algorithmus 6.1.7
    $R \leftarrow (X, I_{\mathbf{r}})$
  else ▷ $(t,s) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $(s,r) \notin \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$
    $\kappa_R = \emptyset$, $X \leftarrow 0 \in \mathbb{R}^{\mathbf{t}\times\kappa_R}$, $Y \leftarrow 0 \in \mathbb{R}^{\mathbf{r}\times\kappa_R}$, $R \leftarrow (X,Y)$
    for $\check{t} \in \mathrm{chil}^*(t)$, $\check{s} \in \mathrm{chil}^*(s)$, $\check{r} \in \mathrm{chil}^*(r)$ **do**
      H2Matrix_Produkt_RkMatrizen($\check{t}$, $\check{s}$, $\check{r}$, A, B, $\widetilde{R}$, *opt*)
      $(\widetilde{X}, \widetilde{Y}) \leftarrow \widetilde{R}$
      $\widehat{X} \leftarrow 0 \in \mathbb{R}^{\mathbf{t}\times\kappa_{\widetilde{R}}}$
      $\widehat{X}_{|\check{\mathbf{t}}\times\kappa_{\widetilde{R}}} \leftarrow \widetilde{X}$
      $\widehat{Y} \leftarrow 0 \in \mathbb{R}^{\mathbf{r}\times\kappa_{\widetilde{R}}}$
      $\widehat{Y}_{|\check{\mathbf{r}}\times\kappa_{\widetilde{R}}} \leftarrow \widetilde{Y}$
      RkMatrix_Summe($(\widehat{X}, \widehat{Y})$, R, *opt*) ▷ Algorithmus 2.4.5
    end for
  end if
end function

Um den Aufwand von Algorithmus 6.2.2 abzuschätzen, schätzen wir einerseits die Anzahl der rekursiven Aufrufe für die Algorithmen 6.2.2 und 6.2.1 ab und andererseits den Aufwand für jeden Aufruf, wobei wir den Aufwand für die rekursiven Aufrufe weglassen. Zuerst definieren wir einen Blockbaum, der die rekursiven Aufrufe widerspiegelt.

Definition 6.2.2
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ Blockbäume. Wir definieren den *Produktbaum* $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}} = (V, \mathrm{chil})$ durch die Menge der Knoten

$$V := \{((t,s),(s',r')) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \times \mathcal{T}_{\mathcal{J}\times\mathcal{K}} \mid s = s' \wedge \mathrm{level}(b) = \mathrm{level}(b')\}$$

und für alle $(b,b') \in V$ die Menge der Kinder

$$\mathrm{chil}(b,b') := \begin{cases} (\mathrm{chil}(b) \times \mathrm{chil}(b')) \cap V & , \text{falls } \mathrm{chil}(b) \neq \emptyset \neq \mathrm{chil}(b') \\ \emptyset & \text{sonst} \end{cases}.$$

Algorithmus 6.2.2 Die Funktion berechnet approximativ das Produkt zweier $\mathcal{H}^2$-Matrizen $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_A, W_A, N_A, S_A)$ und $B = \mathcal{H}^2(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}, V_B, W_B, N_B, S_B)$ mit Rangverteilungen κ_A und λ_A bzw. κ_B und λ_B und addiert das Ergebnis zu einer weiteren $\mathcal{H}^2$-Matrix $C = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{K}}, V_C, W_C, N_C, S_C)$. Die Matrix C wird dabei mit dem Ergebnis $C + AB$ überschrieben.

```
function H2Matrix_Produkt(t, s, r, A, B, C, Z_C, opt)
    if (t,s) ∈ T_{I×J}\L_{I×J} ∧ (s,r) ∈ T_{J×K}\L_{J×K} ∧ (t,r) ∈ T_{I×K}\L_{I×K} then
        for ť ∈ chil*(t), š ∈ chil*(s), ř ∈ chil*(r) do
            H2Matrix_Produkt(ť, š, ř, A, B, C, Z_C, opt)
        end for
    else
        H2Matrix_Produkt_RkMatrizen(t, s, r, A, B, R, opt)
                                                  ▷ Algorithmus 6.2.1
        Niedrigrangupdate((t,r), C, Z_C, R, opt)  ▷ Algorithmus 5.3.8
    end if
end function
```

Der Schnitt mit V in der Definition der Kinder sorgt dafür, dass der Spaltencluster des ersten Blocks gleich dem Zeilenclusters des zweiten Blocks ist. Die Bedingung, dass die Blöcke jeweils auf gleichem Level sind, wird bereits dadurch erfüllt, dass ein Knoten im Produktbaum nur Kinder besitzt, falls beide zugehörigen Blöcke keine Blätter sind. Als Nächstes untersuchen wir, wie sich Summen über den Produktbaum abschätzen lassen.

Lemma 6.2.3
Es seien $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ Blockbäume und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ der Produktbaum aus Definition 6.2.2. Dann ist $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ ein Baum. Falls die Blockbäume $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ jeweils $c_{sp,\mathcal{I}\times\mathcal{J}}$- bzw. $c_{sp,\mathcal{J}\times\mathcal{K}}$-schwachbesetzt sind, gilt für alle $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$

$$\#\{(s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}} \,|\, ((t,s),(s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} \;\leq\; c_{sp,\mathcal{J}\times\mathcal{K}} \tag{6.2}$$

und für alle $(s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$

$$\#\{(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \,|\, ((t,s),(s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} \;\leq\; c_{sp,\mathcal{I}\times\mathcal{J}}. \tag{6.3}$$

In diesem Fall erhalten wir zusätzlich für die verschiedenen Summen über den Produktbaum die folgenden Abschätzungen:

$$\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}} \;\leq\; c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}} \min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}, \#\mathcal{T}_{\mathcal{K}}\}, \tag{6.4}$$

$$\begin{aligned} \sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}\times\mathbf{s}} \;&\leq\; c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, (p_{\mathcal{I}}\, \#\mathcal{T}_{\mathcal{I}} + p_{\mathcal{J}}\, \#\mathcal{T}_{\mathcal{J}}), \\ \sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{s}\times\mathbf{r}} \;&\leq\; c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, (p_{\mathcal{J}}\, \#\mathcal{T}_{\mathcal{J}} + p_{\mathcal{K}}\, \#\mathcal{T}_{\mathcal{K}}), \end{aligned} \tag{6.5}$$

$$\begin{aligned}
\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{I}} \#\mathcal{T}_{\mathcal{I}},\\
\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{s}} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{J}} \#\mathcal{T}_{\mathcal{J}},\\
\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{r}} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{K}} \#\mathcal{T}_{\mathcal{K}},
\end{aligned} \tag{6.6}$$

$$\begin{aligned}
\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathbf{t} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{I}} \#\mathcal{I},\\
\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathbf{s} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{J}} \#\mathcal{J} \quad \text{und}\\
\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathbf{r} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{K}} \#\mathcal{K}.
\end{aligned} \tag{6.7}$$

Beweis: Zuerst zeigen wir, dass $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ ein Baum mit Wurzel $r_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}} := (r_{\mathcal{I}\times\mathcal{J}}, r_{\mathcal{J}\times\mathcal{K}})$ ist. Es sei $(b,b') = ((t,s),(s,r)) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$, $(b_i)_{i=0}^m = (t_i,s_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu b und $(b'_i)_{i=0}^{m'} = (t'_i,s'_i)_{i=0}^{m'}$ der Pfad von $r_{\mathcal{J}\times\mathcal{K}}$ zu b'. Dann gilt $m = \operatorname{level}(b) = \operatorname{level}(b') = m'$. Wir zeigen, dass $(b_i,b'_i)_{i=0}^m$ ein Pfad von $r_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ zu (b,b') ist.
Für alle $i \in \{0,\ldots,m\}$ gilt $\operatorname{level}(b_i) = i = \operatorname{level}(b'_i)$, $b_i \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $b'_i \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$. Es sei $(\hat{s}_i)_{i=0}^{\operatorname{level}(s)}$ der Pfad von $r_{\mathcal{J}}$ zu s. Wegen der Definition der Kinder bei Blockbäumen gilt $s_i = \hat{s}_i = s'_i$ für alle $i \in \{0,\ldots,\operatorname{level}(s)\}$ und $s_i = s = s'_i$ für alle $i \in \{\operatorname{level}(s)+1,\ldots,m\}$. Also ist $(b_i,b'_i) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ für alle $i \in \{0,\ldots,m\}$ und $(b_i,b'_i) \in \operatorname{chil}((b_{i-1},b'_{i-1}))$ für alle $i \in \{1,\ldots,m\}$. Damit ist $(b_i,b'_i)_{i=0}^m$ ein Pfad von $r_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ zu (b,b').
Wir müssen zusätzlich die Eindeutigkeit des Pfades nachweisen. Dazu sei $(\hat{b}_i,\hat{b}'_i)_{i=0}^{\hat{m}}$ ein Pfad von $r_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ zu (b,b'). Nach Definition der Kinder ist dann $(\hat{b}_i)_{i=0}^{\hat{m}}$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu b und $(\hat{b}'_i)_{i=0}^{\hat{m}}$ der Pfad von $r_{\mathcal{J}\times\mathcal{K}}$ zu b'. Damit gilt $(\hat{b}_i,\hat{b}'_i)_{i=0}^{\hat{m}} = (b_i,b'_i)_{i=0}^m$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ ist ein Baum.
Die Blockbäume $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ seien $c_{sp,\mathcal{I}\times\mathcal{J}}$- bzw. $c_{sp,\mathcal{J}\times\mathcal{K}}$-schwachbesetzt. Dann gilt für alle $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$

$$\#\{(s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}} \,|\, ((t,s),(s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} \leq \#\{r \in \mathcal{T}_{\mathcal{K}} \,|\, (s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}\} \leq c_{sp,\mathcal{J}\times\mathcal{K}}$$

und somit (6.2). Für alle $(s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ folgt analog (6.3).
(6.4): Mit (6.2) und Lemma 3.3.6 folgt für die Mächtigkeit des Produktbaums

$$\begin{aligned}
\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}} &\leq \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\{(s,r) \,|\, ((t,s),(s,r)) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\}\\
&\leq \#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}} \leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}} \min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}\}
\end{aligned}$$

und analog mit (6.3)

$$\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}} \leq \#\mathcal{T}_{\mathcal{J}\times\mathcal{K}}\, c_{sp,\mathcal{I}\times\mathcal{J}} \leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}} \min\{\#\mathcal{T}_{\mathcal{J}}, \#\mathcal{T}_{\mathcal{K}}\}.$$

Wir nehmen das Minimum der beiden Abschätzungen und erhalten die Behauptung.
(6.5): Mit (6.2) erhalten wir

$$\begin{aligned}\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}\times\mathbf{s}} &= \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}\times\mathbf{s}} \\ &\leq c_{sp,\mathcal{J}\times\mathcal{K}} \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{t}\times\mathbf{s}}.\end{aligned}$$

Wir schreiben den Teilbaum als Nachfahren der Wurzel und nutzen aus, dass ein Block $(\check{t},\check{s})$ genau dann Nachfahre von $(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ist, wenn (t,s) Vorfahre von $(\check{t},\check{s})$ ist. Damit können wir die Summen entsprechend umsortieren und erhalten

$$\begin{aligned}\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}\times\mathbf{s}} &= c_{sp,\mathcal{J}\times\mathcal{K}} \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \sum_{(\check{t},\check{s})\in\operatorname{desc}(t,s)} 1 \\ &= c_{sp,\mathcal{J}\times\mathcal{K}} \sum_{(\check{t},\check{s})\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \sum_{(t,s)\in\operatorname{pred}(\check{t},\check{s})} 1 \\ &\leq c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{I}\times\mathcal{J}}\, \#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}.\end{aligned}$$

Mit Lemma 3.3.6 und $p_{\mathcal{I}\times\mathcal{J}} \leq \max\{p_{\mathcal{I}}, p_{\mathcal{J}}\}$ folgt

$$\begin{aligned}\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}\times\mathbf{s}} &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{I}\times\mathcal{J}}\, \min\{\#\mathcal{T}_{\mathcal{I}},\, \#\mathcal{T}_{\mathcal{J}}\} \\ &\leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, (p_{\mathcal{I}}\, \#\mathcal{T}_{\mathcal{I}} + p_{\mathcal{J}}\, \#\mathcal{T}_{\mathcal{J}}).\end{aligned}$$

Analog können wir mit (6.3) die Summe über $\#\mathcal{T}_{\mathbf{s}\times\mathbf{r}}$ abschätzen und erhalten die Aussage.
(6.6): Wie im Beweis von (6.5) folgt mit (6.2)

$$\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathcal{T}_{\mathbf{t}} \leq c_{sp,\mathcal{J}\times\mathcal{K}} \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{t}}.$$

Mit der Schwachbesetztheit von $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und einer ähnlichen Umsortierung von Summen wie beim Beweis von (6.5) erhalten wir

$$\begin{aligned}\sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{t}} &\leq c_{sp,\mathcal{I}\times\mathcal{J}} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\mathcal{T}_{\mathbf{t}} = c_{sp,\mathcal{I}\times\mathcal{J}} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \sum_{\check{t}\in\operatorname{desc}(t)} 1 \\ &= c_{sp,\mathcal{I}\times\mathcal{J}} \sum_{\check{t}\in\mathcal{T}_{\mathcal{I}}} \sum_{t\in\operatorname{pred}(\check{t})} 1 \leq c_{sp,\mathcal{I}\times\mathcal{J}} \sum_{\check{t}\in\mathcal{T}_{\mathcal{I}}} p_{\mathcal{I}} = c_{sp,\mathcal{I}\times\mathcal{J}} p_{\mathcal{I}} \#\mathcal{T}_{\mathcal{I}}.\end{aligned}$$

Die Summen der $\#\mathcal{T}_{\mathbf{s}}$ bzw. $\#\mathcal{T}_{\mathbf{r}}$ über $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ können wir auf die gleiche Art mit (6.3) abschätzen und erhalten so die Behauptung.
(6.7): Wie beim Beweis von (6.6) folgt mit (6.2)

$$\sum_{((t,s),(s,r))\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}} \#\mathbf{t} \leq c_{sp,\mathcal{J}\times\mathcal{K}} \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathbf{t} \leq c_{sp,\mathcal{I}\times\mathcal{J}}\, c_{sp,\mathcal{J}\times\mathcal{K}}\, p_{\mathcal{I}}\, \#\mathcal{I},$$

wobei die zweite Ungleichung nach Lemma 3.3.6 gilt. Die Aussage folgt mit (6.3) per analoger Abschätzung für die Summe der $\#\mathbf{s}$ bzw. $\#\mathbf{r}$ über $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$. ∎

Weil die Elemente im Produktbaum genau den Aufrufen in Algorithmus 6.2.2 inklusive der Aufrufe von Algorithmus 6.2.1 entsprechen, können wir mit Lemma 6.2.3 den Aufwand für die $\mathcal{H}^2$-Matrix-Multiplikation berechnen. In Theorem 6.2.4 schätzen wir den Aufwand für den Algorithmus ab.

Theorem 6.2.4
Es seien $H_A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_A, W_A, N_A, S_A)$, $H_B = \mathcal{H}^2(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}, V_B, W_B, N_B, S_B)$ und $H_C = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{K}}, V_C, W_C, N_C, S_C)$ mit $c_{sp,A}$-, $c_{sp,B}$- bzw. $c_{sp,C}$-schwachbesetzten und strikt zulässigen Blockbäumen. Alle lokalen Ränge während des Algorithmus 6.2.2 seien durch $k_{\max}$ beschränkt, $n_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\mathbf{t}, \max_{s\in\mathcal{T}_\mathcal{J}} \#\mathbf{s}, \max_{r\in\mathcal{T}_\mathcal{K}} \#\mathbf{r}\}$ sei die maximale Größe eines Blattclusters und

$$c_{\text{chil}} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\operatorname{chil}(t), \max_{s\in\mathcal{T}_\mathcal{J}} \#\operatorname{chil}(s), \max_{r\in\mathcal{T}_\mathcal{K}} \#\operatorname{chil}(r)\}$$

sei die maximale Anzahl von Söhnen eines Clusters. Wir definieren

$$\begin{aligned}
c_1 &:= (8c_{apr}+32)c_{\text{chil}}^3 k_{\max}^3, \quad c_2 := \max\{2k_{\max}^2, 2k_{\max}n_{\max}, (4c_{qr}+8)c_{\text{chil}}^3 k_{\max}^2\},\\
c_3 &:= \max\{2k_{\max}^3, 2n_{\max}^3\}, \quad c_4 := (12+4c_{qr}+4c_{apr})k_{\max}^2 \quad \text{und}\\
c_5 &:= (33+17c_{qr}+9c_{qr}c_{sp}+9c_{\|.\|}c_{sp,C}+72c_{sp,C}+4c_{apr})\max\{k_{\max}^3, n_{\max}^3\}.
\end{aligned}$$

Dann ist der Aufwand für die Aufrufe von Algorithmus 6.2.1 innerhalb von Algorithmus 6.2.2 höchstens

$$\begin{aligned}
&c_1 c_{sp,A}c_{sp,B}\min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}, \#\mathcal{T}_\mathcal{K}\} \;+\; c_2 c_{sp,A}c_{sp,B}(p_\mathcal{I}\#\mathcal{I} + 2p_\mathcal{J}\#\mathcal{J} + p_\mathcal{K}\#\mathcal{K})\\
&\qquad +\; 2c_3 c_{sp,A}c_{sp,B}(p_\mathcal{I}\#\mathcal{T}_\mathcal{I} + 2p_\mathcal{J}\#\mathcal{T}_\mathcal{J} + p_\mathcal{K}\#\mathcal{T}_\mathcal{K})
\end{aligned}$$

und der Aufwand für die Niedrigrangupdates in Algorithmus 6.2.2 höchstens

$$c_4 c_{sp,A}c_{sp,B}(p_\mathcal{I}\#\mathcal{I} + p_\mathcal{K}\#\mathcal{K}) \;+\; c_5 c_{sp,A}c_{sp,B}(p_\mathcal{I}\#\mathcal{T}_\mathcal{I} + p_\mathcal{K}\#\mathcal{T}_\mathcal{K}).$$

Beweis: [vergleiche [17, Theorem 6]] Zuerst betrachten wir die Aufrufe von Algorithmus 6.2.1. Dabei wollen wir jeweils den Aufwand für $((t,s),(s,r)) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ ohne die rekursiven Aufrufe abschätzen und betrachten dazu die einzelnen Fälle.
1. Fall: Es sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Dann wird Algorithmus 6.1.10 aufgerufen und benötigt nach Lemma 6.1.10 maximal

$$\begin{aligned}
&2k_{\max}^2(\#\mathbf{t} + 2\#\mathbf{s} + \#\mathbf{r}) \;+\; 2k_{\max}^3(\#\mathcal{T}_\mathbf{t} + 2\#\mathcal{T}_\mathbf{s} + \#\mathcal{T}_\mathbf{r})\\
&\qquad\qquad +\; 2\max\{n_{\max}^2, k_{\max}^2\}k_{\max}\#\mathcal{T}_{\mathbf{s}\times\mathbf{r}}
\end{aligned}$$

Operationen.
2. Fall: Es sei $(t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$. Dann wird Algorithmus 6.1.7 aufgerufen. Der Aufwand für

diesen ist nach Lemma 6.1.7 wegen $t \in \mathcal{L}_{\mathcal{I}}$ beschränkt durch

$$\begin{aligned}&\big(2k_{\max}(\#\mathbf{s}+\#\mathbf{r}) \;+\; 2k_{\max}^2(\#\mathcal{T}_{\mathbf{s}}+\#\mathcal{T}_{\mathbf{r}}) \;+\; 2\max\{n_{\max}^2,k_{\max}^2\}\#\mathcal{T}_{\mathbf{s}\times\mathbf{r}}\big)\;\#\mathbf{t}\\ &\leq\; 2k_{\max}n_{\max}(\#\mathbf{s}+\#\mathbf{r}) \;+\; 2k_{\max}^2n_{\max}(\#\mathcal{T}_{\mathbf{s}}+\#\mathcal{T}_{\mathbf{r}}) \;+\; 2\max\{n_{\max}^2,k_{\max}^2\}n_{\max}\#\mathcal{T}_{\mathbf{s}\times\mathbf{r}}.\end{aligned}$$

3. Fall: Es sei $(s,r) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Dann gilt analog zum 1. Fall die Schranke

$$\begin{aligned}2k_{\max}^2(\#\mathbf{t}+2\#\mathbf{s}+\#\mathbf{r}) \;+\; 2k_{\max}^3(\#\mathcal{T}_{\mathbf{t}}+2\#\mathcal{T}_{\mathbf{s}}+\#\mathcal{T}_{\mathbf{r}})\\ +\; 2\max\{n_{\max}^2,k_{\max}^2\}k_{\max}\#\mathcal{T}_{\mathbf{t}\times\mathbf{s}}.\end{aligned}$$

4. Fall: Es sei $(s,r) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$. Analog zum 2. Fall folgt wegen $r \in \mathcal{L}_{\mathcal{K}}$ für den Aufwand die Schranke

$$2k_{\max}n_{\max}(\#\mathbf{t}+\#\mathbf{s}) \;+\; 2k_{\max}^2n_{\max}(\#\mathcal{T}_{\mathbf{t}}+\#\mathcal{T}_{\mathbf{s}}) \;+\; 2\max\{n_{\max}^2,k_{\max}^2\}n_{\max}\#\mathcal{T}_{\mathbf{t}\times\mathbf{s}}.$$

5. Fall: Es sei $(t,s) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $(s,r) \notin \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$. Es seien $t' \in \operatorname{chil}^*(t)$, $s' \in \operatorname{chil}^*(s)$ und $r' \in \operatorname{chil}^*(r)$. Dann wird zuerst der Algorithmus 6.2.1 rekursiv für (t',s',r') aufgerufen und danach werden die Matrizen $\widehat{X}$ und $\widehat{Y}$ aufgestellt. Dabei werden für den aktuellen Aufruf des Algorithmus keine Operationen benötigt. Anschließend addiert Algorithmus 2.4.5 $(\widehat{X},\widehat{Y})$ zu der Niedrigrangmatrix R in weniger als

$$(c_{qr}+2)4k_{\max}^2(\#\mathbf{t}+\#\mathbf{r})+(c_{apr}+4)8k_{\max}^3$$

Operationen (siehe Lemma 2.4.13). Damit können wir den Aufwand im fünften Fall durch

$$\#\operatorname{chil}^*(t)\#\operatorname{chil}^*(s)\#\operatorname{chil}^*(r)\,\big((c_{qr}+2)4k_{\max}^2(\#\mathbf{t}+\#\mathbf{r})+(c_{apr}+4)8k_{\max}^3\big)$$

beschränken. Also werden in jedem Block des Produktbaums höchstens

$$c_1 \;+\; c_2(\#\mathbf{t}+2\#\mathbf{s}+\#\mathbf{r}) \;+\; c_3(\#\mathcal{T}_{\mathbf{t}}+2\#\mathcal{T}_{\mathbf{s}}+\#\mathcal{T}_{\mathbf{r}}) \;+\; c_3(\#\mathcal{T}_{\mathbf{t}\times\mathbf{s}}+\#\mathcal{T}_{\mathbf{t}\times\mathbf{s}})$$

Operationen benötigt, um die Niedrigrangmatrizen aufzustellen. Mit Lemma 6.2.3 folgt die erste Aufwandsabschätzung für den Aufwand des Aufstellens der Niedrigrangmatrizen. Für den Aufwand der Niedrigrangupdates gehen wir wie bei der ersten Abschätzung vor. Der Algorithmus 6.2.2 ruft den Algorithmus 5.3.8 für das Niedrigrangupdate höchstens einmal für jeden Block des Produktbaums auf. Der Aufwand für solch einen Aufruf für einen Block $((t,s),(r,s)) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ mit $(t,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ ist nach Theorem 5.3.20 durch

$$\begin{aligned}&(12+4c_{qr}+4c_{apr})k_{\max}^2(\#\mathbf{t}+\#\mathbf{r})\\ &\quad+\; (33+17c_{qr}+9c_{qr}c_{sp}+9c_{\|.\|}c_{sp}+72c_{sp}+4c_{apr})k_{\max}^3(\#\mathcal{T}_{\mathbf{t}}+\#\mathcal{T}_{\mathbf{r}})\\ &\quad+\; 2c_{sp}\max\{k_{\max}^2,\,n_{\max}^2\}k_{\max}(\#\mathcal{T}_{\mathbf{t}}+\#\mathcal{T}_{\mathbf{s}})\\ &\leq\; c_4(\#\mathbf{t}+\#\mathbf{s})+c_5(\#\mathcal{T}_{\mathbf{t}}+\#\mathcal{T}_{\mathbf{s}})\end{aligned}$$

beschränkt. Mit Lemma 6.2.3 folgt die Behauptung. ∎

Im Theorem 6.2.4 haben wir versucht, die Abschätzung relativ genau zu halten, um eine Einschätzung der unterschiedlichen Teile des Algorithmus am Aufwand zu ermöglichen. Dabei ist es besonders wichtig, dass die Konstanten c_1, c_2, c_3 und c_4 unabhängig von den c_{sp}-Konstanten sind. Nur c_5 beinhaltet noch die Konstante $c_{sp,C}$. Unter gewissen Annahmen können wir die Aufwandsabschätzung kürzer fassen.

Bemerkung 6.2.5
Es existieren zwei Konstanten $\tilde{c}_1$ und $\tilde{c}_2$ derart, dass für alle Probleme, die die Annahmen aus Bemerkung 3.4.7 erfüllen, der Aufwand für die Berechnung der Niedrigrangmatrizen in Algorithmus 6.2.2 maximal

$$\tilde{c}_1 c_{sp,A} c_{sp,B} k_{\max}^2 (\log(\mathcal{I})\#\mathcal{I} + \log(\mathcal{J})\#\mathcal{J} + \log(\mathcal{K})\#K)$$

Operationen und die Berechnung der Niedrigrangupdates höchstens

$$\tilde{c}_2 c_{sp,A} c_{sp,B} c_{sp,C} k_{\max}^2 (\log(\mathcal{I})\#\mathcal{I} + \log(\mathcal{J})\#\mathcal{J} + \log(\mathcal{K})\#K)$$

Operationen benötigt. Die Aufwand für die Matrix-Matrix-Multiplikation liegt somit in $\mathcal{O}(k_{\max}^2(\log(n)n))$.

Für die Berechnung des Niedrigrangupdates steht in der Aufwandsabschätzung in Theorem 6.2.4 ein zusätzlichen Faktor $c_{sp,C}$ im Vergleich zum Aufstellen der Niedrigrangmatrizen. Insbesondere bei dreidimensionalen Problemen können die c_{sp}-Konstanten groß werden. In solchen Fällen führt der zusätzliche Faktor $c_{sp,C}$ zu einem deutlichen Mehraufwand. Dies motiviert den Ansatz in Kapitel 7, bei dem wir das Niedrigrangupdate nur im Teilbaum $\mathcal{T}_{\tilde{\mathfrak{b}}}$ ohne die Verwendung der Blockzeilen durchführen.
Die Berechnung der Multiplikation ist approximativ. Wir verzichten an dieser Stelle auf eine detaillierte Fehleranalyse. Allerdings können wir den Fehler für jedes Niedrigrangupdate und das Kürzen der Niedrigrangmatrizen im Algorithmus 6.2.1 jeweils beschränken. Im Gegensatz zur Fehleranalyse für das Niedrigrangupdate können wir dabei keine Orthogonalitäten zwischen den einzelnen Fehlern erwarten. Allerdings kann der Gesamtfehler einfach per Dreiecksungleichung durch die Summe der einzelnen Fehler abgeschätzt werden.
Den Aufwand für die Inversion und Dreieckszerlegungen wollen wir später gegen den Aufwand für die Matrix-Matrix-Multiplikation abschätzen. Dieses Vorgehen wird auch in [26] für die Inversion von $\mathcal{H}$-Matrizen und [17] für die LR-Zerlegung von $\mathcal{H}^2$-Matrizen verwendet. Weil die adaptiven Ränge sich während der Matrix-Matrix-Multiplikation wesentlich von den Rängen während der Inversion oder der Dreieckszerlegung unterscheiden können, vergleichen wir den Aufwand zur Matrix-Matrix-Multiplikation mit konstantem Rang. Da der Aufwand monoton wachsend von den lokalen Rängen abhängt, können wir den Aufwand für den maximalen lokalen Rang als konstanten Rang als obere Schranke verwenden.

Definition 6.2.6
Es seien c_{sp}-schwachbesetzte Blockbäume $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$, $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ und ein lokaler Rang $k \in \mathbb{N}$ gegeben. Dann bezeichne $\mathcal{N}_{MMM}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathcal{T}_{\mathcal{J}\times\mathcal{K}}, \mathcal{T}_{\mathcal{I}\times\mathcal{K}}, k)$ den Aufwand für die Matrix-Matrix-Multiplikation mit Algorithmus 6.2.2 für $\mathcal{H}^2$-Matrizen zu den Blockbäumen $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ mit konstantem lokalem Rang k, wobei während des Algorithmus der adaptive Rang immer gleich k gewählt werde.

Bemerkung 6.2.7
Die Matrix-Matrix-Multiplikation für $\mathcal{H}^2$-Matrizen zu Blockbäumen $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ ist durch $\mathcal{N}_{MMM}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathcal{T}_{\mathcal{J}\times\mathcal{K}}, \mathcal{T}_{\mathcal{I}\times\mathcal{K}}, k_{\max})$ beschränkt, wobei $k_{\max}$ wie üblich den maximalen lokalen Rang bezeichnet. Diese Schranke gilt, weil der Aufwand monoton wachsend im lokalen Rang ist.

6.3 Inversion von $\mathcal{H}^2$-Matrizen

In diesem Abschnitt beschreiben wir die Berechnung der Inversen für $\mathcal{H}^2$-Matrizen. Dabei verwenden wir eine Darstellung der Inversen mit Hilfe des Schur-Komplements, wie sie in [31] und [26] für $\mathcal{H}$-Matrizen oder in [12, Section 10.5] für $\mathcal{H}^2$-Matrizen verwendet wird. Die Existenz der Darstellung der Inversen als hierarchische Matrizen ist in verschiedenen Veröffentlichungen untersucht. Für elliptische partielle Differentialgleichung ist in [3] gezeigt, dass die Inverse gewisser FEM-Steifigkeitsmatrizen sich als $\mathcal{H}$-Matrix darstellen lassen. Eine Erweiterung des Ergebnisses auch für $\mathcal{H}^2$-Matrizen ist in [11] bewiesen. Beide Arbeiten verwenden den kontinuierlichen Lösungsoperator. In [21] wird der diskrete Lösungsoperator untersucht, um das Ergebnis für $\mathcal{H}$-Matrizen zu beweisen. Auf ähnliche Weise ist in [20] die Approximierbarkeit der Inversen des diskreten Einfachschichtpotentials für BEM gezeigt. Diese Aussagen bilden die theoretische Grundlage zur Berechnung der Inversen mit $\mathcal{H}^2$-Matrizen.
Wir gehen im Folgenden davon aus, dass die vollbesetzten Matrizen, soweit notwendig, invertierbar sind. Insbesondere nehmen wir an, dass alle Diagonalblöcke des Blockbaums unzulässig sind. Für das Schur-Komplement $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ eines Blocks mit 2×2 Kindern erhalten wir

$$\begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} := \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}^{-1} = \begin{pmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}S^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}S^{-1} \\ -S^{-1}A_{21}A_{11}^{-1} & S^{-1} \end{pmatrix}.$$

Daraus ergeben sich folgende Schritte für die Berechnung der Inversen:

(i) Berechne $B_{11} \leftarrow (A_{11})^{-1}$

(ii) Berechne die Hilfsgrößen $H_{12} \leftarrow (A_{11})^{-1}A_{12} = B_{11}A_{12}$ und $H_{21} \leftarrow A_{21}(A_{11})^{-1} = A_{21}B_{11}$

(iii) Stelle das Schur-Komplement $A_{22} \leftarrow S = A_{22} - A_{21}A_{11}^{-1}A_{12} = A_{22} - A_{21}H_{12}$ auf

(iv) Berechne $B_{22} \leftarrow (A^{-1})_{22} = S^{-1}$

(v) Berechne $B_{12} \leftarrow (A^{-1})_{12} = -A_{11}^{-1}A_{12}S^{-1} = -H_{12}B_{22}$ und
$B_{21} \leftarrow (A^{-1})_{21} = -S^{-1}A_{21}A_{11}^{-1} = -B_{22}H_{21}$

(vi) Berechne $B_{11} \leftarrow (A^{-1})_{11} = (A_{11})^{-1} + (A_{11})^{-1}A_{12}S^{-1}A_{21}(A_{11})^{-1} = B_{11} - H_{12}B_{21}$

Also benötigen wir als arithmetische Operationen zusätzlich zu der Inversion nur die Matrix-Matrix-Multiplikationen in der in Abschnitt 6.2 vorgestellten Form. Zusätzlich müssen wir im Gegensatz zur Multiplikation die Kinder anordnen. Wir gehen deshalb im weiteren davon aus, dass die Menge der Kinder für alle Cluster eine Anordnung besitzt. Durch diese sind dann auch die Blöcke geordnet.

Bemerkung 6.3.1
Bei der Berechnung der Inversen für $\mathcal{H}$-Matrizen kann die ursprüngliche Matrix mit dem Ergebnis überschrieben werden. Bei unserem Ansatz der Arithmetik für $\mathcal{H}^2$-Matrizen würde dies ein Problem ergeben, weil der Block A_{11}^{-1} sich dann die Zeilenbasis mit A_{12} und die Spaltenbasis mit A_{21} teilen müsste. Dies kann zu höheren Rängen führen.
Für die Durchführbarkeit des Algorithmus problematischer ist allerdings die unterschiedliche Größenordnung der verschiedenen Teilmatrizen. Dies kann während der Approximation innerhalb des Niedrigrangupdates zu übermäßig großen Fehlern führen. In Experimenten können wir beobachten, dass dies für die 1. Strategie zu derart großen Fehlern führen kann, dass die Ergebnisse keine gute Approximation für die Inverse liefern. Deshalb speichern wir das Ergebnis in einer neuen Matrix B.

Weil wir uns nicht auf Clusterbäume mit jeweils zwei Kindern beschränken wollen, müssen wir uns überlegen, wie wir mit Clustern mit mehr als zwei Kindern umgehen. (Im Fall eines Kinds rufen wir die Funktion direkt rekursiv auf.) Es sei $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ und $\operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\}$. Dann definieren wir für alle $\ell \in \{1, \ldots, \tau\}$ die Teilmatrizen

$$\widetilde{A}_{\ell\ell} := \begin{pmatrix} A_{\ell\ell} & \ldots & A_{\ell\tau} \\ \vdots & \ddots & \vdots \\ A_{\tau\ell} & \ldots & A_{\tau\tau} \end{pmatrix}, \quad \widetilde{A}_{\ell *} := \begin{pmatrix} A_{\ell\ell} & \ldots & A_{\ell\tau} \end{pmatrix} \quad \widetilde{A}_{*\ell} := \begin{pmatrix} A_{\ell\ell} \\ \vdots \\ A_{\tau\ell} \end{pmatrix}$$

Analog definieren wir die Teilmatrizen für B. Dann können wir obigen Ansatz bis zur Berechnung des Schur-Komplements $\widetilde{S}_{22} := \widetilde{A}_{22} - \widetilde{A}_{21}(A_{11})^{-1}\widetilde{A}_{12}$ anwenden. Wie oben beschrieben speichern wir dieses in der Matrix $\widetilde{A}_{22}$, welche wir entsprechend Schritt (iv) invertieren müssen. Die Inverse berechnen wir, indem wir obigen Ansatz auf $\widetilde{A}_{22}$ anwenden.
Also führen wir die Schritte (i)-(iii) für $i = 1$ bis $\tau - 1$ und die Matrizen $\widetilde{A}_{\ell\ell}$ und $\widetilde{B}_{\ell\ell}$ mit der Blockunterteilung

$$\widetilde{A}_{\ell\ell} = \begin{pmatrix} A_{\ell\ell} & \widetilde{A}_{\ell *} \\ \widetilde{A}_{*\ell} & \widetilde{A}_{(\ell+1)(\ell+1)} \end{pmatrix} \quad \text{und} \quad \widetilde{B}_{\ell\ell} = \begin{pmatrix} B_{\ell\ell} & \widetilde{B}_{\ell *} \\ \widetilde{B}_{*\ell} & \widetilde{B}_{(\ell+1)(\ell+1)} \end{pmatrix}$$

durch, wobei $\widetilde{A}_{(\ell+1)(\ell+1)}$ jeweils mit dem Schur-Komplement überschrieben wird. Danach können wir Schritt (iv) für $\widetilde{A}_{\tau\tau} = A_{\tau\tau}$ und $\widetilde{B}_{\tau\tau} = B_{\tau\tau}$ per Rekursion berechnen. Schließlich

können wir für $\ell = \tau - 1$ bis 1 die Schritte (v) und (vi) für die Matrizen $\widetilde{A}_{\ell\ell}$ und $\widetilde{B}_{\ell\ell}$ mit obiger Blockunterteilung durch. Dieser Ansatz führt zu Algorithmus 6.3.1.

Algorithmus 6.3.1 Die Funktion berechnet die Inverse für eine $\mathcal{H}^2$-Matrix $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V, W, N, S)$. Die Matrix A wird mit Hilfsgrößen überschrieben und das Ergebnis wird in der $\mathcal{H}^2$-Matrix B zum Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ gespeichert, wobei diese mit 0 initialisiert sein muss.

```
function H2Matrix_Inversion(t, A, Z_A, B, Z_B, opt)
  if {t_1, ..., t_τ} := chil(t) ≠ ∅ then
    for ℓ = 1 to τ − 1 do
      H2Matrix_Inversion(t_ℓ, A, Z_A, B, Z_B, opt)
      for i ∈ {ℓ + 1, ..., τ} do
        H_{ℓ,i} ← H²(T_{t_ℓ × t_i}, 0, 0, 0, 0)
        H2Matrix_Produkt(t_ℓ, t_ℓ, t_i, B, A, H_{ℓ,i}, 0, opt)
        H_{i,ℓ} ← H²(T_{t_i × t_ℓ}, 0, 0, 0, 0)
        H2Matrix_Produkt(t_i, t_ℓ, t_ℓ, A, B, H_{i,ℓ}, 0, opt)
        for j ∈ {ℓ + 1, ..., τ} do
          H2Matrix_Produkt(t_i, t_ℓ, t_j, −H_{i,ℓ}, A, A, Z_A, opt)
        end for
      end for
    end for
    H2Matrix_Inversion(t_τ, A, Z_A, B, Z_B, opt)
    for ℓ = τ − 1 to 1 do
      for i ∈ {ℓ + 1, ..., τ} do
        for j ∈ {ℓ + 1, ..., τ} do
          H2Matrix_Produkt(t_i, t_j, t_ℓ, −B, H_{j,ℓ}, B, Z_B, opt)
          H2Matrix_Produkt(t_ℓ, t_j, t_i, −H_{ℓ,j}, B, B, Z_B, opt)
        end for
        H2Matrix_Produkt(t_ℓ, t_i, t_ℓ, −H_{ℓ,i}, B, B, Z_B, opt)
                                                  ▷ Algorithmus 6.2.2
      end for
    end for
  else                                  ▷ t ∈ L_I und (t, t) ∈ L⁻_{I×J}
    B ← A^{-1}
  end if
end function
```

Weil der Algorithmus 6.3.1 im Wesentlichen aus Matrix-Matrix-Multiplikationen besteht, schätzen wir den Aufwand für die Inversion gegen den Aufwand des Matrix-Produkts ab.

Theorem 6.3.2
Es sei $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V, W, N, S)$ und $(t,t) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$. Alle lokalen Ränge während des Aufrufs von Algorithmus 6.3.1 für A seien durch $k_{\max}$ beschränkt. Für alle $t \in \mathcal{L}_{\mathcal{I}}$ sei der Aufwand von Algorithmus 6.3.1 für eine vollbesetzte Matrix im Block

(t,t) kleiner als $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max})$. Dann ist der Aufwand für die Inversion von A mit Algorithmus 6.3.1 kleiner als $\mathcal{N}_{MMM}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, \mathcal{T}_{\mathcal{I}\times\mathcal{I}}, \mathcal{T}_{\mathcal{I}\times\mathcal{I}}, k_{\max})$.

Beweis: [vergleiche [26, Theorem 2.29]] Für den Beweis zeigen wir für alle $t \in \mathcal{T}_{\mathcal{I}}$ per Induktion über $\#\operatorname{desc}(t) \in \mathbb{N}_{>0}$, dass der Aufwand für die Inversion von $A_{|\mathbf{t}\times\mathbf{t}}$ mit Algorithmus 6.3.1 kleiner als $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max})$ ist. Es sei $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) = 1$. Dann gilt $t \in \mathcal{L}_{\mathcal{I}}$ und die Behauptung folgt nach Voraussetzung.
Es gelte die Aussage für alle $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) \leq n \in \mathbb{N}_{>0}$. Es sei $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) = n+1 > 1$. Dann gilt $t \notin \mathcal{L}_{\mathcal{I}}$ und für alle $\breve{t} \in \operatorname{chil}(t)$ ist $\#\operatorname{desc}(\breve{t}) < \#\operatorname{desc}(t) = n+1$. Also gilt die Induktionsbehauptung für alle $\breve{t} \in \operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\}$.
Um die Aussage für t zu beweisen, betrachten wir die rekursiven Aufrufe für die Algorithmen 6.2.2 und 6.3.1. Da $(t,t) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{I}}$ ist, gilt für die Multiplikation von $A_{\mathbf{t}\times\mathbf{t}}$ mit sich selbst

$$\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max}) = \sum_{i,j,k\in\{1,\ldots,\tau\}} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_j}, \mathcal{T}_{\mathbf{t}_j\times\mathbf{t}_k}, \mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_k}, k_{\max}).$$

Für die Inversion untersuchen wir, für welche Kombinationen von Clustern der Algorithmus 6.2.2 aufgerufen wird. In der ersten Schleife in Algorithmus 6.3.1 werden für alle $\ell \in \{1, \ldots, \tau-1\}$ die Multiplikation für die Cluster

$$\begin{aligned}\{(t_\ell, t_\ell, t_i) \,|\, i \in \{\ell+1, \ldots, \tau\}\} \;&\dot\cup\; \{(t_i, t_\ell, t_\ell) \,|\, i \in \{\ell+1, \ldots, \tau\}\} \\ &\dot\cup\; \{(t_i, t_\ell, t_j) \,|\, i,j \in \{\ell+1, \ldots, \tau\}\}\end{aligned}$$

berechnet. In der zweiten äußeren Schleife werden für alle $\ell \in \{\tau-1, \ldots, 1\}$ die Multiplikation für die Kombinationen

$$\begin{aligned}\{(t_i, t_j, t_\ell) \,|\, i,j \in \{\ell+1, \ldots, \tau\}\} \;&\dot\cup\; \{(t_\ell, t_j, t_i) \,|\, i,j \in \{\ell+1, \ldots, \tau\}\} \\ &\dot\cup\; \{(t_\ell, t_i, t_\ell) \,|\, i \in \{\ell+1, \ldots, \tau\}\}\end{aligned}$$

berechnet. Zusätzlich wird für alle $\ell \in \{1, \ldots, \tau\}$ der Algorithmus 6.3.1 aufgerufen. Aufgrund der Induktionsvoraussetzung können wir den Aufwand für diese Aufrufe für alle $\ell \in \{1, \ldots, \tau\}$ gegen $c\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, k_{\max})$ abschätzen. Damit folgt für die Inversion im Cluster t die Aufwandsabschätzung

$$\sum_{i,j,k\in\{1,\ldots,\tau\}} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_j}, \mathcal{T}_{\mathbf{t}_j\times\mathbf{t}_k}, \mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_k}, k_{\max}) = \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max}).$$

■

Bemerkung 6.3.3
Indem wir die Inversion der vollbesetzten Diagonalblöcke in Algorithmus 6.3.1 mit Hilfe der LR-Zerlegung und des Vorwärts- und Rückwärtseinsetzens berechnen, können wir die Bedingung in Theorem 6.3.2 erfüllen (siehe [42, Chapter 3]).
Aus Theorem 6.3.2 folgt zusammen mit Theorem 6.2.4 eine Aufwandsabschätzung für die Inversion mit Algorithmus 6.3.1.

Das Theorem 6.3.2 legt einen engen Zusammenhang zwischen dem Aufwand für die Inversion und die Matrix-Multiplikation nahe. Für Berechnungen mit adaptiven Rängen, die typischerweise genutzt werden, können allerdings die lokalen Ränge und somit auch der Aufwand stark voneinander abweichen.

6.4 Vorwärtseinsetzen für $\mathcal{H}^2$-Matrizen

Nachdem wir die Inversion für $\mathcal{H}^2$-Matrizen untersucht haben, wollen wir als Nächstes Dreieckszerlegungen von $\mathcal{H}^2$-Matrizen untersuchen. Weil sich die Dreieckszerlegungen typischerweise schneller als die Inverse berechnen lassen und sich auch das Lösen per Vorwärts- und Rückwärtseinsetzen in den meisten Fällen in einem ähnlichen Aufwand wie die Matrix-Vektor-Multiplikation durchführen lässt, werden approximative Dreieckszerlegungen häufig als Vorkonditionierer verwendet. Da wir für den Begriff der Dreiecksmatrix eine Ordnung benötigen, definieren wir $\mathcal{H}^2$-Matrizen mit Dreiecksgestalt.

Definition 6.4.1
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix. Für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$ sei die Menge der Kinder $\operatorname{chil}(t)$ angeordnet und für alle $t \in \mathcal{L}_{\mathcal{I}}$ sei die Menge $\mathbf{t}$ angeordnet. Dann sagen wir, dass H *untere Dreiecksgestalt* besitzt, falls

(i) für alle $t \in \mathcal{T}_{\mathcal{I}}$ der zugehörige Diagonal-Block unzulässig ist, d.h. $(t,t) \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}}\backslash\mathcal{L}^+_{\mathcal{I}\times\mathcal{I}}$,

(ii) für alle $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$ mit $\operatorname{chil}(t) = \{t_1, \ldots, t_\tau\}$ die Blöcke oberhalb der Diagonalen gleich Null sind, d.h. $H_{|\mathbf{t}_i\times\mathbf{t}_j} = 0$ für alle $\{(t_i, t_j) \in \operatorname{chil}(t,t) \,|\, i < j\}$, und

(iii) für alle $t \in \mathcal{L}_{\mathcal{I}}$ der Blatt-Block untere Dreiecksgestalt hat, d.h. $L_{i,j} = 0$ für alle $\{(i,j) \in \mathbf{t} \times \mathbf{t} \,|\, i < j\}$.

Analog definieren wir die *obere Dreiecksgestalt* für $\mathcal{H}^2$-Matrizen.

Die Definition ist an die späteren Algorithmen angelehnt. In folgender Bemerkung stellen wir die Verbindung zu Dreiecksmatrizen im üblichen Sinne her.

Bemerkung 6.4.2
Durch die Bedingungen aus Definition 6.4.1 können wir eine Anordnung der Indexmenge $\mathcal{I}$ konstruieren, bezüglich der eine $\mathcal{H}^2$-Matrix, die untere Dreiecksgestalt besitzt, eine untere Dreiecksmatrix repräsentiert.

Um $\mathcal{H}^2$-Matrizen in Dreiecksgestalt effizient darzustellen, müssen wir die Nullblöcke speziell behandeln.

Bemerkung 6.4.3
Bei $\mathcal{H}$-Matrizen werden Null-Blöcke typischerweise mit Rang null dargestellt. Damit erfordern diese Blöcke (fast) keinen Aufwand für die Speicherung und für die Arithmetik.

Bei $\mathcal{H}^2$-Matrizen ist der Rang durch die Clusterbasen gegeben. Also wird für einen Null-Block $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ bei einfacher Implementierung die Kopplungsmatrix $S_{(t,s)} = 0 \in \mathbb{R}^{\kappa_t \times \lambda_s}$ gespeichert. Bei diesem Ansatz erfordert der Nullblock genauso viel Speicher- und Rechenaufwand wie ein Nicht-Null-Block. Deshalb sollte bei der Implementierung darauf geachtet werden, dass die Null-Blöcke markiert sind, damit die Kopplungsmatrizen $S_{(t,s)}$ nicht gespeichert werden müssen und bei Algorithmen ignoriert werden können.

Bevor wir Dreieckszerlegungen für $\mathcal{H}^2$-Matrizen untersuchen, betrachten wir in diesem Abschnitt das Vorwärtseinsetzen $Lx = y$, wobei L eine quadratische untere Dreiecksmatrix, y ein gegebener Vektor und x der gesuchte Vektor ist. Dies benötigen wir nicht nur für das Lösen eines Gleichungssystems in Dreiecksgestalt, sondern auch während der Berechnung der Dreieckszerlegungen in Abschnitt 6.5. Dabei beschreiben wir den Ansatz aus [12, Abschnitt 10.3].
In einem Diagonalblock (t,t) der $\mathcal{H}^2$-Matrix L in unterer Dreiecksgestalt mit $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ und $\operatorname{chil}(t) = \{t_1, t_2\}$ erhalten wir die Gleichung

$$\begin{pmatrix} L_{11}x_1 \\ L_{21}x_1 + L_{22}x_2 \end{pmatrix} = \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_1 \end{pmatrix}.$$

Eine direkte Umsetzung dieses Ansatzes führt zu den Schritten:

(i) Berechne $L_{11}x_1 = y_1$ rekursiv

(ii) Berechne das Produkt $y_2 \leftarrow y_2 - L_{21}x_1$

(iii) Berechne $L_{22}x_2 = y_2 - L_{21}x_1$ rekursiv

Dabei müsste auf jedem Level die Matrix-Vektor-Multiplikation mit linearem Aufwand ausgeführt werden. Dies führt zu einem suboptimalen log-linearen Gesamtaufwand für das Vorwärtseinsetzen. Der Effekt entsteht durch die mehrfache Transformation der Vektoren mit den Clusterbasen. Um dies zu vermeiden, verwenden wir ähnlich zur Matrix-Vektor-Multiplikation Familien von Hilfsvektoren derart, dass wir für den aktuellen Cluster $t \in \mathcal{T}_\mathcal{I}$ den Vektor $y_{|\mathbf{t}}$ in der geschachtelten Darstellung

$$y_{|\mathbf{t}} = y_{0|\mathbf{t}} + \sum_{\check{t}\in\operatorname{desc}(t)} V_{\check{t}} y_{\check{t}}$$

speichern und für den Ergebnisvektor zusätzlich zu x auch die Hilfsvektoren $x_{\check{t}} := W_{\check{t}}^T x$ für alle $\check{t} \in \operatorname{desc}(t)$ berechnen.
Für den Fall, dass $t \in \mathcal{L}_\mathcal{I}$ ein Blatt ist, können wir $y_{|\mathbf{t}} = y_{0|\mathbf{t}} + V_t y_t$ direkt auswerten. Anschließend berechnen wir $L_{\mathbf{t}\times\mathbf{t}} x_{|\mathbf{t}} = y_{|\mathbf{t}}$ per Vorwärtseinsetzen für vollbesetzte Matrizen. Um den oben beschriebenen Ansatz für die Rekursion zu verwenden, müssen wir dann noch den Hilfsvektor $x_t = W_t^T x = W_t^T x_{|\mathbf{t}}$ berechnen.
Falls $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ kein Blatt ist, gehen wir rekursiv vor. Bevor wir die Funktion rekursiv aufrufen, müssen wir die geschachtelte Darstellung für $y_{|\mathbf{t}_1}$ berechnen. Dank der

geschachtelten Darstellung der Clusterbasen erhalten wir für y_{t_1} die Darstellung

$$y_{|\mathbf{t}_1} \;=\; y_{0|\mathbf{t}_1} + \sum_{\check{t}\in\operatorname{desc}(t)} V_{\check{t}|\mathbf{t}_1} y_{\check{t}} \;=\; y_{0|\mathbf{t}_1} + V_{t_1} E_{t_1} y_t + \sum_{\check{t}\in\operatorname{desc}(t_1)} V_{\check{t}} y_{\check{t}}.$$

Wir müssen also lediglich $E_{t_1} y_t$ zu y_{t_1} addieren. Dann können wir für das Lösen von $L_{|\mathbf{t}_1\times\mathbf{t}_1} x_{|\mathbf{t}_1} = y_{|\mathbf{t}_1}$ den Algorithmus für das Vorwärtseinsetzen rekursiv aufrufen. Bei diesem Aufruf wird nicht nur $x_{|\mathbf{t}_1}$ berechnet, sondern auch die Hilfsvektoren $x_{\check{t}}$ für alle $\check{t} \in \operatorname{desc}(t_1)$.

Algorithmus 6.4.1 [siehe [12, Algorithmus 58]] Die Funktion löst für eine $\mathcal{H}^2$-Matrix $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V_L, W_L, N_L, S_L)$ mit unterer Dreiecksgestalt $Lx = y$ per Vorwärtseinsetzen, wobei y der gegebene Vektor und x der gesuchten Vektor ist. Die Hilfsvektoren seien mit $y_t = 0 \in \mathbb{R}^{\kappa_t}$ und $x_t = 0 \in \mathbb{R}^{\lambda_t}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ initialisiert.

```
function H2Matrix_Vektor_Vorwärtseinsetzen(t, L, y, (y_ť)_{ť∈T_t}, x, (x_ť)_{ť∈T_t})
    if chil(t) = ∅ then
        y_|t ← y_|t + V_t y_t
        x_|t ← L^{-1}_{|t×t} y_|t
        x_t ← W_t^T x_|t
    else                                                    ▷ chil(t) ≠ ∅
        {t_1, ..., t_τ} ← chil(t)
        for i = 1 to τ do
            y_{t_i} ← y_{t_i} + E_{|t_i} y_t
            for j ∈ {1, ..., i − 1} do
                Mult_Block((t_i, t_j), −N, −S, x, (x_ť)_{ť∈T_{t_j}}, y, (y_ť)_{ť∈T_{t_i}})
                                                            ▷ Algorithmus 6.1.4
            end for
            H2Matrix_Vektor_Vorwärtseinsetzen(t_i, L, y, (y_ť)_{ť∈T_{t_i}}, x, (x_ť)_{ť∈T_{t_i}})
            x_t ← x_t + F_{t_i} x_{t_i}
        end for
    end if
end function
```

Im nächsten Schritt müssen wir $L_{|\mathbf{t}_2\times\mathbf{t}_1} x_{|\mathbf{t}_1}$ von $y_{|\mathbf{t}_2}$ abziehen. Weil wir $x_{|\mathbf{t}_1}$ zusammen mit den Hilfsvektoren $x_{\check{t}}$, $\check{t} \in \operatorname{desc}(t_1)$, berechnet haben und $y_{|\mathbf{t}_2}$ in der geschachtelten Darstellung speichern, können wir die Matrix-Vektor-Multiplikation mit Algorithmus 6.1.4 berechnen (vergleiche die Matrix-Vektor-Multiplikation in Abschnitt 6.1). Nach dem Aufruf haben wir

$$y_{|\mathbf{t}_2} - L_{|\mathbf{t}_2\times\mathbf{t}_1} x_{|\mathbf{t}_1} \;=\; y_{0|\mathbf{t}_2} + \sum_{\check{t}\in\operatorname{desc}(t_2)} V_{\check{t}} y_{\check{t}}$$

als rekursive Darstellung gespeichert und können $x_{|\mathbf{t}_2}$ samt der Hilfsvektoren $x_{\check{t}}$, $\check{t} \in \operatorname{desc}(t_2)$, per rekursiven Aufruf des Vorwärtseinsetzens für die Matrix $L_{|\mathbf{t}_2\times\mathbf{t}_2}$ berechnen.

Damit haben wir $x_{|\mathbf{t}}$ und $x_{\check{t}}$ für alle $t \in \operatorname{desc}(t)\backslash\{t\}$ und es bleibt x_t aufzustellen. Durch die geschachtelte Darstellung der Clusterbasis erhalten wir für $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ die Formel

$$x_t \;=\; W_t^T x \;=\; \sum_{\check{t}\in\operatorname{chil}(t)} F_{\check{t}}^T W_{\check{t}}^T x \;=\; \sum_{\check{t}\in\operatorname{chil}(t)} F_{\check{t}}^T x_{\check{t}}.$$

Dieses Vorgehen führt zu Algorithmus 6.4.1. Dabei speichern wir y_0 in y und initialisieren $y_t = 0 \in \mathbb{R}^{\kappa_t}$ und $x_t = 0 \in \mathbb{R}^{\lambda_t}$ für alle $t \in \mathcal{T}_\mathcal{I}$. Außerdem ist Algorithmus 6.4.1 derart formuliert, dass die Anzahl der Kinder eines Clusters auch ungleich zwei sein kann. Den Aufwand schätzen wir in Lemma 6.4.4 ab.

Lemma 6.4.4
Es sei $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit unterer Dreiecksgestalt. Dann ist der Aufwand von Algorithmus 6.4.1 höchstens so groß wie die Matrix-Vektor-Multiplikation für L mit Algorithmus 6.1.6.

Beweis: Die Operationen für die Hilfsvektoren x_t und y_t, $t \in \mathcal{T}_\mathcal{I}$, entsprechen der Vorwärts- und Rückwärtstransformation mit den Algorithmen 6.1.3 und 6.1.5 während der Matrix-Vektor-Multiplikation. Außerdem wird für alle Blöcke unterhalb der Diagonalen der Algorithmus 6.1.4 aufgerufen. Diese Aufrufe werden auch während der Matrix-Vektor-Multiplikation durch die Rekursion des für die Wurzel aufgerufenen Algorithmus 6.1.4 ausgeführt. In den Blättern auf der Diagonalen wird das Vorwärtseinsetzen für vollbesetzte Matrizen durchgeführt, welches höchstens so viele Operationen wie die Matrix-Vektor-Multiplikation benötigt (siehe [42, Chapter 3]). Damit folgt die Behauptung. ■

Mit Hilfe des Vorwärtseinsetzens für Vektoren können wir dies auch für Matrizen berechnen. Dabei gehen wir wie für die Matrix-Matrix-Multiplikation spaltenweise vor und erhalten den Algorithmus 6.4.2, dessen Aufwand wir in Lemma 6.4.5 abschätzen.

Algorithmus 6.4.2 Die Funktion berechnet $X = L^{-1}Y$ per Vorwärtseinsetzen für eine $\mathcal{H}^2$-Matrix L in unterer Dreiecksgestalt und eine Matrix $Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$.

```
function H2Matrix_Matrix_Vorwärtseinsetzen(L, Y, X)
    for j ∈ J do
        for t ∈ T_I do
            y_ť ← 0 ∈ R^{κ_t}
            x_ť ← 0 ∈ R^{λ_t}
        end for
        H2Matrix_Vektor_Vorwärtseinsetzen(r_I, L, y^j, (y_ť)_{ť∈T_I}, x^j, (x_ť)_{ť∈T_I})
                                                          ▷ Algorithmus 6.4.1
    end for
end function
```

Lemma 6.4.5
Es sei $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit unterer Dreiecksgestalt. Dann ist der Aufwand von Algorithmus 6.4.2 für ein $Y \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ höchstens so groß wie der Aufwand für die Matrix-Matrix-Multiplikation von L und Y mit Algorithmus 6.1.7.

Beweis: Weil der Aufwand von Algorithmus 6.4.2 nur durch die Aufrufe von Algorithmus 6.4.1 entsteht, folgt die Abschätzung aus Lemma 6.4.4. ∎

Das Ergebnis des Vorwärtseinsetzens einer uniformen Matrix wollen wir wie bei der Multiplikation als Niedrigrangmatrix speichern. Deshalb konvertieren wir die uniforme Matrix zuerst in eine Niedrigrangmatrix $R = AB^T$, wenden dann Algorithmus 6.4.2 auf den ersten Faktor an und erhalten $L^{-1}U = L^{-1}R = (L^{-1}A)B^T$. Damit erhalten wir Algorithmus 6.4.3, dessen Aufwand wir in Lemma 6.4.6 abschätzen.

Algorithmus 6.4.3 Die Funktion berechnet $R = L^{-1}U$ für eine $\mathcal{H}^2$-Matrix L in unterer Dreiecksgestalt und eine uniforme Matrix $U = V_U S_U W_U^T$, wobei das Ergebnis als Niedrigrangmatrix $R = AB^T$ gespeichert wird.

```
function H2Matrix_Uniform_Vorwärtseinsetzen(L, U, R)
    Uniform_RkMatrix_Konvertieren(U, (Ã, B))      ▷ Algorithmus 6.1.9
    H2Matrix_Matrix_Vorwärtseinsetzen(L, Ã, A)    ▷ Algorithmus 6.4.2
    R ← (A, B)
end function
```

Lemma 6.4.6
Es sei $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V_L, W_L, S_L, N_L)$ eine $\mathcal{H}^2$-Matrix mit unterer Dreiecksgestalt. Weiter seien $(V_{U,s})_{s\in\mathcal{T}_\mathcal{I}}$ und $(W_{U,r})_{r\in\mathcal{T}_\mathcal{J}}$ Clusterbasen mit Rangverteilungen κ_U und λ_U, $S_U \in \mathbb{R}^{\kappa_{U,r_\mathcal{I}}\times\lambda_{U,r_\mathcal{J}}}$ und $U = V_{U,r_\mathcal{I}} S_{U,\mathcal{I}\times\mathcal{J}} W_{U,r_\mathcal{I}}^T$. Dann benötigt Algorithmus 6.4.3 für L und U maximal so viele Operationen wie die entsprechende Matrix-Matrix-Multiplikation mit Algorithmus 6.1.10.

Beweis: Zuerst wird in beiden Algorithmen der Algorithmus 6.1.9 aufgerufen. Danach ruft Algorithmus 6.4.3 den Algorithmus 6.4.2. Dieser benötigt nach Lemma 6.4.6 höchstens so viele Operationen, wie der von Algorithmus 6.1.10 aufgerufene Algorithmus 6.1.7. Damit folgt die Behauptung. ∎

Mit Hilfe dieser Funktionen können wir das Vorwärtseinsetzen für $\mathcal{H}^2$-Matrizen formulieren (vergleiche [17]). Dabei gehen wir davon aus, dass das Ergebnis X in der gleichen Struktur wie die Ausgangsmatrix Y dargestellt wird. Falls L und Y in 2×2 Kinder unterteilt sind, erhalten wir

$$\begin{aligned}\begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix} &= \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix} \\ &= \begin{pmatrix} L_{11}X_{11} & L_{11}X_{12} \\ L_{21}X_{11} + L_{22}X_{21} & L_{21}X_{12} + L_{22}X_{22} \end{pmatrix}.\end{aligned}$$

Weil beide Spalten analog zu behandeln sind, gehen wir nur auf die Erste ein. Für diese müssen wir folgende drei Schritte durchführen:

(i) Berechne per Vorwärtseinsetzen $X_{11} \leftarrow L_{11}^{-1} Y_{11}$

(ii) Berechne die Hilfsgröße $H_{21} \leftarrow Y_{21} - L_{21}X_{11}$

(iii) Berechne per Vorwärtseinsetzen $X_{21} \leftarrow L_{22}^{-1}H_{21} = L_{22}^{-1}(Y_{12} - L_{21}X_{11})$

Zusätzlich zu dem Fall, dass beide Blöcke (t,t) und (t,s) unterteilt sind, müssen wir die Fälle, dass einer der Blöcke ein Blatt ist, untersuchen.

Algorithmus 6.4.4 Die Funktion berechnet per Vorwärtseinsetzen $X = LY$ für gegebene $\mathcal{H}^2$-Matrizen $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V_L, W_L, N_L, S_L)$ und $Y = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_Y, W_Y, N_Y, S_Y)$, wobei L untere Dreiecksgestalt habe und $(L_{\mathbf{t}\times\mathbf{t}})^{-1}$ für alle $t \in \mathcal{L}_{\mathcal{I}}$ gegeben sei. Das Ergebnis wird als $\mathcal{H}^2$-Matrix $X = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_X, W_X, N_X, S_X)$ gespeichert, die als Nullmatrix initialisiert sei. Die Matrix Y wird teilweise mit Hilfsgrößen überschrieben.

---

function H2Matrix_Vorwärtseinsetzen(t, s, L, $((L_{\mathbf{t}\times\mathbf{t}})^{-1})_{t\in\mathcal{T}_{\mathcal{I}}}$, Y, Z_Y, X, Z_X, *opt*)
  if $(t,t) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{I}} \wedge (t,s) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ **then**
    $\{s_1, \ldots, s_\sigma\} \leftarrow \operatorname{chil}(s)^*$, $\{t_1, \ldots, t_\tau\} \leftarrow \operatorname{chil}(t)$
    for $i = 1$ **to** τ **do**
      for $j \in \{1, \ldots, \sigma\}$ **do**
        H2Matrix_Vorwärtseinsetzen(t_i, s_j, L, $((L_{\mathbf{t}\times\mathbf{t}})^{-1})_{t\in\mathcal{T}_{\mathcal{I}}}$, Y, Z_Y, X, Z_X, *opt*)
        for $\ell \in \{i+1, \ldots, \tau\}$ **do**
          H2Matrix_Produkt(t_ℓ, t_i, s_j, $-L$, X, Y, Z_Y, *opt*)
                ▷ Algorithmus 6.2.2
        end for
      end for
    end for
  else if $(t,t) \in \mathcal{L}_{\mathcal{I}\times\mathcal{I}}$ **then** ▷ $t \in \mathcal{L}_{\mathcal{I}}$, $(t,t) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{I}}$
    $B \leftarrow 0 \in \mathbb{R}^{\mathbf{s}\times\mathbf{t}}$
    H2Matrix_Matrix_Multiplikation($(Y_{\mathbf{t}\times\mathbf{s}})^T$, $(L_{\mathbf{t}\times\mathbf{t}})^{-T}$, B)
                ▷ Algorithmus 6.1.7
    Niedrigrangupdate((t,s), X, Z_X, $(I_{\mathbf{t}}, B)$, *opt*) ▷ Algorithmus 5.3.8
  else if $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ **then** ▷ $t \notin \mathcal{L}_{\mathcal{I}}$, $(t,s) \notin \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$
    H2Matrix_Uniform_Vorwärtseinsetzen(t, L, $(V_{Y,t}, S_{Y,(t,s)}, W_{Y,s})$, R)
                ▷ Algorithmus 6.4.3
    Niedrigrangupdate((t,s), X, Z_X, R, *opt*)
  end if
end function

---

1. Fall: Es sei $(t,t) \in \mathcal{L}_{\mathcal{I}\times\mathcal{I}}$. Weil wir annehmen, dass L untere Dreiecksgestalt besitzt, folgt $(t,t) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{I}}$. In der in [17] betrachteten vereinfachten Situation folgt in diesem Fall $(t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ und die Berechnung kann vollbesetzt durchgeführt werden. In der Praxis können wir dies typischerweise nicht sicherstellen, weil wir für $\mathcal{H}^2$-Matrizen strikt

zulässige Blockbäume verwenden. Um den Aufwand trotzdem gegen die Matrix-Matrix-Multiplikation abschätzen zu können, nehmen wir an, dass für den Block (t,t) die Inverse $(L_{\mathbf{t}\times\mathbf{t}})^{-1}$ bereits berechnet ist. Dann können wir ähnlich zur Matrix-Matrix-Multiplikation vorgehen. Wir berechnen zuerst $B := (Y_{\mathbf{t}\times\mathbf{s}})^T (L_{\mathbf{t}\times\mathbf{t}})^{-T}$ und danach addieren wir B als Niedrigrangmatrix $(I_{\mathbf{t}}, B)$ zu X mit Hilfe des Niedrigrangupdates aus Kapitel 5.
2. Fall: Es sei $(t,t) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{I}}$ und $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Dann ist $t \notin \mathcal{L}_{\mathcal{I}}$ und somit $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}} = \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. In diesem Fall berechnen wir $R := (L_{\mathbf{t}\times\mathbf{t}})^{-1} Y_{\mathbf{t}\times\mathbf{s}}$ mit Algorithmus 6.4.3 und addieren dann R mit Hilfe des Niedrigrangupdate zu X.
Damit sind alle Fälle behandelt, wobei wir bei der Rekursion vom Fall mit 2 Kindern wie bei der Inversion zum allgemeinen Fall übergehen. Daraus ergibt sich der rekursive Algorithmus 6.4.4 für die Berechnung der Vorwärtseinsetzens für Matrizen im $\mathcal{H}^2$-Matrix-Format. Den Aufwand für die Berechnungen schätzen wir in Lemma 6.4.7 gegen den Aufwand von entsprechenden Matrix-Produkten ab.

Lemma 6.4.7
Es seien $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V_L, W_L, N_L, S_L)$ und $Y = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_Y, W_Y, N_Y, S_Y)$ zwei $\mathcal{H}^2$-Matrizen, wobei L untere Dreiecksgestalt habe und $(L_{\mathbf{t}\times\mathbf{t}})^{-1}$ für alle $t \in \mathcal{L}_{\mathcal{I}}$ gegeben sei. Dann ist der Aufwand von Algorithmus 6.4.2, aufgerufen für L und Y, höchstens $\mathcal{N}_{MMM}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, k_{\max})$, wobei $k_{\max}$ den maximalen lokalen Rang während des Algorithmus 6.4.4 bezeichne.

Beweis: Wir zeigen für alle $t \in \mathcal{T}_{\mathcal{I}}$ per Induktion über $\#\operatorname{desc}(t)$, dass für alle $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ der Aufwand für Algorithmus 6.4.4 aufgerufen für die Teilmatrizen $L_{\mathbf{t}\times\mathbf{t}}$ und $Y_{\mathbf{t}\times\mathbf{s}}$ höchstens $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, k_{\max})$ ist.
Es sei $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) = 1$ und $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Dann gilt $t \in \mathcal{L}_{\mathcal{I}}$ und $(t,t) \in \mathcal{L}_{\mathcal{I}}$ und Algorithmus 6.4.4 ruft erst Algorithmus 6.1.7 und dann Algorithmus 5.3.8 auf. Damit ist in diesem Fall der Aufwand höchsten $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, k_{\max})$.
Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Induktionsbehauptung gilt. Weiter sei $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) = n+1 > 1$ und $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Dann ist $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$. Also gilt $\operatorname{desc}(\check{t}) \leq n$ und somit die Induktionsbehauptung für alle $\check{t} \in \{t_1, \ldots, t_\tau\} := \operatorname{chil}(t) \neq \emptyset$.
Wir unterscheiden die beiden Fälle, ob der Block (t,s) ein Blatt ist oder nicht. Falls der Block ein Blatt ist, muss $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ gelten, weil $t \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ gilt und somit $(t,s) \notin \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ folgt.
1. Fall: Es sei $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.
Dann wird erst Algorithmus 6.4.3 aufgerufen, der nach Lemma 6.4.6 weniger Operationen als der Aufruf von Algorithmus 6.1.10 für die Multiplikation der gleichen Matrizen benötigt. Anschließend wird wie für die Matrix-Matrix-Multiplikation das Niedrigrangupdate berechnet. Also benötigt in diesem Fall Algorithmus 6.4.4 höchstens so viele Operationen wie Algorithmus 6.2.2, d. h. maximal $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, k_{\max})$.
2. Fall: Es sei $(t,s) \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$.
Nach Induktionsbehauptung können wir die rekursiven Aufrufe von Algorithmus 6.4.4 für die Cluster $\check{t} \in \operatorname{chil}(t)$ und $\check{s} \in \operatorname{chil}^*(s)$ durch $\mathcal{N}_{MMM}(\mathcal{T}_{\check{\mathbf{t}}\times\check{\mathbf{t}}}, \mathcal{T}_{\check{\mathbf{t}}\times\check{\mathbf{s}}}, \mathcal{T}_{\check{\mathbf{t}}\times\check{\mathbf{s}}}, k_{\max})$ beschränken. Weil wir die Aufrufe des Algorithmus 6.2.2 für das Matrix-Produkt auch entsprechend

abschätzen können, erhalten wir für den Aufwand die obere Schranke

$$\sum_{\substack{\check{t}\in\mathrm{chil}(t)\\ \check{s}\in\mathrm{chil}^*(s)}} \mathcal{N}_{MMM}(\mathcal{T}_{\check{\mathbf{t}}\times\check{\mathbf{t}}}, \mathcal{T}_{\check{\mathbf{t}}\times\check{\mathbf{s}}}, \mathcal{T}_{\check{\mathbf{t}}\times\check{\mathbf{s}}}, k_{\max}) + \sum_{\substack{i\in\{1,\dots,\tau\}\\ j\in\{1,\dots,\sigma\}\\ \ell\in\{i+1,\dots,\tau\}}} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_i}, \mathcal{T}_{\mathbf{t}_i\times\mathbf{s}_j}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{s}_j}, k_{\max})$$

$$= \sum_{\substack{i\in\{1,\dots,\tau\}\\ j\in\{1,\dots,\sigma\}\\ \ell\in\{i,\dots,\tau\}}} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_i}, \mathcal{T}_{\mathbf{t}_i\times\mathbf{s}_j}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{s}_j}, k_{\max}) \;\leq\; \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, \mathcal{T}_{\mathbf{t}\times\mathbf{s}}, k_{\max}).$$

■

Bei der tatsächlichen algorithmischen Umsetzung können wir ohne die Inversion der Diagonalblöcke auskommen, indem wir das Vorwärtseinsetzen für vollbesetzte Matrizen verwenden. Dabei ist der Fall, dass $(t,t) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{I}}$ und $(t,s) \notin \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ ist, allerdings nicht so einfach gegen den Aufwand für das Matrix-Produkt abzuschätzen. Deshalb bietet es sich für die theoretische Betrachtung an, den obigen Ansatz zu wählen.

6.5 Dreieckszerlegungen von $\mathcal{H}^2$-Matrizen

In diesem Abschnitt untersuchen wir, wie Dreieckszerlegungen für $\mathcal{H}^2$-Matrizen konstruiert werden können. Dabei verwenden wir den gleichen blockweisen Ansatz, wie er bei $\mathcal{H}$-Matrizen verwendet wird. Die Anwendung von Dreieckszerlegungen im Kontext von $\mathcal{H}$-Matrizen wird in [1, 2, 20, 21, 27, 28] untersucht, wobei im dritten und vierten Artikel insbesondere spezielle Clusterstrategien für FEM-Steifigkeitsmatrizen betrachtet werden. Für $\mathcal{H}^2$-Matrizen wird in [17] die LR-Zerlegung basierend auf Niedrigrangmatrizen beschrieben und der Aufwand abgeschätzt. Dabei ist zu beachten, dass die Faktoren L und R als einzelne $\mathcal{H}^2$-Matrizen und nicht wie bei $\mathcal{H}$-Matrizen zusammen in der Struktur der Ausgangsmatrix A gespeichert werden. Dies liegt vor allem daran, dass wir nicht davon ausgehen können, dass L und R mit den gleichen Clusterbasen effizient darstellbar sind und aufgrund der Normierung des Faktors L typischerweise unterschiedliche Größenordnungen besitzen. Die Cholesky-Zerlegung für $\mathcal{H}^2$-Matrizen in der Form $A = LDL^T$ wird in [4] beschrieben und für die Berechnung von Eigenwerten von symmetrischen $\mathcal{H}^2$-Matrizen mit Hilfe eines Slicing-the-Spectrum-Ansatzes verwendet. Deshalb beschränken wir uns an dieser Stelle auf die Beschreibung der Cholesky-Zerlegung in der Form $A = LL^T$, weil die unterschiedlichen Arten der Dreieckszerlegung auf den gleichen Ideen basieren. Wie schon bei der Inversion gehen wir dabei davon aus, dass die zu berechnenden Zerlegungen für die vollbesetzten Matrizen existieren.
Im Falle einer symmetrischen, positiv definiten Matrix kann die Cholesky-Zerlegung berechnet werden. Wir wollen diese approximativ im $\mathcal{H}^2$-Matrix-Format berechnen, wobei wir davon ausgehen, dass die ursprüngliche Matrix A bereits als $\mathcal{H}^2$-Matrix vorliegt. Den Faktor L wollen wir in der gleichen Blockstruktur wie A speichern. Wie bei der Inversion

betrachten wir zuerst den Fall, dass die Matrizen in 2×2 Kinder unterteilten sind. Für diesen erhalten wir die zu erfüllende Gleichung

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ & L_{22}^T \end{pmatrix} = \begin{pmatrix} L_{11}L_{11}^T & L_{11}L_{21}^T \\ L_{21}L_{11}^T & L_{21}L_{21}^T + L_{22}L_{22}^T \end{pmatrix}.$$

Dabei sind die Gleichungen $A_{12} = L_{11}L_{21}^T$ und $A_{21} = L_{21}L_{11}^T$ aufgrund der Symmetrie von A äquivalent. Wir gehen davon aus, dass die Matrix A in unterer Dreiecksgestalt gespeichert ist (aufgrund der Symmetrie müssen die Einträge oberhalb der Diagonalen nicht gespeichert werden). Dies hat den Vorteil, dass die Clusterbasen kleinere Teile beschreiben müssen und wir mit potentiell kleineren Rängen arbeiten können. Deshalb verwenden wir nur Blöcke unterhalb der Diagonalen zur Beschreibung des Algorithmus. Für die Berechnung der Cholesky-Zerlegung erhalten wir folgende Schritte:

(i) Berechne die Cholesky-Zerlegung $L_{11}L_{11}^T \leftarrow A_{11}$

(ii) Berechne $L_{21}^T \leftarrow L_{11}^{-1}A_{21}^T$ per Vorwärtseinsetzen

(iii) Berechne die Hilfsgröße $H_{22} \leftarrow A_{22} - L_{21}L_{21}^T$

(iv) Berechne die Cholesky-Zerlegung $L_{22}L_{22}^T \leftarrow H_{22} = A_{22} - L_{21}L_{21}^T$

Also benötigen wir nur das Matrix-Produkt aus Abschnitt 6.2 und das Vorwärtseinsetzen aus Abschnitt 6.4. Für die Hilfsmatrix entspricht

$$H_{22} = A_{22} - L_{21}L_{21}^T = A_{22} - A_{21}L_{11}^{-T}L_{11}^{-1}A_{21}^T = A_{22} - A_{21}A_{11}^{-1}A_{12} = S$$

dem Schur-Komplement S. Dieses speichern wir wie bei der Inversion im Block A_{22}.
Mit diesem Vorgehen erhalten wir Algorithmus 6.5.1, wobei wir wie bei er Inversion vom Fall mit 2 Kindern zum allgemeinen Fall übergehen. Den Aufwand schätzen wir in Theorem 6.5.1 wie bei der Inversion gegen den Aufwand für das Matrix-Produkt ab.

Theorem 6.5.1
Es sei $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit $(t,t) \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}} \backslash \mathcal{L}^+_{\mathcal{I}\times\mathcal{I}}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$. Der maximale lokale Rang während des Aufrufs von Algorithmus 6.5.1 sei $k_{\max}$. Der Aufwand für die Berechnung der Cholesky-Zerlegung und die Invertierung des resultierenden L-Faktors für eine vollbesetzte Matrix $X \in \mathbb{R}^{\mathbf{t}\times\mathbf{t}}$ sei für alle $t \in \mathcal{L}_{\mathcal{I}}$ kleiner als $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max})$. Dann ist der Aufwand für die Cholesky-Zerlegung von A mit Algorithmus 6.5.1 kleiner als $\mathcal{N}_{MMM}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, \mathcal{T}_{\mathcal{I}\times\mathcal{I}}, \mathcal{T}_{\mathcal{I}\times\mathcal{I}}, k_{\max})$

Beweis: Wir zeigen, dass für alle $t \in \mathcal{T}_{\mathcal{I}}$ der Aufwand für die Cholesky-Zerlegung von A mit Algorithmus 6.5.1 kleiner als $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max})$ ist, per Induktion über $\#\operatorname{desc}(t) \in \mathbb{N}_{>0}$. Für $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) = 1$ gilt $t \in \mathcal{L}_{\mathcal{I}}$. Also berechnet der Algorithmus 6.5.1 erst die Cholesky-Zerlegung der vollbesetzten Matrix $N_{A,(t,t)} = N_{L,(t,t)}N_{L,(t,t)}^T$ und danach die Inverse von $N_{L,(t,t)}$. Nach Voraussetzung benötigt dies höchstens $\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, \mathcal{T}_{\mathbf{t}\times\mathbf{t}}, k_{\max})$ Operationen.
Es sei $n \in \mathbb{N}_{>0}$ derart, dass die Behauptung für alle $t \in \mathcal{T}_{\mathcal{I}}$ mit $\#\operatorname{desc}(t) \leq n$ gilt. Weiter

Algorithmus 6.5.1 Die Funktion berechnet die Cholesky-Zerlegung für eine $\mathcal{H}^2$-Matrix $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V_A, W_A, N_A, S_A)$. Die Matrix A wird mit Hilfsgrößen überschrieben und das Ergebnis wird in der $\mathcal{H}^2$-Matrix $L = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, V_L, W_L, N_L, S_L)$ gespeichert, wobei diese mit 0 initialisiert sein muss.

function H2MATRIX_CHOLESKY(t, A, Z_A, L, $((L_{\mathbf{t}\times\mathbf{t}})^{-1})_{t\in\mathcal{L}_\mathcal{I}}$, Z_L, *opt*)
  if $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$ **then**
    $\{t_1, \ldots, t_\tau\} \leftarrow \operatorname{chil}(t)$
    for $\ell = 1$ **to** τ **do**
      H2MATRIX_CHOLESKY(t_ℓ, A, Z_A, L, $((L_{\mathbf{t}\times\mathbf{t}})^{-1})_{t\in\mathcal{L}_\mathcal{I}}$, Z_L, *opt*)
      for $j \in \{\ell+1, \ldots, \tau\}$ **do**
        H2MATRIX_VORWÄRTSEINSETZEN(t_ℓ, t_j, L, $((L_{\mathbf{t}\times\mathbf{t}})^{-1})_{t\in\mathcal{L}_\mathcal{I}}$, A^T, Z_A
. L^T, Z_{L^T}, *opt*) ▷ Algorithmus 6.4.4
      end for
      for $i, j \in \{\ell+1, \ldots, \tau\}$ **do**
        H2MATRIX_PRODUKT(t_i, t_ℓ, t_j, $-L$, L^T, A, Z_A, *opt*)
        ▷ Algorithmus 6.2.2
      end for
    end for
  else ▷ $t \in \mathcal{L}_\mathcal{I}$ und $(t,t) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{I}}$
    $N_{L,(t,t)} N^T_{L,(t,t)} \leftarrow N_{A,(t,t)}$
    $L^{-1}_{\mathbf{t}\times\mathbf{t}} \leftarrow N^{-1}_{L,(t,t)}$
  end if
end function

sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{desc}(t) = n+1$. Dann ist $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$. Es gilt $\#\operatorname{desc}(\breve{t}) \leq n$ und somit die Induktionsbehauptung für alle $\breve{t} \in \operatorname{chil}(t) =: \{t_1, \ldots, t_\tau\}$. Also ist der Aufwand für die rekursiven Aufrufe von Algorithmus 6.5.1 durch

$$\sum_{\ell=1}^{\tau} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, k_{\max})$$

beschränkt. Die Aufrufe von Algorithmus 6.4.4 für das Vorwärtseinsetzen benötigen nach Lemma 6.4.7 maximal

$$\sum_{\ell=1}^{\tau} \sum_{j=\ell+1}^{\tau} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_\ell}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_j}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_j}, k_{\max})$$

Operationen. Die Multiplikationen mit Algorithmus 6.2.2 benötigen maximal

$$\sum_{\ell=1}^{\tau} \sum_{i=\ell+1}^{\tau} \sum_{j=\ell+1}^{\tau} \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_\ell}, \mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_j}, \mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_j}, k_{\max})$$

Operationen. Zusammen erhalten wir die obere Schranke

$$\sum_{\ell=1}^{\tau}\sum_{i=\ell}^{\tau}\sum_{j=\ell}^{\tau}\mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_\ell},\mathcal{T}_{\mathbf{t}_\ell\times\mathbf{t}_j},\mathcal{T}_{\mathbf{t}_i\times\mathbf{t}_j},k_{\max}) \;\leq\; \mathcal{N}_{MMM}(\mathcal{T}_{\mathbf{t}\times\mathbf{t}},\mathcal{T}_{\mathbf{t}\times\mathbf{t}},\mathcal{T}_{\mathbf{t}\times\mathbf{t}},k_{\max}). \tag{6.8}$$

Damit folgt die Behauptung. ∎

Bemerkung 6.5.2
Indem wir die Inversion des L-Faktors per Vorwärtseinsetzens berechnen, können wir die Bedingung in Theorem 6.5.1 erfüllen (siehe [42, Chapter 3]).
Aus Theorem 6.5.1 folgt zusammen mit Theorem 6.2.4 eine Aufwandsabschätzung für die Cholesky-Zerlegung mit Algorithmus 6.3.1.

Für vollbesetzte Matrizen können wir den Aufwand für die Cholesky-Zerlegung gegen 1/6 des Aufwands für das Matrix-Produkt abschätzen. Ein solcher Faktor ist für $\mathcal{H}^2$-Matrizen nicht zu erwarten, weil bereits für das Vorwärtseinsetzen für $\mathcal{H}^2$-Matrizen mit Vektoren mehr als die Hälfte der Operationen der entsprechenden Matrix-Vektor-Multiplikation notwendig sind (siehe Algorithmus 6.4.1). Allerdings ist in (6.8) zu sehen, dass an dieser Stelle der Aufwand grob abgeschätzt wird. Wir verzichten hier auf weitergehende Untersuchung.
In diesem Kapitel haben wir Algorithmen vorgestellt, die das Matrix-Produkt, die Inversion und Cholesky-Zerlegungen (exemplarisch für die Dreieckszerlegungen) für $\mathcal{H}^2$-Matrizen in log-linearem Aufwand berechnen. Dabei sparen wir im Vergleich zu den Aufwandsabschätzungen für $\mathcal{H}$-Matrizen einen Logarithmus als Faktor ein (siehe [26, Theorem 2.24]).
Im nächsten Kapitel beschreiben wir eine Modifikation des Niedrigrangupdates, das insbesondere für Probleme mit großen Konstanten für die Schwachbesetztheit der Blockbäume den Aufwand reduziert.

Bemerkung 6.5.3
Aufgrund der Symmetrie der Ausgangsmatrix können wir darauf verzichten, die Blöcke oberhalb der Diagonalen zu speichern. Bei der Implementierung in [39] werden diese als Nullblöcke, wie sie in Bemerkung 6.4.3 beschrieben sind, gespeichert.
Für eine höhere Effizienz kann die Berechnung des Schur-Komplements derart gestaltet werden, dass die Blöcke oberhalb der Diagonalen nicht mit berechnet werden. Dies ist aktuell in [39] noch nicht umgesetzt.

7 Niedrigrangupdates für rekursive Algorithmen

Das Niedrigrangupdate aus Kapitel 5 haben wir derart gestaltet, dass es unabhängig von der Reihenfolge der Updates ist. Allerdings sind die in Kapitel 6 vorgestellten Algorithmen zur Arithmetik alle rekursiv formuliert. Deshalb gehen wir bei den in diesem Kapitel vorgestellten Varianten davon aus, dass die Niedrigrangupdates innerhalb einer Funktion (z. B. Matrix-Matrix-Multiplikation) ausgeführt werden, die den Blockbaum der $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$, für die die Updates berechnet werden, rekursiv durchläuft. (Bei der Multiplikation $C \leftarrow C + AB$ entspricht H der Zielmatrix C.) Diese Tatsache wollen wir in diesem Kapitel nutzen, um einerseits auf die Projektionen in den Clusterbasen und Gewichten zu verzichten (siehe die Definitionen 5.3.1 und 5.3.4) und zweitens den Aufwand für das Niedrigrangupdate in einem Block $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ auf die Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$, $\mathcal{T}_{\tilde{\mathbf{s}}}$ und $\mathcal{T}_{\tilde{\mathbf{b}}}$ ohne Durchlaufen der entsprechenden Blockzeilen und -spalten beschränken.
Das erste Ziel erreichen wir in Abschnitt 7.1 durch die 1. Variante. Diese nutzt den rekursiven Durchlauf, um statt mit projizierten Clusterbasen und Gewichten mit orthogonalen Clusterbasen und adaptiven Gewichten zu arbeiten. Außerdem kommt diese Variante mit weniger gespeicherten Hilfsgrößen aus. Um dies zu erreichen, definieren wir zusätzliche Hilfsfunktionen für das Durchlaufen des Blockbaums.
Mit der 2. Variante aus Abschnitt 7.2 erhalten wir die Vorteile der 1. Variante und beschränken zusätzlich den Aufwand auf die Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$, $\mathcal{T}_{\tilde{\mathbf{s}}}$ und $\mathcal{T}_{\tilde{\mathbf{b}}}$. Das Niedrigrangupdate aus Kapitel 5 durchläuft die Blockzeilen bzw. -spalten der beteiligten Cluster. Dies führt dazu, dass für die Aufwandsabschätzung in Theorem 5.3.20 die c_{sp}-Konstante für die $\mathcal{H}^2$-Matrix des Updates auftaucht. Dies vermeiden wir bei der 2. Variante dadurch, dass alle Berechnungen nur in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ zum aktuellen Blocks $\tilde{b} = (\tilde{t}, \tilde{s})$ durchgeführt werden.
Dazu führen wir spezielle Speicherformate für die Kopplungsmatrizen und die notwendigen Hilfsgrößen ein, die darauf basieren, dass wir die Matrizen nur im aktuellen Block $\tilde{b}$ aktualisieren. Dies führt zwar zu aufwendigeren Hilfsfunktionen beim Durchlaufen des Blockbaums, hat aber den Vorteil, dass die Blockzeilen/-spalten während des Updates nicht durchlaufen werden. Dies spart uns einen Faktor c_{sp} bei der Aufwandsabschätzung, welcher sich vor allem bei dreidimensionalen Problemen bemerkbar macht.
In Abschnitt 7.3 stellen wir den Algorithmus für das Matrix-Produkt $C \leftarrow C + AB$ mit dem Niedrigrangupdate in der 2. Variante vor. Dieser ist anders strukturiert als der Algorithmus 6.2.2 in Kapitel 6. Um den Aufwand für die Hilfsfunktionen zu reduzieren, durchlaufen wir den Blockbaum der Zielmatrix C nur einmal rekursiv und speichern

die Matrixpaare, die jeweils zu einem Block addiert werden müssen, in einer Liste. In Kombination mit dem neuen Update führt das dazu, dass in der Aufwandsabschätzung für die Updates während der Multiplikation nur noch die c_{sp}-Konstante der Zielmatrix C und diese einfach vorkommt.
Zur Veranschaulichung der Funktionsweise der beiden Varianten geben wir jeweils einen Prototyp für einen rekursiven Algorithmus an, der die neuen Niedrigrangupdates verwendet. Dabei verwenden wir Prolog und Epilog-Funktionen die vor bzw. nach den rekursiven Aufrufen verwendet werden, um die von den Updates benötigten Hilfsgrößen aktuell zu halten. Um die Prototypen in den Abschnitten 7.1 und 7.2 sowie das Matrix-Produkt Abschnitt 7.3 übersichtlich zu gestalten, verwenden wir die verkürzte Notation aus Bemerkung 6.2.1 für die Formulierung die Update-, Prolog- und Epilog-Algorithmen.

7.1 1. Variante

Bei der 1. Variante vermeiden wir, mit projizierten Clusterbasen und Gewichten zu arbeiten. Wie wir beim allgemeinen Update gesehen haben, können wir nicht ohne zusätzlichen Aufwand sicherstellen, in allen Clustern orthogonale Clusterbasen und adaptive Gewichte zu haben. Allerdings werden diese auch nicht für das lokale Niedrigrangupdate benötigt. Sobald wir adaptive Gewichte für die Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ berechnet haben, können wir die adaptiven Clusterbasen und die lokalen Projektionen wie in Abschnitt 5.3 berechnen. Für die adaptiven Gewichte (siehe Definition 4.4.17) im Teilbaum benötigen wir die adaptiven Gewichte der Eltern, falls diese existieren, und orthogonale Gewichte für den Teil der Clusterbasis, der nicht orthogonal ist. Deshalb gehen wir davon aus, dass erstens die Clusterbasen in wesentlichen Teilen orthogonal sind, d. h., dass

$$V_t,\, W_s \text{ für alle } t \in \mathcal{T}_{\mathcal{I}} \backslash (\operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\}),\, s \in \mathcal{T}_{\mathcal{J}} \backslash (\operatorname{pred}(\tilde{s}) \backslash \{\tilde{s}\}) \text{ orthogonal} \tag{7.1}$$

und zweitens

$$\text{adaptive Gewichte } Z_{\mathcal{I},t},\, Z_{\mathcal{J},s} \text{ für alle } t \in \operatorname{pred}(\tilde{t}) \backslash \{\tilde{t}\},\, s \in \operatorname{pred}(\tilde{s}) \backslash \{\tilde{s}\} \text{ gegeben} \tag{7.2}$$

sind. Für das einzelne Update benötigen wir in (7.1) lediglich $t \in \mathcal{T}_{\mathcal{I}} \backslash (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ und $s \in \mathcal{T}_{\mathcal{J}} \backslash (\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s}))$ und in (7.2) nur $t = \operatorname{par}(\tilde{t})$ und $s = \operatorname{par}(\tilde{s})$. Die stärkeren Voraussetzungen benötigen wir, damit (7.1) beim Abwärtslaufen im Baum und (7.2) beim Aufwärtslaufen im Baum erhalten bleibt. Bevor wir zeigen, dass die beiden Voraussetzungen ausreichen, um das Niedrigrangupdate lokal zu berechnen, beweisen wir eine technische Aussage zu den erweiterten Clusterbasen (siehe Definition 5.1.1).

Lemma 7.1.1
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Falls V_t für alle $t \in \mathcal{T}_{\mathcal{I}} \backslash (\mathcal{T}_{\tilde{\mathbf{t}}} \cup \operatorname{pred}(\tilde{t}))$ und W_s

für alle $s \in \mathcal{T}_\mathcal{J} \backslash (\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s}))$ orthogonal sind, folgt für alle $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$

$$\begin{aligned} t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \notin \mathcal{T}_{\tilde{\mathbf{s}}} &\Rightarrow \widetilde{W}(W,B,\tilde{s})_s \text{ ist orthogonal und} \\ t \notin \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} &\Rightarrow \widetilde{V}(V,A,\tilde{t})_t \text{ ist orthogonal.} \end{aligned}$$

Beweis: Es sei $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ mit $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Weil $(\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ ist, folgt aus Lemma 3.3.14 (iv), dass $s \notin (\operatorname{pred}(\tilde{s})\backslash\{\tilde{s}\})$ gilt. Somit gilt $s \in \mathcal{T}_\mathcal{J}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s}))$ und nach Definition 5.1.1 ist $\widetilde{W}(W,B,\tilde{s})_s = W_s$ orthogonal.
Die zweite Aussage folgt analog. ∎

Wir wollen die adaptiven Gewichte wie in Abschnitt 5.3 durch Algorithmus 5.3.7 berechnen. Im folgenden Lemma zeigen wir, dass die projizierten Gewichte aus Lemma 5.3.7 unter den Voraussetzungen (7.1) und (7.2) adaptive Gewichte sind, falls für die Teilbäume $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ orthogonale Gewichte gegeben sind.

Lemma 7.1.2
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix, $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung, $\tilde{b} = (\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{t}\times\tilde{s}}$ und $\widetilde{H} := \widetilde{H}(H,R,\tilde{b}) =: \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{V}, \widetilde{W}, \widetilde{N}, \widetilde{S})$. Die Spaltenbasis W_s sei für alle $s \in \mathcal{T}_\mathcal{J}\backslash(\mathcal{T}_{\tilde{\mathbf{s}}} \cup \operatorname{pred}(\tilde{s}))$ orthogonal. Weiter seien Skalierungen $\omega, \tilde{\omega} \in \mathbb{R}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}_{>0}$ und $\theta, \tilde{\theta} \in \mathbb{R}^{\mathcal{T}_\mathcal{I}}_{>0}$ mit $\omega_b = \tilde{\omega}_b$ für alle $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{b}}}$ und $\theta_t = \tilde{\theta}_t$ für alle $t \in \mathcal{T}_\mathcal{I}\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$ gegeben. Es seien $(\widetilde{O}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Gewichte für die Clusterbasis $\widetilde{W}$ im Teilbaum $\mathcal{T}_{\tilde{\mathbf{s}}}$ und, falls $\tilde{t} \neq r_\mathcal{I}$ ist, für $\acute{t} := \operatorname{par}(t)$ ein adaptives Gewicht $Z_{\acute{t}}$ von H zu ω und θ gegeben. Dann ist für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ die Matrix $\widetilde{Z}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \widetilde{H}, \widetilde{V}, \tilde{\omega}, \tilde{\theta})_t \in \mathbb{R}^{\tilde{\kappa}_t\times\tilde{\zeta}_t}$ aus Lemma 5.3.12 ein adaptives Gewicht von $\widetilde{H}$ zu $\tilde{\omega}$ und $\tilde{\theta}$ mit Rangverteilung $\tilde{\zeta}$.

Beweis: Nach Lemma 5.3.12 berechnet Algorithmus 5.3.7 projizierte Gewichte. Dies folgt aus Lemma 5.3.9. Nach Bemerkung 5.3.11 handelt es sich dabei um adaptive Gewichte, falls für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und alle $s \in \operatorname{row}(t)$ die Clusterbasis $\widetilde{W}_s$ orthogonal ist oder ein orthogonales Gewicht O_s berechnet wurde und $Z_{\acute{t}}$ ein adaptives Gewicht ist. Letzteres folgt aus der Voraussetzung. Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)$.
1. Fall: Es sei $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann ist nach Voraussetzung ein orthogonales Gewicht $\widetilde{O}_s$ gegeben.
2. Fall: Es sei $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. In diesem Fall ist nach Lemma 7.1.1 $\widetilde{W}_s$ orthogonal. Damit folgt die Behauptung. ∎

Also genügen die Voraussetzungen (7.1) und (7.2) sowie orthogonale Gewichte in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$, um adaptive Gewichte mit Algorithmus 5.3.7 zu berechnen. In Algorithmus 7.1.1 definieren wir das Niedrigrangupdate für die 1. Variante.
Diese entspricht im Wesentlichen der allgemeinen Variante aus Abschnitt 5.3, wobei wir allerdings auf die Neuberechnung der projizierten Gewichte in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ verzichten können.

Algorithmus 7.1.1 Die Funktion berechnet das Niedrigrangupdate einer $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ für eine Rang-$\mathcal{K}$-Matrix $R = AB^T$ für die 1. Variante. Wir verwenden die verkürzte Notation nach Bemerkung 6.2.1.

function NIEDRIGRANGUPDATE_VARIANTE1($\tilde{b}$, H, Z, R, *opt*)
  $(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V,E), (W,F), N, S) \leftarrow H$
  LOKALE_ORTHOGONALE_GEWICHTE($\tilde{t}$, V, κ, A, $\mathcal{K}$, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$) ▷ Algorithmus 5.3.1
  LOKALE_ORTHOGONALE_GEWICHTE($\tilde{s}$, W, λ, B, $\mathcal{K}$, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$)
  SKALIERUNG($\tilde{b}$, S, κ, λ, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$, *opt*, $\omega_{\mathcal{I}}$) ▷ Algorithmus 4.5.2
  SKALIERUNG($\tilde{b}^T$, S^T, λ, κ, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$, *opt*, $\omega_{\mathcal{J}}$)
  LOKALE_PROJIZIERTE_GEWICHTE($\tilde{t}, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \tilde{t}, E, S, \kappa, \mathcal{K}, O_{\mathcal{I}}, \rho_{\mathcal{I}}, Z_{\mathcal{I}}, \omega_{\mathcal{I}}, \theta_{\mathcal{I}}$)
  LOKALE_PROJIZIERTE_GEWICHTE($\tilde{s}, \mathcal{T}^T_{\mathcal{I}\times\mathcal{J}}, \tilde{s}, F, S^T, \lambda, \mathcal{K}, O_{\mathcal{J}}, \rho_{\mathcal{J}}, Z_{\mathcal{J}}, \omega_{\mathcal{J}}, \theta_{\mathcal{J}}$)
  ▷ Algorithmus 5.3.2
  LOKALE_ADAPTIVE_CLUSTERBASIS($\tilde{t}$, $\tilde{t}$, V, κ, A, $\mathcal{K}$, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$, $\overline{\kappa}$, $C_{\mathcal{I}}$, *opt*, $\epsilon_{\mathcal{I}}$)
  LOKALE_ADAPTIVE_CLUSTERBASIS($\tilde{s}$, $\tilde{s}$, W, λ, B, $\mathcal{K}$, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$, $\overline{\lambda}$, $C_{\mathcal{J}}$, *opt*, $\epsilon_{\mathcal{J}}$)
  ▷ Algorithmus 5.3.3
  LOKALE_H2-MATRIX-PROJEKTION($\tilde{b}$, N, S, κ, λ, A, B, $\mathcal{K}$, $C_{\mathcal{I}}$, $\overline{\kappa}$, $C_{\mathcal{J}}$, $\overline{\lambda}$)
  ▷ Algorithmus 5.3.6
end function

Wie in Theorem 5.3.20 für die allgemeine Variante zeigen wir in Lemma 7.1.3, dass sich der Aufwand durch die Größen der Teilbäume beschränken lässt und dass die Voraussetzungen des Niedrigrangupdates durch das Update erhalten bleiben.

Lemma 7.1.3 (Aufwand des Niedrigrangupdates)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem, strikt zulässigem Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und für die Clusterbasen V und W gelte (7.1). Weiter sei $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$. Für die Skalierungen $\omega_{\mathcal{I}} \in \mathbb{R}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}_{>0}$ und $\theta_{\mathcal{I}} \in \mathbb{R}^{\mathcal{T}_{\mathcal{I}}}_{>0}$ bzw. $\omega_{\mathcal{J}} \in \mathbb{R}^{(\mathcal{L}^T_{\mathcal{I}\times\mathcal{J}})^+}_{>0}$ und $\theta_{\mathcal{J}} \in \mathbb{R}^{\mathcal{T}_{\mathcal{J}}}_{>0}$ gelte (7.2). Es seien $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t, \#\mathcal{K}, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s\}$ und $n_{\max} := \max\{\max_{t\in\mathcal{L}_{\mathcal{I}}} \#\mathbf{t}, \max_{s\in\mathcal{L}_{\mathcal{J}}} \#\mathbf{s}\}$. Dann benötigt der Algorithmus 5.3.8 für die Berechnung von $\overline{H} \approx H + R$ höchstens

$$\begin{aligned}
&(12 + 4c_{qr} + 4c_{apr})k^2_{\max}(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) \\
&+ (30 + 16c_{qr} + 8c_{qr}c_{sp} + 8c_{\|.\|}c_{sp} + 64c_{sp} + 4c_{apr})k^3_{\max}(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}) \\
&+ c_{sp}\max\{k^2_{\max}, n^2_{\max}\}k_{\max}(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})
\end{aligned}$$

Operationen. Die $\mathcal{H}^2$-Matrix-Darstellung von H wird mit der von $\overline{H}$ mit Clusterbasen $\overline{V}$ und $\overline{W}$ überschrieben, die (7.1) erfüllen. Dann erfüllen die alten Gewichte $Z_{\mathcal{I}}$ und $Z_{\mathcal{J}}$ auch für die neue Matrix $\overline{H}$ die Bedingung (7.2) zu den Skalierungen $\omega_{\mathcal{I}}$, $\omega_{\mathcal{J}}$, $\theta_{\mathcal{I}}$ und $\theta_{\mathcal{J}}$.

BEWEIS: Um $\overline{H}$ zu berechnen, wird Algorithmus 7.1.1 im Block $\tilde{b}$ aufgerufen. Die Berechnung der orthogonalen Gewichte für $\widetilde{H} := \widetilde{H}(H, R, \tilde{b})$ in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ benötigt nach

Lemma 5.3.8 höchstens

$$4c_{qr}\, k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (4 + 8c_{qr})\, k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen. Die Skalierungen werden nach Lemma 4.5.7 mit Algorithmus 4.5.2 in nicht mehr als $8(7 + c_{\|\cdot\|})c_{sp}k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$ Operationen berechnet. Nach Lemma 5.3.9 und Bemerkung 5.3.11 berechnet Algorithmus 5.3.7 adaptive Gewichte $\widetilde{H}$ zu den Skalierungen $\tilde{\omega}_{\mathcal{I}}$, $\tilde{\omega}_{\mathcal{J}}$, $\tilde{\theta}_{\mathcal{I}}$ und $\tilde{\theta}_{\mathcal{J}}$, wobei diese außerhalb der Teilbäume $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ gleich den alten gesetzt werden. Für den Aufwand gilt nach Lemma 5.3.13 die obere Schranke

$$(6c_{sp} + 8c_{qr}c_{sp} + 8c_{qr} + 8)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}).$$

Danach werden die neuen Clusterbasen $\overline{V}$ und $\overline{W}$ in $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ in höchstens

$$(4c_{apr} + 16)k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (8c_{apr} + 36)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen aufgestellt (siehe Lemma 5.3.15). Dabei werden die Ergebnisse in V bzw. W gespeichert und gleichzeitig die Basiswechsel C und D berechnet. Also sind die neuen Clusterbasen $\overline{V}$ und $\overline{W}$ in V bzw. W gespeichert und nach Lemma 5.2.1 erfüllen die Clusterbasen $\overline{V}$ und $\overline{W}$ die Bedingung (7.1). Mit den Basiswechseln werden die Nahfeld- und Kopplungsmatrizen der bezüglich $\overline{V}$ und $\overline{W}$ lokal projizierten Matrix $\overline{H}$ mit Algorithmus 5.3.6 nach Lemma 5.3.17 in weniger als

$$2c_{sp}(2k_{\max}^3 + \max\{6k_{\max}^2,\, n_{\max}^2\}k_{\max})(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen berechnet. Dabei werden die alten Kopplungs- und Nahfeldmatrizen mit den neuen überschrieben. Zusammen folgt die Schranke für den Gesamtaufwand von Algorithmus 7.1.1.
Wegen Lemma 5.2.5 und Lemma 5.2.1 gilt für alle $t \in \operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und die zum adaptiven Gewicht gehörige orthogonale Matrix P_t

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \overline{H}, \omega_{\mathcal{I}}, \theta_{\mathcal{I}})_t &= (\Pi_{V_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}})X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega_{\mathcal{I}}, \theta_{\mathcal{I}})_t \\ &= (\Pi_{V_{\tilde{t}}} + \Pi_{\mathbf{t}\backslash\tilde{\mathbf{t}}})V_t Z_{\mathcal{I},t} P_{\mathcal{I},t} \;=\; \overline{V}_t Z_{\mathcal{I},t} P_{\mathcal{I},t}. \end{aligned}$$

Also sind die alten Gewichte für die Zeilenbasis auch adaptive Gewichte für die neue Matrix. Analog folgt die Aussage für die Gewichte der Spaltenbasis. ∎

Bemerkung 7.1.4
Der Fehler für das Niedrigrangupdate in der 1. Variante lässt sich wie für das Niedrigrangupdate aus Kapitel 5 abschätzen und es gelten die Fehlerschranken aus Abschnitt 5.4. Weil wir für das Update in der 1. Variante adaptive Gewichte verwenden, gilt (5.17) mit $\hat{\epsilon} = 0$ und in der Fehlerabschätzung aus Lemma 5.4.2 entfällt der Term $1/(1 - \hat{\epsilon})$.

Um diese Variante des Niedrigrangupdates umzusetzen, müssen wir sicherstellen, dass die Voraussetzungen (7.1) und (7.2) erfüllt sind. Vor dem Aufruf der rekursiven Funktion können wir die Voraussetzungen folgendermaßen initialisieren.

Bemerkung 7.1.5 (Initialisierung von (7.1) und (7.2))
Die Initialisierung der Bedingung (7.1) erreichen wir durch die Orthogonalisierung der $\mathcal{H}^2$-Matrix-Darstellung mit Algorithmus 4.2.3. Weil die Wurzeln $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ keine Vorfahren außer sich selbst haben, ist die Bedingung (7.2) für den Block $\tilde{b} = r_{\mathcal{I}\times\mathcal{J}}$ trivial.

Im nächsten Schritt definieren wir den Algorithmus 7.1.3, der die Voraussetzungen beim Abwärtslaufen im Baum sicherstellt. Dieser berechnet lediglich adaptive Gewichte für die aktuellen Cluster $\tilde{t}$ und $\tilde{s}$. In Lemma 7.1.6 zeigen wir, dass dies ausreicht. Zuvor stellen wir mit Algorithmus 7.1.2 die Berechnung eines adaptiven Gewichts für einen Cluster vor, wobei davon ausgegangen wird, dass alle beteiligten Spaltenclusterbasen orthogonal sind. Es handelt sich dabei um eine Adaption von Algorithmus 4.4.2.

Algorithmus 7.1.2 Die Funktion berechnet ein adaptives Gewicht zu ω und θ für den Cluster t, vorausgesetzt, dass das adaptive Gewicht für das Elter $\operatorname{par}(t)$ bekannt ist und W_s für alle $s \in \operatorname{row}(t)$ orthogonal ist.

---

function ADAPTIVES_GEWICHT_CLUSTER(t, E, S, κ, λ, ω, θ, Z, ζ)
  $\ell \leftarrow \dot{\bigcup}_{s\in\operatorname{row}(t)} \lambda_s$
  if $t \neq r_{\mathcal{I}}$ **then**
    $\acute{t} \leftarrow \operatorname{par}(t)$
    $\ell \leftarrow \ell \dot{\cup} \zeta_{\acute{t}}$
  end if
  $\widehat{Z}_t \in \mathbb{R}^{\kappa_t \times \ell}$
  for $s \in \operatorname{row}(t)$ **do**
    $\widehat{Z}_{t_{|\kappa_t\times\lambda_s}} \leftarrow \frac{1}{\omega_{(t,s)}} S_{(t,s)}$
  end for
  if $t \neq r_{\mathcal{I}}$ **then**
    $\widehat{Z}_{t_{|\kappa_t\times\zeta_{\acute{t}}}} \leftarrow \frac{1}{\theta_t} E_t Z_{\acute{t}}$
  end if
  QR-ZERLEGUNG($\widehat{Z}_t^T$, ℓ, ℓ, κ_t, P_t, Z_t^T, ζ_t) ▷ Algorithmus 2.4.1
end function

Lemma 7.1.6
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen $\kappa = (\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ und $\lambda = (\lambda_s)_{s\in\mathcal{T}_{\mathcal{J}}}$. Es seien $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und die Voraussetzungen (7.1) und (7.2) erfüllt. Falls $\tilde{t} \neq r_{\mathcal{I}}$ ist, gelte für die Rangverteilungen des adaptiven Gewichts $\#\zeta_{\mathcal{I},\operatorname{par}(\tilde{t})} \leq \#\kappa_{\operatorname{par}(\tilde{t})}$ und, falls $\tilde{s} \neq r_{\mathcal{J}}$ ist, gelte für die Rangverteilungen des adaptiven Gewichts $\#\zeta_{\mathcal{J},\operatorname{par}(\tilde{s})} \leq \#\kappa_{\operatorname{par}(\tilde{s})}$. Dann benötigt der Algorithmus 7.1.3 aufgerufen in $\tilde{b}$ höchstens

$$2(c_{sp} + c_{sp}c_{qr} + c_{qr} + 3)k_{\max}^3$$

Operationen, wobei $k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s\}$ sei. Außerdem sind die Voraussetzungen (7.1) und (7.2) nach dem Aufruf von Algorithmus 7.1.3 für alle $\breve{b} \in \operatorname{chil}(\tilde{b})$ erfüllt.

Algorithmus 7.1.3 Die Funktion stellt sicher, dass die Voraussetzungen (7.1) und (7.2) für alle Kinder $\breve{b} \in \text{chil}(\tilde{b})$ erfüllt sind. Dabei wird davon ausgegangen, dass die Voraussetzungen für $\tilde{b}$ erfüllt sind.

function PROLOG_VARIANTE1($\tilde{b}$, H, Z, *opt*)
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$
  $(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V, E), (W, F), N, S) \leftarrow H$
  if $\tilde{t} \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ **then**
    ADAPTIVES_GEWICHT_CLUSTER($\tilde{t}$, E, S, κ, λ, $\omega_{\mathcal{I}}$, $\theta_{\mathcal{I}}$, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$)
      ▷ Algorithmus 7.1.2
  end if
  if $\tilde{s} \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{L}_{\mathcal{J}}$ **then**
    ADAPTIVES_GEWICHT_CLUSTER($\tilde{s}$, F, S^T, λ, κ, $\omega_{\mathcal{J}}$, $\theta_{\mathcal{J}}$, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$)
      ▷ Algorithmus 7.1.2
  end if
end function

BEWEIS: Es sei $\breve{b} = (\breve{t}, \breve{s}) \in \text{chil}(\tilde{b})$. Wir betrachten die Zeilencluster. Der Beweis für die Spaltencluster verläuft analog. Es gilt $\breve{t} \in \text{chil}^*(\tilde{t})$.
1. Fall: Es sei $\tilde{t} \in \mathcal{L}_{\mathcal{I}}$. In diesem Fall führt Algorithmus 7.1.3 für die Zeilencluster keine Berechnungen aus. Es gilt $\breve{t} = \tilde{t}$. Insbesondere gilt $\text{pred}(\breve{t})\backslash\{\breve{t}\} = \text{pred}(\tilde{t})\backslash\{\tilde{t}\}$ und $\mathcal{T}_{\mathcal{I}}\backslash(\text{pred}(\breve{t})\backslash\{\breve{t}\}) = \mathcal{T}_{\mathcal{I}}\backslash(\text{pred}(\tilde{t})\backslash\{\tilde{t}\})$. Also sind die Voraussetzungen (7.1) und (7.2) bezüglich der Zeilencluster auch für $\breve{b}$ erfüllt.
2. Fall: Es sei $\tilde{t} \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$. Dann gilt $\breve{t} \in \text{chil}(\tilde{t})$ und $\mathcal{T}_{\mathcal{I}}\backslash(\text{pred}(\breve{t})\backslash\{\breve{t}\}) \subseteq \mathcal{T}_{\mathcal{I}}\backslash(\text{pred}(\tilde{t})\backslash\{\tilde{t}\})$. Somit überträgt sich die Voraussetzung (7.1) für die Zeilen auf $\breve{b}$.
Wegen der Voraussetzung (7.1) ist W_s für alle $s \in \mathcal{T}_{\mathcal{J}}\backslash(\text{pred}(\tilde{s})\backslash\{\tilde{s}\})$ orthogonal. Nach Lemma 3.3.4 (iv) gilt $s \notin \text{pred}(\tilde{s})\backslash\{\tilde{s}\}$ für alle $s \in \text{row}(\tilde{t})$. Damit ist W_s orthogonal und $O_s := I_{\lambda_s}$ ein orthogonales Gewicht für alle $s \in \text{row}(\tilde{t})$. Also berechnet Algorithmus 7.1.3 in diesem Fall ein adaptives Gewicht für $\tilde{t}$ (siehe Lemma 4.4.19 und Bemerkung 4.4.20). Mit $\text{pred}(\breve{t})\backslash\{\breve{t}\} = (\text{pred}(\tilde{t})\backslash\{\tilde{t}\})\dot{\cup}\{\tilde{t}\}$ folgt die Voraussetzung (7.2) für die Zeilencluster. Der Algorithmus 7.1.2 skaliert die Kopplungsmatrizen $S_{(\tilde{t},s)}$ für alle $s \in \text{row}(\tilde{t})$. Dies benötigt maximal $\sum_{s\in\text{row}(\tilde{t})} \#\kappa_{\tilde{t}}\#\lambda_s \leq c_{sp}k_{\max}^2 \leq c_{sp}k_{\max}^3$ Operationen. Falls das Elter $\acute{t} := \text{par}(\tilde{t})$ existiert, kommt die Berechnung des skalierten Produkts $\frac{1}{\theta_{\acute{t}}}E_t Z_{\acute{t}}$ in höchstens $3\#\kappa_{\tilde{t}}\#\kappa_{\acute{t}}\#\zeta_{\acute{t}} \leq 3k_{\max}^3$ Operationen hinzu. Für die Berechnung der QR-Zerlegung werden nach Lemma 2.4.3 maximal

$$c_{qr}\#\ell\#\kappa_t \min\{\#\ell, \#\kappa_t\} \;\leq\; c_{qr}(c_{sp}k_{\max} + k_{\max})k_{\max}^2 \;=\; c_{qr}(c_{sp}+1)k_{\max}^3.$$

Für die Spaltenbasis können wir den Aufwand analog abschätzen und die Aussage folgt. ■

Mit Algorithmus 7.1.3 können wir sicherstellen, dass die Voraussetzungen (7.1) und (7.2) beim Abwärtslaufen im Blockbaum erhalten bleiben. Nach Lemma 7.1.3 werden die Bedingungen auch durch das Niedrigrangupdate der 1. Variante erhalten. Weil wir den

Algorithmus 7.1.3 nur einmal für die Rekursion aufrufen wollen und zwischen den rekursiven Aufrufen für die verschiedenen Kinder keine weiteren Hilfsberechnungen vornehmen wollen, müssen wir sichergehen, dass die Voraussetzungen (7.1) und (7.2) beim Wechsel von einem Kind zum nächsten erhalten bleibt.

Lemma 7.1.7
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $\breve{b} = (\breve{t}, \breve{s}) \in \operatorname{chil}(\tilde{b})$. Weiter seien die Voraussetzungen (7.1) und (7.2) für $\breve{b}$ erfüllt. Dann sind (7.1) und (7.2) für alle $b \in \operatorname{chil}(\tilde{b})$ erfüllt.

Beweis: Es sei $b = (t, s) \in \operatorname{chil}(\tilde{b})$. Wir betrachten wieder die Zeilencluster, weil die Aussagen für die Spaltencluster analog gelten.
1. Fall: Es sei $\tilde{t} \in \mathcal{L}_{\mathcal{I}}$. Dann gilt $t = \tilde{t} = \breve{t}$ und somit die Aussage.
2. Fall: Es sei $\tilde{t} \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$. Dann gilt $\operatorname{pred}(t)\backslash\{t\} = \operatorname{pred}(\tilde{t}) = \operatorname{pred}(\breve{t})\backslash\{\breve{t}\}$. Damit stellen die Voraussetzungen (7.1) und (7.2) für b und $\breve{b}$ bezüglich der Zeilencluster dieselben Bedingungen an die gleichen Mengen. Also folgt die Behauptung. ∎

Es bleibt, die Voraussetzungen beim Aufwärtslaufen im Blockbaum sicherzustellen. Dazu reicht es aus, die Clusterbasen $V_{\tilde{t}}$ und $W_{\tilde{s}}$ des aktuellen Blocks zu orthogonalisieren. (Wir zeigen dies in Lemma 7.1.8). Weil wir davon ausgehen können, dass die Clusterbasen der Kinder orthogonal sind, reicht ein Schritt (ohne Rekursion) von Algorithmus 4.2.1 aus. Dazu definieren wir Algorithmus 7.1.4.

Algorithmus 7.1.4 Die Funktion orthogonalisiert die Clusterbasis V im Cluster $t \in \mathcal{T}_{\mathcal{I}}\backslash\mathcal{L}_{\mathcal{I}}$ unter der Voraussetzung, dass sie für die Kinder orthogonal ist. Dabei wird die alte Clusterbasis V mit der neuen $\overline{V}$ überschrieben und der Basiswechsel C_t mitberechnet.

```
function ORTHOGONALE_CLUSTERBASIS_CLUSTER(t, E, κ, C, κ̄)
    ℓ ← ⨄_{ť∈chil(t)} κ_ť
    V̂_t ∈ ℝ^{ℓ×κ_t}
    for ť ∈ chil(t) do
        V̂_{t|ρ_ť×κ_t} ← E_ť
    end for
    QR-ZERLEGUNG(V̂_t, ℓ, ℓ, κ_t, Q̂_t, C_t, κ̄_t)        ▷ siehe Algorithmus 2.4.1
    for ť ∈ chil(t) do
        E_ť ← Q̂_{t|ρ_ť×κ̄_t}
    end for
    if t ≠ r_I then
        E_t̃ ← C_t E_t̃ ∈ ℝ^{κ̄_t×κ_t̃}
    end if
end function
```

Nachdem wir die Clusterbasis im Cluster t orthogonalisiert haben, müssen wir noch die $\mathcal{H}^2$-Matrix-Darstellung, d. h. die Kopplungsmatrizen, anpassen. Diese Anpassung können

wir wie bei der Orthogonalisierung einer $\mathcal{H}^2$-Matrix durch einfaches Multiplizieren der Kopplungsmatrizen mit den von Algorithmus 7.1.4 berechneten Basiswechseln vornehmen. Dabei entsteht wie bei der Orthogonalisierung einer $\mathcal{H}^2$-Matrix-Darstellung kein Fehler, weil das Bild der alten Clusterbasis im Bild der im Cluster orthogonalisierten Clusterbasis enthalten ist (siehe Lemma 4.2.8).

Algorithmus 7.1.5 Die Funktion aktualisiert die Kopplungsmatrizen durch die Multiplikation mit dem Basiswechsel C. Dabei wird für alle $s \in \operatorname{row}(t)$ die alte Kopplungsmatrix $S_{(t,s)}$ jeweils mit der neuen $\overline{S}_{(t,s)}$ überschrieben.

function AKTUALISIERE_KOPPLUNGSMATRIZEN_CLUSTER(t, S, κ, λ, C, $\overline{\kappa}$)
  for $s \in \operatorname{row}(t)$ **do**
    $S_s \leftarrow C_s S_s \in \mathbb{R}^{\overline{\kappa}_t \times \lambda_s}$
  end for
end function

Mit den Algorithmen 7.1.4 und 7.1.5 können wir also dafür sorgen, dass die Voraussetzung (7.1) auch beim Aufwärtslaufen im Blockbaum erfüllt bleibt. Wie beim Prolog müssen wir dabei unterscheiden, ob ein Cluster ein Blatt ist oder nicht. Dies fassen wir in Algorithmus 7.1.6 zusammen.

Algorithmus 7.1.6 Die Funktion stellt sicher, dass die Voraussetzungen (7.1) und (7.2) für dir $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ nach der Rekursion für die Kinder $\breve{b} \in \operatorname{chil}(\tilde{b})$ wieder für $\tilde{b}$ erfüllt werden. Dabei wird davon ausgegangen, dass die Voraussetzungen für die Kinder erfüllt sind.

function EPILOG_VARIANTE1($\tilde{b}$, H)
  $(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V, E), (W, F), N, S) \leftarrow H$
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$
  if $\tilde{t} \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ **then**
    ORTHOGONALE_CLUSTERBASIS_CLUSTER($\tilde{t}$, E, κ, C, $\overline{\kappa}$) ▷ Algorithmus 7.1.4
    AKTUALISIERE_KOPPLUNGSMATRIZEN_CLUSTER($\tilde{t}$, S, κ, λ, C, $\overline{\kappa}$)
      ▷ Algorithmus 7.1.5
  end if
  if $\tilde{s} \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{L}_{\mathcal{J}}$ **then**
    ORTHOGONALE_CLUSTERBASIS_CLUSTER($\tilde{s}$, F, λ, D, $\overline{\lambda}$) ▷ Algorithmus 7.1.4
    AKTUALISIERE_KOPPLUNGSMATRIZEN_CLUSTER($\tilde{s}$, S^T, λ, κ, D, $\overline{\lambda}$)
      ▷ Algorithmus 7.1.5
  end if
end function

In Lemma 7.1.8 zeigen wir, dass Algorithmus 7.1.6 die Voraussetzungen (7.1) und (7.2) von den Kindern auf den aktuellen Block überträgt. Dabei ist zu beachten, dass die Voraussetzungen für alle Kinder nach Lemma 7.1.7 äquivalent sind. Zusätzlich schätzen wir den Aufwand für den Algorithmus ab.

Lemma 7.1.8
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen $\kappa = (\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $\lambda = (\lambda_s)_{s\in\mathcal{T}_\mathcal{J}}$. Es seien $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $\breve{b} = (\breve{t}, \breve{s}) \in \operatorname{chil}(\tilde{b})$. Weiter seien die Voraussetzungen (7.1) und (7.2) für $\breve{b}$ erfüllt. Dann gelten nach dem Aufruf von Algorithmus 7.1.6 im Block $\tilde{b}$ die Voraussetzungen (7.1) und (7.2) für $\tilde{b}$ und der Algorithmus benötigt dafür höchstens

$$(c_{qr}(\#\operatorname{chil}(\tilde{t}) + \#\operatorname{chil}(\tilde{s})) + 4 + 4c_{sp})k_{\max}^3$$

Operationen, wobei $k_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\}$ sei.

Beweis: Wir betrachten wieder die Zeilencluster. Die Aussage für die Spaltencluster verläuft analog. Wir unterscheiden die beiden Fälle aus Algorithmus 7.1.6.
1. Fall: Es sei $\tilde{t} \in \mathcal{L}_\mathcal{I}$. Dann gilt $\tilde{t} = \breve{t}$ und die Voraussetzungen (7.1) und (7.2) gelten für den Zeilencluster $\tilde{t}$.
2. Fall: Es sei $\tilde{t} \in \mathcal{T}_\mathcal{I}\backslash\mathcal{L}_\mathcal{I}$. Dann ist $\breve{t} \in \operatorname{chil}(\tilde{t})$ und es gilt $\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\} \subset \operatorname{pred}(\breve{t})\backslash\{\breve{t}\}$. Also überträgt sich die Voraussetzung (7.2) an die adaptiven Gewichte von $\breve{t}$ auf $\tilde{t}$.
Außerdem gilt $\mathcal{T}_\mathcal{I}\backslash(\operatorname{pred}(\tilde{t})\backslash\{\tilde{t}\}) = (\mathcal{T}_\mathcal{I}\backslash(\operatorname{pred}(\breve{t})\backslash\{\breve{t}\}))\dot\cup\{\tilde{t}\}$. Also müssen wir, um die Voraussetzung (7.1) an die Orthogonalität der Clusterbasis V zu erfüllen, $V_{\tilde{t}}$ durch eine orthogonale Clusterbasis ersetzen und die übrige Darstellung der $\mathcal{H}^2$-Matrix anpassen.
Der Algorithmus 7.1.4 berechnet einen Schritt von Algorithmus 4.2.1. Dabei können wir die Basiswechsel aus den rekursiven Aufrufen in Algorithmus 4.2.1 durch Identitäten ersetzen, weil die Clusterbasen in den Kindern unverändert bleiben. Weil nach Voraussetzung (7.1) mit Lemma 7.1.7 die Clusterbasis V_t orthogonal für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}\backslash\{\tilde{t}\}$ ist, folgt wie in Lemma 4.2.8, dass die neue Clusterbasis in $\tilde{t}$ orthogonal und $\operatorname{Bild}(V_{\tilde{t}}) \subseteq \operatorname{Bild}(\overline{V}_{\tilde{t}})$ ist. Falls $\tilde{t} \neq r_\mathcal{I}$ ist, gilt dank der neuen Transfermatrix $\overline{E}_{\tilde{t}}$ für das Elter $\acute{t} := \operatorname{par}(\tilde{t})$, mit der wir $E_{\tilde{t}}$ überschreiben,

$$V_{\acute{t}|\tilde{\mathbf{t}}} \;=\; V_{\acute{t}}E_{\tilde{t}} \;=\; \overline{V}_{\acute{t}}\overline{V}_{\acute{t}}^T V_{\acute{t}}E_{\tilde{t}} \;=\; \overline{V}_{\acute{t}}C(V,\overline{V})_t E_{\tilde{t}} \;=\; \overline{V}_{\acute{t}}\overline{E}_{\tilde{t}}$$

und die Clusterbasis V bleibt in allen Clustern $t \in \mathcal{T}_\mathcal{I}\backslash\{\tilde{t}\}$ unverändert. Somit muss nur die Blockzeile von $\tilde{t}$ angepasst werden. Für alle $s \in \operatorname{row}(\tilde{t})$ gilt

$$H_{|\tilde{\mathbf{t}}\times\mathbf{s}} \;=\; V_{\tilde{t}}S_{(t,s)}W_s^T \;=\; \overline{V}_{\tilde{t}}C(V,\overline{V})_t S_{(t,s)}W_s^T \;=\; \overline{V}_{\tilde{t}}\overline{S}_{(t,s)}W_s^T.$$

Also erhalten wir eine $\mathcal{H}^2$-Matrix-Darstellung, deren Clusterbasis $\overline{V}$ die Voraussetzung (7.1) für den Block $\tilde{b}$ bezüglich der Zeilencluster erfüllt.
Abschließend betrachten wir noch den Aufwand. Falls $\tilde{t} \notin \mathcal{L}_\mathcal{I}$ ist, wird in Algorithmus 7.1.4 die Matrix $\widehat{V}$ aufgestellt, wobei keine arithmetischen Operationen benötigt werden. Der Aufwand für die QR-Zerlegung läßt sich mit Lemma 2.4.3 durch

$$c_{qr}\#\ell\#\kappa_t \min\{\#\ell, \#\kappa_t\} \;\leq\; c_{qr}\sum_{t'\in\operatorname{chil}(\tilde{t})} \#\kappa_{t'}\#\kappa_t^2 \;\leq\; c_{qr}k_{\max}^3\#\operatorname{chil}(\tilde{t})$$

beschränken. Für die neue Rangverteilung in $\tilde{t}$ gilt $\#\overline{\kappa}_{\tilde{t}} \leq \#\kappa_{\tilde{t}}$. Falls $\tilde{t} \neq r_{\mathcal{I}}$ ist, wird zusätzlich $C_{\tilde{t}}E_{\tilde{t}}$ berechnet, was höchstens $2\#\overline{\kappa}_{\tilde{t}}\#\kappa_{\tilde{t}}\#\kappa_{\mathrm{par}(\tilde{t})} \leq 2k_{\max}^3$ Operationen benötigt. Der Aufwand für die Berechnung der neuen Kopplungsmatrizen erfordert maximal

$$\sum_{s\in\mathrm{row}(\tilde{t})} 2\#\overline{\kappa}_{\tilde{t}}\#\kappa_{\tilde{t}}\#\lambda_s \;\leq\; \sum_{s\in\mathrm{row}(\tilde{t})} 2k_{\max}^3 \;=\; 2c_{sp}k_{\max}^3$$

Operationen. Der Aufwand für die Spaltenbasis lässt sich analog abschätzen. ∎

Mit Hilfe der Algorithmen 7.1.3 und 7.1.6 können wir sicherstellen, dass die Voraussetzungen (7.1) und (7.2) während der Rekursion erhalten bleiben. Um die Verwendung dieser Hilfsfunktionen zusammen mit den Niedrigrangupdates der 1. Variante zu veranschaulichen, definieren wir einen Prototypen für eine rekursive Funktion mit Niedrigrangupdates (siehe Algorithmus 7.1.7).

Algorithmus 7.1.7 Prototyp für einen rekursiven Algorithmus unter Verwendung des Niedrigrangupdates in der 1.Variante für eine $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$. Dabei verwenden wir die zusammengefasste Notation für Gewichte und Optionen nach Bemerkung 6.2.1

```
function PROTOTYP_VARIANTE1(b, H, Z, opt)
    (t, s) ← b
    (T_{I×J}, V, W, N, S) ← H
    Berechnung der Rang-K-Matrix R = AB^T ∈ R^{I×J}_{t×s}
    NIEDRIGRANGUPDATE_VARIANTE1(b, H, Z, R, opt)        ▷ Algorithmus 7.1.1
    if chil(b) ≠ ∅ then
        PROLOG_VARIANTE1(b̃, H, opt, Z)                  ▷ Algorithmus 7.1.3
        for b̆ ∈ chil(b) do
            PROTOTYP_VARIANTE1(b̆, H, Z, opt)
        end for
        EPILOG_VARIANTE1(b̃, H)                          ▷ Algorithmus 7.1.6
    end if
end function
```

Falls wir die lokalen Ränge durch $k_{\max}$ und die Anzahl der Söhne durch c_{chil} beschränken können, ist der Aufwand für die Hilfsfunktionen für jeden Block in $\mathcal{O}(k_{\max}^3)$. Nach Lemma 3.3.6 folgt, dass der Aufwand der Hilfsfunktionen während des Prototyps sich in $\mathcal{O}(nk_{\max}^2)$ befindet. Der Aufwand für das Niedrigrangupdate liegt nach Lemma 7.1.3 in $\mathcal{O}((\#\mathbf{t} + \#\mathbf{s})k_{\max}^2)$ und das Aufstellen der Niedrigrangmatrix hat typischerweise keine höhere Komplexität. Damit erhalten wir für den Aufwand dieses Teils des Prototyps nach Lemma 3.3.6 eine obere Schranke in $\mathcal{O}(n\log(n)k_{\max}^2)$. Im Vergleich zur Berechnung des Niedrigrangupdates inklusive der Niedrigrangmatrix haben die Hilfsfunktionen (Prolog und Epilog) also eine geringere Komplexität.

7.2 2. Variante

Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, für die wir Niedrigrangupdates durchführen wollen. Wie bei der 1. Variante gehen wir davon aus, dass die Updates während eines Algorithmus berechnet werden, der den Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ rekursiv durchläuft. Dabei wollen wir wie bei der 1. Variante beim Durchlaufen des Blockbaums die Voraussetzungen (7.1) und (7.2) für den aktuellen Block $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ erfüllen. Zusätzlich wollen wir bei der 2. Variante das Update derart gestalten, dass die Berechnungen nur in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$, $\mathcal{T}_{\tilde{\mathbf{s}}}$ und $\mathcal{T}_{\tilde{\mathbf{b}}}$ vorgenommen werden. Insbesondere wollen wir auf das Durchlaufen der Blockzeilen und -spalten verzichten.
Um dies zu gewährleisten, müssen wir einerseits die Informationen aus $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{b}}}$, die wir für das Update benötigen, in geeigneter Form vorberechnen. Bei diesen Informationen handelt es sich um die adaptiven Gewichte für die Eltern von $\tilde{t}$ und $\tilde{s}$ aus Bedingung (7.2) und die Kopplungsmatrizen zu zulässigen Blöcken $(t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ mit $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \operatorname{row}(t)\backslash\mathcal{T}_{\tilde{\mathbf{s}}}$ bzw. $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ und $t \in \operatorname{col}(s)\backslash\mathcal{T}_{\tilde{\mathbf{t}}}$, wobei die Kopplungsmatrizen geeignet zusammengefasst werden müssen.
Andererseits müssen die Änderungen des Updates, die Teile der Matrix in $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{b}}}$ betreffen, so gespeichert werden, dass sie später effizient angewendet werden können. Dabei handelt es sich um die Basiswechsel innerhalb von $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$, die wir nur auf Kopplungsmatrizen zu zulässigen Blättern in $\mathcal{T}_{\tilde{\mathbf{b}}}$ sofort anwenden wollen. Um sie später auch auf Kopplungsmatrizen zu zulässigen Blättern $(t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{T}_{\tilde{\mathbf{b}}}$ anwenden zu können, müssen wir sie in geeigneter Form speichern.
Zuerst betrachten wir, wie die Veränderungen des Niedrigrangupdates derart aufgeteilt werden können, dass nur Veränderungen in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ ohne Durchlaufen der Blockzeilen/-spalten berechnet werden. Dazu passen wir die Clusterbasen und die Kopplungsmatrizen in $\mathcal{T}_{\tilde{\mathbf{b}}}$ sofort an. Die Kopplungsmatrizen außerhalb des aktuellen Teilbaums $\mathcal{T}_{\tilde{\mathbf{b}}}$ wollen wir erst später aktualisieren. Dies führt zu der in Abbildung 7.1 beschriebenen Aufteilung der Berechnungen.
Um diesen Ansatz zu verwirklichen, müssen wir die Basiswechsel für die späteren Anpassungen speichern. Während der rekursiven Algorithmen werden diverse Niedrigrangupdates berechnet. Deshalb wäre es ineffizient alle Basiswechsel für die späteren Anpassungen direkt zu speichern, weil wir potentiell sehr viele Basiswechsel speichern müssten. An dieser Stelle nutzen wir aus, dass wir, statt Basiswechsel $C_{t,i}$, $i \in \{1, \dots, \mu\}$, für einen Zeilencluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ nacheinander auf eine Kopplungsmatrix $S_{(t,s)}$ anzuwenden, auch erst das Produkt $C_t = \prod_{i=1}^{\mu} C_{t,i}$ der Basiswechsel berechnen und anschließend das Produkt mit der Kopplungsmatrix multiplizieren können:

$$C_{t,\mu}(C_{t,\mu-1}(\dots(C_{t,2}(C_{t,1}S_{t,s}))\dots)) \;=\; \left(\prod_{i=1}^{\mu} C_{t,i}\right) S_{t,s} \;=:\; C_t S_{t,s}.$$

Das hat den Vorteil, dass wir nur eine Matrix C_t speichern müssen und nicht alle Basiswechsel. Falls ein weiterer Basiswechsel hinzukommt, können wir diesen einfach von links zu C_t multiplizieren. Außerdem vermeiden wir Aufwand, weil wir, statt potentiell viele Kopplungsmatrizen mit jedem der Basiswechsel zu multiplizieren, vorerst nur C_t mit dem

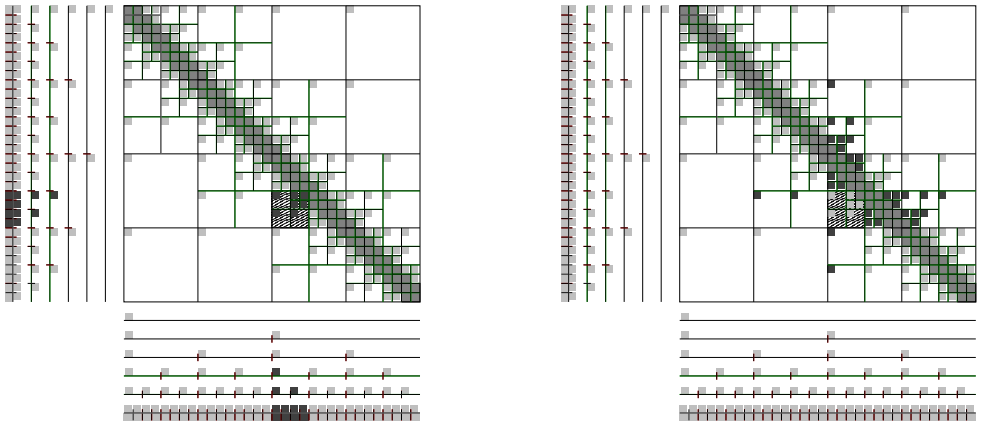


Abbildung 7.1: Aufteilung des Niedrigrangupdates: Links sind die Matrizen markiert, die sofort angepasst werden, und rechts die Matrizen, die später aktualisiert werden.

Basiswechsel und dann später die Kopplungsmatrizen einmal mit C_t multiplizieren. Deshalb speichern wir für jedes $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ das Produkt der Basiswechsel, die aus Niedrigrangupdates in Blöcken in $\mathcal{T}_{\tilde{\mathbf{b}}}$ stammen, und bezeichnen diese mit C_t. Analog speichern wir für alle $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ das Produkt von Basiswechseln D_s. Diese Produkte von Basiswechseln müssen auf die Kopplungsmatrizen angewendet werden, die in der rechten Grafik in Abbildung 7.1 markiert sind. Damit können wir für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ die aktuelle Kopplungsmatrix S_b aus der aktuell gespeicherten $\widehat{S}_b$ durch

$$S_b \;=\; \begin{cases} \widehat{S}_b & \text{, falls } b \in \mathcal{T}_{\tilde{\mathbf{b}}} \\ C_t \widehat{S}_b & \text{, falls } b \notin \mathcal{T}_{\tilde{\mathbf{b}}} \wedge t \in \mathcal{T}_{\tilde{\mathbf{t}}} \\ \widehat{S}_b D_s & \text{, falls } b \notin \mathcal{T}_{\tilde{\mathbf{b}}} \wedge s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ \widehat{S}_b & \text{sonst} \end{cases}$$

berechnen. Dabei ist zu beachten, dass für alle $(t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ nach Lemma 3.3.14 (iii) genau dann $(t, s) \in \mathcal{T}_{\tilde{\mathbf{b}}}$ gilt, falls $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ ist. Es sei an dieser Stelle darauf hingewiesen, dass die Kopplungsmatrizen somit außerhalb des aktuellen Teilbaums $\mathcal{T}_{\tilde{\mathbf{b}}}$ nie mit beiden Basiswechseln multipliziert werden müssen.
Um die Berechnungen auch während des Durchlaufens des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ lokal im aktuellen Block zu halten, dürfen wir beim Übergang zum Elter-Block $\acute{b} := \mathrm{par}(\tilde{b})$ nur die Kopplungsmatrizen zu Blöcken in $\mathrm{desc}(\acute{b})\backslash\mathcal{T}_{\tilde{\mathbf{b}}}$ aktualisieren. Und beim Übergang zu einem Kind $\breve{b} \in \mathrm{chil}(b)$ müssen wir die Produkte der Basiswechsel, die zum aktuellen Block gehören, weiterhin speichern. Deshalb speichern wir für jeden Vorfahren entsprechende Produkte von Basiswechseln. Dies führt zu dem rekursiven Speicherformat für die Kopplungsmatrizen, dass wir im folgenden Abschnitt beschreiben.

7.2.1 Rekursives Speicherformat der Kopplungsmatrizen

In diesem Abschnitt stellen wir ein rekursives Speicherformat für Kopplungsmatrizen vor. Dabei nutzen wir den oben beschriebenen Ansatz, die Basiswechsel zusammenzufassen, die noch auf Kopplungsmatrizen außerhalb des Teilbaums $\mathcal{T}_{\tilde{\mathbf{b}}}$ angewendet werden müssen. Wir erläutern zuerst die Darstellung, bevor wir sie exakt definieren. Danach geben wir an, wie sich die Darstellung durch ein Niedrigrangupdate, das Abwärtslaufen im Baum oder das Aufwärtslaufen im Baum verändert.
Wie bereits erwähnt, sammeln wir für jeden Vorfahren $\hat{b} \in \operatorname{pred}(\tilde{b})$ des aktuellen Blocks die Basiswechsel, die auf Kopplungsmatrizen außerhalb des Blocks angewendet werden müssen. Um dies besser beschreiben zu können, bezeichnen wir den Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$ mit $(\tilde{b}_i)_{i=0}^m = (\tilde{t}_i, \tilde{s}_i)_{i=0}^m$. Dann speichern wir für alle $i \in \{1, \dots, m\}$ die Familien $(C_{i,t})_{t\in\operatorname{desc}(\tilde{t}_i)}$ und $(D_{i,s})_{s\in\operatorname{desc}(\tilde{s}_i)}$, die die Basiswechsel beschreiben, die auf Kopplungsmatrizen außerhalb von $\operatorname{desc}(\tilde{b}_i)$ angewendet werden müssen. Für $i = 0$ speichern wir keine Basiswechsel, weil $\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \setminus \operatorname{desc}(\tilde{b}_0) = \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \setminus \operatorname{desc}(r_{\mathcal{I}\times\mathcal{J}}) = \emptyset$ gilt.
Zur besseren Beschreibung der Basiswechsel, die auf eine Kopplungsmatrix angewendet werden, definieren wir Hilfsmengen. Diese beschreiben für $i \in \{0, \dots, m\}$ jeweils den zugehörigen Teilbaum $\operatorname{desc}(\tilde{b}_i)$, wobei für $i \in \{0, \dots, m-1\}$ der Teil des Kindes $\operatorname{desc}(\tilde{b}_{i+1})$ ausgenommen ist. Die Mengen sind in Abbildung 7.2 an einem Beispiel illustriert.

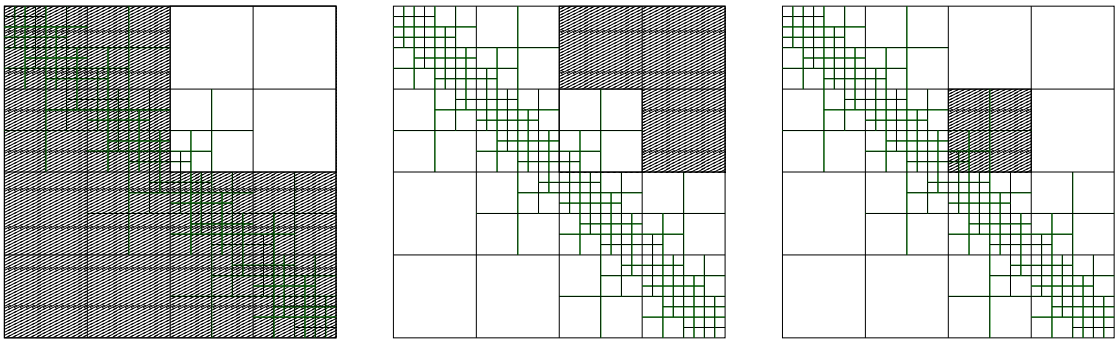
Definition 7.2.1
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. (Es gilt $m = \operatorname{level}(\tilde{b})$.) Dann definieren wir die Mengen

$$\mathcal{M}_{\tilde{b},i} := \operatorname{desc}(\tilde{b}_i) \setminus \operatorname{desc}(\tilde{b}_{i+1}) \quad \text{für alle } i \in \{0, \dots, m-1\} \quad \text{und} \quad \mathcal{M}_{\tilde{b},m} := \mathcal{T}_{\tilde{\mathbf{b}}}.$$

Zusätzlich bezeichnen wir die zulässigen Blätter dieser Mengen mit $\mathcal{M}^+_{\tilde{b},i} := \mathcal{M}_{\tilde{b},i} \cap \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ für alle $i \in \{0, \dots, m\}$.

Bemerkung 7.2.2
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Dann bilden die Mengen $\mathcal{M}_{\tilde{b},i}$, $i \in \{0, \dots, m\}$, eine Partition von $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Entsprechend bilden die Mengen $\mathcal{M}^+_{\tilde{b},i}$, $i \in \{0, \dots, m\}$, eine Partition von $\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.

Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Dann existiert nach Bemerkung 7.2.2 genau ein $i \in \{0, \dots, m\}$ mit $b \in \mathcal{M}_{\tilde{b},i}$. Weil $(\tilde{b}_i)_{i=0}^m$ ein Pfad ist, gilt $b \in \mathcal{M}_{\tilde{b},i} \subseteq \operatorname{desc}(\tilde{b}_i) \subseteq \operatorname{desc}(\tilde{b}_j)$ für alle $j \in \{0, \dots, i\}$ und, falls $i \neq m$ ist, $b \notin \operatorname{desc}(\tilde{b}_{i+1}) \supseteq \operatorname{desc}(\tilde{b}_j)$ für alle $j \in \{i+1, \dots, m\}$. Weil die Basiswechsel nur auf den Teil des Blockbaums außerhalb des zugehörigen Blocks $\tilde{b}_j$ angewendet werden sollen, müssen wir für die Kopplungsmatrix S_b nur die Basiswechsel zu den Blöcken $\tilde{b}_j$ mit $j \in \{i+1, \dots, m\}$ berücksichtigen.
Auf der anderen Seite müssen die Basiswechsel $(C_{j,\check{t}})_{\check{t}\in\operatorname{desc}(t_j)}$ zum Block $\tilde{b}_j$ nur dann auf S_b angewendet werden, wenn $t \in \operatorname{desc}(t_j)$ ist. Es sei darauf hingewiesen, dass ansonsten für t keine Matrix $C_{j,t}$ gespeichert ist, weil S_b in solchen Fällen nicht von den Niedrigrangupdates im Block $\tilde{b}_j$ verändert wird. Entsprechendes gilt für die Basiswechsel in



(a) $\mathcal{M}_{\tilde{b},0}$ (b) $\mathcal{M}_{\tilde{b},1}$ (c) $\mathcal{M}_{\tilde{b},2}$

Abbildung 7.2: Die Abbildungen zeigen die Hilfsmengen aus Definition 7.2.1 für einen Block $\tilde{b}$ mit $\text{level}(\tilde{b}) = 2$.

den Spaltenclustern. Um mit diesen beiden Überlegungen formal zu beschreiben, welche Basiswechsel auf S_b angewendet werden müssen, definieren wir folgende Hilfsgrößen.

Definition 7.2.3
Es sei $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(\tilde{b}_i)_{i=0}^m = (\tilde{t}_i, \tilde{s}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Dann sei für alle $b \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ der Index $i_{\tilde{b},b} \in \{0,\dots,m\}$ derart, dass $b \in \mathcal{M}_{\tilde{b},i_{\tilde{b},b}}$ gilt. Des Weiteren sei

$$\begin{aligned} i_{\tilde{b},t} &:= \max\{j \in \{0,\dots,m\} \mid t \in \text{desc}(\tilde{t}_j)\} \quad \text{für alle } t \in \mathcal{T}_{\mathcal{I}} \text{ und} \\ i_{\tilde{b},s} &:= \max\{j \in \{0,\dots,m\} \mid s \in \text{desc}(\tilde{s}_j)\} \quad \text{für alle } s \in \mathcal{T}_{\mathcal{J}}. \end{aligned}$$

Wir verzichten bei der Definition $i_{\tilde{b},s}$ darauf, mit dem transponierten Block $\tilde{b}^T$ oder einer speziellen Kennzeichnung für die Spaltencluster zu arbeiten, weil die Bedeutung der beiden Größen $i_{\tilde{b},t}$ und $i_{\tilde{b},s}$ grundsätzlich durch den Kontext deutlich werden sollte.

Bemerkung 7.2.4
Weil die Mengen $\mathcal{M}_{\tilde{b},i}$, $i \in \{0,\dots,m\}$, eine Partition von $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ bildet ist $i_{\tilde{b},b}$ für alle $b \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ wohldefiniert. Außerdem gilt $b \in \text{desc}(b_j)$ für alle $j \in \{0,\dots,i_{\tilde{b},b}\}$ und $b \notin \text{desc}(b_j)$ für alle $j \in \{i_{\tilde{b},b}+1,\dots,m\}$, da $(\tilde{b}_i)_{i=0}^m$ ein Pfad ist.
Wegen $\mathcal{T}_{\mathcal{I}} = \text{desc}(r_{\mathcal{I}}) = \text{desc}(t_0)$ ist $i_{\tilde{b},t}$ für alle $t \in \mathcal{T}_{\mathcal{I}}$ definiert. Außerdem gilt $t \in \text{desc}(t_j)$ für alle $j \in \{0,\dots,i_{\tilde{b},t}\}$, weil $(\tilde{b}_i)_{i=0}^m$ ein Pfad ist, und nach Definition $t \notin \text{desc}(t_j)$ für alle $j \in \{i_{\tilde{b},t}+1,\dots,m\}$. Entsprechendes gilt für $i_{\tilde{b},s}$, $s \in \mathcal{T}_{\mathcal{J}}$.

Mit diesen Hilfsgrößen und den vorherigen Überlegungen erhalten wir, dass auf eine Kopplungsmatrix S_b mit $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ die Basiswechsel $C_{j,t}$ mit $j \in \{i_{\tilde{b},b}+1,\dots,i_{\tilde{b},t}\}$ und $D_{j,s}$ mit $j \in \{i_{\tilde{b},b}+1,\dots,i_{\tilde{b},s}\}$ angewendet werden müssen. Weil die Basiswechsel während eines rekursiven Algorithmus berechnet werden und wir davon ausgehen, dass die Kopplungsmatrizen im aktuellen Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ nicht mehr angepasst werden müssen, sind

$(C_{1,\check{t}})_{\check{t}\in\operatorname{desc}(t_1)}$ und $(D_{1,\check{s}})_{\check{s}\in\operatorname{desc}(s_1)}$ die ältesten und $(C_{m,\check{t}})_{\check{t}\in\operatorname{desc}(t_m)}$ und $(D_{m,\check{s}})_{\check{s}\in\operatorname{desc}(s_m)}$ die neuesten Basiswechsel. Die Basiswechsel müssen in der Reihenfolge des Alters angewendet werden, wobei mit den ältesten begonnen wird. Dieses Vorgehen ergibt die folgende Definition.

Definition 7.2.5 (Rekursive Darstellung von Kopplungsmatrizen)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m = ((\tilde{t}_i, \tilde{s}_i))_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Für alle $i \in \{0,\dots,m\}$ seien Rangverteilungen κ_i auf $\operatorname{desc}(\tilde{t}_i)$ und λ_i auf $\operatorname{desc}(\tilde{s}_i)$ sowie Familien von Matrizen S_i mit $S_{i,b} \in \mathbb{R}^{\kappa_{i,t}\times\lambda_{i,s}}$, $b = (t,s) \in \mathcal{M}^+_{\tilde{b},i}$, gegeben. Weiter seien für alle $i \in \{1,\dots,m\}$ Familien von Matrizen C_i mit $C_{i,t} \in \mathbb{R}^{\kappa_{i,t}\times\kappa_{i-1,t}}$, $t \in \operatorname{desc}(\tilde{t}_i)$, und D_i mit $D_{i,s} \in \mathbb{R}^{\lambda_{i,s}\times\lambda_{i-1,s}}$, $s \in \operatorname{desc}(\tilde{s}_i)$, gegeben. Falls für alle $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$

$$S_b = \underbrace{\left(\prod_{j=i_b+1}^{i_t} C_{j,t}\right)}_{\in\mathbb{R}^{\kappa_{i_t}\times\kappa_{i_b}}} S_{i_b,b} \underbrace{\left(\prod_{j=i_b+1}^{i_s} D_{j,s}\right)^T}_{\in\mathbb{R}^{\lambda_{i_s}\times\lambda_{i_b}}} \tag{7.3}$$

gilt, wobei $i_b := i_{\tilde{b},b}$, $i_t := i_{\tilde{b},t}$ und $i_s := i_{\tilde{b},s}$ seien, ist das Tupel $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine *rekursive Darstellung der Kopplungsmatrizen* S für den Block $\tilde{b}$. Wir schreiben auch $C_{b_i} := C_i$ und $C_{b_i^T} := D_i$.

Für die Produkte in Definition 7.2.5 verwenden wir die bereits erwähnte Konvention, dass die Matrix zum unteren Index rechts und die Matrix zum oberen Index links steht. Als Abbildungen betrachtet, werden sie also auf ein rechts stehendes Objekt in der Reihenfolge des Produkts angewendet. Falls der untere Index größer als der obere Index ist, setzen wir das Produkt gleich einer passenden Identität. Zuerst prüfen wir, ob die Kopplungsmatrizen im Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ aktuell sind.

Bemerkung 7.2.6
Weil $t \in \operatorname{desc}(\tilde{t})$ und $s \in \operatorname{desc}(\tilde{s})$ für $b = (t,s) \in \mathcal{M}^+_{\tilde{b},m}$ gilt (siehe Lemma 3.3.14 (iii)), ist in diesem Fall $i_{\tilde{b},b} = i_{\tilde{b},t} = i_{\tilde{b},s} = m$ und es folgt, dass beide Produkte in Definition 7.2.5 nach Konvention die Identität sind. Also gilt $S_b = S_{m,b}$ für alle $b \in \mathcal{M}^+_{\tilde{b},m} = \mathcal{L}^+_{\tilde{\mathbf{b}}}$. Für den Teilbaum des aktuellen Blocks $\mathcal{T}_{\tilde{\mathbf{b}}}$ ist die Darstellung somit aktuell.

Damit wir die rekursive Darstellung praktisch umsetzen können, müssen wir uns überlegen, wie sich die Darstellung speichern lässt.

Bemerkung 7.2.7
Wegen Bemerkung 7.2.2 können wir die Matrizen $S_{i,b}$ für alle $i \in \{0,\dots,m\}$ und $b \in \mathcal{M}^+_{\tilde{b},i}$ anstatt der Matrix S_b speichern. Dabei bleibt die Abschätzung für den Speicheraufwand in Lemma 3.4.25 gültig, solange alle lokalen Ränge $\#\kappa_{i,t}$ und $\#\lambda_{i,s}$ durch $k_{\max}$ beschränkt sind.

Zusätzlich müssen wir die Basiswechsel $C_{\tilde{b}_i,t} := C_{i,t}$ und $D_{\tilde{b}_i,s} := D_{i,s}$ für alle $i \in \{1,\ldots,m\}$, $t \in \operatorname{desc}(\tilde{t}_i)$ und $s \in \operatorname{desc}(\tilde{s}_i)$ speichern. Der Aufwand dafür ist höchstens

$$k_{\max}^2 \left(\sum_{i=1}^{m} \# \operatorname{desc}(\tilde{t}_i) + \# \operatorname{desc}(\tilde{s}_i) \right),$$

falls die lokalen Ränge durch $k_{\max}$ beschränkt sind.

Wir müssen, wie bereits erwähnt, mit i_t statt m als obere Schranke arbeiten, da $C_{j,t}$ nur für $t \in \operatorname{desc}(\tilde{t}_j)$ definiert ist. Der Grund dafür liegt in der Tatsache, dass lokale Projektionen für einen Block $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ nur die Kopplungsmatrizen zu Blöcken mit Zeilenclustern in $\operatorname{desc}(t)$ oder Spaltenclustern in $\operatorname{desc}(s)$ verändern.
Als Nächstes untersuchen wir das Verhalten der rekursiven Darstellung während der Rekursion. Als Erstes geben wir eine Konstruktion einer rekursiven Darstellung für ein Kind aus der aktuellen Darstellung an, um das Abwärtslaufen im Blockbaum zu ermöglichen. Danach beschreiben wir die notwendigen Veränderungen, die durch ein lokales Niedrigrangupdate entstehen. Und zum Abschluss des Abschnitts konstruieren wir aus der aktuellen Darstellung eine Darstellung für den Elter, um das Aufwärtslaufen im Blockbaum zu erklären. An dieser Stelle untersuchen wir die drei Schritte formal ohne Angabe von Algorithmen, um uns auf die Funktionsweise der rekursiven Darstellung zu konzentrieren.
Es sei $\breve{b} \in \operatorname{chil}(\tilde{b})$. Weil die Kopplungsmatrizen im Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ aktuell sind, gilt dies auch für den Teilbaum $\mathcal{T}_{\breve{\mathbf{b}}}$ des Kindes. Also können die Kopplungsmatrizen direkt übernommen werden. Weil für das Kind noch keine Niedrigrangupdates berechnet wurden, können die Hilfsmatrizen für die Basiswechsel im Teilbaum $\mathcal{T}_{\breve{\mathbf{b}}}$ mit Identitäten initialisiert werden. Die restlichen Hilfsmatrizen können von der rekursiven Darstellung des aktuellen Blocks übernommen werden. Damit erhalten wir die Darstellung in Lemma 7.2.8.

Lemma 7.2.8
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m = ((\tilde{t}_i, \tilde{s}_i))_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Weiter sei eine rekursive Darstellung $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ der Kopplungsmatrizen S in $\tilde{b}$ gegeben.
Es sei $\breve{b} \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ und $\breve{b} = (\breve{t}, \breve{s}) \in \operatorname{chil}(\tilde{b})$. Wir definieren $\breve{S}_i := S_i$ für alle $i \in \{0,\ldots,m-1\}$, $\breve{S}_{m,b} = S_{m,b}$ für alle $b \in \mathcal{M}^+_{\breve{b},m}$, $\breve{S}_{m+1,b} = S_{m,b}$ für alle $b \in \mathcal{M}^+_{\breve{b},m+1}$, $\breve{C}_i := C_i$ und $\breve{D}_i := D_i$ für alle $i \in \{1,\ldots,m\}$, $\breve{C}_{m+1,t} := I_{\kappa_{m,t}}$ für alle $t \in \operatorname{desc}(\breve{t})$ und $\breve{D}_{m+1,s} := I_{\lambda_{m,s}}$ für alle $s \in \operatorname{desc}(\breve{s})$. Dann ist $((\breve{S}_i)_{i=0}^{m+1}, (\breve{C}_i)_{i=1}^{m+1}, (\breve{D}_i)_{i=1}^{m+1})$ eine rekursive Darstellung der Kopplungsmatrizen S im Block $\breve{b}$.

Beweis: Wir untersuchen die Bedingung (7.3) aus Definition 7.2.5 für den Block $\breve{b}$ und die oben definierten Matrizen. Es sei $b \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Zuerst halten wir fest, dass nach Definition $i_{\tilde{b},b} = \min\{i_{\breve{b},b}, m\}$, $i_{\tilde{b},t} = \min\{i_{\breve{b},t}, m\}$ und $i_{\tilde{b},s} = \min\{i_{\breve{b},s}, m\}$ gilt. Nach der obigen Definition der Familien $\breve{S}_i$ folgt

$$S_{i_{\tilde{b},b},b} \;=\; S_{\min\{i_{\breve{b},b},\,m\},b} \;=\; \breve{S}_{i_{\breve{b},b},b}.$$

Falls $b \in \mathcal{M}_{\breve{b},m+1} = \operatorname{desc}(\breve{b}) \subset \mathcal{T}_{\tilde{\mathbf{b}}}$ ist, gilt nach Bemerkung 7.2.6

$$S_b = \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t}\right) S_{i_{\tilde{b},b},b} \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}} D_{j,s}\right)^T = S_{m,b}$$
$$= \breve{S}_{m+1,b} = \left(\prod_{j=i_{\breve{b},b}+1}^{i_{\breve{b},t}} \breve{C}_{j,t}\right) \breve{S}_{i_{\breve{b},b},b} \left(\prod_{j=i_{\breve{b},b}+1}^{i_{\breve{b},s}} \breve{D}_{j,s}\right)^T.$$

Es sei also im Folgenden $b \notin \operatorname{desc}(\breve{b})$. Dann ist $i_{\tilde{b},b} = i_{\breve{b},b} \in \{0,\dots,m\}$. Wegen $\breve{C}_{m+1,t} = I_{\kappa_{m,t}}$ für alle $t \in \operatorname{desc}(\breve{t})$ folgt

$$\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t} = \prod_{j=i_{\tilde{b},b}+1}^{\min\{i_{\breve{b},t},m\}} C_{j,t} = \prod_{j=i_{\breve{b},b}+1}^{\min\{i_{\breve{b},t},m\}} \breve{C}_{j,t} = \prod_{j=i_{\breve{b},b}+1}^{i_{\breve{b},t}} \breve{C}_{j,t}.$$

Die entsprechende Gleichheit folgt analog für das rechte Produkt in (7.3). Damit erhalten wir

$$S_b = \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t}\right) S_{i_{\tilde{b},b},b} \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}} D_{j,s}\right)^T = \left(\prod_{j=i_{\breve{b},b}+1}^{i_{\breve{b},t}} \breve{C}_{j,t}\right) \breve{S}_{i_{\breve{b},b},b} \left(\prod_{j=i_{\breve{b},b}+1}^{i_{\breve{b},s}} \breve{D}_{j,s}\right)^T$$

und die Behauptung folgt. ∎

Während der Algorithmen wollen wir stets nur die rekursive Darstellung des aktuellen Blocks speichern. Deshalb betrachten wir, wie die alte Darstellung durch die im Kind überschrieben werden kann.

Bemerkung 7.2.9
Weil wir die Kopplungsmatrizen $S_{i,b}$ der rekursiven Darstellung anstatt der echten Kopplungsmatrizen speichern und die Matrizen $\breve{S}_{m+1,b}$ in Lemma 7.2.8 gleich $S_{m,b}$ gesetzt werden, müssen wir an den Kopplungsmatrizen keine Änderungen vornehmen. Genauso übernehmen wir die Familien C_j und D_j für alle $j \in \{0,\dots,m\}$. Die Matrizen $\breve{C}_{m+1,t}$, $t \in \operatorname{desc}(\breve{t})$, und $\breve{D}_{m+1,s}$, $s \in \operatorname{desc}(\breve{s})$, werden als Identitäten angelegt.

Nach dem Abwärtslaufen im Blockbaum betrachten wir das Niedrigrangupdate. Das Ziel ist eine rekursive Darstellung für die neu berechnete $\mathcal{H}^2$-Matrix. Dabei werden die Kopplungsmatrizen im Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ direkt angepasst. Um die rekursive Darstellung für die Kopplungsmatrizen außerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$ zu aktualisieren, multiplizieren wir die Familien C_m und D_m mit den entsprechenden Kopplungsmatrizen. Die restlichen Matrizen der rekursiven Darstellung bleiben unverändert.

Lemma 7.2.10
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ mit Rangverteilungen κ und λ, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ mit Rang-$\mathcal{K}$-Darstellung sowie $\widetilde{H} = \widetilde{H}(H, R, \tilde{b})$ mit Clusterbasen $\widetilde{V}$

und $\widetilde{W}$. Weiter seien $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen, $\overline{V}$ und $\overline{W}$ die Clusterbasen nach Lemma 5.2.1 sowie $\overline{H} := \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[\widetilde{H}]$. Es sei $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$ und $((S_i)_{i=0}^m,(C_i)_{i=1}^m,(D_i)_{i=1}^m)$ eine rekursive Darstellung von S.
Wir definieren $\overline{S}_i := S_i$ für alle $i\in\{0,\dots,m-1\}$, $\overline{S}_{m,b} := \overline{S}_b$ für alle $b\in\mathcal{M}_{\tilde{b},m}^+ = \mathcal{L}_{\tilde{\mathbf{b}}}^+$, $\overline{C}_i := C_i$ und $\overline{D}_i := D_i$ für alle $i\in\{1,\dots,m-1\}$, $\overline{C}_{m,t} := C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t}C_{m,t}$ für alle $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\overline{D}_{m,s} := C(\widetilde{W},\overline{W})_{s|\overline{\lambda}_s\times\lambda_s}D_{m,s}$ für alle $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$. Dann ist $((\overline{S}_i)_{i=0}^m,(\overline{C}_i)_{i=1}^m,(\overline{D}_i)_{i=1}^m)$ eine rekursive Darstellung der Kopplungsmatrizen $\overline{S}$ von $\overline{H}$ (siehe Lemma 5.3.16).

Beweis: Es sei $b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+$. Wir untersuchen die vier Fälle aus der Darstellung von $\overline{S}$ aus Lemma 5.3.16.
1. Fall: Es sei $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt $b\in\mathcal{L}_{\tilde{\mathbf{b}}}^+ = \mathcal{M}_{\tilde{b},m}^+$ und nach Bemerkung 7.2.6

$$\overline{S}_b \;=\; \overline{S}_{m,b} \;=\; \left(\prod_{j=i_{\tilde{b},t}+1}^{i_{\tilde{b},t}}\overline{C}_{j,t}\right)\overline{S}_{i_{\tilde{b},b},b}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}}\overline{D}_{j,s}\right)^T.$$

2. Fall: Es sei $t\in\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\notin\mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt $b\notin\mathcal{L}_{\tilde{\mathbf{b}}}^+ = \mathcal{M}_{\tilde{b},m}^+$ und somit $i_{\tilde{b},b}\in\{0,\dots,m-1\}$. Wegen $i_{\tilde{b},t}=m\geq i+1$, $i_{\tilde{b},s}<m$ und $C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t}C_{m,t}=\overline{C}_{m,t}$ folgt

$$\begin{aligned}
\overline{S}_b \;&=\; C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t}S_b \;=\; C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}}C_{j,t}\right)S_{i_{\tilde{b},b},b}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}}D_{j,s}\right)^T\\
&=\; C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t}C_{m,t}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}-1}C_{j,t}\right)S_{i_{\tilde{b},b},b}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}}D_{j,s}\right)^T\\
&=\; \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}}\overline{C}_{j,t}\right)\overline{S}_{i_{\tilde{b},b},b}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}}\overline{D}_{j,s}\right)^T.
\end{aligned}$$

3. Fall: Es sei $t\notin\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\in\mathcal{T}_{\tilde{\mathbf{s}}}$. In diesem Fall folgt die Aussage analog zum 2. Fall, wobei der Basiswechsel $C(\widetilde{W},\overline{W})_{s|\overline{\lambda}_s\times\lambda_s}$ zur Matrix $D_{m,s}$ hinzumultipliziert wird.
4. Fall: Es sei $t\notin\mathcal{T}_{\tilde{\mathbf{t}}}$ und $s\notin\mathcal{T}_{\tilde{\mathbf{s}}}$. Dann gilt $b\notin\mathcal{L}_{\tilde{\mathbf{b}}}^+ = \mathcal{M}_{\tilde{b},m}^+$ und $i_{\tilde{b},b}\in\{0,\dots,m-1\}$. Wegen $i_{\tilde{b},t}<m$ und $i_{\tilde{b},s}<m$ folgt

$$\begin{aligned}
\overline{S}_b \;=\; S_b \;&=\; \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}}C_{j,t}\right)S_{i_{\tilde{b},b},b}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}}D_{j,s}\right)^T\\
&=\; \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}}\overline{C}_{j,t}\right)\overline{S}_{i_{\tilde{b},b},b}\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}}\overline{D}_{j,s}\right)^T.
\end{aligned}$$

Also ist $((\overline{S}_i)_{i=0}^m,(\overline{C}_i)_{i=1}^m,(\overline{D}_i)_{i=1}^m)$ eine rekursive Darstellung von $\overline{S}$. ∎

Wir können die durch die lokale $\mathcal{H}^2$-Matrix-Projektion notwendigen Anpassungen der rekursiven Darstellung also durch Berechnungen in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{b}}}$, $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ absolvieren.
Als Nächstes untersuchen wir, wie wir aus einer gegebenen rekursiven Darstellung für den aktuellen Block $\tilde{b} \neq r_{\mathcal{I}\times\mathcal{J}}$ eine solche Darstellung für das Elter $\acute{b} = \operatorname{par}(\tilde{b})$ definieren können. Die Kopplungsmatrizen müssen nur für den Teilbaum des Elter ohne den Teil für den aktuellen Block aktualisiert werden, also in $\operatorname{desc}(\acute{b}) \setminus \operatorname{desc}(\tilde{b}) = \mathcal{M}^+_{\tilde{b},m-1}$. Außerdem müssen die Basiswechsel aus $\mathcal{T}_{\tilde{\mathbf{b}}}$ nach außen weitergegeben werden. Dadurch dass die Basiswechsel in Form von Produkten vorliegen, genügt es, die Familien C_{m-1} und D_{m-1} für den Elterblock in den Cluster in $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ mit den Familien C_m und D_m im aktuellen Block zu multiplizieren.

Lemma 7.2.11
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m = ((\tilde{t}_i, \tilde{s}_i))_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Weiter sei eine rekursive Darstellung $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ der Kopplungsmatrizen S in $\tilde{b}$ gegeben.
Es sei $\tilde{b} \neq r_{\mathcal{I}\times\mathcal{J}}$ und $\acute{b} = (\acute{t}, \acute{s}) := \operatorname{par}(\tilde{b})$. Wir definieren $\acute{S}_i := S_i$ für alle $i \in \{0, \dots, m-2\}$, $\acute{C}_i := C_i$ und $\acute{D}_i := D_i$ für alle $i \in \{1, \dots, m-2\}$. Für alle $b \in \operatorname{desc}(\acute{b}) \cap \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, $t \in \operatorname{desc}(\acute{t})$ und $s \in \operatorname{desc}(\acute{s})$ sei

$$\acute{S}_{m-1,b} := \begin{cases} S_{m,b} & \text{, falls } b \in \mathcal{M}^+_{\tilde{b},m} \\ C_{m,t} S_{m-1,b} & \text{, falls } b = (t,s) \in \mathcal{M}^+_{\tilde{b},m-1} \wedge t \in \mathcal{T}_{\tilde{\mathbf{t}}} \\ S_{m-1,b} D^T_{m,t} & \text{, falls } b = (t,s) \in \mathcal{M}^+_{\tilde{b},m-1} \wedge s \in \mathcal{T}_{\tilde{\mathbf{s}}} \\ S_{m-1,b} & \text{, falls } b = (t,s) \in \mathcal{M}^+_{\tilde{b},m-1} \wedge t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \wedge s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \end{cases},$$

$$\acute{C}_{m-1,t} := \begin{cases} C_{m-1,t} & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \\ C_{m,t} C_{m-1,t} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}} \end{cases}$$

und

$$\acute{D}_{m-1,t} := \begin{cases} D_{m-1,t} & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \\ D_{m,t} D_{m-1,t} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}} \end{cases}.$$

Dann ist $((\acute{S}_i)_{i=0}^{m-1}, (\acute{C}_i)_{i=1}^{m-1}, (\acute{D}_i)_{i=1}^{m-1})$ eine rekursive Darstellung der Kopplungsmatrizen S im Block $\acute{b}$.

Beweis: Es sei $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$.
1. Fall: Es sei $b \in \mathcal{M}^+_{\tilde{b},m} \subseteq \mathcal{M}^+_{\acute{b},m-1}$. Nach Bemerkung 7.2.9 folgt

$$S_b = S_{i_{\tilde{b},b},b} = \acute{S}_{i_{\acute{b},b},b} = \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},t}} \acute{C}_{j,t} \right) \acute{S}_{i_{\check{b},b},b} \left(\prod_{j=i_{\check{b},b}+1}^{i_{\check{b},s}} \acute{D}_{j,s} \right)^T.$$

2. Fall: Es sei $b \notin \mathcal{M}^+_{\tilde{b},m}$. Dann gilt $i_{\tilde{b},b} = i_{\check{b},b} \in \{0,\dots,m-1\}$. Zuerst unterscheiden wir, ob der Zeilencluster an lokalen Projektionen in $\mathcal{T}_{\tilde{\mathbf{b}}}$ beteiligt war oder nicht.
2. 1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Weil $(t,s) \notin \mathcal{T}_{\tilde{\mathbf{b}}}$ ist, gilt $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ (siehe Lemma 3.3.14 (iii)). Also gilt $i_{\acute{b},t} = m-1 = i_{\tilde{b},t} - 1$ und $i_{\acute{b},s} = i_{\tilde{b},s} \in \{0,\dots,m-1\}$. Für den dritten Term in (7.3) für $\acute{b}$ und $\tilde{b}$ gilt somit

$$\left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}} D_{j,s}\right)^T = \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},s}} \acute{D}_{j,s}\right)^T.$$

Für die Betrachtung des zweiten und dritten Terms machen wir die weitere Fallunterscheidung, ob der Basiswechsel $C_{m,t}$ direkt auf die Kopplungsmatrizen angewendet oder zu den nächsten Basiswechseln multipliziert wird.
2. 1. 1. Fall: Es sei $b \in \mathcal{M}^+_{\tilde{b},m-1} \subset \mathcal{M}^+_{\acute{b},m-1}$, also $i_{\tilde{b},b} = i_{\check{b},b} = m-1$. Dann gilt $i_{\acute{b},t} = m-1 < m = i_{\tilde{b},b}+1 = i_{\tilde{b},t}$. Also ist der erste Term in (7.3) für $\acute{b}$ gleich der Identität und für $\tilde{b}$ gleich $C_{m,t}$. Damit folgt

$$\begin{aligned} S_b &= \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t}\right) S_{i_{\tilde{b},b},b} \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}} D_{j,s}\right)^T = C_{m,t} S_{m-1,b} \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},s}} \acute{D}_{j,s}\right)^T \\ &= \acute{S}_{i_{\acute{b},b},b} \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},s}} \acute{D}_{j,s}\right)^T = \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},t}} \acute{C}_{j,t}\right) \acute{S}_{i_{\acute{b},b},b} \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},s}} \acute{D}_{j,s}\right)^T. \end{aligned}$$

2. 1. 2. Fall: Es sei $b \notin \mathcal{M}^+_{\tilde{b},m-1}$, also $i_{\tilde{b},b} = i_{\check{b},b} \in \{0,\dots,m-2\}$. Dann ist $\acute{S}_{i_{\acute{b},b},b} = S_{i_{\tilde{b},b},b}$. Wegen $i_{\tilde{b},t} = m$ und $i_{\acute{b},t} = m-1$ hat der erste Term in (7.3) für $\tilde{b}$ mindestens zwei Faktoren und für $\acute{b}$ mindestens einen Faktor. Damit gilt für die ersten Terme in (7.3)

$$\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t} = C_{m,t} C_{m-1,t} \prod_{j=i_{\tilde{b},b}+1}^{m-2} C_{j,t} = \acute{C}_{m-1,t} \prod_{j=i_{\acute{b},b}+1}^{m-2} \acute{C}_{j,t} = \prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},t}} \acute{C}_{j,t}.$$

Also sind alle drei Terme in (7.3) jeweils gleich und es folgt

$$S_b = \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t}\right) S_{i_{\tilde{b},b},b} \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}} D_{j,s}\right)^T = \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},t}} \acute{C}_{j,t}\right) \acute{S}_{i_{\check{b},b},b} \left(\prod_{j=i_{\check{b},b}+1}^{i_{\check{b},s}} \acute{D}_{j,s}\right)^T.$$

2. 2. Fall: Es sei $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann ist $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$. Der Beweis für diesen Fall verläuft analog zum 2. 1. Fall.
2. 3. Fall: Es sei $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$ und $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$. Dann sind $i_{\acute{b},t} = i_{\tilde{b},t}$ und $i_{\acute{b},s} = i_{\tilde{b},s}$. Es folgt direkt

$$S_b = \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},t}} C_{j,t}\right) S_{i_{\tilde{b},b},b} \left(\prod_{j=i_{\tilde{b},b}+1}^{i_{\tilde{b},s}} D_{j,s}\right)^T = \left(\prod_{j=i_{\acute{b},b}+1}^{i_{\acute{b},t}} \acute{C}_{j,t}\right) \acute{S}_{i_{\check{b},b},b} \left(\prod_{j=i_{\check{b},b}+1}^{i_{\check{b},s}} \acute{D}_{j,s}\right)^T.$$

Damit ist $((\acute{S}_i)_{i=0}^{m-1}, (\acute{C}_i)_{i=1}^{m-1}, (\acute{D}_i)_{i=1}^{m-1})$ eine rekursive Darstellung für die Kopplungsmatrizen S im Block $\acute{b}$. ∎

In Lemma 7.2.11 wird deutlich, warum wir die Darstellung durch Produkte von Matrizen $C_{j,t}$ und $D_{j,s}$ statt des direkten Speicherns wählen. Die von uns gewählte Darstellung erlaubt es, die komplette Darstellung beim Übergang zum Elterblock $\acute{b}$ durch die Anpassung der Kopplungsmatrizen in $\mathcal{T}_{\acute{b}}$ und der zum Block $\acute{b}$ gehörigen Basiswechsel zu aktualisieren. Beim direkten Speichern der Basiswechsel für jede Menge $\mathcal{M}_{\tilde{b},i}$ müssten immer alle C_i in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und D_i in $\mathcal{T}_{\tilde{\mathbf{s}}}$ aktualisiert werden. Dies würde zusätzlich den Level des aktuellen Blocks in den Aufwand bringen.

7.2.2 Lokale und externe Gewichte

Bevor wir das rekursive Speicherformat für die Hilfsgrößen einführen, stellen wir in diesem Abschnitt die sogenannten lokalen und externen Gewichte vor, die wir zur Berechnung der adaptiven Gewichte innerhalb des Teilbaums $\mathcal{T}_{\tilde{\mathbf{b}}}$ benötigen. Dazu untersuchen wir, wie wir adaptive Gewichte für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ derart berechnen können, dass wir dabei nur auf Kopplungsmatrizen $S_{(t,s)}$ mit $(t,s) \in \mathcal{T}_{\tilde{\mathbf{b}}}$ zugreifen müssen. Um Fallunterscheidungen zu vermeiden, sei $t \neq r_{\mathcal{I}}$. (Für $t = r_{\mathcal{I}}$ entfällt der Term für das Gewicht des Vaters.) Dann können wir adaptive Gewichte durch die QR-Zerlegung der Matrix $\hat{Z}_t^T$ mit

$$\hat{Z}_t \;:=\; \left(\sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T \quad E_t Z_{\acute{t}}\right)$$

mit adaptivem Gewicht $Z_{\acute{t}}$ des Elter $\acute{t} := \operatorname{par}(t)$ und der Blockzeile $\{s_1, \ldots, s_\sigma\} := \operatorname{row}(t)$ berechnen (siehe Anschnitt 4.5). Bei den bisherigen Berechnungen haben wir zuerst durch Verwendung von orthogonalen Gewichten die Anzahl der Zeilen für den Teil $\sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T$ der Blockzeile reduziert (vgl. Lemma 4.4.19). Bei dem in diesem Teilabschnitt beschriebenen Ansatz wollen wir sogenannte lokale Gewichte verwenden. Diese sind bereits in den Bibliotheken [14, 39] implementiert, wurden bisher allerdings noch in keinen theoretischen Arbeiten behandelt.

Definition 7.2.12 (Lokale Gewichte)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\kappa = (\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ die Rangverteilung von V und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Weiter sei eine Skalierung $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ gegeben. Eine Familie von Matrizen $L_t \in \mathbb{R}^{\kappa_t\times\eta_t}$, $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$, mit Rangverteilung $(\eta_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ nennen wir *lokale Gewichte* von H zu ω, falls für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ eine orthogonale Matrix $Q_t \in \mathbb{R}^{\mathcal{J}\times\eta_t}_{\xi_t\times\eta_t}$ mit

$$L_t Q_t^T \;=\; \sum_{s\in\operatorname{row}(t)} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T$$

existiert.

Die lokalen Gewichte fassen also alle Kopplungsmatrizen einer Blockzeile zusammen und komprimieren diese durch eine QR-Zerlegung. Mit dem lokalen Gewicht können wir eine Variante zur Berechnung des adaptiven Gewichts angeben, das ohne Durchlauf der Blockzeile auskommt (siehe Lemma 7.2.23).

Algorithmus 7.2.1 Die Funktion berechnet aus dem lokalen Gewicht L_t und ggf. einem adaptiven Gewicht des Elters $Z_{\acute{t}}$ ein adaptives Gewicht Z_t.

function ADAPTIVES_GEWICHT_SCHNELL(t, E, κ, L, η, θ, Z, ζ)
  if $t \neq r_{\mathcal{I}}$ **then**
    $\acute{t} := \operatorname{par}(t)$
    $\ell \leftarrow \eta_t \dot\cup \zeta_{\acute{t}}$
    $\widehat{Z}_t \in \mathbb{R}^{\kappa_t \times \ell}$
    $\widehat{Z}_{t|\kappa_t \times \eta_t} \leftarrow L_t$
    $\widehat{Z}_{t|\kappa_t \times \zeta_{\acute{t}}} \leftarrow \frac{1}{\theta_t} E_t Z_{\acute{t}}$
    QR-ZERLEGUNG($\widehat{Z}_t^T, \ell, \ell, \kappa_t, \breve{P}_t, Z_t^T, \zeta_t$)
  else
    $Z_t \leftarrow L_t$
  end if
end function

Mit Hilfe dieser Funktion adaptieren wir später die Prolog-Funktion für die 1. Variante in Algorithmus 7.1.3 so, dass sie nur Gewichte der Elter und lokale Gewichte in den aktuellen Clustern benötigt und keine Zugriffe auf Blöcke außerhalb des aktuellen Clusterbaums. Um Algorithmus 7.2.1 anzuwenden, benötigen wir eine Methode, mit der wir lokale Gewichte effizient berechnen können. Eine Möglichkeit wäre, wie bisher die Blockzeile $\operatorname{row}(t)$ zu durchlaufen. Dabei müssten wir typischerweise auch auf Kopplungsmatrizen $S_{(t,s)}$ zu Blöcken $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathfrak{b}}}$ außerhalb des Teilbaums $\mathcal{T}_{\tilde{\mathfrak{b}}}$ zugreifen. Weil wir dies für die 2. Variante verhindern wollen, teilen wir die Blockzeile in den Anteil im Teilbaum $\{s \in \operatorname{row}(t) | (t,s) \in \mathcal{T}_{\tilde{\mathfrak{b}}}\}$ und den Anteil außerhalb $\{s \in \operatorname{row}(t) | (t,s) \notin \mathcal{T}_{\tilde{\mathfrak{b}}}\}$ auf. Für diese Aufteilung definieren wir das sogenannte externe Gewicht.

Definition 7.2.13 (Externe Gewichte)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\kappa = (\kappa_t)_{t\in\mathcal{T}_{\mathcal{I}}}$ die Rangverteilung von V und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Weiter sei eine Skalierung $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ gegeben. Eine Familie von Matrizen $L_{\tilde{b},t} \in \mathbb{R}^{\kappa_t \times \eta_{\tilde{b},t}}$, $t \in \mathcal{T}_{\tilde{\mathfrak{t}}}$, mit Rangverteilung $(\eta_{\tilde{b},t})_{t\in\mathcal{T}_{\tilde{\mathfrak{t}}}}$ nennen wir *externe Gewichte* von H zum Block $\tilde{b}$, falls für alle $t \in \mathcal{T}_{\tilde{\mathfrak{t}}}$ eine orthogonale Matrix $Q_{\tilde{b},t} \in \mathbb{R}^{\mathcal{J}\times\eta_t}_{\xi_{\tilde{b},t}\times\eta_t}$, $\xi_{\tilde{b},t} := \dot\bigcup_{\substack{s\in\operatorname{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathfrak{b}}}}} \mathbf{s}$, mit

$$L_{\tilde{b},t} Q_{\tilde{b},t}^T = \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathfrak{b}}}}} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T = \sum_{\substack{(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\\(t,s)\notin\mathcal{T}_{\tilde{\mathfrak{b}}}}} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T \tag{7.4}$$

existiert.

Weil wir die externen Gewichte während der Rekursion Stück für Stück aufbauen wollen, ist es wichtig, für die Wurzel des Blockbaums externe Gewichte definieren zu können.

Bemerkung 7.2.14 (Initialisierung externer Gewichte)
Für $\tilde{b} = r_{\mathcal{I}\times\mathcal{J}}$ ist $\mathcal{T}_{\tilde{\mathbf{b}}} = \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und somit die rechte Seite in (7.4) gleich null. Wir können also für alle $t \in \mathcal{T}_{\mathcal{I}}$ die Nullmatrix $0 \in \mathbb{R}^{\kappa_t \times \eta_{r_{\mathcal{I}\times\mathcal{J}},t}}$ als externes Gewicht verwenden.

Wir haben bereits gesehen, dass sich mit Hilfe der lokalen Gewichte schnell adaptive Gewichte berechnen lassen. Beim Abwärtslaufen im Baum müssen wir, um die Voraussetzung (7.2) zu erfüllen, adaptive Gewichte für den aktuellen Zeilen- und Spaltencluster berechnen. Nach folgender Betrachtung können wir dabei direkt die externen Gewichte verwenden.

Bemerkung 7.2.15 (Externe und lokale Gewichte)
Es sei $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ mit $\tilde{t} \notin \mathcal{L}_{\mathcal{I}}$. Dann müssen wir vor dem rekursiven Aufruf für die Kinder ein adaptives Gewicht für $\tilde{t}$ berechnen. Für alle $\check{b} = (\check{t}, \check{s}) \in \mathcal{T}_{\tilde{\mathbf{b}}} \backslash \{\tilde{b}\}$ gilt $\check{t} \in \operatorname{desc}(\tilde{t}) \backslash \{\tilde{t}\}$. Also existiert kein zulässiges Blatt $(t, s) \in \mathcal{L}^+_{\tilde{\mathbf{b}}}$ mit $t = \tilde{t}$. Deshalb ist ein externes Gewicht $L_{\tilde{b},\tilde{t}}$ auch ein lokales Gewicht für $\tilde{t}$.

Algorithmus 7.2.1 und Bemerkung 7.2.15 ermöglichen uns, während des Abwärtslaufens im Blockbaum die effiziente Berechnung der adaptiven Gewichte aus den externen Gewichten ohne Zugriff auf Blöcke außerhalb des aktuellen Teilbaums. Für eine direkte Umsetzung würden wir

$$\text{externe Gewichte } L_{\hat{b},t},\ L_{\hat{b}^T,s} \text{ für alle } \hat{b} = (\hat{t}, \hat{s}) \in \operatorname{pred}(\tilde{b}),\ t \in \mathcal{T}_{\tilde{\mathbf{t}}},\ s \in \mathcal{T}_{\tilde{\mathbf{s}}} \tag{7.5}$$

benötigen. Deshalb beschäftigen wir uns zunächst mit der Berechnung von externen Gewichten.
Nach Bemerkung 7.2.14 sind für die Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ die externen Gewichte durch Nullmatrizen gegeben. Um die externen Gewichte schnell zu berechnen, wollen wir eine Möglichkeit finden, externe Gewichte für einen Kindblock $\check{b} = (\check{t}, \check{s}) \in \operatorname{chil}(\tilde{b}))$ effizient, d. h. innerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$, aus externen Gewichten in $\tilde{b}$ zu berechnen. Weil die Berechnungen alle nur im aktuellen Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ verlaufen sollen, betrachten wir die Menge der relevanten Blöcke für externe Gewichte für $\tilde{b}$ und $\check{b}$. Wegen $\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \operatorname{desc}(\check{b}) = (\mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{b}}}) \dot\cup (\mathcal{T}_{\tilde{\mathbf{b}}} \backslash \operatorname{desc}(\check{b}))$ genügt es, für alle $t \in \operatorname{desc}(\check{t})$ zu den alten externen Gewichten $L_{\tilde{b},t}$ jeweils die Blöcke

$$\{(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} |\, (t,s) \in \mathcal{T}_{\tilde{\mathbf{b}}} \backslash \operatorname{desc}(\check{b})\} \;=\; \{(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} |\, \exists s' \in \operatorname{chil}^*(\tilde{s}) \backslash \{\check{s}\} : s \in \operatorname{desc}(s')\}$$

hinzuzufügen. Bei der Gleichung nutzen wir, dass für $t \in \operatorname{desc}(\check{t})$ wegen Lemma 3.3.14 (iii) $(t,s) \operatorname{desc}(\check{b})$ genau dann gilt, wenn $s \in \operatorname{desc}(\check{s})$ gilt. Wir können also die Teilbäume zu den Blöcke $(\check{t}, s')$ mit $s' \in \operatorname{chil}^*(\tilde{s}) \backslash \{\check{s}\}$ durchlaufen und die zulässigen Blöcke zu den jeweiligen externen Gewichten hinzufügen. Insbesondere müssen wir, falls $\tilde{s} \in \mathcal{L}_{\mathcal{I}}$ gilt, keine weiteren Blöcke hinzufügen und könne direkt die alten externen Gewichte übernehmen.
Als Nächstes beschreiben wir eine Möglichkeit, das externe Gewicht in $\check{b}$ aus einem externen Gewicht zu $\tilde{b}$ durch Hinzufügen einzelner Blöcke zu berechnen. Dazu sei $t \in \operatorname{desc}(\check{t})$

und $\{1,\dots,s_\sigma\} := \mathrm{row}(t)$. Diese sei derart angeordnet, dass $\{s_{\breve{\sigma}},\dots,s_\sigma\} := \mathrm{row}(t) \cap \mathcal{T}_{\breve{\mathbf{s}}}$ und $\{s_{\breve{\sigma}},\dots,s_\sigma\} := \mathrm{row}(t) \cap \mathrm{desc}(\breve{s})$ gelte. Zur Berechnung des externen Gewichts $L_{\breve{b},t}$ benötigen wir nur den Faktor R einer QR-Zerlegung der Matrix $\sum_{i=1}^{\breve{\sigma}-1} \frac{1}{\omega_{(t,\tilde{s}_i)}} S_{(t,\tilde{s}_i)} W_{\tilde{s}_i}^T$, da $\{1,\dots,s_{\breve{\sigma}}-1\} = \mathrm{row}(t) \setminus \mathrm{desc}(\breve{s})$ gilt. Wie wir bereits mehrfach ausgenutzt haben, können wir dazu erst eine QR-Zerlegung für einen Teil berechnen, die orthogonale Matrix herausziehen und anschließend eine QR-Zerlegung vom Rest berechnen. Für das externe Gewicht bedeutet das, dass wir mit $L_{\breve{b},t}^{\tilde{\sigma}-1} = L_{\tilde{b},t}$ anfangen und dann rekursiv für alle $i \in \{\tilde{\sigma},\dots,\breve{\sigma}-1\} = \mathrm{row}(t) \cap (\mathcal{T}_{\breve{\mathbf{s}}} \setminus \mathrm{desc}(\breve{(s)}))$ die Matrix $L_{\breve{b},t}^{i}$ durch die QR-Zerlegung von $\begin{pmatrix} S_{(t,\tilde{s}_i)} W_{\tilde{s}_i}^T & L_{t,i-1} \end{pmatrix}$ berechnen. Auf diese Weise erhalten wir ein externes Gewicht $L_{\breve{b},t} := L_{\breve{b},t}^{\breve{\sigma}-1}$ von $\breve{b}$ in t. (Später führen wir einen entsprechenden Beweis.)
Statt dabei für alle $t \in \mathcal{T}_{\mathcal{I}}$ die Blockzeile $\mathrm{row}(t)$ zu durchlaufen, können wir, wie oben beschrieben, auch den Teil des Blockbaums $\mathcal{T}_{\breve{\mathbf{b}}}$ außerhalb des Teilbaums $\mathrm{desc}(\breve{b})$ durchlaufen und jeden Block $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+$ dem externen Gewicht von t hinzufügen. Dazu definieren wir den Algorithmus 7.2.2. Bei diesem gehen wir davon aus, dass die Clusterbasis W in den relevanten Clustern orthogonal ist.

Algorithmus 7.2.2 Die Funktion fügt für jeden zulässigen Block (t,s) im Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ die Matrix $S_{(t,s)} W_s^T$ dem zugehörigen Gewicht L_t hinzu. Dabei nehmen wir an, dass W_s orthogonal ist.

```
function GEWICHT_ERWEITERN_BAUM(b̃, S, κ, λ, ω, L, η)
    if b̃ ∈ L⁺_{I×J} then
        (t̃, s̃) ← b̃
        ℓ ← η_t ∪ λ_s
        L̂_t ∈ ℝ^{κ_t × ℓ}
        L̂_{t|κ_t×η_t} ← L_t
        L̂_{t|κ_t×ρ_t} ← (1/ω_{(t,s)}) S_{(t,s)}
        QR-ZERLEGUNG(L̂_t^T, ℓ, ℓ, κ_t, P̆_t, L_t^T, η_t)
    else if chil(b̃) ≠ ∅ then
        for ŏ ∈ chil(b̃) do
            GEWICHT_ERWEITERN_BAUM(b̃, S, κ, λ, ω, L, η)
        end for
    end if
end function
```

Mit Hilfe dieser Funktion können wir die zulässigen Blöcke eines Teilbaums zu gegebenen Gewichten hinzufügen. Falls der aktuelle Block $\tilde{b} = (\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ kein Blatt ist, müssen wir für den rekursiven Aufruf im Kind $\breve{b} = (\breve{t},\breve{s}) \in \mathrm{chil}(\tilde{b})$ externe Gewichte $L_{\breve{b},t}$ für alle $t \in \mathrm{desc}(\breve{t})$ berechnen. Dazu legen wir $L_{\breve{b},t}$ als Kopie von $L_{\tilde{b},t}$ für alle $t \in \mathrm{desc}(\breve{t})$ an und rufen den Algorithmus 7.2.2 für $L_{\breve{b}}$ und $(\breve{t},s)$ für alle $s \in \mathrm{chil}^*(\tilde{s}) \setminus \{\breve{s}\}$ auf. Danach sind in $L_{\breve{b}}$ externe Gewichte für den Block $\breve{b}$ gespeichert. Mit Hilfe dieses Vorgehens definieren wir später Algorithmus 7.2.5, der die innere Prolog-Funktion für die 2. Variante

des Niedrigrangupdates beschreibt. In dem zugehörigen Lemma 7.2.24 untersuchen wir die Eigenschaften formal.
Weil sich mit den lokalen Niedrigrangupdates auch Kopplungsmatrizen außerhalb des aktuellen Blocks verändern, müssen auch die externen Gewichte angepasst werden. Würden wir die externen Gewichte wie in (7.5) direkt speichern, müssten wir beim Aufwärtslaufen die externen Gewichte $L_{\hat{b},t}$ und $L_{\hat{b}^T,s}$ für alle $\hat{b} = (\hat{t}, \hat{s}) \in \operatorname{pred}(\tilde{b})$, $t \in \mathcal{T}_{\hat{\mathbf{t}}}$ und $s \in \mathcal{T}_{\hat{\mathbf{s}}}$ aktualisieren. Dies würde den Level des aktuellen Blocks $\tilde{b}$ in den Aufwand bringen. Deshalb verwenden wir wie für die Kopplungsmatrizen eine rekursive Darstellung.

7.2.3 Rekursive Darstellung der externen Gewichte

In diesem Abschnitt stellen wir die rekursive Darstellung der externen Gewichte vor. Diese basiert auf einer rekursiven Darstellung für die Kopplungsmatrizen, wobei die Familien C_i, $i \in \{1, \dots, m\}$ verwendet werden.

Definition 7.2.16 (Rekursive Darstellung von externen Gewichten)
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ eine Skalierung, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m = ((\tilde{t}_i, \tilde{s}_i))_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Weiter sei eine rekursive Darstellung $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ von S mit Rangverteilungen κ_i auf $\operatorname{desc}(\tilde{t}_i)$ und λ_i auf $\operatorname{desc}(\tilde{s}_i)$ für alle $i \in \{0, \dots, m\}$ gegeben. Weiter seien für alle $i \in \{0, \dots, m\}$ Rangverteilungen η_i auf $\operatorname{desc}(\tilde{t}_i)$ und Familien von Matrizen L_i mit $L_{i,t} \in \mathbb{R}^{\kappa_{i,t}\times\eta_{i,t}}$, $t \in \operatorname{desc}(\tilde{t}_i)$, gegeben. Falls für alle $i \in \{0, \dots, m\}$ und alle $t \in \operatorname{desc}(\tilde{t}_i)$

$$\Bigg(\underbrace{\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t}}_{\in \mathbb{R}^{\kappa_{i_{\tilde{b},t},t}\times\kappa_{i,t}}} \Bigg) L_{i,t} \tag{7.6}$$

ein externes Gewicht von H in $\tilde{b}_i$ zu ω ist, bezeichnen wir $(L_i)_{i=0}^m$ als *rekursive externe Gewichte* von H zu ω im Block $\tilde{b}$. Wir schreiben auch $L_{b_i} := L_i$ und für externe Gewichte der transponierten Matrix $L_{b_i}^T$ (also für die Spaltenbasis).

Wie in Definition 7.2.5 verwenden wir für die rekursiven externen Gewichte Produkte von Basiswechseln. Nach Bemerkung 7.2.6 sind diese Produkte für $i = m$ gleich der Identität. Daraus folgt Bemerkung 7.2.17.

Bemerkung 7.2.17
Im aktuellen Block $\tilde{b}$ ($i = m$) sind die rekursiven externen Gewichte $L_{m,t}$ für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ externe Gewichte von H in $\tilde{b}$ zu ω.

Um rekursive externe Gewichte für die Spaltenbasis zu betrachten, benötigen wir eine rekursive Darstellung für die transponierten Kopplungsmatrizen S^T, die zur transponierten $\mathcal{H}^2$-Matrix H^T gehören (siehe Lemma 3.4.26). Die Darstellung aus Bemerkung 7.2.18 folgt direkt, indem wir beide Seiten in (7.3) transponieren.

Bemerkung 7.2.18
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$ und $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine rekursive Darstellung von S. Dann ist durch die Familien $((S_i^T)_{i=0}^m, (D_i)_{i=1}^m, (C_i)_{i=1}^m)$ eine rekursive Darstellung von S^T gegeben.

Für diese Form der externen Gewichte untersuchen wir das Durchlaufen des Blockbaums. Dabei gehen wir sehr ähnlich zu der rekursiven Darstellung der Kopplungsmatrizen vor. Wir fangen mit dem Abwärtslaufen im Blockbaum an.

Lemma 7.2.19
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ eine Skalierung, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block und $(\tilde{b}_i)_{i=0}^m = ((\tilde{t}_i, \tilde{s}_i))_{i=0}^m$ der Pfad von der Wurzel $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$. Weiter seien eine rekursive Darstellung $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ der Kopplungsmatrizen S in $\tilde{b}$ und rekursive externe Gewichte $(L_i)_{i=0}^m$ von H zu ω im Block $\tilde{b}$ gegeben.
Es sei $\tilde{b} \notin \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$, $\breve{b} \in \operatorname{chil}(\tilde{b})$ und $((\breve{S}_i)_{i=0}^{m+1}, (\breve{C}_i)_{i=1}^{m+1}, (\breve{D}_i)_{i=1}^{m+1})$ die rekursive Darstellung von S aus Lemma 7.2.8. Es sei $L_{m+1,t}$ für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ ein externes Gewicht von H zu ω in $\breve{b}$. Dann sind $(L_i)_{i=0}^{m+1}$ rekursive externe Gewichte von H zu ω im Block $\breve{b}$.

Beweis: Für alle $t \in \operatorname{desc}(\breve{t})$ sind die Matrizen $\breve{C}_{m+1,t}$ aus Lemma 7.2.8 gleich der Identität. Damit gilt für alle $i \in \{0, \ldots, m\}$ und $t \in \operatorname{desc}(\tilde{t}_i)$

$$\left(\prod_{j=i+1}^{i_{\breve{b},t}} \breve{C}_{j,t}\right) L_{i,t} = \left(\prod_{j=i+1}^{\min\{i_{\breve{b},t}, m\}} \breve{C}_{j,t}\right) L_{i,t} = \left(\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t}\right) L_{i,t}.$$

Der letzte Term der Gleichungskette ist nach Voraussetzung ein externes Gewicht von H zu ω in $\tilde{b}_i$. Also gilt in diesem Fall die Behauptung.
Für $i = m+1$ ist das Produkt über die Matrizen $\breve{C}_{j,t}$ wegen $i+1 = m+2 > i_{\breve{b},t}$ gleich der Identität und $L_{m+1,t}$ für alle $t \in \operatorname{desc}(\breve{t})$ per Definition ein externes Gewicht. Damit folgt die Behauptung. ∎

Nachdem wir das Abwärtslaufen im Blockbaum für die rekursiven externen Gewichte betrachtet haben, beschäftigen wir uns mit dem Niedrigrangupdate.

Lemma 7.2.20
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$ mit Rang-$\mathcal{K}$-Darstellung und $\widetilde{H} := \widetilde{H}(H, R, \tilde{b})$. Weiter seien $(\overline{V}_t)_{t\in\mathcal{T}_{\tilde{\mathbf{t}}}}$ und $(\overline{W}_s)_{s\in\mathcal{T}_{\tilde{\mathbf{s}}}}$ orthogonale Clusterbasen, $\overline{V}$ und $\overline{W}$ die Clusterbasen nach Lemma 5.2.1 und $\overline{H} := \Pi_{\mathcal{T}_{\mathcal{I}\times\mathcal{J}},\tilde{b},\overline{V},\overline{W}}[\widetilde{H}]$. Es sei $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$, $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine rekursive Darstellung von S und $(L_i)_{i=0}^m$ rekursive externe Gewichte von H zu $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ in $\tilde{b}$.
Wir definieren $\overline{L}_i := L_i$ für alle $i \in \{0, \ldots, m-1\}$ und $\overline{L}_{m,t} := C(\widetilde{V}, \overline{V})_{t|\overline{\kappa}_t\times\kappa_t} L_{m,t}$ für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann sind $(\overline{L}_i)_{i=0}^m$ rekursive externe Gewichte von $\overline{H}$ zu ω in $\tilde{b}$ basierend auf der rekursiven Darstellung $((\overline{S}_i)_{i=0}^m, (\overline{C}_i)_{i=1}^m, (\overline{D}_i)_{i=1}^m)$ von $\overline{S}$ aus Lemma 7.2.10.

Beweis: Wir untersuchen zuerst die externen Gewichte für den aktuellen Block.
1. Fall: Es sei $i = m$.
Weiter sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Für alle $s \in \mathrm{row}(t)$ mit $b = (t,s) \notin \mathcal{T}_{\tilde{\mathbf{b}}}$ gilt nach Lemma 3.3.14 (iii) und (iv) $s \notin \mathcal{T}_{\tilde{\mathbf{s}}} \cup \mathrm{pred}(\tilde{s})$ und zusammen mit Lemma 5.3.16 folgt $\overline{S}_b = C(V,\overline{V})_t S_b$ sowie mit Lemma 5.2.1 $\overline{W}_s = W_s$. Nach Bemerkung 7.2.17 ist $L_{m,t}$ ein externes Gewicht von H zu ω in $\tilde{b}$. Also existiert eine orthogonale Matrix $Q_{\tilde{b},t}$ mit

$$\begin{aligned}\sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)}\overline{S}_{(t,s)}\overline{W}_s^T &= C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t} \sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)} S_{(t,s)} W_s^T \\ &= C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t} L_{m,t} Q_{\tilde{b},t}^T = \overline{L}_{m,t} Q_{\tilde{b},t}^T.\end{aligned}$$

Also ist $\overline{L}_{m,t}$ ein externes Gewicht von $\overline{H}$ zu ω in $\tilde{b}$. Nach Bemerkung 7.2.6 gilt somit die Bedingung (7.6) für $i = m$ und alle $t \in \mathrm{desc}(\tilde{t}_m) = \mathcal{T}_{\tilde{\mathbf{t}}}$.
2. Fall: Es sei $i \in \{0,\ldots,m-1\}$.
Es sei $t \in \mathrm{desc}(\tilde{t}_i)$. Dann folgt für alle $s \in \mathrm{row}(t)$ mit $b = (t,s) \notin \mathrm{desc}(\tilde{b}_i)$ aus Lemma 3.3.14 (iii) und (iv) $s \notin \mathrm{desc}(\tilde{s}_i) \cup \mathrm{pred}(\tilde{s}_i) \supseteq \mathcal{T}_{\tilde{\mathbf{s}}} \cup \mathrm{pred}(\tilde{s})$ und mit Lemma 5.2.1 auch $\overline{W}_s = W_s$. Wir unterscheiden zusätzlich die Fälle, ob die Kopplungsmatrix bezüglich der Projektion in $\mathcal{T}_{\tilde{\mathbf{b}}}$ angepasst werden muss oder nicht.
2. 1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$.
Nach Lemma 5.2.4 gilt $\overline{S}_b = C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t} S_b$ wegen $s \notin \mathcal{T}_{\tilde{\mathbf{s}}}$ für alle $s \in \mathrm{row}(t)$ mit $b = (t,s) \notin \mathrm{desc}(\tilde{b}_i)$. Aus der Definition 7.2.16 der rekursiven externen Gewichte folgt die Existenz einer orthogonalen Matrix $Q_{\tilde{b},t}$ mit

$$\begin{aligned}\sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)}\overline{S}_{(t,s)}\overline{W}_s^T &= C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t} \sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)} S_{(t,s)} W_s^T \\ &= C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t} \left(\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t}\right) L_{i,t} Q_{\tilde{b},t}^T.\end{aligned}$$

Nach Definition gilt $L_{i,t} = \overline{L}_{i,t}$ und $C_{j,t} = \overline{C}_{j,t}$ für $j \in \{i+1,\ldots,i_{\tilde{b},t}-1\} \subseteq \{0,\ldots,m-1\}$. Zusammen mit $C(\widetilde{V},\overline{V})_{t|\overline{\kappa}_t\times\kappa_t} C_{m,t} = \overline{C}_{m,t}$ und $i_{\tilde{b},t} = m \geq i+1$ folgt

$$\sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)}\overline{S}_{(t,s)}\overline{W}_s^T = \left(\prod_{j=i+1}^{i_{\tilde{b},t}} \overline{C}_{j,t}\right) \overline{L}_{i,t} Q_{\tilde{b},t}^T.$$

Also gilt in diesem Fall die Bedingung (7.6).
2. 2. Fall: Es sei $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$.
Für alle $s \in \mathrm{row}(t)$ mit $(t,s) \notin \mathrm{desc}(\tilde{b}_i)$ folgt $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ und mit Lemma 5.2.4, dass $\overline{S}_b = S_b$ ist. Mit der orthogonalen Matrix $Q_{\tilde{b}_i,t}$ aus der Definition der externen Gewichte und

$i_{\tilde{b},t}, i \in \{0,\dots,m-1\}$ folgt

$$\sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)}\overline{S}_{(t,s)}\overline{W}_s^T = \sum_{\substack{s\in\mathrm{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega(t,s)}S_{(t,s)}W_s^T = \left(\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t}\right)L_{i,t}Q_{\tilde{b},t}^T$$
$$= \left(\prod_{j=i+1}^{i_{\tilde{b},t}} \overline{C}_{j,t}\right)\overline{L}_{i,t}Q_{\tilde{b},t}^T.$$

Damit ist die Bedingung (7.6) auch in diesem Fall erfüllt und die Behauptung gilt. ■

Zum Abschluss dieses Abschnitts betrachten wir das Aufwärtslaufen im Blockbaum. Dies verläuft ähnlich wie für die rekursive Darstellung der Kopplungsmatrizen. Allerdings werden die Matrizen $L_{m-1,t}$ für jeden Cluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ aktualisiert und die Basiswechsel aus Lemma 7.2.11 übernommen.

Lemma 7.2.21
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$, $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$ und $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$. Weiter seien eine rekursive Darstellung $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ von S und rekursive externe Gewichte $(L_i)_{i=0}^m$ von H zu ω in $\tilde{b}$ gegeben.
Es sei $\tilde{b} \neq r_{\mathcal{I}\times\mathcal{J}}$, $\acute{b} = (\acute{t}, \acute{s}) = \mathrm{par}(\tilde{b})$ und $((\acute{S}_i)_{i=0}^m, (\acute{C}_i)_{i=1}^m, (\acute{D}_i)_{i=1}^m)$ die rekursive Darstellung von S in $\acute{b}$ aus Lemma 7.2.11. Wir definieren $\acute{L}_i := L_i$ für alle $i \in \{0,\dots,m-2\}$ und für alle $t \in \mathrm{desc}(\acute{t})$

$$\acute{L}_{m-1,t} = \begin{cases} L_{m-1,t} & \text{, falls } t \notin \mathcal{T}_{\tilde{\mathbf{t}}} \\ C_{m,t}L_{m,t} & \text{, falls } t \in \mathcal{T}_{\tilde{\mathbf{t}}} \end{cases}.$$

Dann sind $(\acute{L}_i)_{i=0}^m$ rekursive externe Gewichte von H zu ω im Block $\acute{b}$.

Beweis: Es sei $i \in \{0,\dots,m-1\}$ und $t \in \mathrm{desc}(\tilde{t}_i)$. Wir unterscheiden die Fälle, ob Blöcke mit Zeilencluster t von Projektionen in $\mathcal{T}_{\tilde{\mathbf{b}}}$ beeinflusst werden oder nicht.
1. Fall: Es sei $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt $i_{\acute{b},t} = m-1 = i_{\tilde{b},t} - 1$. Als Nächstes unterscheiden wir, ob $i = m-1$ ist und das externe Gewicht direkt aktualisiert wird oder $i \in \{0,\dots,m-2\}$ ist und das externe Gewicht indirekt durch die Definition von $C_{m-1,t}$ angepasst wird.
1. 1. Fall: Es sei $i = m-1$. Dann ist $i+1 = i_{\tilde{b},t} = m > m-1 = i_{\acute{b},t}$ und

$$\left(\prod_{j=i+1}^{i_{\acute{b},t}} \acute{C}_{j,t}\right)\acute{L}_{i,b} = \acute{L}_{m-1,t} = C_{m,t}L_{m,t} = \left(\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t}\right)L_{i,t}$$

ein externes Gewicht von H zu ω in $\tilde{b}_i$.
1. 2. Fall: Es sei $i \in \{0,\dots,m-2\}$. Dann ist $i+1 \leq i_{\acute{b},t} = m-1 = i_{\tilde{b},t} - 1$ und

$$\left(\prod_{j=i+1}^{i_{\acute{b},t}} \acute{C}_{j,t}\right)\acute{L}_{i,b} = C_{m,t}C_{m-1,t}\left(\prod_{j=i+1}^{i_{\acute{b},t}-1} C_{j,t}\right)L_{i,b} = \left(\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t}\right)L_{i,t}$$

ein externes Gewicht von H zu ω in $\tilde{b}_i$.
2. Fall: Es sei $t \notin \mathcal{T}_{\tilde{\mathbf{t}}}$. Dann gilt $i_{\acute{b},t} = i_{\tilde{b},t} \in \{0, \dots, m-1\}$ und

$$\left(\prod_{j=i+1}^{i_{\acute{b},t}} \acute{C}_{j,t} \right) \acute{L}_{i,t} \;=\; \left(\prod_{j=i+1}^{i_{\tilde{b},t}} C_{j,t} \right) L_{i,t}$$

ist ein externes Gewicht von H zu ω in $\tilde{b}_i$. ∎

Damit haben wir ein Möglichkeit gefunden, rekursive Darstellungen von Kopplungsmatrizen und rekursive externe Gewichte während des rekursiven Durchlaufens eines Blockbaums zu erhalten.

Bemerkung 7.2.22
Die rekursiven externen Gewichte für die Spaltenbasis, also für die transponierte $\mathcal{H}^2$-Matrix H^T, geben uns die Lemmata 7.2.19, 7.2.21 und 7.2.20 die Möglichkeit der Anpassung während der Rekursion. Dies folgt daraus, dass die Anpassungen für die rekursiven Darstellungen mit den Lemmata 7.2.8, 7.2.11 und 7.2.10 für die Basiswechsel der Zeilen- und Spaltenbasis analog verlaufen.

Als Nächstes beschäftigen wir uns mit der algorithmischen Umsetzung des Niedrigrangupdates in der 2. Variante.

7.2.4 Algorithmische Umsetzung

Nach unseren Vorbetrachtungen stellen wir nun die 2. Variante unseres rekursiven Updates vor. Dabei fügen wir im Vergleich zur 1. Variante noch eine Prolog- und eine Epilog-Funktion innerhalb der Schleife für die Kinder hinzu. Wir beginnen mit der äußeren Prolog-Funktion, die entsprechend zu Algorithmus 7.1.3 die Voraussetzungen (7.1) und (7.2) für die Kinder sicherstellt. Wie bereits erwähnt, verwenden wir dazu Algorithmus 7.2.1, um aus den lokalen Gewichten ein adaptives Gewicht zu berechnen. Nach Bemerkung 7.2.15 sind für die relevanten Fälle externe Gewichte gleich lokalen Gewichten. Dies führt zu Algorithmus 7.2.4, dessen Aufwand und Eigenschaften wir in Lemma 7.2.23 untersuchen.

Lemma 7.2.23
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ ein Nicht-Blatt-Block, $\omega_{\mathcal{I}}, \omega_{\mathcal{J}} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$, $\theta_{\mathcal{I}} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ und $\theta_{\mathcal{J}} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{J}}}$ Skalierungen. Es gelten die Voraussetzungen (7.1) und (7.2) im Block $\tilde{b}$. Falls $\tilde{t} \neq r_{\mathcal{I}}$ gilt, sei $\#\zeta_{\mathcal{I},\mathrm{par}(\tilde{t})} \leq \#\kappa_{\mathrm{par}(\tilde{t})}$ und, falls $\tilde{s} \neq r_{\mathcal{J}}$ gilt, sei $\#\zeta_{\mathcal{J},\mathrm{par}(\tilde{s})} \leq \#\lambda_{\mathrm{par}(\tilde{s})}$. Weiter seien $L_{\tilde{b}}$ und $L_{\tilde{b}^T}$ lokale Gewichte in $\tilde{t}$ für H zu $\omega_{\mathcal{I}}$ mit weniger als $\#\kappa_{\tilde{t}}$ Spalten bzw. in $\tilde{s}$ für H^T zu $\omega_{\mathcal{J}}$ mit weniger als $\#\lambda_{\tilde{s}}$ Spalten. Dann benötigt Algorithmus 7.2.4 aufgerufen in $\tilde{b}$ höchstens

$$(4c_{qr} + 4)k_{\max}^3$$

Algorithmus 7.2.3 Prototyp für einen rekursiven Algorithmus, der das Niedrigrangupdate in der 2.Variante verwendet. Dabei sind $L_{\tilde{b}}$ $L_{\tilde{b}^T}$ die rekursiven externen Gewichte sowie $C_{\tilde{b}}$ und $C_{\tilde{b}^T}$ die Basiswechsel der rekursiven Darstellung.

function PROTOTYP_VARIANTE2($\tilde{b}$, H, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, $C_{\tilde{b}}$, $C_{\tilde{b}^T}$, *opt*, ϵ_I, $\epsilon_{\mathcal{J}}$, *str*)
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$
  $(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V, E), (W, F), N, S) \leftarrow H$
  Berechnungen der Rang-$\mathcal{K}$-Matrix $R = AB^T \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$
  NIEDRIGRANGUPDATE_VARIANTE2($\tilde{b}$, H, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, R, *opt*, ϵ_I, $\epsilon_{\mathcal{J}}$, *str*)
    ▷ Algorithmus 7.2.11
  if chil($\tilde{b}$) $\neq \emptyset$ **then**
    PROLOG_AUSSEN_VARIANTE2($\tilde{b}$, E, κ, F, λ, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, $\theta_{\mathcal{I}}$, $\theta_{\mathcal{J}}$)
      ▷ Algorithmus 7.2.4
    for $\breve{b} \in$ chil($\tilde{b}$) **do**
      PROLOG_INNEN_VARIANTE2($\tilde{b}$, $\breve{b}$, S, κ, λ, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $L_{\breve{b}}$, $L_{\breve{b}^T}$, $C_{\breve{b}}$, $C_{\breve{b}^T}$)
        ▷ Algorithmus 7.2.5
      PROTOTYP_VARIANTE2($\breve{b}$, H, $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, $L_{\breve{b}}$, $L_{\breve{b}^T}$, $C_{\breve{b}}$, $C_{\breve{b}^T}$, *opt*, ϵ_I, $\epsilon_{\mathcal{J}}$, *str*)
      EPILOG_INNEN_VARIANTE2($\tilde{b}$, $\breve{b}$, $C_{\breve{b}}$, $C_{\breve{b}^T}$, S, κ, λ, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$)
        ▷ Algorithmus 7.2.13
    end for
    EPILOG_VARIANTE2($\tilde{b}$, E, κ, F, λ, $C_{\tilde{b}}$, $D_{\tilde{b}}$) ▷ Algorithmus 7.1.6
  end if
end function

Algorithmus 7.2.4 Die Funktion stellt sicher, dass die Voraussetzungen (7.1) und (7.2) für alle Kinder $\breve{b} \in$ chil($\tilde{b}$) erfüllt sind. Dabei wird davon ausgegangen, dass die Voraussetzungen für den aktuellen Block $\tilde{b} = (\tilde{t}, \tilde{s})$ erfüllt und lokale Gewichte für den Fall $\tilde{t} \notin \mathcal{L}_{\mathcal{I}}$ bzw. $\tilde{s} \notin \mathcal{L}_{\mathcal{J}}$ gegeben sind.

function PROLOG_AUSSEN_VARIANTE2($\tilde{b}$, E, κ, F, λ, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, $\theta_{\mathcal{I}}$, $\theta_{\mathcal{J}}$)
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$
  if $\tilde{t} \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ **then**
    ADAPTIVE_GEWICHTE_SCHNELL($\tilde{t}$, E, κ, $L_{\tilde{b}}$, $\eta_{\mathcal{I}}$, $\theta_{\mathcal{I}}$, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$)
      ▷ Algorithmus 7.2.1
  end if
  if $\tilde{s} \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{L}_{\mathcal{J}}$ **then**
    ADAPTIVE_GEWICHTE_SCHNELL($\tilde{s}$, F, λ, $L_{\tilde{b}^T}$, $\eta_{\mathcal{J}}$, $\theta_{\mathcal{J}}$, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$)
      ▷ Algorithmus 7.2.1
  end if
end function

Operationen, wobei $k_{\max} := \max_{t \in \mathcal{T}_\mathcal{I}} \#\kappa_t$ sei. Außerdem sind die Voraussetzungen (7.1) und (7.2) mit $\#\zeta_{\mathcal{I},t} \leq \#\kappa_t$ und $\#\zeta_{\mathcal{J},s} \leq \#\lambda_s$ nach dem Aufruf von Algorithmus 7.2.4 für alle $\breve{b} \in \operatorname{chil}(b)$ erfüllt.

Beweis: Wie wir im Beweis von Lemma 7.1.6 gesehen haben, reicht es aus, im Falle von $\tilde{t} \notin \mathcal{L}_\mathcal{I}$ ein adaptives Gewicht in $\tilde{t}$ und im Falle von $\tilde{s} \notin \mathcal{L}_\mathcal{J}$ ein adaptives Gewicht in $\tilde{s}$ zu berechnen. Falls einer der Cluster ein Blatt ist, brauchen wir für diesen keine weiteren Größen berechnen. Wir betrachten den Zeilencluster $\tilde{t}$.
Für den Fall $\tilde{t} = r_\mathcal{I}$ ist das lokale Gewicht $L_{\tilde{t}}$ gleichzeitig ein adaptives Gewicht und die Behauptung folgt (siehe die Definitionen 4.4.17, 4.4.15 und 7.2.12). Weil der Algorithmus in diesem Fall keine Berechnungen vornimmt, gilt die Aufwandsabschätzung.
Es sei also o. E. d. A. $\tilde{t} \neq r_\mathcal{I}$ und $\acute{t} := \operatorname{par}(\tilde{t})$. Dann existieren orthogonale Matrizen $Q_{\tilde{t}} \in \mathbb{R}^{\mathcal{J} \times \eta_{\tilde{t}}}_{\xi_{\tilde{t}} \times \eta_{\tilde{t}}}$ und $P_{\acute{t}} \in \mathbb{R}^{\mathcal{J} \times \zeta_{\acute{t}}}_{\xi_{\acute{t}} \times \zeta_{\acute{t}}}$ derart, dass für die adaptive totale Clusterbasis

$$\begin{aligned} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} &= \sum_{s \in \operatorname{row}(\tilde{t})} \frac{1}{\omega_{(\tilde{t},s)}} V_{\tilde{t}} S_{(\tilde{t},s)} W_s^T + \frac{1}{\theta_{\tilde{t}}} X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\acute{t}|\tilde{\mathbf{t}}} \\ &= V_{\tilde{t}} \begin{pmatrix} L_{\tilde{t}} & E_{\tilde{t}} Z_{\acute{t}} \end{pmatrix} \begin{pmatrix} Q_{\tilde{t}} & P_{\acute{t}} \end{pmatrix}^T =: V_{\tilde{t}} \widehat{Z}_{\tilde{t}} \widehat{P}_{\tilde{t}}^T \end{aligned}$$

gilt. Dabei ist die Matrix $\widehat{P}_{\tilde{t}}$ nach Bemerkung 2.3.15 orthogonal. Mit der durch Algorithmus 7.2.1 berechneten QR-Zerlegung folgt

$$X(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, H, \omega, \theta)_{\tilde{t}} = V_{\tilde{t}} \widehat{Z}_{\tilde{t}} \widehat{P}_{\tilde{t}}^T = V_{\tilde{t}} Z_{\tilde{t}} \breve{P}_{\tilde{t}}^T \widehat{P}_{\tilde{t}}^T = V_{\tilde{t}} Z_{\tilde{t}} P_{\tilde{t}}^T.$$

Die Matrix $P_{\tilde{t}}$ ist als Produkt von orthogonalen Matrizen orthogonal. Also ist $Z_{\tilde{t}}$ ein adaptives Gewicht. Weil $Z_{\tilde{t}} \in \mathbb{R}^{\kappa_{\tilde{t}} \times \zeta_{\tilde{t}}}$ durch die QR-Zerlegung der Matrix $\widehat{Z}_{\tilde{t}} \in \mathbb{R}^{\kappa_{\tilde{t}} \times (\eta_{\tilde{t}} \dot\cup \zeta_{\acute{t}})}$ entsteht, gilt $\#\zeta_{\tilde{t}} = \min\{\#\kappa_{\tilde{t}}, \#\eta_{\tilde{t}} + \#\zeta_{\acute{t}}\} \leq \#\kappa_{\tilde{t}}$.
Als Nächstes schätzen wir den Aufwand ab. Das Aufstellen der Matrix $\widehat{Z}_{\tilde{t}}$ kostet höchstens

$$2\#\kappa_{\tilde{t}}\#\kappa_{\acute{t}}\#\zeta_{\acute{t}} \leq 2\#\kappa_{\tilde{t}}\#\kappa_{\acute{t}}\#\kappa_{\acute{t}} \leq 2k_{\max}^3$$

Operationen. Danach wird die QR-Zerlegung der $(\eta_{\tilde{t}} \cup \zeta_{\acute{t}}) \times \kappa_{\tilde{t}}$-Matrix $\widehat{Z}_{\tilde{t}}^T$ in weniger als

$$c_{qr}\#(\eta_{\tilde{t}} \cup \zeta_{\acute{t}})\#\kappa_t \min\{\#(\eta_{\tilde{t}} \cup \zeta_{\acute{t}}), \#\kappa_{\tilde{t}}\} \leq c_{qr}(\#\kappa_{\tilde{t}} + \#\kappa_{\acute{t}})\#\kappa_{\tilde{t}}\#\kappa_{\tilde{t}} \leq 2c_{qr}k_{\max}^3$$

Operationen berechnet. Die Aussagen folgen analog für die Spaltencluster. Damit folgt die Aussage. ∎

Damit sind die Voraussetzungen (7.1) und (7.2) beim Abwärtslaufen im Blockbaum erfüllt. Als Nächstes wenden wir uns den rekursiven Darstellungen der Kopplungsmatrizen und den rekursiven externen Gewichten zu. Diese berechnen wir für die Kinder mit Hilfe der inneren Prolog-Funktion in Algorithmus 7.2.5, deren Eigenschaften wir in Lemma 7.2.24 untersuchen.

Algorithmus 7.2.5 Die Funktion berechnet für ein Kind $\breve{b} \in \operatorname{chil}(\tilde{b})$ die externen Gewichte $L_{\breve{b}}$ und $L_{\breve{b}^T}$ aus den externen Gewichten des aktuellen Blocks $L_{\tilde{b}}$ und $L_{\tilde{b}^T}$. Dabei gehen wir davon aus, dass alle relevanten Clusterbasen orthogonal sind.

function PROLOG_INNEN_VARIANTE2($\tilde{b}$, $\breve{b}$, S, κ, λ, ω, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $L_{\breve{b}}$, $L_{\breve{b}^T}$, $C_{\breve{b}}$, $C_{\breve{b}^T}$)
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$, $(\breve{t}, \breve{s}) \leftarrow \breve{b}$
  for $t \in \operatorname{desc}(\breve{t})$ **do**
    $L_{\breve{b},t} \leftarrow L_{\tilde{b},t} \in \mathbb{R}^{\kappa_t \times \eta_{\tilde{b},t}}$
    $C_{\breve{b},t} \leftarrow I_{\kappa_t} \in \mathbb{R}^{\kappa_t \times \kappa_t}$
  end for
  for $s \in \operatorname{chil}^*(\tilde{s}) \backslash \{\breve{s}\}$ **do**
    GEWICHT_ERWEITERN_BAUM($(\breve{t}, s)$, S, κ, λ, ω, $L_{\breve{b}}$, $\eta_{\breve{b}}$) ▷ Algorithmus 7.2.2
  end for
  for $s \in \operatorname{desc}(\breve{s})$ **do**
    $L_{\breve{b}^T,s} \leftarrow L_{\tilde{b}^T,s} \in \mathbb{R}^{\lambda_s \times \eta_{\tilde{b}^T,s}}$
    $C_{\breve{b}^T,s} \leftarrow I_{\lambda_s} \in \mathbb{R}^{\lambda_s \times \lambda_s}$
  end for
  for $t \in \operatorname{chil}^*(\tilde{t}) \backslash \{\breve{t}\}$ **do**
    GEWICHT_ERWEITERN_BAUM($(\breve{s}, t)$, S^T, λ, κ, ω, $L_{\breve{b}^T}$, $\eta_{\breve{b}^T}$)
  end for
end function

Lemma 7.2.24
Es seien $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ ein Nicht-Blatt-Block, $\breve{b} \in \operatorname{chil}(\tilde{b})$ ein Kind dieses Blocks und $\omega_{\mathcal{I}}, \omega_{\mathcal{J}} \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$, $\theta_{\mathcal{I}} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{I}}}$ und $\theta_{\mathcal{J}} \in \mathbb{R}_{>0}^{\mathcal{T}_{\mathcal{J}}}$ Skalierungen. Weiter seien $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$, $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine rekursive Darstellung von S, $(L_{\mathcal{I},i})_{i=0}^m$ rekursive externe Gewichte von H zu $\omega_{\mathcal{I}}$ in $\tilde{b}$ und $(L_{\mathcal{J},i})_{i=0}^m$ rekursive externe Gewichte von H^T zu $\omega_{\mathcal{J}}$ in $\tilde{b}^T$. Es gelten die Voraussetzungen (7.1) und (7.2) im Block $\breve{b}$. Dann benötigt Algorithmus 7.2.5 aufgerufen mit $\tilde{b}$ und $\breve{b}$ höchstens

$$(4c_{qr} + 2)k_{\max}^3 \left(\sum_{s \in \operatorname{chil}^*(\tilde{s}) \backslash \{\breve{s}\}} \#\mathcal{L}^+_{\mathbf{t}\times\mathbf{s}} + \sum_{t \in \operatorname{chil}^*(\tilde{t}) \backslash \{\breve{t}\}} \#\mathcal{L}^+_{\mathbf{t}\times\breve{\mathbf{s}}} \right)$$

Operationen, wobei $k_{\max} := \max_{t \in \mathcal{T}_{\mathcal{I}}} \#\kappa_t$ sei. Die berechneten Matrizen $L_{\breve{b},t}$, $t \in \operatorname{desc}(\breve{t})$, und $L_{\breve{b},s}$, $s \in \operatorname{desc}(\breve{s})$, sind externe Gewichte und die Matrizen $C_{\breve{b},t}$ und $D_{\breve{b},s}$ sind als Identitäten angelegt. Somit sind eine rekursive Darstellung der Kopplungsmatrizen und rekursive externe Gewichte nach dem Aufruf von Algorithmus 7.2.5 für $\breve{b}$ gegeben.

BEWEIS: Wir betrachten das Gewicht $L_{\breve{b},t}$ für einen Zeilencluster $t \in \operatorname{desc}(\breve{t})$. Der Beweis für die Spaltencluster verläuft analog. Zu Beginn von Algorithmus 7.2.5 werden in $L_{\breve{b},t}$ die externen Gewichte zum Block $\tilde{b}$ gespeichert. Diese Matrix bezeichnen wir mit $L^{(0)}_{\breve{b},t}$.

Danach werden nacheinander Blöcke hinzugefügt. Es sei $p \in \mathbb{N}$ die Anzahl der Aufrufe von Algorithmus 7.2.2 für zulässige Blätter mit Zeilencluster t. Für alle $i \in \{1, \ldots, p\}$ bezeichne $(t, \tilde{s}_i)$ den i-ten dieser Blöcke und $L_{\check{b},t}^{(i)}$ die Matrix, mit der $L_{\check{b},t}$ nach dem i-ten dieser Aufrufe überschrieben wird. Wir zeigen, dass für alle $i \in \{0, \ldots, p\}$ eine orthogonale Matrix $Q_i \in \mathbb{R}^{\mathcal{J} \times \eta_{t,i}}_{\xi_{t,i} \times \eta_{t,i}}$ mit $\xi_{t,i} := \xi_{\check{b},t} \dot\cup \left(\dot\bigcup_{j=1}^{i} \mathbf{s}_i \right)$ und

$$L_{\check{b},t}^{(i)} Q_i^T = \sum_{\substack{s \in \mathrm{row}(t) \\ (t,s) \notin \mathcal{T}_{\check{\mathbf{b}}}}} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T + \sum_{j=1}^{i} \frac{1}{\omega_{(t,s_j)}} S_{(t,s_j)} W_{s_j}^T$$

existiert. Weil $L_{\check{b},t}^{(0)}$ gleich dem externen Gewicht $L_{\check{b},t}$ ist, gilt die Aussage nach Definition 7.2.13. Es gelte die Aussage für $i \in \{0, \ldots, p-1\}$. Dann gilt

$$\begin{aligned} \sum_{\substack{s \in \mathrm{row}(t) \\ (t,s) \notin \mathcal{T}_{\check{\mathbf{b}}}}} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T + \sum_{j=1}^{i+1} \frac{1}{\omega_{(t,s_j)}} S_{(t,s_j)} W_{s_j}^T &= L_{\check{b},t}^{(i)} Q_i^T + \frac{1}{\omega_{(t,s_{i+1})}} S_{(t,s_{i+1})} W_{s_{i+1}}^T \\ &= \begin{pmatrix} L_{\check{b},t}^{(i)} & \frac{1}{\omega_{(t,s_{i+1})}} S_{(t,s_{i+1})} \end{pmatrix} \begin{pmatrix} Q_i & W_{s_{i+1}} \end{pmatrix}^T. \end{aligned}$$

Nach Lemma 3.3.14 ist $s_{i+1} \notin \mathrm{pred}(\check{s}) \backslash \{\check{s}\}$ und somit ist wegen Voraussetzung (7.1) die Clusterbasis $W_s \in \mathbb{R}^{\mathcal{J} \times \lambda_{s_{i+1}}}_{\mathbf{s}_{i+1} \times \lambda_{s_{i+1}}}$ orthogonal. Also ist die Matrix $\begin{pmatrix} Q_i & W_{s_{i+1}} \end{pmatrix} =: \tilde{Q}_{i+1} \in \mathbb{R}^{\mathcal{J} \times (\eta_{t,i} \cup \lambda_{s_{i+1}})}_{\xi_{t,i+1} \times (\eta_{t,i} \cup \lambda_{s_{i+1}})}$ nach Bemerkung 2.3.15 orthogonal. Im $(i+1)$-ten Schritt wird dann eine QR-Zerlegung

$$\begin{pmatrix} L_{\check{b},t}^{(i)} & \frac{1}{\omega_{(t,s_{i+1})}} S_{(t,s_{i+1})} \end{pmatrix}^T = \hat{Q}_{i+1} \left(L_{\check{b},t}^{(i)} \right)^T$$

berechnet. Die Matrix $Q_{i+1} := \tilde{Q}_{i+1} \hat{Q}_{i+1} \in \mathbb{R}^{\mathcal{J} \times \eta_{t,i+1}}_{\xi_{t,i+1} \times \eta_{t,i+1}}$ ist als Produkt orthogonaler Matrizen orthogonal und erfüllt

$$\sum_{\substack{s \in \mathrm{row}(t) \\ (t,s) \notin \mathcal{T}_{\check{\mathbf{b}}}}} \frac{1}{\omega_{(t,s)}} S_{(t,s)} W_s^T + \sum_{j=1}^{i+1} \frac{1}{\omega_{(t,s_j)}} S_{(t,s_j)} W_{s_j}^T = L_{\check{b},t}^{(i)} Q_{i+1}^T.$$

Um zu zeigen, dass das von Algorithmus 7.2.5 berechnete Gewicht $L_{\check{b},t} = L_{\check{b},t}^{(m)}$ ein externes Gewicht für den Block $\check{b}$ ist, müssen wir noch

$$\{s \in \mathrm{row}(t) \,|\, (t,s) \notin \mathrm{desc}(\check{b})\} = \{s \in \mathrm{row}(t) \,|\, (t,s) \notin \mathcal{T}_{\check{\mathbf{b}}}\} \dot\cup \{\tilde{s}_i \,|\, i \in \{1, \ldots, p\}\}$$

beweisen. Dafür reicht es, die Mengengleichheit

$$\{\tilde{s}_i \,|\, i \in \{1, \ldots, p\}\} = \{s \in \mathrm{row}(t) \,|\, (t,s) \in \mathcal{T}_{\check{\mathbf{b}}} \backslash \mathrm{desc}(\check{b})\}$$

zu zeigen.
"$\subseteq$": Es sei $s \in \{\tilde{s}_i \,|\, i \in \{1,\dots,p\}\}$. Dann ist $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, also $s \in \operatorname{row}(t)$. Außerdem ist $s \in \operatorname{desc}(s')$ für ein $s' \in \operatorname{chil}^*(\tilde{s})\backslash\{\breve{s}\}$. Weil die Pfade in Bäumen eindeutig sind, ist $s \notin \operatorname{desc}(\breve{s})$ und insbesondere $(t,s) \notin \operatorname{desc}(\breve{b})$. Da $(t,s) \in \mathcal{T}_{\tilde{\mathbf{b}}}$ gilt, folgt die erste Inklusion.
"$\supseteq$": Es sei $s \in \{s \in \operatorname{row}(t) \,|\, (t,s) \in \mathcal{T}_{\tilde{\mathbf{b}}}\backslash \operatorname{desc}(\breve{b})\}$. Dann ist $(t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und somit $(t,s) \neq \tilde{b}$. Also existiert ein Kind $b' = (t',s') \in \operatorname{chil}(\tilde{b})$ mit $(t,s) \in \operatorname{desc}(b')$. Wegen $t \in \operatorname{desc}(\breve{t})$ und der Eindeutigkeit der Pfade gilt $t' = \breve{t}$. Andererseits folgt aus $(t,s) \notin \operatorname{desc}(\breve{b})$, dass $s' \neq \breve{s}$ gilt. Damit ist $s' \in \operatorname{chil}^*(\tilde{s})\backslash\{\breve{s}\}$ und die zweite Inklusion folgt.
Als Nächstes schätzen wir den Aufwand ab. Weil wir für den Aufwand nur die arithmetischen Operationen zählen, können wir die Initialisierung der externen Gewichte und der Basiswechsel im Block $\breve{b}$ ignorieren. Es sei $s \in \operatorname{chil}^*(\tilde{s})\backslash\{\breve{s}\}$ und $(t',s') \in \mathcal{L}^+_{\breve{\mathbf{t}}\times\mathbf{s}}$. Dann berechnet Algorithmus 7.2.2 zuerst die Skalierung $\frac{1}{\omega_{(t',s')}} S_{(t',s')}$ in weniger als $\#\kappa_{t'}\#\lambda_{s'} \leq k_{\max}^2 \leq k_{\max}^3$ Operationen. Der Aufwand der anschließenden QR-Zerlegung ist durch

$$c_{qr}\#\ell\#\kappa_{t'} \min\{\#\ell, \#\kappa_{t'}\} \leq c_{qr}(\#\eta_{t'} + \#\lambda_{s'})\#\kappa_{t'}^2 \leq 2c_{qr}k_{\max}^3$$

beschränkt. Durch Aufsummieren über alle Cluster $s \in \operatorname{chil}^*(\tilde{s})\backslash\{\breve{s}\}$ und zulässigen Blätter $(t',s') \in \mathcal{L}^+_{\breve{\mathbf{t}}\times\mathbf{s}}$ folgt die Aufwandsabschätzung für die Zeilencluster.
Für die Spaltencluster folgen die Aussagen analog unter Verwendung des transponierten Blockbaums. ∎

Bemerkung 7.2.25
Es sei $\tilde{b} = (\tilde{t},\tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\backslash\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Dann wird Algorithmus 7.2.5, wie im Prototyp 7.2.3 beschrieben, während der Rekursion innerhalb der Schleife für die Kinder aufgerufen. Der Aufwand dafür lässt sich mit Lemma 7.2.24 abschätzen. Dazu betrachten wir die Summe

$$\begin{aligned}
\sum_{(\breve{t},\breve{s})\in\operatorname{chil}(\tilde{b})} \sum_{s\in\operatorname{chil}^*(\tilde{s})\backslash\{\breve{s}\})} \#\mathcal{L}^+_{\breve{\mathbf{t}}\times\mathbf{s}} &= \sum_{\breve{t}\in\operatorname{chil}^*(\tilde{t})} \sum_{\breve{s}\in\operatorname{chil}^*(\tilde{s})} \sum_{s\in\operatorname{chil}^*(\tilde{s}\backslash\{\breve{s}\})} \#\mathcal{L}^+_{\breve{\mathbf{t}}\times\mathbf{s}} \\
&= \sum_{\breve{t}\in\operatorname{chil}^*(\tilde{t})} \sum_{\breve{s}\in\operatorname{chil}^*(\tilde{s})} (\#\operatorname{chil}^*(\tilde{s}) - 1)\#\mathcal{L}^+_{\breve{\mathbf{t}}\times\breve{\mathbf{s}}} \\
&= (\#\operatorname{chil}^*(\tilde{s}) - 1)\#\mathcal{L}^+_{\tilde{\mathbf{b}}}.
\end{aligned}$$

Analog erhalten wir für die zweite Summe in Lemma 7.2.24

$$\sum_{(\breve{t},\breve{s})\in\operatorname{chil}(\tilde{b})} \sum_{t\in\operatorname{chil}^*(\tilde{t}\backslash\{\breve{t}\})} \#\mathcal{L}^+_{\mathbf{t}\times\breve{\mathbf{s}}} = (\#\operatorname{chil}^*(\tilde{t}) - 1)\#\mathcal{L}^+_{\tilde{\mathbf{b}}}.$$

Statt über $\#\mathcal{L}^+_{\mathbf{t}\times\breve{\mathbf{s}}}$ können wir auch die Terme $\#\mathcal{T}_{\mathbf{t}\times\breve{\mathbf{s}}}$ summieren und erhalten dann auf der rechten Seite $\mathcal{T}_{\tilde{\mathbf{b}}}$.

Nachdem wir das Abwärtslaufen während der Rekursion erläutert haben, beschäftigen wir uns als Nächstes mit dem Fall, dass im aktuellen Block $\tilde{b}$ ein Niedrigrangupdate berechnet

werden soll. Dann müssen wir zuerst aus den externen Gewichten $L_{\tilde{b}}$ lokale Gewichte im Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ berechnen. Weil wir diese für die erweiterte Matrix $\widetilde{H} = \widetilde{H}(H, R, \tilde{b})$ berechnen, definieren wir dafür eine Adaption von Algorithmus 7.2.2 für $\widetilde{H}$. Weil die erweiterten Clusterbasen i. A. nicht orthogonal sind, gehen wir davon aus, dass wir bereits orthogonale Gewichte für die erweiterten Clusterbasen in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ berechnet haben (siehe Algorithmus 5.3.1).

Algorithmus 7.2.6 Die Funktion berechnet die lokalen Gewicht für die Cluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$. Dabei werden die externen Gewichte $L = L_{\tilde{b}}$ durch lokale Gewichte überschrieben.

function LOKALE_GEWICHTE_UPDATE($\tilde{b}$, S, κ, λ, $\mathcal{K}$, $\tilde{\omega}$, O, ρ, L, η)
  if $\tilde{b} \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ **then**
    $\ell \leftarrow \eta_t \cup \rho_s$
    $\widehat{L}_t \in \mathbb{R}^{(\kappa_t \cup \mathcal{K})\times\ell}$
    $\widehat{L}_{t|(\kappa_t\cup\mathcal{K})\times\eta_t} \leftarrow L_t$
    $\widehat{L}_{t|\kappa_t\times\rho_s} \leftarrow \frac{1}{\tilde{\omega}_{(t,s)}} S_{(t,s)} O^T_{s|\rho_s\times\lambda_s}$
    $\widehat{L}_{t|\mathcal{K}\times\rho_s} \leftarrow \frac{1}{\tilde{\omega}_{t,s}} (O_{s|\rho_s\times\mathcal{K}})^T$
    QR-ZERLEGUNG($\widehat{L}_t^T, \ell, \ell, \kappa_t \cup \mathcal{K}, \breve{P}_t, L_t^T, \eta_t$)
  else if $\tilde{b} \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ **then**
    for $\breve{b} \in \mathrm{chil}(\tilde{b})$ **do**
      LOKALE_GEWICHTE_UPDATE($\breve{b}$, S, κ, λ, $\mathcal{K}$, $\tilde{\omega}$, O, ρ, L, η)
    end for
  end if
end function

In Algorithmus 7.2.6 wird deutlich, dass das externe Gewicht, mit dem gestartet wird, aus $\mathbb{R}^{(\kappa_t\cup\mathcal{K})\times\eta_t}$ sein muss. Die externen Gewichte vor dem Niedrigrangupdate sind allerdings in $\mathbb{R}^{\kappa_t\times\eta_t}$. Dieses Problem kann durch den einfachen Algorithmus 7.2.7 gelöst werden, der keine arithmetischen Operationen benötigt und vor Algorithmus 7.2.6 aufgerufen wird.

Algorithmus 7.2.7 Die Funktion legt die lokalen Gewichte für die erweiterte Matrix $\widetilde{H}$ für den Teilbaum $\mathcal{T}_{\tilde{\mathbf{t}}}$ an. Dabei gehen wir davon aus, das externe Gewichte $L_{\tilde{b}}$ gegeben sind.

function LOKALE_GEWICHTE_ERWEITERN(t, κ, $\mathcal{K}$, $L_{\tilde{b}}$, $\eta_{\tilde{b}}$, L, η)
  $L = 0 \in \mathbb{R}^{(\kappa_t\cup\mathcal{K})\times\eta_t}$
  $L_{|\kappa_t\times\eta_t} \leftarrow L_{\tilde{b}}$
  for $\breve{t} \in \mathrm{chil}(t)$ **do**
    LOKALE_GEWICHTE_ERWEITERN($\breve{t}$, κ, $L_{\tilde{b}}$, $\eta_{\tilde{b}}$, $\mathcal{K}$, L, η)
  end for
end function

Der Algorithmus 7.2.6 ermöglicht es uns, lokale Gewichte für die erweiterte $\mathcal{H}^2$-Matrix im Teilbaum $\mathcal{T}_{\tilde{\mathbf{b}}}$ zu berechnen. Um daraus adaptive Gewichte zu berechnen, nutzen wir eine

Adaption des Algorithmus 7.2.1 für das Niedrigrangupdate. Dabei ist es entscheidend, dass die Voraussetzung (7.2) erfüllt ist, damit wir im Cluster $\tilde{t}$ beginnen können.

Algorithmus 7.2.8 Die Funktion berechnet die adaptiven Gewichte der erweiterte $\mathcal{H}^2$-Matrix für die Cluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ unter Verwendung lokaler Gewichte L.

function ADAPTIVE_GEWICHTE_UPDATE($\tilde{t}$, E, $\kappa.\mathcal{K}$, $\tilde{\theta}$, L, η, Z, ζ)
  if $\tilde{t} = r_{\mathcal{I}}$ **then**
    $\widehat{Z}_t \leftarrow L_t \in \mathbb{R}^{(\kappa_{\tilde{t}} \dot\cup \mathcal{K})\times\eta}$
  else
    $\acute{t} := \operatorname{par}(t)$
    $\ell \leftarrow \eta_t \dot\cup \zeta_{\acute{t}}$
    $\widehat{Z}_t \in \mathbb{R}^{(\kappa_t \dot\cup \mathcal{K})\times\ell}$
    $\widehat{Z}_{t|(\kappa_t \dot\cup \mathcal{K})\times\eta_t} \leftarrow L_t$
    $\widehat{Z}_{t|\kappa_t\times\zeta_{\acute{t}}} \leftarrow \frac{1}{\theta_t} E_t Z_{\acute{t}|\kappa_{\acute{t}}\times\zeta_{\acute{t}}}$
    $\widehat{Z}_{t|\mathcal{K}\times\zeta_{\acute{t}}} \leftarrow \frac{1}{\theta_t} Z_{\acute{t}|\mathcal{K}\times\zeta_{\acute{t}}}$
    QR-ZERLEGUNG($\widehat{Z}_t^T, \ell, \ell, \kappa_t, \breve{P}_t, Z_t^T, \zeta_t$)
    for $\breve{t} \in \operatorname{chil}(\tilde{t})$ **do**
      ADAPTIVE_GEWICHTE_UPDATE($\breve{t}$, E, κ, $\mathcal{K}$, $\tilde{\theta}$, L, η, Z, ζ)
    end for
  end if
end function

Damit haben wir eine Möglichkeit, erst die lokalen Gewichte und dann die adaptiven Gewichte für die erweiterte Matrix $\widetilde{H}$ in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ zu berechnen. Anschließend können wir die adaptiven Clusterbasen lokal in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\mathcal{T}_{\tilde{\mathbf{s}}}$ berechnen. Weil wir für das Update Berechnungen nur im Teilblock $\mathcal{T}_{\tilde{\mathbf{b}}}$ vornehmen wollen, nutzen wir Algorithmus 5.3.5 statt Algorithmus 5.3.6, um die Kopplungsmatrizen innerhalb von $\mathcal{T}_{\tilde{\mathbf{b}}}$ anzupassen. Schließlich müssen wir noch die rekursiven Darstellungen aktualisieren. Nach Lemma 7.2.10 und Lemma 7.2.20 müssen wir die Basiswechsel und die externen Gewichte mit den Basiswechseln $C(\widetilde{V}, \overline{V})_{t|\overline{\kappa}_t\times\kappa_t}$ bzw. $C(\widetilde{W}, \overline{W})_{s|\overline{\lambda}_s\times\lambda_s}$ multiplizieren. Deshalb schneiden wir in Algorithmus 7.2.9 die zu $\mathcal{K}$ gehörigen Spalten der Basiswechsel ab.

Algorithmus 7.2.9 Die Funktion schneidet den hinteren Teil der Basiswechsel ab.

function BASISWECHSEL_ABSCHNEIDEN($\tilde{t}$, $\overline{\kappa}$, κ, C)
  $C_t \leftarrow C_{t|\overline{\kappa}_t\times\kappa_t} \in \mathbb{R}^{\overline{\kappa}_t\times\kappa_t}$
  for $\breve{t} \in \operatorname{chil}(\tilde{t})$ **do**
    BASISWECHSEL_ABSCHNEIDEN($\breve{t}$, $\overline{\kappa}$, κ, C)
  end for
end function

Mit den abgeschnittenen Basiswechseln können wir die Basiswechsel und die externen Gewichte aktualisieren. Diese Aufgabe erledigt der Algorithmus 7.2.10. Weil in beiden Fällen eine Familie von Matrizen für einen Clusterbaum mit einer anderen solchen Familie

multipliziert wird, benötigen wir nur eine Funktion.

Algorithmus 7.2.10 Die Funktion multipliziert die Familie von Matrizen X im Clusterbaum mit den Basiswechseln C. Dabei wird das Ergebnis wieder in X gespeichert.

```
function BASISWECHSEL_CLUSTERBAUM(t, C, κ̄, κ, X, η)
    X_t ← C_{t|κ̄_t×κ_t} X_t ∈ ℝ^{κ̄_t × η_t}
    for ť ∈ chil(t) do
        BASISWECHSEL_CLUSTERBAUM(ť, C, κ̄, κ, X, η)
    end for
end function
```

Damit können wir das Niedrigrangupdate für rekursive Algorithmen in der 2. Variante definieren. Es ist derart gestaltet, dass es die Voraussetzungen (7.1) und (7.2) erhält und die rekursiven Darstellungen aktualisiert. Dies und eine Aufwandsabschätzung zeigen wir in Theorem 7.2.26.

Theorem 7.2.26
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, S, N)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem, strikt zulässigem Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und für die Clusterbasen V und W gelte (7.1). Weiter sei $R = AB^T$ eine Niedrigrangmatrix mit Rang-$\mathcal{K}$-Darstellung und $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $R \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}_{\tilde{\mathbf{t}}\times\tilde{\mathbf{s}}}$. Es gelten (7.2) für gegebene Skalierungen $\omega_{\mathcal{I}} \in \mathbb{R}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}_{>0}$ und $\theta_{\mathcal{I}} \in \mathbb{R}^{\mathcal{T}_{\mathcal{I}}}_{>0}$ bzw. $\omega_{\mathcal{J}} \in \mathbb{R}^{(\mathcal{L}^T_{\mathcal{I}\times\mathcal{J}})^+}_{>0}$ und $\theta_{\mathcal{J}} \in \mathbb{R}^{\mathcal{T}_{\mathcal{J}}}_{>0}$. Weiter seien $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$, $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine rekursive Darstellung von S, $(L_{\mathcal{I},i})_{i=0}^m$ rekursive externe Gewichte von H zu $\omega_{\mathcal{I}}$ in $\tilde{b}$ und $(L_{\mathcal{J},i})_{i=0}^m$ rekursive externe Gewichte von H^T zu $\omega_{\mathcal{J}}$ in $\tilde{b}^T$. Wir definieren die maximale Blattgröße $n_{\max} := \max\{\max_{t\in\mathcal{L}_{\mathcal{I}}} \#\mathbf{t}, \max_{s\in\mathcal{L}_{\mathcal{J}}} \#\mathbf{s}\}$ und den maximalen lokalen Rang

$$k_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_t, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_s, \max_{t\in\mathcal{T}_{\mathcal{I}}} \#\kappa_{m-1,t}, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\lambda_{m-1,s}, \mathcal{K}, \max_{t\in\mathcal{T}_{\mathcal{I}}} \#\overline{\kappa}_t, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\overline{\lambda}_s\}.$$

Dann benötigt der Algorithmus 7.2.11 für die Berechnung von $\overline{H} \approx H + X$ höchstens

$$\begin{aligned}&(4c_{apr} + 4c_{qr} + 12)k^2_{\max}(\#\tilde{t} + \#\tilde{s}) + (4c_{apr} + 24c_{qr} + 40)k^3_{\max}(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})\\ &+ \left((8c_{\|.\|} + 32c_{qr} + 72)k^3_{\max} + 2k_{\max}\max\{3k^2_{\max}, n^2_{\max}\}\right)\#\mathcal{T}_{\tilde{\mathbf{b}}}\end{aligned}$$

Operationen. Die $\mathcal{H}^2$-Matrix-Darstellung von H wird mit der von $\overline{H}$ mit Clusterbasen $\overline{V}$ und $\overline{W}$ überschrieben, die (7.1) erfüllen. Dann erfüllen die alten Gewichte $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, $L_{\tilde{b}}$ und $L_{\tilde{b}^T}$ auch für die neue Matrix $\overline{H}$ und die Skalierungen $\omega_{\mathcal{I}}$, $\omega_{\mathcal{J}}$, $\theta_{\mathcal{I}}$ und $\theta_{\mathcal{J}}$ die Bedingung (7.2). Außerdem sind die rekursiven Darstellungen und rekursiven externen Gewichte aktualisiert für die neue $\mathcal{H}^2$-Matrix $\overline{H}$.

Beweis: Zuerst werden mit Algorithmus 5.3.1 nach Lemma 5.3.8 orthogonale Gewichte für die erweiterten Clusterbasen in weniger als

$$4c_{qr}k^2_{\max}(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (4 + 8c_{qr})k^3_{\max}(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Algorithmus 7.2.11 Die Funktion berechnet das Niedrigrangupdate einer $\mathcal{H}^2$-Matrix $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ für eine Rang-$\mathcal{K}$-Matrix $R = AB^T$ für die 2. Variante.

function NIEDRIGRANGUPDATE_VARIANTE2($\tilde{b}$, H, $C_{\tilde{b}}$, $C_{\tilde{b}^T}$, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $Z_{\mathcal{I}}$, $Z_{\mathcal{J}}$, R, opt, ϵ, ω, str)

$(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, (V, E), (W, F), N, S) \leftarrow H$, $(A, B) \leftarrow R$

LOKALE_ORTHOGONALE_GEWICHTE($\tilde{t}$, V, κ, A, $\mathcal{K}$, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$)

LOKALE_ORTHOGONALE_GEWICHTE($\tilde{s}$, W, λ, B, $\mathcal{K}$, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$) ▷ Algorithmus 5.3.1

SKALIERUNG($\tilde{b}$, S, κ, λ, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$, opt, str, $\omega_{\mathcal{I}}$)

SKALIERUNG($\tilde{b}^T$, S^T, λ, κ, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$, opt, str, $\omega_{\mathcal{J}}$) ▷ Algorithmus 4.5.2

LOKALE_GEWICHTE_ERWEITERN($\tilde{t}$, κ, $\mathcal{K}$, $L_{\tilde{b}}$, $\eta_{\tilde{b}}$, $L_{\mathcal{I}}$, $\eta_{\mathcal{I}}$)

LOKALE_GEWICHTE_ERWEITERN($\tilde{s}$, λ, $\mathcal{K}$, $L_{\tilde{b}^T}$, $\eta_{\tilde{b}^T}$, $L_{\mathcal{J}}$, $\eta_{\mathcal{J}}$) ▷ Algorithmus 7.2.7

LOKALE_GEWICHTE_UPDATE($\tilde{b}$, S, κ, λ, $\mathcal{K}$, $\tilde{\omega}$, $O_{\mathcal{I}}$, $\rho_{\mathcal{I}}$, $L_{\mathcal{I}}$, $\eta_{\mathcal{I}}$)

LOKALE_GEWICHTE_UPDATE($\tilde{b}$, S^T, λ, κ, $\mathcal{K}$, $\tilde{\omega}$, $O_{\mathcal{J}}$, $\rho_{\mathcal{J}}$, $L_{\mathcal{J}}$, $\eta_{\mathcal{J}}$) ▷ Algorithmus 7.2.6

ADAPTIVE_GEWICHTE_UPDATE($\tilde{t}$, $\widetilde{E}$, $\tilde{\theta}$, $L_{\mathcal{I}}$, $\eta_{\mathcal{I}}$, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$)

ADAPTIVE_GEWICHTE_UPDATE($\tilde{s}$, $\widetilde{F}$, $\tilde{\theta}$, $L_{\mathcal{J}}$, $\eta_{\mathcal{J}}$, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$) ▷ Algorithmus 7.2.8

LOKALE_ADAPTIVE_CLUSTERBASIS($\tilde{t}$, $\tilde{t}$, V, κ, A, $\mathcal{K}$, $Z_{\mathcal{I}}$, $\zeta_{\mathcal{I}}$, $\overline{\kappa}$, $C_{\mathcal{I}}$, opt, $\epsilon_{\mathcal{I}}$)

LOKALE_ADAPTIVE_CLUSTERBASIS($\tilde{s}$, $\tilde{s}$, W, λ, B, $\mathcal{K}$, $Z_{\mathcal{J}}$, $\zeta_{\mathcal{J}}$, $\overline{\lambda}$, $C_{\mathcal{J}}$, opt, $\epsilon_{\mathcal{J}}$) ▷ Algorithmus 5.3.3

H2-MATRIX-PROJEKTION_INNERHALB($\tilde{b}$, N, S, κ, λ, A, B, $\mathcal{K}$, $C_{\mathcal{I}}$, $\overline{\kappa}$, $C_{\mathcal{J}}$, $\overline{\lambda}$) ▷ Algorithmus 5.3.5

BASISWECHSEL_ABSCHNEIDEN($\breve{t}$, $\overline{\kappa}$, κ, $C_{\mathcal{I}}$)

BASISWECHSEL_ABSCHNEIDEN($\breve{s}$, $\overline{\lambda}$, λ, $C_{\mathcal{J}}$) ▷ Algorithmus 7.2.9

BASISWECHSEL_CLUSTERBAUM($\tilde{t}$, $C_{\mathcal{I}}$, $\overline{\kappa}$, κ, $C_{\tilde{b}}$, $\kappa_{\acute{b}}$)

BASISWECHSEL_CLUSTERBAUM($\tilde{s}$, $C_{\mathcal{J}}$, $\overline{\lambda}$, λ, $D_{\tilde{b}^T}$, $\lambda_{\acute{b}^T}$) ▷ Algorithmus 7.2.10

BASISWECHSEL_CLUSTERBAUM($\tilde{t}$, $C_{\mathcal{I}}$, $\overline{\kappa}$, κ, $L_{\tilde{b}}$, $\eta_{\tilde{b}}$)

BASISWECHSEL_CLUSTERBAUM($\tilde{s}$, $D_{\mathcal{J}}$, $\overline{\lambda}$, λ, $L_{\tilde{b}^T}$, $\eta_{\tilde{b}^T}$) ▷ Algorithmus 7.2.10

end function

Operationen berechnet. Der Aufwand für die Berechnung der Skalierungen ist nach Lemma 4.5.7 durch $8(c_{\|.\|}+7)k_{\max}^3\#\mathcal{T}_{\tilde{\mathbf{b}}}$ beschränkt. Mit der Definition 5.1.4 der erweiterten Kopplungsmatrizen gilt für alle $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$

$$\begin{aligned}\begin{pmatrix} L_{\tilde{b},t} \\ 0_{\mathcal{K}\times\eta_{\tilde{b},t}} \end{pmatrix} Q_{\tilde{b},t}^T &= \begin{pmatrix} \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega_{(t,s)}} S_{(t,s)} \\ 0_{\mathcal{K}\times\eta_{\tilde{b},t}} \end{pmatrix} = \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega_{(t,s)}} \begin{pmatrix} S_{(t,s)} \\ 0_{\mathcal{K}\times\eta_{\tilde{b},t}} \end{pmatrix} \\ &= \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\notin\mathcal{T}_{\tilde{\mathbf{b}}}}} \frac{1}{\omega_{(t,s)}} \widetilde{S}(H,R,\tilde{b}).\end{aligned}$$

Also gibt der Algorithmus 7.2.7 externe Gewichte für die erweiterte $\mathcal{H}^2$-Matrix $\widetilde{H}$ zum Block $\tilde{b}$ in $L_{\mathcal{I}}$ zurück. Genauso ist nach dem Aufruf von Algorithmus 7.2.7 für die Spaltencluster ein externes Gewicht für $\widetilde{H}^T$ in $\tilde{b}^T$ in $L_{\mathcal{J}}$ gespeichert. Wie im Beweis von Lemma 7.2.24 folgt, dass nach den beiden Aufrufen von Algorithmus 7.2.6 lokale Gewichte für die erweiterte Matrix in $L_{\mathcal{I}}$ und $L_{\mathcal{J}}$ gespeichert sind. Der Unterschied zu Algorithmus 7.2.2 besteht darin, dass wir statt der Kopplungsmatrizen S das Produkt der erweiterten Kopplungsmatrizen $\widetilde{S}$ mit dem zugehörigen orthogonalen Gewicht $O_{\mathcal{I}}$ bzw. $O_{\mathcal{J}}$ verwenden müssen, wobei wir $\widetilde{S}$ wie in Abschnitt 5.3 nicht explizit aufstellen. Der Algorithmus 7.2.6 berechnet für $L_{\mathcal{I}}$ in jedem zulässigen Blatt $(t,s) \in \mathcal{L}_{\tilde{\mathbf{b}}}^+$ das Produkt $(1/\omega_{(t,s)})S_{(t,s)}(O_{s|\rho_s\times\lambda_s})^T$ und die Skalierung $(1/\omega_{(t,s)})(O_{s|\rho_s\times\mathcal{K}})^T$ in weniger als

$$\begin{aligned}3\#\kappa_t\#\lambda_s\#\rho_s + \#\mathcal{K}\#\rho_s &\leq 3\#\kappa_t\#\lambda_s(\#\lambda_s+\#\mathcal{K}) + \#\mathcal{K}(\#\lambda_s+\#\mathcal{K}) \\ &\leq 6k_{\max}^3 + 2k_{\max}^2 \leq 8k_{\max}^3\end{aligned}$$

Operationen. Zusätzlich wird die QR-Zerlegung von $\widehat{L}_t^T$ in weniger als

$$\begin{aligned}c_{qr}\#\ell(\#\kappa_t+\#\mathcal{K})\min\{\#\ell,(\#\kappa_t+\#\mathcal{K})\} &\leq c_{qr}(\#\eta_t+\#\rho_s)(\#\kappa_t+\#\mathcal{K})^2 \\ &\leq c_{qr}((\#\kappa_t+\#\mathcal{K})+(\#\lambda_s+\#\mathcal{K}))4k_{\max}^2 \\ &\leq 16c_{qr}k_{\max}^3\end{aligned}$$

Operationen berechnet. Analog folgt die Aufwandsabschätzung für die Berechnung von $L_{\mathcal{J}}$ mit Algorithmus 7.2.6. Der Aufwand für beide Aufrufe zusammen ist somit durch $(16+32c_{qr})k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{b}}}^+)$ beschränkt.

Der Algorithmus 7.2.8 berechnet für jeden Cluster in $\mathcal{T}_{\tilde{\mathbf{t}}}$ und $\widetilde{H}$ die gleichen Größen wie Algorithmus 7.2.1 für $\mathcal{T}_{\tilde{\mathbf{t}}}$ und H. Weil durch die Rekursion und die vorherigen Berechnungen jeweils sichergestellt ist, dass ein adaptives Gewicht für das jeweilige Elter und ein lokales Gewicht des aktuellen Clusters gegeben sind, berechnet Algorithmus 7.2.8 adaptive Gewichte (vgl. Beweis von 7.2.23). Für $t \in \mathcal{T}_{\tilde{\mathbf{t}}}\backslash\{r_{\mathcal{I}}\}$ und $\acute{t} := \operatorname{par}(t)$ benötigt das Aufstellen der Matrix $\widehat{Z}_t$ höchstens

$$3\#\kappa_t\#\kappa_{\acute{t}}\#\zeta_{\acute{t}} + \#\mathcal{K}\#\zeta_{\acute{t}} \leq 3k_{\max}^2(\#\kappa_{\acute{t}}+\#\mathcal{K}) + k_{\max}(\#\kappa_{\acute{t}}+\#\mathcal{K}) \leq 8k_{\max}^3$$

Operationen. Zusätzlich wird die QR-Zerlegung von $\widehat{Z}_t^T$ in weniger als

$$\begin{aligned} c_{qr}\#\ell(\#\kappa_t + \#\mathcal{K})\min\{\#\ell, (\#\kappa_t + \#\mathcal{K})\} &\leq c_{qr}(\#\eta_t + \#\zeta_{t'})(\#\kappa_t + \#\mathcal{K})^2 \\ &\leq c_{qr}(4k_{\max})(2k_{\max})^2 = 16c_{qr}k_{\max}^3. \end{aligned}$$

Operationen berechnet. Für $t = r_{\mathcal{I}}$ werden keine Berechnungen vorgenommen und für $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ erhalten wir die gleiche Aufwandsabschätzung. Damit gilt für die beiden Aufrufe von Algorithmus 7.2.8 die obere Schranke $(8 + 16c_{qr})k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$.
Die Berechnung der adaptiven Clusterbasen mit Algorithmus 5.3.3 benötigt nach Lemma 5.3.15 höchstens

$$(4c_{apr} + 12)k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (4c_{apr} + 18)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$$

Operationen. Weil wir die alten Clusterbasen V und W in den Teilbäumen $\mathcal{T}_{\tilde{\mathbf{t}}}$ bzw. $\mathcal{T}_{\tilde{\mathbf{s}}}$ mit den neuen überschreiben, erfüllen die nach Algorithmus 5.3.3 gespeicherten Clusterbasen nach Lemma 5.2.1 die Voraussetzung (7.1). Die Voraussetzung (7.2) bleibt, wie im Beweis von Lemma 7.1.3 gezeigt, auch erhalten.
Für die neuen Clusterbasen berechnet Algorithmus 5.3.5 die neuen Kopplungs- und Nahfeldmatrizen innerhalb des Teilbaums von $\mathcal{T}_{\tilde{\mathbf{b}}}$. Dafür werden weniger als

$$\sum_{b \in \mathcal{L}_{\tilde{\mathbf{b}}}^+} 6k_{\max}^3 + \sum_{b \in \mathcal{L}_{\tilde{\mathbf{b}}}^-} 2k_{\max}n_{\max}^2 \leq 2k_{\max}\max\{3k_{\max}^2, n_{\max}^2\}\#\mathcal{T}_{\tilde{\mathbf{b}}}$$

Operationen benötigt (siehe Beweis von Lemma 5.3.17). Damit sind die Matrizen im Block $\tilde{b}$ alle aktuell und (7.3) für $i = m$ erfüllt. Es bleiben noch die rekursive Darstellung für $i \in \{0, \ldots, m-1\}$ und die rekursiven externen Gewichte zu aktualisieren. Mit Algorithmus 7.2.9 erhalten wir die Basiswechsel $C(\widetilde{V}, \overline{V})_{t|\overline{\kappa}_t \times \kappa_t}$ in $C_{\mathcal{I},t}$ und $C(\widetilde{W}, \overline{W})_{s|\overline{\lambda}_s \times \lambda_s}$ in $C_{\mathcal{J},s}$. Dann berechnet Algorithmus 7.2.10 in jedem Cluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ das Produkt des Basiswechsels $C_{\mathcal{I},t}$ aus dem aktuellen Update mit dem externen Gewicht $L_{m,t}$ bzw. mit dem Basiswechsel $C_{m,t}$. Damit ergibt sich für $t \in \mathcal{T}_{\mathcal{I}}$ ein Aufwand von höchstens

$$2\#\overline{\kappa}_t\#\kappa_t\#\eta_t \leq 2k_{\max}^3 \quad \text{bzw.} \quad 2\#\overline{\kappa}_t\#\kappa_t\#\kappa_{m-1,t} \leq 2k_{\max}^3$$

Operationen. Durch die Summe über alle Cluster $t \in \mathcal{T}_{\tilde{\mathbf{t}}}$ und analog über alle $s \in \mathcal{T}_{\tilde{\mathbf{s}}}$ erhalten wir die obere Schranke $8k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}})$. Nach Lemma 7.2.10 und Lemma 7.2.20 sind damit die rekursive Darstellung von S und die rekursiven externen Gewichte für die neue $\mathcal{H}^2$-Matrix $\overline{H}$ angepasst. Durch Summieren der einzelnen Aufwandsschranken erhalten wir die Schranke für den Gesamtaufwand. ∎

Bemerkung 7.2.27
Für das Niedrigrangupdate der 2. Variante gelten wie für die 1. Variante (siehe Bemerkung 7.1.4) die Fehlerabschätzungen aus Abschnitt 5.4, wobei wieder mit orthogonalen Clusterbasen und adaptiven Gewichten gerechnet wird.

Nachdem wir das Niedrigrangupdate beschrieben haben, wenden wir uns dem Aufwärtslaufen im Baum zu. Dabei untersuchen wir zuerst die innere Prolog-Funktion. Während des rekursiven Aufrufs für die verschiedenen Kinder müssen die berechneten Basiswechsel verarbeitet werden. Dazu werden die Kopplungsmatrizen, die Basiswechsel und die externen Gewichte für den Block $\tilde{b}$ mit den Basiswechseln aus dem rekursiven Aufruf für $\breve{b}$ multipliziert. Die Aktualisierung der Basiswechsel und der externen Gewichte berechnen wir wie beim Niedrigrangupdate mit Algorithmus 7.2.10. Für die Anpassung der Kopplungsmatrizen definieren wir den Algorithmus 7.2.12, der alle Kopplungsmatrizen in einem Blockbaum (also in den zulässigen Blättern) von links mit einer zum Zeilencluster gehörigen Matrix multipliziert. Für die Basiswechsel in den Spaltenclustern verwenden wir dann transponierte Blockbäume.

Algorithmus 7.2.12 Die Funktion multipliziert die Familie von Matrizen S in den zulässigen Blättern des Blockbaums mit den Basiswechseln C. Dabei wird das Ergebnis wieder in S gespeichert.

```
function Basiswechsel_Blockbaum(b, C, κ̄, κ, S, λ)
    if b ∈ L⁺_{I×J} then (t, s) ← b
        S_b ← C_{t|κ̄_t × κ_t} S_b ∈ ℝ^{κ̄_t × η_t}
    else if b ∈ T_{I×J} \ L_{I×J} then
        for b̆ ∈ chil(b) do
            Basiswechsel_Blockbaum(b̆, C, κ̄, κ, S, λ)
        end for
    end if
end function
```

Damit können wir die innere Epilog-Funktion definieren. Diese stellt sicher, dass die rekursiven Darstellungen im aktuellen Block entsprechend Lemma 7.2.11 und Lemma 7.2.21 angepasst werden.
Nach dem Aufruf der inneren Epilog-Funktion haben wir wieder eine rekursive Darstellung für den aktuellen Block inklusive rekursiver externer Gewichte. Dies ermöglicht uns, für das nächste Kind die innere Prolog-Funktion und danach die rekursive Funktion für dieses Kind aufzurufen. Außerdem stellen wir sicher, dass nach der Rekursion für die Kinder im aktuellen Block wieder die rekursiven Darstellungen aktuell sind. In Lemma 7.2.28 untersuchen wir die Eigenschaften von Algorithmus 7.2.13.

Lemma 7.2.28
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix, $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \setminus \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$, $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$ und $\breve{b} = (\breve{t}, \breve{s}) \in \operatorname{chil}(\tilde{b})$. Weiter seien $((S_i)_{i=0}^{m+1}, (C_i)_{i=1}^{m+1}, (D_i)_{i=1}^{m+1})$ eine rekursive Darstellung von S in $\breve{b}$ und $(L_{\mathcal{I},i})_{i=0}^{m+1}$ und $(L_{\mathcal{J},i})_{i=0}^{m+1}$ rekursive externe Gewichte von H zu $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ in $\breve{b}$ bzw. von H^T zu ω in $\breve{b}^T$. Dann überschreibt Algorithmus 7.2.13 die rekursiven Darstellungen derart, dass nach dem Aufruf mit $\tilde{b}$ und $\breve{b}$ die Familien $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine rekursive Darstellung von S in $\tilde{b}$ und $(L_{\mathcal{I},i})_{i=0}^m$ und $(L_{\mathcal{J},i})_{i=0}^m$ rekursive externe Gewichte von H zu $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ in $\tilde{b}$ bzw. von H^T zu ω in $\tilde{b}^T$ sind. Der

Algorithmus 7.2.13 Diese Funktion stellt sicher, dass die rekursiven Darstellungen im aktuellen Block $\tilde{b}$ durch die Basiswechsel, die aus dem Block $\breve{b}$ kommen, aktualisiert werden.

function EPILOG_INNEN_VARIANTE2($\tilde{b}$, $\breve{b}$, $C_{\breve{b}}$, $C_{\breve{b}^T}$, S, $L_{\tilde{b}}$, $L_{\tilde{b}^T}$, $C_{\tilde{b}}$, $C_{\tilde{b}^T}$, opt, ϵ, str)
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$, $(\breve{t}, \breve{s}) \leftarrow \breve{b}$
  for $s \in \mathrm{chil}^*(\tilde{s}) \backslash \{\breve{s}\}$ **do**
    BASISWECHSEL_BLOCKBAUM($(\breve{t}, s)$, $C_{\breve{b}}$, $\kappa_{\breve{b}}$, $\kappa_{\tilde{b}}$, S^T, λ) ▷ Algorithmus 7.2.12
  end for
  BASISWECHSEL_CLUSTERBAUM($\tilde{s}$, $C_{\breve{b}}$, $\kappa_{\breve{b}}$, $\kappa_{\tilde{b}}$, $C_{\tilde{b}}$, $\kappa_{\tilde{b}}$) ▷ Algorithmus 7.2.10
  BASISWECHSEL_CLUSTERBAUM($\tilde{s}$, $C_{\breve{b}}$, $\kappa_{\breve{b}}$, $\kappa_{\tilde{b}}$, $L_{\tilde{b}}$, $\eta_{\tilde{b}}$) ▷ Algorithmus 7.2.10
  for $t \in \mathrm{chil}^*(\tilde{t}) \backslash \{\breve{t}\}$ **do**
    BASISWECHSEL_BLOCKBAUM($(\breve{s}, t)$, $C_{\breve{b}^T}$, $\lambda_{\breve{b}}$, $\lambda_{\tilde{b}}$, S^T, κ) ▷ Algorithmus 7.2.12
  end for
  BASISWECHSEL_CLUSTERBAUM($\tilde{t}$, $C_{\breve{b}^T}$, $\lambda_{\breve{b}}$, $\lambda_{\tilde{b}}$, $C_{\tilde{b}^T}$, $\lambda_{\tilde{b}^T}$) ▷ Algorithmus 7.2.10
  BASISWECHSEL_CLUSTERBAUM($\tilde{t}$, $C_{\breve{b}^T}$, $\lambda_{\breve{b}}$, $\lambda_{\tilde{b}}$, $L_{\tilde{b}^T}$, $\eta_{\tilde{b}^T}$) ▷ Algorithmus 7.2.10
end function

Aufwand des Algorithmus ist höchstens

$$2k_{\max}^3 \left(\sum_{s' \in \mathrm{chil}^*(\tilde{s}) \backslash \{\breve{s}\}} \#\mathcal{L}^+_{\breve{\mathbf{t}} \times \mathbf{s}'} + \sum_{t' \in \mathrm{chil}^*(\tilde{t}) \backslash \{\breve{t}\}} \#\mathcal{L}^+_{\mathbf{t}' \times \breve{\mathbf{s}}} \right) \; + \; 4k_{\max}^3 (\#\operatorname{desc}(\breve{t}) + \#\operatorname{desc}(\breve{s})),$$

wobei $k_{\max} := \max\{\max_{t \in \operatorname{desc}(\breve{t})} \kappa_{m+1,t}, \max_{t \in \mathcal{T}_{\tilde{\mathbf{t}}}} \kappa_{m,t}, \max_{s \in \operatorname{desc}(\breve{s})} \lambda_{m+1,s}, \max_{s \in \mathcal{T}_{\tilde{\mathbf{s}}}} \lambda_{m,s}\}$ der maximale lokale Rang sei.

BEWEIS: Nach Lemma 7.2.11 müssen wir für alle $b = (t, s) \in \mathcal{M}^+_{\breve{b},m} = \mathcal{L}^+_{\tilde{\mathbf{b}}} \backslash \operatorname{desc}(\breve{b})$ mit $t \in \operatorname{desc}(\breve{t})$ die Kopplungsmatrix S_b mit dem Basiswechsel $C_{m+1,t} = C_{\breve{b},t}$ multiplizieren. Weil nach Lemma 3.3.14 (iii) und $\tilde{b} \notin \mathcal{L}_{\mathcal{I} \times \mathcal{J}}$

$$\begin{aligned} \{(t,s) \in \mathcal{L}^+_{\tilde{\mathbf{b}}} \backslash \operatorname{desc}(\breve{b}) \,|\, t \in \operatorname{desc}(\breve{t})\} &= \{(t,s) \in \mathcal{L}^+_{\tilde{\mathbf{b}}} \,|\, t \in \operatorname{desc}(\breve{t}) \vee s \notin \operatorname{desc}(\breve{s})\} \\ &= \dot{\bigcup_{s' \in \mathrm{chil}^*(\tilde{s}) \backslash \{\breve{s}\}}} \{(t,s) \in \mathcal{L}^+_{\mathcal{I} \times \mathcal{J}} \,|\, (t,s) \in \operatorname{desc}(\breve{t}, s')\} \end{aligned}$$

gilt, werden diese Produkte durch die Aufrufe von Algorithmus 7.2.12 in Algorithmus 7.2.13 berechnet und in den Kopplungsmatrizen gespeichert. Jedes dieser Produkte benötigt höchstens $2\#\kappa_{m+1,t}\#\kappa_{m,t}\#\lambda_s \leq 2k_{\max}^3$ Operationen. Also ist der Aufwand für diese Berechnungen durch

$$2k_{\max}^3 \sum_{s' \in \mathrm{chil}^*(\tilde{s}) \backslash \{\breve{s}\}} \#\mathcal{L}^+_{\breve{\mathbf{t}} \times \mathbf{s}'}$$

beschränkt. Analog ist der Aufwand für die Aktualisierung der Kopplungsmatrizen bezüglich der Spaltenprojektion mit Algorithmus 7.2.12 kleiner als

$$2k_{\max}^3 \sum_{t' \in \operatorname{chil}^*(\tilde{t}) \setminus \{\check{t}\}} \#\mathcal{L}^+_{\check{\mathbf{s}} \times \mathbf{t}'} \;=\; 2k_{\max}^3 \sum_{t' \in \operatorname{chil}^*(\tilde{t}) \setminus \{\check{t}\}} \#\mathcal{L}^+_{\mathbf{t}' \times \check{\mathbf{s}}}.$$

Zusätzlich müssen nach Lemma 7.2.11 die Basiswechsel $C_{m,t} = C_{\tilde{b},t}$ für alle $t \in \operatorname{desc}(\breve{t})$ mit $C_{m+1,t} = C_{\breve{b},t}$ und $D_{m,s} = C_{\tilde{b}^T,s}$ für alle $s \in \operatorname{desc}(\check{s})$ mit $D_{m+1,s} = C_{\breve{b}^T,s}$ multipliziert werden. Dafür benötigen die beiden entsprechenden Aufrufe von Algorithmus 7.2.10 höchstens

$$\begin{aligned}\sum_{t \in \operatorname{desc}(\breve{t})} 2\#\kappa_{m+1,t}\#\kappa_{m,t}\#\kappa_{m-1,t} + \sum_{s \in \operatorname{desc}(\check{s})} 2\#\lambda_{m+1,s}\#\lambda_{m,s}\#\lambda_{m-1,s} \\ \leq\; 2k_{\max}^3(\#\operatorname{desc}(\breve{t}) + \#\operatorname{desc}(\check{s}))\end{aligned}$$

Operationen. Weil die Ergebnisse in den alten Größen gespeichert werden, ist die rekursive Darstellung derart angepasst, dass $((S_i)_{i=0}^m, (C_i)_{i=1}^m, (D_i)_{i=1}^m)$ eine rekursive Darstellung von S darstellt. Für die rekursiven externen Gewichte müssen nach Lemma 7.2.21 die externen Gewichte $L_{\mathcal{I},m,t} = L_{\tilde{b},t}$ für alle $t \in \operatorname{desc}(\breve{t})$ mit $C_{m+1,t} = C_{\breve{b},t}$ und $L_{\mathcal{J},m,s} L_{\tilde{b}^T,s}$ für alle $s \in \operatorname{desc}(\check{s})$ mit $D_{m+1,s} = C_{\breve{b}^T,s}$ multipliziert werden. Diese Produkte werden mit Algorithmus 7.2.10 in höchstens

$$\begin{aligned}\sum_{t \in \operatorname{desc}(\breve{t})} 2\#\kappa_{m+1,t}\#\kappa_{m,t}\#\eta_{\mathcal{I},m,t} + \sum_{s \in \operatorname{desc}(\check{s})} 2\#\lambda_{m+1,s}\#\lambda_{m,s}\#\eta_{\mathcal{J},m,s} \\ \leq\; 2k_{\max}^3(\#\operatorname{desc}(\breve{t}) + \#\operatorname{desc}(\check{s}))\end{aligned}$$

Operationen berechnet. Also gilt die Behauptung. ∎

Für den ersten Teil der Aufwandsabschätzung haben wir die Summe über alle Söhne $\breve{b} \in \operatorname{chil}(\tilde{b})$ bereits in Bemerkung 7.2.25 untersucht. Als Nächstes wollen wir das Entsprechende für den zweiten Term betrachten.

Bemerkung 7.2.29
Es sei $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \setminus \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Dann gilt

$$\sum_{(\breve{t},\check{s}) \in \operatorname{chil}(\tilde{b})} \#\operatorname{desc}(\breve{t}) \;=\; \sum_{\check{s} \in \operatorname{chil}^*(\tilde{s})} \sum_{\breve{t} \in \operatorname{chil}^*(\tilde{t})} \#\operatorname{desc}(\breve{t}) \;\leq\; \#\operatorname{chil}^*(\tilde{s})\#\mathcal{T}_{\tilde{\mathbf{t}}}.$$

Analog folgt

$$\sum_{(\breve{t},\check{s}) \in \operatorname{chil}(\tilde{b})} \#\operatorname{desc}(\check{s}) \;\leq\; \#\operatorname{chil}^*(\tilde{t})\#\mathcal{T}_{\tilde{\mathbf{s}}}.$$

Für die äußere Prolog-Funktion müssen wir, wie für die Epilog-Funktion der 1. Variante die Clusterbasis im aktuellen Block orthogonalisieren. Aufgrund der rekursiven Darstellung der Kopplungsmatrizen multiplizieren wir diese nicht direkt mit den Basiswechseln, sondern aktualisieren den Basiswechsel. Dies führt zu Algorithmus 7.2.14, dessen Eigenschaften wir in Lemma 7.2.30 untersuchen.

Algorithmus 7.2.14 Diese Funktion stellt sicher, dass die Voraussetzungen (7.1) und (7.2) nach der Rekursion wieder für den aktuellen Block erfüllt sind.

function EPILOG_AUSSEN_VARIANTE2($\tilde{b}$, V, κ, W, λ, $C_{\tilde{b}}$, $C_{\tilde{b}^T}$)
  $(\tilde{t}, \tilde{s}) \leftarrow \tilde{b}$
  if $\tilde{t} \in \mathcal{T}_\mathcal{I} \backslash \mathcal{L}_\mathcal{I}$ **then**
    ORTHOGONALE_CLUSTERBASIS_CLUSTER($\tilde{t}$, E, κ, C, $\overline{\kappa}$) ▷ Algorithmus 7.1.4
    $C_{\tilde{b},\tilde{t}} \leftarrow C_{\tilde{t}} C_{\tilde{b},\tilde{t}} \in \mathbb{R}^{\overline{\kappa}_{\tilde{t}} \times \kappa_{m-1,t}}$
  end if
  if $\tilde{s} \in \mathcal{T}_\mathcal{J} \backslash \mathcal{L}_\mathcal{J}$ **then**
    ORTHOGONALE_CLUSTERBASIS_CLUSTER($\tilde{s}$, F, λ, D, $\overline{\lambda}$) ▷ Algorithmus 7.1.4
    $C_{\tilde{b}^T,\tilde{s}} \leftarrow C_{\tilde{s}} C_{\tilde{b}^T,\tilde{s}} \in \mathbb{R}^{\overline{\lambda}_{\tilde{s}} \times \lambda_{m-1,s}}$
  end if
end function

Lemma 7.2.30
Es sei $H = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V, W, N, S)$ eine $\mathcal{H}^2$-Matrix mit c_{sp}-schwachbesetztem Blockbaum und Rangverteilungen $\kappa = (\kappa_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $\lambda = (\lambda_s)_{s\in\mathcal{T}_\mathcal{J}}$. Es seien $\tilde{b} = (\tilde{t}, \tilde{s}) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$, $(\tilde{b}_i)_{i=0}^m$ der Pfad von $r_{\mathcal{I}\times\mathcal{J}}$ zu $\tilde{b}$ und $\breve{b} = (\breve{t}, \breve{s}) \in \operatorname{chil}(\tilde{b})$. Weiter seien die Voraussetzungen (7.1) und (7.2) für $\breve{b}$ erfüllt. Außerdem seien eine rekursive Darstellung der Kopplungsmatrizen S in $\tilde{b}$ und rekursive externe Gewichte $L_\mathcal{I}$ von H zu $\omega \in \mathbb{R}_{>0}^{\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ in $\tilde{b}$ bzw. $\mathcal{L}_\mathcal{J}$ von H^T zu ω in $\tilde{b}^T$ gegeben. Dann gelten nach dem Aufruf von Algorithmus 7.2.14 im Block $\tilde{b}$ die Voraussetzungen (7.1) und (7.2) für $\tilde{b}$ und die rekursiven Darstellungen werden derart angepasst, dass sie für die neue $\mathcal{H}^2$-Matrix-Darstellung gültig sind. Der Algorithmus 7.2.14 benötigt dafür höchstens

$$(c_{qr}(\#\operatorname{chil}(\tilde{t}) + \#\operatorname{chil}(\tilde{s})) + 4)k_{\max}^3$$

Operationen, wobei $k_{\max} := \max\{\max_{t\in\mathcal{T}_\mathcal{I}} \#\kappa_t, \max_{s\in\mathcal{T}_\mathcal{J}} \#\lambda_s\}$ sei.

Beweis: Die Voraussetzung (7.2) überträgt sich von $\breve{b}$ direkt auf $\tilde{b}$. Außerdem ist die Voraussetzung (7.1) nach den Aufrufen von Algorithmus 7.1.4 für $\tilde{b}$ erfüllt (vergleiche Lemma 7.1.8).
Es bleibt, die Kopplungsmatrizen anzupassen. Dies wird für die 1. Variante in Algorithmus 7.1.6 per Durchlauf der Blockzeile/-spalte realisiert. Für die 2. Variante müssen wir hingegen die rekursive Darstellung anpassen. Der aktuelle Block ist kein Blatt und somit haben wir für $\tilde{b}$ keine Kopplungsmatrix anzupassen. Damit erhalten wir für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$

$$\overline{S}_b := \begin{cases} C(V, \overline{V})_{\tilde{t}} S_b & , \text{falls } t = \tilde{t} \\ S_b C(W, \overline{W})_{\tilde{s}} & , \text{falls } s = \tilde{s} \\ S_b & , \text{sonst} \end{cases}.$$

Weil die Clusterbasen $V_{\tilde{t}}$ und $W_{\tilde{s}}$ nur für den Fall $\tilde{t} \in \mathcal{T}_{\mathcal{I}} \backslash \mathcal{L}_{\mathcal{I}}$ bzw. $\tilde{s} \in \mathcal{T}_{\mathcal{J}} \backslash \mathcal{L}_{\mathcal{J}}$ orthogonalisiert werden, gilt für die neuen Kopplungsmatrizen $\overline{S}_b = S_b$ für alle $b \in \mathcal{T}_{\tilde{\mathbf{b}}}$. Es sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \backslash \mathcal{T}_{\tilde{\mathbf{b}}}$.
1. Fall: Es sei $t = \tilde{t}$. Dann gilt $s \neq \tilde{s}$ und

$$\overline{S}_b \;=\; C(V, \overline{V})_{\tilde{t}} S_b \;=\; C(V, \overline{V})_{\tilde{t}} \left(\prod_{j=i_b}^{i_{\tilde{t}}} C_{j,\tilde{t}} \right) S_{i_b,b} \left(\prod_{j=i_b}^{i_s} D_{j,s} \right)^T .$$

Es gilt $i_{\tilde{t}} = m$. Indem wir den neuen Basiswechsel $\overline{C}_{m,\tilde{t}} := C(V, \overline{V})_{\tilde{t}} C_{m,\tilde{t}}$ setzen und die restlichen Matrizen übernehmen, erhalten wir in diesem Fall eine rekursive Darstellung für S_b.
2. Fall: Es sei $s = \tilde{s}$. Dann gilt $t \neq \tilde{t}$ und analog zum ersten Fall reicht es aus, den neuen Basiswechsel $\overline{D}_{m,\tilde{s}} := C(W, \overline{W})_{\tilde{s}} D_{m,\tilde{s}}$ zu berechnen.
3. Fall: Es sei $t \neq \tilde{t}$ und $s \neq \tilde{s}$. Dann müssen wir keine Anpassungen vornehmen.
In Algorithmus 7.2.14 werden zwei QR-Zerlegungen durch den Algorithmus 7.1.4 berechnet, deren Aufwand sich wie in Lemma 7.1.8 durch $c_{qr}(\#\operatorname{chil}(\tilde{t}) + \#\operatorname{chil}(\tilde{s}))k^3_{\max}$ abschätzen lassen. Die beiden Matrix-Produkte benötigen zusammen höchstens $4k^3_{\max}$ Operationen. Damit folgt die Behauptung. ∎

Insgesamt haben wir sichergestellt, dass bei einem rekursiven Algorithmus in der Art des Prototyps 7.2.3 die notwendigen Hilfsgrößen für das Niedrigrangupdate zur Verfügung stehen, und den notwendigen Rechenaufwand abgeschätzt. Damit können wir einen Algorithmus für das Matrix-Produkt mit Hilfe des Niedrigrangupdates in der 2. Variante formulieren.

7.3 Matrix-Produkt für die 2.Variante

In diesem Abschnitt beschreiben wir, wie das neue Update genutzt werden kann, um das Matrix-Produkt $C + AB$ für Probleme mit großen c_{sp}-Konstanten effizienter zu gestalten. Wir beschränken uns auf das Matrix-Produkt, weil die weiteren arithmetischen Operationen entsprechend angepasst werden können. Der Aufwand dabei kann dann genau wie in Kapitel 6 gegen den der Multiplikation abgeschätzt werden.
Es seien dazu $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_A, W_A, N_A, S_A)$, $B = \mathcal{H}^2(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}, V_B, W_B, N_B, S_B)$, $C = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{K}}, V_C, W_C, N_C, S_C)$ $\mathcal{H}^2$-Matrizen. Wie bereits erwähnt, wollen wir das Matrix-Produkt anders als in Abschnitt 6.2 strukturieren. Dafür definieren wir für alle $(t, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ die Menge $\mathcal{M}(t, r)$ aller Elemente $((t, s), (s, r)) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ des Produktbaums, die während der Berechnung des Matrix-Produkts mit Algorithmus 6.2.2 auftauchen. Weil die Elemente $\mathcal{M}(t, r)$ die rekursiven Aufrufe widerspiegeln, speichern wir diese als Tripel (t, s, r). Dabei lassen wir die Elemente aus, die durch die rekursiven Aufrufe von Algorithmus 6.2.1 aufgerufen werden.
Wir können diese Menge rekursiv beschreiben. Für die Wurzel startet der Algorithmus mit dem Tripel der Wurzeln und wir erhalten $\mathcal{M}(r_{\mathcal{I}}, r_{\mathcal{J}}) := \{(r_{\mathcal{I}}, r_{\mathcal{J}}, r_{\mathcal{K}})\}$.

Es sei $(t,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ und $(t,s,r) \in \mathcal{M}(t,r)$. Falls $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$, $(s,r) \in \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$ oder $(t,r) \in \mathcal{L}_{\mathcal{I}\times\mathcal{K}}$ gilt, wird für (t,s,r) der Algorithmus 6.2.1 zur Berechnung einer Niedrigrangmatrix aufgerufen. Ansonsten wird Algorithmus 6.2.2 rekursiv für alle $(\check{t},\check{s},\check{r})$ mit $\check{t} \in \operatorname{chil}^*(t)$, $\check{s} \in \operatorname{chil}^*(s)$ und $\check{r} \in \operatorname{chil}^*(r)$ aufgerufen.
Für $(t,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{K}}$ und $(\check{t},\check{r}) \in \operatorname{chil}(t,r) = \operatorname{chil}^*(t) \times \operatorname{chil}^*(r)$ erhalten wir also

$$\mathcal{M}(\check{t},\check{r}) \;=\; \dot{\bigcup_{\substack{(t,s,r)\in\mathcal{M}(t,s)\\(t,s)\notin\mathcal{L}_{\mathcal{I}\times\mathcal{J}}\\(s,r)\notin\mathcal{L}_{\mathcal{J}\times\mathcal{K}}}}} \{(\check{t},\check{s},\check{r}) \mid \check{s} \in \operatorname{chil}^*(s)\}.$$

Wir können also die Menge $\mathcal{M}(\check{t},\check{r})$ mit Hilfe der Menge $\mathcal{M}(t,r)$ konstruieren. Damit können wir den Algorithmus 7.3.1 zur Berechnung des $\mathcal{H}^2$-Matrix-Produkts basierend auf der 2. Variante des Niedrigrangupdates formulieren, wobei wir, wie in Kapitel 6, die Übergabeparameter zusammenfassen. Den Aufwand für Algorithmus 7.3.1 schätzen wir in Theorem 7.3.3 ab.

Bemerkung 7.3.1
Um den Algorithmus übersichtlich zu halten, fassen wir die Optionen zum Kürzen, die dabei verwendeten Genauigkeiten, die Strategien zur Berechnung der Skalierungen sowie die Skalierungen in dem Parameter *opt* zusammen. Ebenso fassen wir die Gewichte und Basiswechsel der rekursiven Darstellung von $\mathcal{H}^2$-Matrizen bezüglich der Zeilenbasis und der Spaltenbasis in einer Größe zusammenfassen.

Bevor wir den Aufwand von Algorithmus 7.3.1 (siehe S. 270) abschätzen, beweisen wir noch eine Hilfsaussage. Diese verwenden wir, um die Summen über die Mächtigkeit aller Teilbäume eines Blockbaums zu beschränken.

Lemma 7.3.2
Es sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein c_{sp}-schwachbesetzter Blockbaum. Dann gilt

$$\sum_{b\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{b}} \;\leq\; c_{sp}(p_{\mathcal{I}}\#\mathcal{T}_{\mathcal{I}} + p_{\mathcal{J}}\#\mathcal{T}_{\mathcal{J}}),$$

$$\sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{t}} \;\leq\; c_{sp}p_{\mathcal{I}}\#\mathcal{T}_{\mathcal{I}} \quad \text{und} \quad \sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{s}} \;\leq\; c_{sp}p_{\mathcal{J}}\#\mathcal{T}_{\mathcal{J}}.$$

Beweis: Wir gehen, wie im Beweis von Lemma 6.2.3 vor. Wir schreiben die Teilbäume als Nachfahren und nutzen aus, dass genau dann $\check{b} \in \operatorname{desc}(b)$ gilt, falls $b \in \operatorname{pred}(\check{b})$ ist. Mit Lemma 3.3.6 und Lemma 3.3.14 folgt

$$\begin{aligned}\sum_{b\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{b}} \;&=\; \sum_{b\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\operatorname{desc}(b) \;=\; \sum_{b\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\operatorname{pred}(b) \;\leq\; p_{\mathcal{I}\times\mathcal{J}}\#\mathcal{T}_{\mathcal{I}\times\mathcal{J}}\\ &\leq\; \max\{p_{\mathcal{I}},p_{\mathcal{J}}\}c_{sp}\min\{\#\mathcal{T}_{\mathcal{I}},\#\mathcal{T}_{\mathcal{J}}\} \;\leq\; c_{sp}(p_{\mathcal{I}}\#\mathcal{T}_{\mathcal{I}} + p_{\mathcal{J}}\#\mathcal{T}_{\mathcal{J}}).\end{aligned}$$

Für die Summe der Mächtigkeit der Zeilencluster erhalten wir

$$\begin{aligned}\sum_{(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{T}_{\mathbf{t}} &= \sum_{t\in\mathcal{T}_{\mathcal{I}}} \sum_{s\in\operatorname{row}(t)} \#\mathcal{T}_{\mathbf{t}} \leq \sum_{t\in\mathcal{T}_{\mathcal{I}}} c_{sp}\#\mathcal{T}_{\mathbf{t}} \\ &= c_{sp} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\operatorname{desc}(t) = c_{sp} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \#\operatorname{pred}(t) \leq c_{sp} p_{\mathcal{I}} \#\mathcal{T}_{\mathcal{I}}.\end{aligned}$$

■

Mit Hilfe dieses technischen Hilfsmittels können wir den Aufwand des Matrix-Produkts mit dem Niedrigrangupdate in der 2. Variante abschätzen.

Theorem 7.3.3
Es seien $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_A, W_A, N_A, S_A)$, $B = \mathcal{H}^2(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}, V_B, W_B, N_B, S_B)$ und $C = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{K}}, V_C, W_C, N_C, S_C)$ mit $c_{sp,A}$-, $c_{sp,B}$- bzw. $c_{sp,C}$-schwachbesetzten und strikt zulässigen Blockbäumen. Alle lokalen Ränge während des Algorithmus 7.3.1 seien durch $k_{\max}$ beschränkt, $n_{\max} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\mathbf{t}, \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\mathbf{s}, \max_{r\in\mathcal{T}_{\mathcal{K}}} \#\mathbf{r}\}$ sei die maximale Größe eines Blattclusters und

$$c_{\operatorname{chil}} := \max\{\max_{t\in\mathcal{T}_{\mathcal{I}}} \#\operatorname{chil}(t), \max_{s\in\mathcal{T}_{\mathcal{J}}} \#\operatorname{chil}(s), \max_{r\in\mathcal{T}_{\mathcal{K}}} \#\operatorname{chil}(r)\}$$

sei die maximale Anzahl von Söhnen eines Clusters. Wir definieren

$$\begin{aligned}c_1 &:= (8c_{apr} + 32)(c_{\operatorname{chil}}^3 + 1)k_{\max}^3, \\ c_2 &:= \max\{(4c_{qr} + 10)k_{\max}^2, (4c_{qr} + 10)k_{\max}n_{\max}, (4c_{qr} + 8)(c_{\operatorname{chil}}^3 + 1)k_{\max}^2\}, \\ c_3 &:= \max\{2k_{\max}^3, 2n_{\max}^3\}, \quad c_4 := (12 + 4c_{qr} + 4c_{apr}) \quad \text{und} \\ c_5 &:= (4c_{apr} + (72 + 8c_{\operatorname{chil}})c_{qr} + 12c_{\operatorname{chil}} + 8c_{\|.\|} + 126)\max\{k_{\max}^3, n_{\max}^3\}.\end{aligned}$$

Dann ist der Aufwand für die Berechnung der Niedrigrangmatrizen innerhalb von Algorithmus 7.3.1 höchstens

$$\begin{aligned}c_1 c_{sp,A} c_{sp,B} \min\{\#\mathcal{T}_{\mathcal{I}}, \#\mathcal{T}_{\mathcal{J}}, \#\mathcal{T}_{\mathcal{K}}\} &+ c_2 c_{sp,A} c_{sp,B}(p_{\mathcal{I}}\#\mathcal{I} + 2p_{\mathcal{J}}\#\mathcal{J} + p_{\mathcal{K}}\#\mathcal{K}) \\ &+ 2c_3 c_{sp,A} c_{sp,B}(p_{\mathcal{I}}\#\mathcal{T}_{\mathcal{I}} + 2p_{\mathcal{J}}\#\mathcal{T}_{\mathcal{J}} + p_{\mathcal{K}}\#\mathcal{T}_{\mathcal{K}})\end{aligned}$$

und der Aufwand für die Niedrigrangupdates inklusive der zugehörigen Hilfsfunktionen für das Durchlaufen des Blockbaums in Algorithmus 7.3.1 höchstens

$$c_4 c_{sp,C}(p_{\mathcal{I}}\#\mathcal{I} + p_{\mathcal{K}}\#\mathcal{K}) + c_5 c_{sp,C}(p_{\mathcal{I}}\#\mathcal{T}_{\mathcal{I}} + p_{\mathcal{K}}\#\mathcal{T}_{\mathcal{K}}).$$

Beweis: Der Aufwand für die Berechnung der Niedrigrangmatrizen lässt sich wie in Theorem 6.2.4 abschätzen. Allerdings kommt noch für jedes Element $(t, s, r) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ mit $(t, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ ein zusätzlicher Aufruf von Algorithmus 2.4.5 hinzu. Den zusätzlichen Aufwand können wir mit Lemma 2.4.13 abschätzen. Dies führt zu den leicht veränderten Konstanten c_1 und c_2.

Als Nächstes betrachten wir den Aufwand für die Hilfsfunktionen. Es sei also $b = (t,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{K}}$. Zuerst wird der äußere Prolog mit Algorithmus 7.2.4 aufgerufen. Der Aufwand ist nach Lemma 7.2.23 durch $(4c_{qr}+4)k_{\max}^3$ beschränkt. Innerhalb der Schleife über die Kinder werden für jedes $\breve{b} \in \operatorname{chil}(b)$ die Algorithmen 7.2.5 und 7.2.13 aufgerufen. Deren Aufwand können wir mit Hilfe der Lemmata 7.2.24 und 7.2.28 und den Bemerkungen 7.2.25 und 7.2.29 durch

$$\begin{aligned}(4c_{qr}+2)k_{\max}^3 2(c_{\text{chil}}-1)\#\mathcal{T}_{\mathbf{b}} + 2k_{\max}^3 2(c_{\text{chil}}-1)\#\mathcal{T}_{\mathbf{b}} + 4k_{\max}^3 c_{\text{chil}}(\#\mathcal{T}_{\mathbf{t}} + \#\mathcal{T}_{\mathbf{t}}) \\ = \ (8c_{qr}+8)k_{\max}^3(c_{\text{chil}}-1)\#\mathcal{T}_{\mathbf{b}} + 4k_{\max}^3 c_{\text{chil}}(\#\mathcal{T}_{\mathbf{t}} + \#\mathcal{T}_{\mathbf{s}})\end{aligned}$$

beschränken. Nach der For-Schleife wird noch Algorithmus 7.2.14 aufgerufen, der nach Lemma 7.2.30 höchstens $c_{qr}(\#\operatorname{chil}(t) + \#\operatorname{chil}(s) + 2)k_{\max}^3$ Operationen benötigt. Diesen Aufwand können wir wegen $\#\operatorname{chil}(t) + \#\operatorname{chil}(s) \leq 2\#\operatorname{chil}^*(t)\#\operatorname{chil}^*(s) = 2\#\operatorname{chil}(b)$ auf die Söhne aufteilen und erhalten für jeden Sohn den Anteil $(2c_{qr}+2)k_{\max}^3$.
Indem wir den Aufwand für die äußeren Hilfsfunktionen zum ersten Term der inneren Hilfsfunktionen hinzufügen, erhalten wir für den Aufwand der Hilfsfunktionen die obere Schranke

$$\sum_{b\in\mathcal{T}_{\mathcal{I}\times\mathcal{K}}} \left((8c_{qr}+8)k_{\max}^3 c_{\text{chil}}\#\mathcal{T}_{\mathbf{b}} + 4k_{\max}^3 c_{\text{chil}}(\#\mathcal{T}_{\mathbf{t}} + \#\mathcal{T}_{\mathbf{s}})\right).$$

Für den Aufwand der Niedrigrangupdates in der 2. Variante in einem Block $(t,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ gilt nach Theorem 7.2.26 die obere Schranke

$$\begin{aligned}&(4c_{apr}+4c_{qr}+12)k_{\max}^2(\#\tilde{\mathbf{t}} + \#\tilde{\mathbf{s}}) + (4c_{apr}+40c_{qr}+48)k_{\max}^3(\#\mathcal{T}_{\tilde{\mathbf{t}}} + \#\mathcal{T}_{\tilde{\mathbf{s}}}) \\ &+ \left((8c_{\|.\|}+32c_{qr}+72)k_{\max}^3 + 2k_{\max}\max\{3k_{\max}^2, n_{\max}^2\}\right)\#\mathcal{T}_{\tilde{\mathbf{b}}}.\end{aligned}$$

Die Aussage folgt mit Lemma 7.3.2. ∎

Damit haben wir einen Algorithmus für das Matrix-Produkt, dessen Aufwand für die Niedrigrangupdates nur noch mit $c_{sp,C}$ skaliert. Die Inversion und die Dreieckszerlegungen können wir wie in Kapital 6 gestalten und erhalten auch mit der neuen Variante den Aufwand des Matrix-Produkts als Schranke für den Aufwand. Zum Abschluss der Arbeit stellen wir im nächsten Kapitel noch numerische Ergebnisse und schließen mit einem kurzen Fazit.

Algorithmus 7.3.1 Die Funktion berechnet approximativ das Produkt zweier $\mathcal{H}^2$-Matrizen $A = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, V_A, W_A, N_A, S_A)$ und $B = \mathcal{H}^2(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}, V_B, W_B, N_B, S_B)$ mit Rangverteilungen κ_A und λ_A bzw. κ_B und λ_B und addiert das Ergebnis zu einer weiteren $\mathcal{H}^2$-Matrix $C = \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{K}}, V_C, W_C, N_C, S_C)$. Die Matrix C wird dabei mit dem Ergebnis $C + AB$ überschrieben.

function H2Matrix_Produkt_2(t, r, $\mathcal{M}$, A, B, C, $L_{(t,r)}$, Z, $C_{(t,r)}$, *opt*)
  if $(t,r) \in \mathcal{L}_{\mathcal{I}\times\mathcal{K}}$ **then**
    $A \leftarrow 0 \in \mathbb{R}^{\mathbf{t}\times\emptyset}$, $B \leftarrow 0 \in \mathbb{R}^{\mathbf{r}\times\emptyset}$, $R \leftarrow (A, B)$
    for $(t,s,r) \in \mathcal{M}(t,r)$ **do**
      H2Matrix_Produkt_RkMatrizen(t, s, r, A, B, $\widetilde{R}$, *opt*)
        ▷ Algorithmus 6.2.1
      RkMatrix_Summe($\widetilde{R}, R$, *opt*) ▷ Algorithmus 2.4.5
    end for
    Niedrigrangupdate((t,r), C, $C_{(t,r)}$, $L_{(t,r)}$, Z, R, *opt*) ▷ Algorithmus 7.2.11
  else ▷ $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{K}} \backslash \mathcal{L}_{\mathcal{I}\times\mathcal{K}}$
    for $(t,s,r) \in \mathcal{M}(t,r)$ **do**
      if $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \vee (s,r) \in \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$ **then**
        H2Matrix_Produkt_RkMatrizen(t, s, r, A, B, $\widetilde{R}$, *opt*)
          ▷ Algorithmus 6.2.1
        RkMatrix_Summe($\widetilde{R}, R$, *opt*) ▷ Algorithmus 2.4.5
      else
        for $\breve{t} \in \mathrm{chil}^*(t)$, $\breve{s} \in \mathrm{chil}^*(s)$, $\breve{r} \in \mathrm{chil}^*(r)$ **do**
          $\mathcal{M}(\breve{t}, \breve{r}) \leftarrow \mathcal{M}(\breve{t}, \breve{r}) \dot\cup \{(\breve{t}, \breve{s}, \breve{r})\}$
        end for
      end if
    end for
    Prolog_aussen_Variante2($(\tilde{t}, \tilde{r})$, E, κ, F, λ, $L_{(t,r)}$, Z, *opt*)
      ▷ Algorithmus 7.2.4
    for $(\breve{t}, \breve{r}) \in \mathrm{chil}(t,r)$ **do**
      Prolog_innen_Variante2($(\tilde{t}, \tilde{r})$, $(\breve{t}, \breve{r})$, S, κ, λ, $L_{(\tilde{t},\tilde{r})}$, $L_{(\breve{t},\breve{r})}$, $C_{(\breve{t},\breve{r})}$)
        ▷ Algorithmus 7.2.5
      H2Matrix_Produkt_2($\breve{t}$, $\breve{r}$, $\mathcal{M}, A$, B, C, $L_{(\breve{t},\breve{r})}$, $Z_{\mathcal{I}}$, $C_{(\breve{t},\breve{r})}$, *opt*)
      Epilog_innen_Variante2($(\tilde{t}, \tilde{r})$, $(\breve{t}, \breve{r})$, $C_{(\breve{t},\breve{r})}$, S, $L_{(\tilde{t},\tilde{r})}$, $C_{(\tilde{t},\tilde{r})}$, *opt*)
        ▷ Algorithmus 7.2.13
    end for
    Epilog_Variante2($\tilde{b}$, E, κ, F, λ, $C_{(\tilde{t},\tilde{r})}$) ▷ Algorithmus 7.2.14
  end if
end function

8 Numerische Experimente und Fazit

In diesem Kapitel stellen wir Experimente zur Arithmetik vor. Dabei beginnen wir mit Ergebnissen für den Laplace-Operator im $\mathbb{R}^2$, den wir mit einer Finite Elemente Methode (FEM) diskretisieren. In Abschnitt 8.2 stellen wir ein einfaches Beispiel für eine Integralgleichung auf Kurven im $\mathbb{R}^2$ vor, die wir mit einer Randelementmethode (BEM) diskretisieren. Für die ersten beiden Beispiele verwenden wir das Update aus Kapitel 5. Schließlich zeigen wir Zahlen für das Lösen einer mit BEM diskretisierten Integralgleichung auf der Kugeloberfläche im $\mathbb{R}^3$. Beim letzten Beispiel stellen wir die Niedrigrangupdates aus den Kapiteln 5 und 7 gegenüber. Die Rechenzeiten geben wir in Sekunden (sek) und die Speicheraufwände in Mebibyte (MiB) oder in Kibibyte pro Freiheitsgrad (KiB/n) an. Zum Abschluss des Kapitels fassen wir die Arbeit zusammen und geben einen kleinen Ausblick über mögliche anschließende Forschung.

8.1 FEM in der Ebene

Wir untersuchen die verallgemeinerte Laplace-Gleichung

$$-\operatorname{div}\alpha \operatorname{grad} u \;=\; f \quad \text{in } \Omega, \qquad u \;=\; 0 \quad \text{auf } \Gamma$$

für ein Gebiet $\Omega \subseteq \mathbb{R}^2$ mit Rand $\Gamma = \partial\Omega$ und der Koeffizientenfunktion $\alpha : \Omega \to \mathbb{R}$ mit $\alpha(\Omega) \subseteq [a, b]$ für $a, b \in \mathbb{R}_{>0}$. Für die FEM-Diskretisierung verwenden wir eine global verfeinerte Triangulation mit P1-Elementen.
Für das erste Experiment betrachten wir $\alpha \equiv 1$ und die drei Geometrien, das komplette Quadrat $[-1, 1]$, ein enthaltenes L-Gebiet und ein enthaltenes U-Gebiet (siehe Abbildung

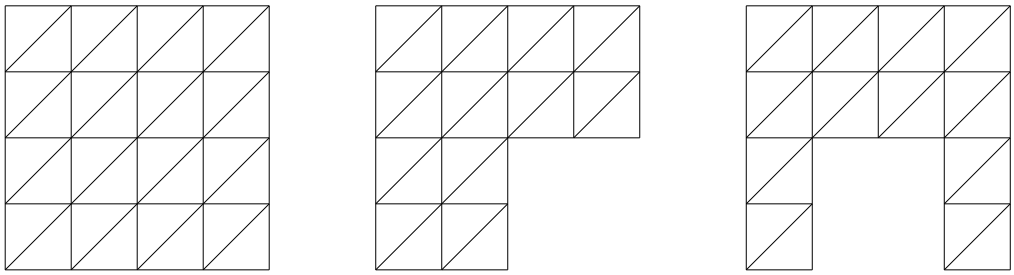
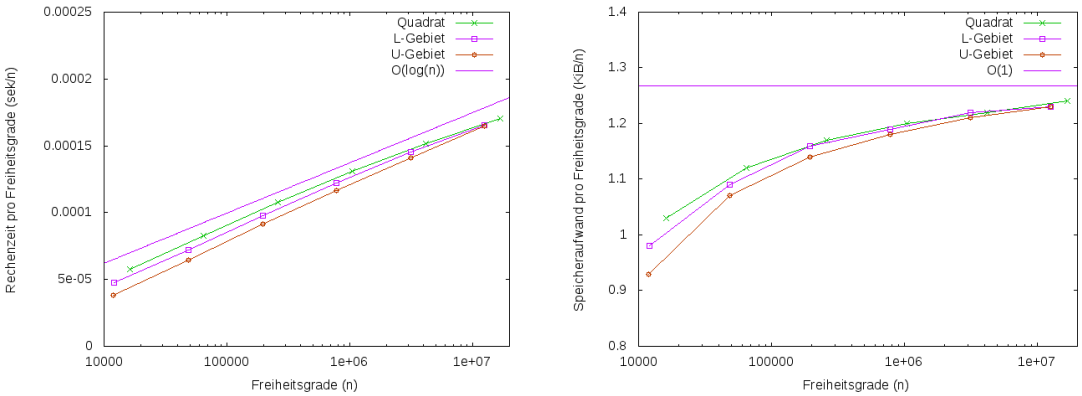


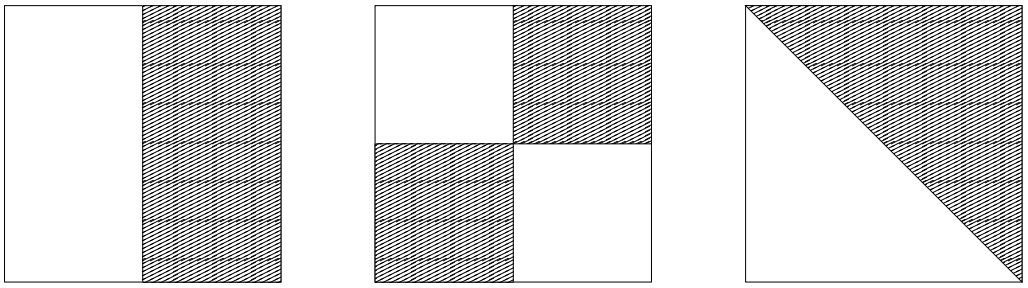
Abbildung 8.1: Beispiele für Gitter für das Einheitsquadrat, L-Gebiet und U-Gebiet.



Quadrat		L-Gebiet		U-Gebiet	
n	Zeit	n	Zeit	n	Zeit
16129	0,9	12033	0,6	11969	0,5
65025	5,4	48641	3,5	48513	3,1
261121	28,1	195585	19,2	195329	17,9
1046529	137,3	784385	95,8	783873	91,6
4190209	634,1	3141633	457,3	3140609	442,1
16769025	2859,1	12574721	2077,5	12572673	2068,9

Abbildung 8.2: In der linken Grafik ist der Zeitaufwand und in der rechten Grafik ist der Speicheraufwand für die Berechnungen der Cholesky-Zerlegung jeweils pro Freiheitsgrad eingezeichnet. In der Tabelle sind für die drei untersuchten Gebiete jeweils die Anzahl der Freiheitsgrade n und die Zeit für die Berechnung der approximativen Cholesky-Zerlegung angegeben.

8.1).Wir verwenden eine Approximation der Cholesky-Zerlegung im $\mathcal{H}^2$-Matrix-Format als Vorkonditionierer für die Rechnungen auf den drei Geometrien. Für das Aufstellen der $\mathcal{H}^2$-Matrizen haben wir die auf Gebietszerlegung basierende Clusterstrategie aus [28] verwendet, wobei wir eine maximale Blattgröße von 32 und $\eta = 4$ gewählt haben. Wir verwenden die Fehlerschranke $50/n$, um einen Fehler $\|I - (LL^T)^{-1}A\|_S \approx 10^{-2}$ zu erhalten. Zur Fehlerkontrolle verwenden wir den blockrelativen Ansatz der Strategie 3 für die Spektralnorm (siehe Lemma 4.5.5). Das vorkonditionierte cg-Verfahren (siehe [30, Chapter 9]) benötigte für jedes der Probleme für einen normierten zufälligen Startvektor und den Nullvektor als Lösung 3 Schritte, um eine Lösung mit euklidischem Fehler kleiner als 10^{-8} zu berechnen.
Die Ergebnisse dieses Experiments legen nahe, dass der Speicheraufwand pro Freiheitsgrad für den berechnete L-Faktor der Cholesky-Zerlegung für große n gegen eine Konstante verläuft (siehe rechte Grafik in Abbildung 8.2). Dieses deutet daraufhin, dass die lokalen Ränge (zumindest im Durchschnitt) für zunehmende Problemgröße nicht anwächst. Der



(a) $\alpha_{1,c}$ (b) $\alpha_{2,c}$ (c) $\alpha_{3,c}$

Abbildung 8.3: Im weißen Bereich haben die Funktionen den Wert 1 und im staffierten Bereich den Wert c.

Rechenaufwand scheint dementsprechend in $\mathcal{O}(n \log(n))$ zu sein (siehe linke Grafik in Abbildung 8.2). Außerdem verhält sich der Rechen- und Speicheraufwand für die verschiedenen Gebiete ähnlich.
Als Zweites untersuchen wir, wie sich der auf der $\mathcal{H}^2$-Matrix-Cholesky-Zerlegung basierende Vorkonditionierer bei springenden Koeffizienten verhält. Als Geometrie verwenden wir wieder das Quadrat $\Omega = [-1, 1]^2$. Für $c \in \mathbb{R}$ definieren wir die drei Koeffizientenfunktionen

$$\alpha_{1,c} : \Omega \to \mathbb{R}, (x, y) \mapsto \begin{cases} 1 & \text{, falls } x < 0 \\ c & \text{sonst} \end{cases},$$

$$\alpha_{2,c} : \Omega \to \mathbb{R}, (x, y) \mapsto \begin{cases} 1 & \text{, falls } xy < 0 \\ c & \text{sonst} \end{cases} \text{ und}$$

$$\alpha_{3,c} : \Omega \to \mathbb{R}, (x, y) \mapsto \begin{cases} 1 & \text{, falls } x < y \\ c & \text{sonst} \end{cases}.$$

In Abbildung 8.3 sind jeweils die Teile des Quadrats markiert, in denen die Funktion gleich c ist. Für das diskretisierte Problem setzen wir die diskrete Koeffizientenfunktion für jedes Dreieck konstant gleich dem Wert im Mittelpunkt. Für das Aufstellen der $\mathcal{H}^2$-

	Zeit für Cholesky-Zerlegung				$\|I - L^{-T}L^{-1}A\|_S$			
n	$\alpha_{1,1}$	$\alpha_{1,10}$	$\alpha_{1,100}$	$\alpha_{1,1000}$	$\alpha_{1,1}$	$\alpha_{1,10}$	$\alpha_{1,100}$	$\alpha_{1,1000}$
16129	1,3	0,9	1,0	1,0	0,062	0,066	0,027	0,024
65025	5,4	5,5	5,6	5,6	0,069	0,032	0,026	0,026
261121	28,1	28,5	29,1	29,8	0,068	0,047	0,031	0,025
1046529	137,9	140,0	143,1	144,6	0,103	0,051	0,042	0,041
4190209	635,3	649,2	661,1	673,1	0,106	0,051	0,042	0,041

Abbildung 8.4: In den Tabellen sind die Ergebnisse für das Problem mit springenden Koeffizienten für $\alpha_{1,c}$ mit $c \in \{1, 10, 100, 1000\}$ aufgeführt.

	Zeit für LL^T		$\|I - L^{-T}L^{-1}A\|_S$		Schritte		Zeit zum Lösen	
n	$\alpha_{2,1000}$	$\alpha_{3,1000}$	$\alpha_{2,1000}$	$\alpha_{3,1000}$	$\alpha_{2,1000}$	$\alpha_{3,1000}$	$\alpha_{2,1000}$	$\alpha_{3,1000}$
16129	1,0	1,0	0,015	0,177	3	4	0,07	0,09
65025	5,6	5,6	0,016	0,199	3	3	0,32	0,32
261121	29,6	29,5	0,015	0,222	3	3	1,33	1,34
1046529	145,4	144,6	0,026	0,092	3	3	5,50	5,54
4190209	673,0	672,8	0,025	0,067	2	3	15,07	22,47

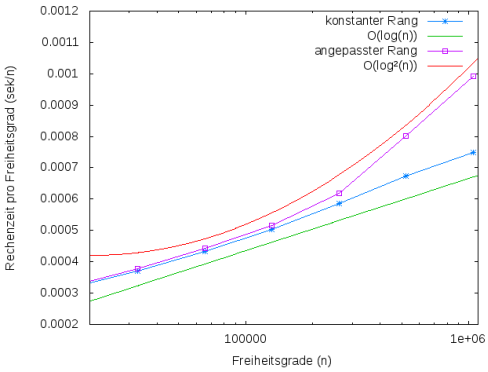
Abbildung 8.5: In den Tabellen sind die Ergebnisse für das Problem mit springenden Koeffizienten für $\alpha_{2,c}$ und $\alpha_{3,c}$ aufgeführt.

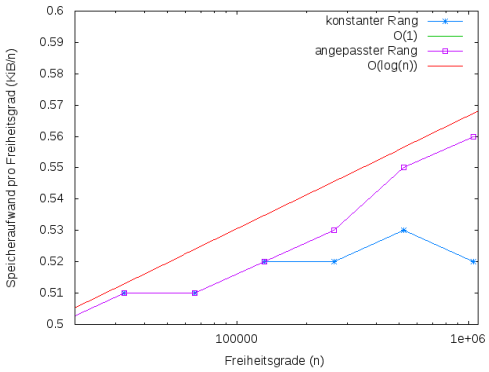
Matrix-Cholesky-Zerlegung verwenden wir die gleichen Techniken und Optionen wie oben. Diesmal verwenden wir einen normierten Zufallsvektor als Lösung und den Nullvektor als Startvektor. Das vorkonditionierte cg-Verfahren lassen wir bis zu einem Fehler von 10^{-8} iterieren.
Als Erstes betrachten wir die erste Koeffizientenfunktion $\alpha_{c,1}$ für $c \in \{1, 10, 100, 1000\}$, wobei $c = 1$ dem ursprünglichen Problem entspricht. In Tabelle 8.4 sehen wir, dass die Rechenzeiten sich zwar etwas erhöhen, aber der Anstieg unter 10% liegt. Dem gegenüber steht eine Verbesserung der Genauigkeit im Term $\|I - L^{-T}L^{-1}A\|_S$. Für alle Größen benötigen wir 3 Iterationsschritte zum Lösen des Gleichungssystems.
Als Zweites präsentieren wir Zahlen für $\alpha_{2,c}$ und $\alpha_{3,c}$. Weil wir für $\alpha_{1,c}$ eine gleichmäßige Entwicklung über das variierte c beobachtet haben, zeigen wir in Tabelle 8.5 nur die Zahlen für $c = 1000$ ($c = 1$ entspricht wieder dem ursprünglichem Problem). Zusätzlich zeigen wir die Anzahl der Iterationsschritte und die Zeiten zum Lösen des Gleichungssystems. Bei den Rechenzeiten sehen wir, dass der Aufwand sich für alle drei Funktionen sehr ähnlich verhält. Für den Fehler $\|I - L^{-T}L^{-1}A\|_S$ beobachten wir, dass dieser für $\alpha_{3,1000}$ insbesondere für die kleineren Dimensionen größer ist als für $\alpha_{2,1000}$. Allerdings wird der Fehler für $\alpha_{3,1000}$ mit zunehmender Anzahl von Freiheitsgraden kleiner und die Anzahl der Iterationsschritte unterscheidet sich maximal um eins. Es zeigt sich somit, dass die approximative $\mathcal{H}^2$-Matrix-Cholesky-Zerlegung sich auch für springende Koeffizienten gut eignet und dabei die Parameter des Verfahrens nicht angepasst werden müssen.

8.2 BEM auf Kurven

Für das zweite Experiment betrachten wir das zweidimensionale Einfachschichtpotential, also den Integraloperator

$$\mathcal{G}[u](x) \;=\; -\frac{1}{2\pi}\int_\Gamma \log\|x-y\|_2 u(y)dy.$$





	konstanter Rang		Rang in $\mathcal{O}(\log(n))$	
n	KiB	sek	KiB	sek
65536	33736	28,5	33717	29,1
131072	68205	65,9	68188	67,6
262144	137619	153,3	139192	162,9
524288	275817	353,6	285868	420,0
1048576	544885	785,7	589196	1040,3

Abbildung 8.6: Die Grafiken zeigen den Rechenaufwand für die Berechnung der Cholesky-Zerlegung und den Speicheraufwand für den L-Faktor jeweils pro Freiheitsgrad wieder. Die Tabelle gibt die Zahlen nochmals absolut wieder. Dabei sind Zahlen für konstanten Rang und für angepassten Rang angegeben.

Dieses betrachten wir auf dem Kreis um den Ursprung $\Gamma := \partial B(0, 1/3)$, den wir durch ein regelmäßiges n-Eck approximieren. Zur Diskretisierung verwenden wir stückweise konstante Basisfunktionen. Die entstehende Matrix V approximieren wir mit Interpolation der Kernfunktion und Rekompression durch eine $\mathcal{H}^2$-Matrix $\widetilde{V}$ und berechnen von dieser Approximation die näherungsweise Cholesky-Zerlegung $\widetilde{L}\widetilde{L}^T$. Dabei verwenden wir die Strategie 1 mit relativem Fehler bezüglich der Spektralnorm, weil sich in Experimenten gezeigt hat, dass diese zu besseren Ergebnissen als die Strategie 3 führt. Die Genauigkeit für das Kürzen wählen wir in diesem Fall ein $\epsilon \approx 1/n$.
Damit sich der Fehler der $\mathcal{H}^2$-Matrix-Approximation wie der Diskretisierungfehler verhält, müssen wir für die $\mathcal{H}^2$-Matrix-Approximation den Rang in $\mathcal{O}(\log(n))$ wählen. Um die Entwicklung in n nochmals getrennt aufzuzeigen, geben wir Rechnungen für $\mathcal{H}^2$-Matrix-Approximationen mit konstantem lokalen Rang und angepasstem lokalen Rang an (für die Berechnung der Cholesky-Zerlegung ist der lokale Rang jeweils adaptiv gewählt).
Bei der Berechnung für $\widetilde{V}$ mit konstantem Rang scheint auch der berechnete L-Faktor linearen Speicheraufwand zu haben. Dies deutet auf im Wesentlichen konstante lokale Ränge hin. Dementsprechend ist der Aufwand für die Cholesky-Zerlegung in $\mathcal{O}(n \log(n))$. Bei dem angepassten lokalen Rang hat der L-Faktor einen Speicheraufwand in $\mathcal{O}(n \log(n))$.

Nach den Aufwandsabschätzungen würden wir für einen lokalen Rang in $\mathcal{O}(\log(n))$ einen Rechenaufwand in $\mathcal{O}(n \log^3(n))$ erwarten. Die Ergebnisse in diesem Fall scheinen sich allerdings eher wie $\mathcal{O}(n \log^2(n))$ zu verhalten.

8.3 BEM auf Oberflächen

Als letztes Experiment betrachten wir die Laplace-Gleichung auf einem Gebiet $\Omega \subset \mathbb{R}^3$. Wir wollen die Neumann-Werte auf dem Rand $\Gamma = \partial\Omega$ aus den Dirichlet-Daten berechnen. Dazu verfolgen wir den Ansatz aus [12, Sections 10.1 und 10.2]. Als Beispiel betrachten wir die Einheitskugel, die wir per Bisektion einer Doppelpyramide approximieren. Dadurch erhalten wir ein beschränktes Polyedergebiet Ω, dessen Rand Γ aus Dreiecken besteht. Damit erhalten wir eine Triangulation $\{\tau_i \,|\, i \in \mathcal{I}\}$ des Randes Γ und definieren für alle $i \in \mathcal{I}$ die stückweise konstanten Basisfunktionen

$$\phi_i : \Gamma \to \mathbb{R}, \quad x \;\mapsto\; \phi(x) \;:=\; \begin{cases} 1 & \text{, falls } x \in \tau_i \\ 0 & \text{sonst} \end{cases}$$

für die Neumann-Daten. Für die Dirichlet-Daten definieren wir die stückweise linearen Basisfunktionen in jedem Knoten $(x_j)_{j \in \mathcal{J}}$ der Triangulation, die für alle $j \in \mathcal{J}$ durch

$$\begin{aligned} \psi_j(x_k) \;&=\; \begin{cases} 1 & \text{, falls } j = k \\ 0 & \text{sonst} \end{cases} && \text{für alle } k \in \mathcal{J} \text{ und} \\ \psi_{j|\tau_j} &\text{ ist linear} && \text{für alle } i \in \mathcal{I} \end{aligned}$$

charakterisiert sind. Dann besteht das mit BEM diskretisierte Problem aus dem linearen Gleichungssystem

$$Vx \;=\; \left(K + \frac{1}{2}M\right) y \tag{8.1}$$

mit dem diskretisierten Einfachschichtpotential $V \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$,

$$V_{i,j} \;=\; \int_\Gamma \phi_i(x) \int_\Gamma \frac{1}{4\pi\|x-y\|_2} \phi_j(y) dy\, dx \quad \text{für alle } i,j \in \mathcal{I},$$

dem diskretisierten Doppelschichtpotential $K \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$,

$$K_{i,j} \;=\; \int_\Gamma \phi_i(x) \int_\Gamma \frac{\langle x-y, \eta(y)\rangle}{4\pi\|x-y\|_2^3} \psi_j(y) dy\, dx \quad \text{für alle } i \in \mathcal{I},\, j \in \mathcal{J},$$

wobei η die Normale von Γ sei, und der Massematrix $M \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$. Der Koeffizientenvektor y repräsentiert die Galerkin-Approximation der Dirichlet-Daten $u_{D,h}$ auf unserer diskretisierten Oberfläche und ist durch

$$u_{D,h} \;=\; \sum_{j \in \mathcal{J}} y_j \psi_j$$

	Speicheraufwand				Rechenzeit			
n	$\widetilde{V}$	$\widetilde{W}$	L_1	L_2	$\widetilde{V}$	$\widetilde{W}$	$L_1L_1^T$	$L_2L_2^T$
2048	9	7	5	5	6,9	15,7	45,3	17,6
8192	49	40	26	25	35,4	103,8	262,9	96,4
32768	277	214	128	126	197,0	564,5	1233,0	458,4
131072	1228	961	589	584	834,2	2512,0	7227,0	2587,0
524288	6180	4984	2847	2773	3781,0	11340,0	32800,0	11220,0

Abbildung 8.7: Die Tabelle gibt die Rechenzeit für das Aufstellen der Systemmatrizen $\widetilde{V}$ und $\widetilde{W}$ sowie der approximierten Cholesky-Zerlegung an, wobei das Update aus Kapitel 5 (L_1) mit der 2. Variante aus Kapitel 7 (L_2) verglichen wird.

gegeben. Das diskrete Einfachschichtpotential V ist symmetrisch und positiv definit. Also existiert eine eindeutig Lösung $x \in \mathbb{R}^I$ von (8.1). Diese repräsentiert die Galerkin-Approximation

$$u_{N,h} \;=\; \sum_{i \in \mathcal{I}} x_i \phi_i$$

der Neumann-Daten auf unserer diskretisierten Oberfläche. Damit ist unser diskretes Problem beschrieben. Die Matrizen approximieren wir mit Hilfe der in [13] beschriebenen Techniken durch $\mathcal{H}^2$-Matrizen $\widetilde{V}$, $\widetilde{W}$ und $\widetilde{M}$ mit an den Diskretisierungsfehler angepassten Genauigkeiten. Danach berechnen wir die approximierte Cholesky-Zerlegung und lösen das Gleichungssystem mit Hilfe des vorkonditionierten cg-Verfahrens.
In der Tabelle 8.7 zeigen wir Zahlen für den Speicheraufwand und den Rechenaufwand des Aufbaus der $\mathcal{H}^2$-Matrix-Approximation des Einfachschichtpotential $\widetilde{V}$ und des Doppelschichtpotentials $\widetilde{W}$. Außerdem sind zum Vergleich die Ergebnisse für die Cholesky-Zerlegung mit dem ersten Ansatz des Niedrigrangupdates aus Kapitel 5 ($L_1L_1^T$) und für die Cholesky-Zerlegung mit dem Niedrigrangupdate in der 2. Variante aus Kapitel 7 ($L_2L_2^T$) aufgeführt. Wie schon bei dem zweidimensionalen BEM-Problem verwenden wir die Strategie 1 (siehe Lemma 4.5.2). Der Fehler der Cholesky-Zerlegung gemessen im Term $\|I - \widetilde{L}^{-T}\widetilde{L}^{-1}\widetilde{V}\|_S$ per Vektoriteration ist ≈ 0.1. Dazu verwenden wir eine Genauigkeit beim Kürzen von $h_{\max}/100 \in \mathcal{O}(1/\sqrt{n})$ mit der maximalen Höhe der Dreiecke $h_{\max}$.
Für die L-Faktoren der Cholesky-Zerlegung erwarten wir etwas mehr als die Hälfte des Speichers einer normalen $\mathcal{H}^2$-Matrix mit entsprechenden Rängen, weil wir für die $\mathcal{H}^2$-Matrizen mit unterer Dreiecksgestalt die Blöcke über der Diagonalen nicht speichern (siehe Bemerkung 6.4.3). Für größere Problemdimensionen sehen wir, dass der Speicherbedarf für die L-Faktoren weniger als die Hälfte des Speicheraufwands der Matrix $\widetilde{V}$. Weil wir für beide Matrizen die gleiche Blockstruktur verwenden, deutet dies auf niedrigere Ränge in den L-Faktoren hin.
Beim Vergleich der Speicheraufwände der L-Faktoren für die beiden Varianten des Niedrigrangupdates fällt auf, dass die Variante aus Kapitel 5 etwas mehr Speicher benötigt. Dies deutet auf entsprechend höhere Ränge hin. Vermutlich liegt die Ursache hierfür

	$u_{D,1}$			$u_{D,2}$	$u_{D,3}$
n	Schritte	Zeit	L^2-Fehler	L^2-Fehler	L^2-Fehler
2048	3	0,07	$1{,}25 \cdot 10^{-1}$	$2{,}36 \cdot 10^{-2}$	$1{,}84 \cdot 10^{-1}$
8192	3	0,31	$6{,}22 \cdot 10^{-2}$	$1{,}16 \cdot 10^{-2}$	$8{,}96 \cdot 10^{-2}$
32768	3	1,19	$3{,}09 \cdot 10^{-2}$	$5{,}63 \cdot 10^{-3}$	$4{,}43 \cdot 10^{-2}$
131072	3	5,11	$1{,}55 \cdot 10^{-2}$	$2{,}88 \cdot 10^{-3}$	$2{,}21 \cdot 10^{-2}$
524288	3	19,50	$7{,}76 \cdot 10^{-3}$	$1{,}49 \cdot 10^{-3}$	$1{,}11 \cdot 10^{-2}$
2097152	3	92,43	$3{,}90 \cdot 10^{-3}$	$7{,}57 \cdot 10^{-4}$	$5{,}54 \cdot 10^{-3}$

Abbildung 8.8: Die Tabelle zeigt die Rechenzeit zum Lösen per vorkonditioniertem cg-Verfahren und den L^2-Fehler der berechneten Lösung für die drei verschiedenen Dirichlet-Daten.

bei der Verwendung der projizierten Gewichte, die dafür sorgen, dass Teile approximiert werden, die bereits durch Projektionen weggefallen sind.
Des Weiteren zeigt die Tabelle 8.7, dass die Arithmetik mit dem neuen Update aus Kapitel 7 um einen Faktor zwischen $2,5$ und 3 schneller ist als die Variante aus Kapitel 5. Dabei sind die Genauigkeiten der berechneten Cholesky-Zerlegungen fast gleich. (In den Experimenten zeigt sich, dass die neue Variante etwas genauer ist. Dies entspricht den theoretischen Aussagen aus Bemerkung 7.2.27.) Außerdem zeigt sich, dass die Cholesky-Zerlegung mit dem Updates in der 2. Variante ungefähr die gleiche Rechenzeit benötigt wie das Aufstellen der Matrix $\widetilde{W}$, aber circa die dreifache Zeit für das Aufstellen von $\widetilde{V}$.
Als Nächstes wenden wir uns dem Lösen des Gleichungssystems zu. Dabei verwenden wir drei verschiedene Dirichlet-Daten:

$$\begin{aligned} u_{D,1} : \Gamma \to \mathbb{R}, \quad x &\mapsto x_1^2 - \frac{1}{2}x_2^2 - \frac{1}{2}x_3^2, \\ u_{D,2} : \Gamma \to \mathbb{R}, \quad x &\mapsto \frac{1}{\|x - \begin{pmatrix} \frac{6}{5} & \frac{6}{5} & \frac{6}{5} \end{pmatrix}^T \|_2}, \\ u_{D,3} : \Gamma \to \mathbb{R}, \quad x &\mapsto \frac{1}{\|x - \begin{pmatrix} 1 & \frac{1}{4} & 1 \end{pmatrix}^T \|_2}. \end{aligned}$$

Für diese drei Dirichlet-Daten verwenden wir zum Lösen das vorkonditionierten cg-Verfahren, wobei wir die Cholesky-Zerlegung der 2. Variante mit obigen Parametern als Vorkonditionierer nutzen. In Tabelle 8.8 geben wir den L^2-Fehler der berechneten Neumann-Daten im Vergleich zu den richtigen Neumann-Daten an. Weil für alle drei Dirichlet-Daten die Anzahl der Iterationen und somit Rechenzeiten für die Berechnung der Lösungen gleich sind, zeigen wir in Abbildung 8.8 diese beiden Größen nur für $u_{D,1}$.
Bei der Betrachtung der Fehler wird deutlich, dass sich der Fehler von einer Verfeinerungsstufe zur nächsten halbiert. Da sich die Gitterweite $h_{\max}$ auch halbiert und sich der Diskretisierungsfehler in $\mathcal{O}(h_{\max})$ befindet erhalten wir also die optimale Ordnung der Fehlerreduktion. Außerdem zeigt sich, dass die Anzahl der Iterationen für alle Gitterweiten konstant bleibt. Dies erreichen wir dadurch, dass wir den Fehler der Cholesky-Zerlegung bei $\|I - \widetilde{L}^{-T}\widetilde{L}^{-1}\widetilde{V}\|_S \approx 0.1$ halten.

	Rechenaufwand			Speicheraufwand		
n	$h_{\max}/100$	$h_{\max}/10$	$h_{\max}$	$h_{\max}/100$	$h_{\max}/10$	$h_{\max}$
2048	17,6	11,5	6,6	5,2	4,1	3,6
8192	96,4	57,4	32,9	25,0	19,5	17,4
32768	458,4	251,2	146,1	126,1	103,0	95,0
131072	2587,0	1576,0	976,1	583,5	480,5	436,0
524288	11220,0	6750,0	4051,0	2773,0	2423,0	2234,0
2097152	61480,0	38410,0	19810,0	12490,0	11030,0	10100,0

Abbildung 8.9: Die Tabelle vergleicht den Aufwand für die Cholesky-Zerlegung zu verschiedenen Genauigkeiten.

Des Weiteren wird deutlich, dass das Lösen des Gleichungssystems deutlich weniger Rechenzeit benötigt als die Berechnung des Vorkonditionierers. Je nach Anzahl der rechten Seiten, für die wir das Gleichungssystem lösen wollen, kann es von Interesse sein, die Berechnung des Vorkonditionierers auf Kosten der Qualität zu beschleunigen. Dazu betrachten wir, wie sich der Vorkonditionierer für geringere Genauigkeiten verhält und führen die Experimente für die Fehlertoleranzen $h_{\max}/10$ und $h_{\max}$ nochmals durch. Die Ergebnisse sind in Abbildung 8.9 zusammengefasst, wobei wir wieder das Niedrigrangupdate in der 2. Variante und die Strategie 1 verwenden.
Wir können durch eine Verwendung der Cholesky-Zerlegung mit der geringeren Genauigkeit $h_{\max}$ die Rechenzeit auf circa ein Drittel des Aufwands für die Genauigkeit $h_{\max}/100$ reduzieren. Damit besitzt die Berechnung des Vorkonditionierers für die Matrix $\widetilde{V}$ ungefähr den gleichen Aufwand wie das Aufstellen der selbigen.
Als Nächstes betrachten wir, wie sich die geringere Genauigkeit auf das Verhalten des vorkonditionierten cg-Verfahrens auswirkt. Dazu betrachten wir die Fehlergröße $\|I - \widetilde{L}^{-T}\widetilde{L}^{-1}\widetilde{V}\|_S$, die Anzahl der Iterationsschritte und die Zeit zum Lösen des Gleichungssystems. Bei den letzten beiden Größen bilden wir die Summe über die drei Dirichlet-Daten. Die L^2-Fehler geben wir in Abbildung 8.10 nicht an, weil sich diese wie in Abbildung 8.8 verhalten.
Das Lösen des Gleichungssystems ist bei der Cholesky-Zerlegung mit Genauigkeit $h_{\max}$

	$\|I - \widetilde{L}^{-T}\widetilde{L}^{-1}\widetilde{V}\|_S$			Iterationsschritte			Zeit zum Lösen		
n	ϵ_{100}	ϵ_{10}	ϵ_1	ϵ_{100}	ϵ_{10}	ϵ_1	ϵ_{100}	ϵ_{10}	ϵ_1
2048	0,047	0,257	2,556	9	11	23	0,21	0,21	0,43
8192	0,074	0,512	5,082	9	14	30	1,24	1,54	2,41
32768	0,088	0,619	7,268	9	14	31	3,57	4,77	10,41
131072	0,100	1,019	8,443	9	12	23	15,33	17,21	32,39
524288	0,097	1,016	7,981	9	12	22	58,34	70,68	127,86
2097152	0,111	1,125	7,872	9	11	21	277,14	309,19	617,90

Abbildung 8.10: Die Tabelle vergleicht das Verhalten des Vorkonditionierers für die Genauigkeiten $\epsilon_{100} := h_{\max}/100$, $\epsilon_{10} := h_{\max}/10$ und $\epsilon_1 := h_{\max}$.

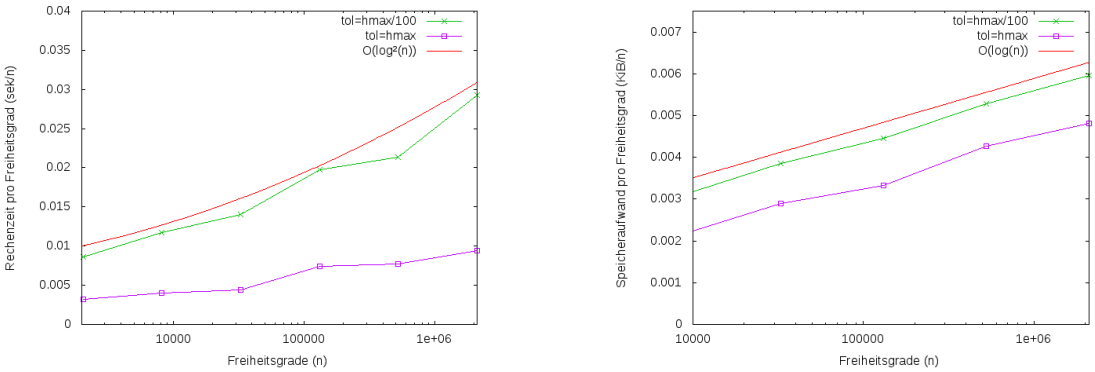


Abbildung 8.11: Die Grafiken zeigen den Rechenaufwand für die Cholesky-Zerlegung und den Speicheraufwand für den berechneten L-Faktor zu den Genauigkeiten $h_{\max}/100$ und $h_{\max}$.

zwar mehr als doppelt so langsam als die für $h_{\max}/100$. Weil für die Genauigkeit $h_{\max}$ die Zeit zum Lösen des Gleichungssystems für eine rechte Seite nur ein bis zwei Prozent der Rechenzeit für das Aufstellen des Vorkonditionierers benötigt, lohnt sich die Verwendung der geringeren Genauigkeit in vielen Fällen trotzdem.
Wir können also mit unserem Verfahren das Gleichungssystem effizient lösen, wobei zum Beispiel für $n = 524288$ die Zeit für das Lösen für drei rechte Seiten inklusive des Aufstellens des Vorkonditionierers weniger als ein Drittel der Zeit des Aufstellens der Problemmatrizen $\widetilde{V}$ und $\widetilde{W}$ benötigt.
Um die Komplexität der beiden Größen zu diskutieren, visualisieren wir die Rechenzeit und den Speicheraufwand. In Abbildung 8.11 sehen wir, dass sich der Speicheraufwand wie $\mathcal{O}(n \log(n))$ verhält, was einen gemittelten lokalen Rang in $\mathcal{O}(\log(n))$ andeutet. Allerdings scheint sich der Rechenaufwand eher wie $\mathcal{O}(n \log^2(n))$ zu verhalten. Dies ist wieder ein Logarithmus weniger, als wir nach den Aufwandsabschätzungen erwarten.
Um zu zeigen, dass die vorgestellten Verfahren nicht nur für simple Geometrie verwendet werden können, präsentieren wir ein Experiment für eine Kurbelwelle (siehe Abbildung 8.12). Dabei verwenden wir wieder das Verfahren aus [13] zum Aufstellen der $\mathcal{H}^2$-Matrizen

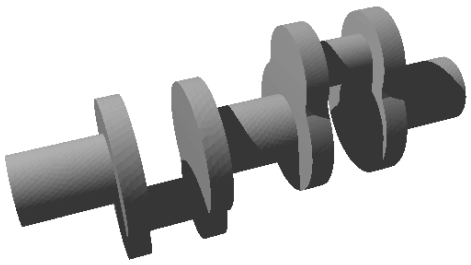


Abbildung 8.12: Auf dem Bild ist die verwendete Kurbelwelle dargestellt.

	Rechenaufwand			Speicheraufwand		
n	$\widetilde{V}$	$\widetilde{W}$	LL^T	$\widetilde{V}$	$\widetilde{W}$	L
1748	144	346	9	13	10	5
6992	1396	3299	46	106	88	31
27968	9992	26710	255	631	658	147
111872	56880	171000	1272	2966	3384	666

Abbildung 8.13: In der Tabelle sind der Rechen- und Speicheraufwand für das Aufstellen von $\widetilde{V}$ und $\widetilde{W}$ sowie die Cholesky-Zerlegung aufgeführt.

$\widehat{V}$ und $\widehat{W}$ und die approximierte Cholesky-Zerlegung als Vorkonditionierer. Zur Berechnung der Cholesky-Zerlegung verwenden wir die Strategie 1 mit der Genauigkeit $h_{\max}/10$. Wie wir in Tabelle 8.13 sehen, ist das Aufstellen der Problemmatrizen mit den von uns verwendeten Verfahren für diese Geometrie verglichen mit der Sphäre (siehe Tabelle 8.7) wesentlich aufwendiger. Wenn wir hingegen den Rechenaufwand der Cholesky-Zerlegung für $n = 111872$ in Tabelle 8.13 mit der für $n = 131072$ und Fehlerschranke $h_{\max}$ in Tabelle 8.9 vergleichen, stellen wir fest, dass der Rechenaufwand pro Freiheitsgrad auf der Kurbelwelle sogar etwas geringer ist. Das Aufstellen des Vorkonditionierers für die Kurbelwelle kann in weniger als einem Prozent der Zeit berechnet werden, die für das Aufstellen der Problemmatrizen $\widetilde{V}$ und $\widetilde{W}$ benötigt wird.
Das vorkonditionierte cg-Verfahren verhält sich für die Kurbelwelle ähnlich wie für die Sphäre (siehe Tabelle 8.14). Dabei betrachten wir die gleichen Dirichlet-Daten wie für die Sphäre. Das Lösen des Gleichungssystems für die drei rechten Seiten benötigt weniger als 2% der Zeit für das Aufstellen der Cholesky-Zerlegung. Nach dem Aufstellen des Vorkonditionierers kann das Gleichungssystem für verschiedene rechte Seiten effizient berechnet werden.

	Schritte	Zeit	$u_{D,1}$	$u_{D,2}$	$u_{D,3}$
1748	9	0,20	$8{,}81 \cdot 10^{-2}$	$9{,}48 \cdot 10^{-3}$	$3{,}64 \cdot 10^{-2}$
6992	10	1,01	$3{,}13 \cdot 10^{-2}$	$4{,}16 \cdot 10^{-3}$	$1{,}62 \cdot 10^{-2}$
27968	10	5,07	$1{,}21 \cdot 10^{-2}$	$1{,}99 \cdot 10^{-3}$	$7{,}63 \cdot 10^{-3}$
111872	11	20,85	$4{,}68 \cdot 10^{-3}$	$1{,}25 \cdot 10^{-3}$	$3{,}88 \cdot 10^{-3}$

Abbildung 8.14: Die Tabelle zeigt die Iterationsschritte und Rechenzeit summiert über alle drei rechte Seiten und den L^2-Fehler der berechneten Lösung für die drei verschiedenen Dirichlet-Daten.

8.4 Fazit

Wir haben in dieser Arbeit eine Möglichkeit zur Berechnung der approximativen $\mathcal{H}^2$-Matrix-Arithmetik basierend auf lokalen Niedrigrangupdates vorgestellt. Bei dem vorgestellten

Ansatz ist keinerlei Vorwissen zu den Clusterbasen des Ergebnisses notwendig, da diese adaptiv während der Berechnungen aufgestellt werden. Die dabei verwendeten Niedrigrangupdates basieren auf der Rekompression aus [9], die wir in Kapitel 4 zusammengefasst haben. Entsprechend den Fehlerabschätzungen für die Rekompression haben wir in Kapitel 5 gezeigt, dass auch der Fehler für die Niedrigrangupdates kontrolliert werden kann. Dabei können wir den Fehler global, global relativ oder blockrelativ beschränken. Außerdem entsteht der Fehler nicht in der ganzen Matrix, sondern durch den lokalen Ansatz des Updates nur in einem Teil der Matrix.
Dieser lokale Ansatz des Updates führt auch dazu, dass wir den Aufwand linear in der Anzahl der Zeilen und der Spalten beschränken können. Dies ermöglicht uns das Matrix-Produkt für $\mathcal{H}^2$-Matrizen unter gewissen Voraussetzungen mit einem Aufwand in $\mathcal{O}(n \log(n) k^2)$ zu berechnen. Für die Berechnung der approximativen Inversen und Cholesky-Zerlegung haben wir den Aufwand durch den der Matrix-Matrix-Multiplikation beschränkt (siehe Kapitel 6). Damit haben wir in den Aufwandsschranken für die Arithmetik einen Faktor $\log(n)$ gegenüber der $\mathcal{H}$-Matrix-Arithmetik aus [26] eingespart. Dies entspricht der Einsparung beim Speicheraufwand und Rechenaufwand für die Matrix-Vektor-Multiplikation. Zusätzlich haben wir in Kapitel 7 eine Variante des $\mathcal{H}^2$-Matrix-Produkts vorgestellt, das den Aufwand für dreidimensionale BEM-Problem deutlich reduziert. Dabei bleibt der Fehler wie für die Variante in Kapitel 5 kontrollierbar.
Die numerischen Ergebnisse in diesem Kapitel bestätigen die theoretischen Ergebnisse. Wir können mit Hilfe der approximierten Cholesky-Zerlegung einen Vorkonditionierer aufstellen, der es ermöglicht, die Gleichungssysteme effizient mit dem vorkonditionierten cg-Verfahren zu lösen. Insbesondere kann nach der Berechnung des Vorkonditionierers das Gleichungssystem für verschiedene rechte Seiten gelöst werden.
Für die betrachteten FEM-Probleme hat sich der Vorkonditionierer als robust bezüglich der Geometrien und springender Koeffizienten erwiesen. Dabei müssen die Parameter für die verschiedenen Probleme nicht angepasst werden (abgesehen von dem Faktor $1/n$ in der Genauigkeit). Dies ist für praktische Anwendung wichtig, weil das Verfahren keine aufwendigen Anpassungen von Parametern benötigt.
Bei den dreidimensionalen BEM-Problemen hat sich die Arithmetik mit dem Update aus Kapitel 7 als effizient erwiesen. Auch bei diesem Experimenten hat sich gezeigt, dass die Parameter für die Kugel auch für die Kurbelwelle verwendet werden können. Außerdem scheint das Aufstellen des Problems für kompliziertere Geometrien wie die Kurbelwelle deutlich zeitintensiver zu werden, die Berechnung der Cholesky-Zerlegung allerdings nicht. Dies führt dazu, dass bei der Kurbelwelle das Lösen des Problems im Vergleich zum Aufstellen nur einen Bruchteil der Zeit benötigt.
Für die zukünftige Arbeit ist eine Überprüfung der Effizienz der Implementierung vorzunehmen. Zum Beispiel kann versucht werden, die Dreiecksgestalt der Matrizen, die aus QR-Zerlegungen entstehen, konsequent auszunutzen. Auch ein effizienter Umgang mit dem Speicher, insbesondere die Vermeidung von Speicheranforderungen, kann die Algorithmen beschleunigen. Des Weiteren könnte es sich als effizient erweisen, kleine Blöcke zeitweise als vollbesetzte Matrizen zu behandeln, weil diese auf aktuellen Rechnerarchitekturen sehr effizient behandelt werden können. Nach den Berechnungen können diese dann wieder in das $\mathcal{H}^2$-Matrix-Format konvertiert werden.

Außerdem sollte noch untersucht werden, in wieweit sich der Ansatz aus Abschnitt 7 mit den Mengen $\mathcal{M}(\check{t}, \check{r})$ eignet, den Aufwand auch für andere arithmetischen Operationen als das $\mathcal{H}^2$Matrix-Produkt zu verringern. Des Weiteren können wir bei dem $\mathcal{H}^2$-Matrix-Produkt dafür sorgen, dass nur noch in den Blättern Updates durchgeführt werden, indem wir die in Nicht-Blatt-Blöcken berechneten Niedrigrangmatrizen auf die Kinder aufteilen. Weil dafür mehr Additionen von Niedrigrangmatrizen entstehen, muss geprüft werden, ob dies die Algorithmen wesentlich beschleunigt.
In diesem Zusammenhang stellt sich auch die Frage, ob teilweise statt der Niedrigrangmatrizen im Rang-k-Format semi-uniforme Matrizen verwendet werden sollten, bei denen einer der Faktoren A oder B aus der entsprechenden Clusterbasis V bzw. W besteht. Diese werden im Ansatz der Arithmetik aus [12, Chapter 8] verwendet. Dabei könnte insbesondere der Aufwand für das Aufstellen der Niedrigrangmatrizen reduziert werden. Auch mit diesen Ansätzen lässt sich der Aufwand voraussichtlich nicht auf lineare Komplexität reduzieren, weil in fast jedem Block eine semi-uniforme Matrix in linearer Komplexität berechnet werden muss.
Bei der in Kapitel 7 vorgestellten $\mathcal{H}^2$-Matrix-Multiplikation verwenden wir zwei Techniken, um den Einfluss der c_{sp}-Konstanten auf den Aufwand zu reduzieren. Einerseits verwenden wir die 2. Variante des Niedrigrangupdates aus Kapitel 7, bei dem die Blockzeilen und -spalten nicht durchlaufen werden. Andererseits nutzen wir die Hilfsmengen $\mathcal{M}(t, r)$, um die Zielmatrix nur einmal rekursiv zu durchlaufen und die Anzahl der Updates zu reduzieren. Es bleibt die Frage, wie groß der jeweilige Einfluss auf die Rechenzeit der beiden Techniken ist. In diesem Zusammenhang wäre eine Umsetzung der 1. Variante aus Kapitel 7 mit Hilfe der Mengen $\mathcal{M}(\check{t}, \check{r})$ von Interesse.
Eine weitere Herausforderung für die Weiterentwicklung der vorgestellten Algorithmen ist die mittlerweile in vielen Bereichen der Numerik besonders wichtige Frage der parallelen Umsetzung. Für $\mathcal{H}$-Matrizen gibt es z.B. für die Berechnung der LR-Zerlegung für FEM-Matrizen Arbeiten (siehe [27, 38]). Weil die Darstellung der Niedrigrangmatrizen in den zulässigen Blättern bei den $\mathcal{H}^2$-Matrizen im Gegensatz zu den $\mathcal{H}$-Matrizen von anderen Teilen der Matrix abhängt, kann dieser Ansatz nicht direkt auf die $\mathcal{H}^2$-Arithmetik mit lokalen Niedrigrangupdates übertragen werden. Beim ersten Ansatz aus Kapitel 5 sollte es genügen, dass Updates nicht gleichzeitig für die gleichen Zeilen oder die gleichen Spalten berechnet werden, da die verschiedenen Updates keine nachträgliche Anpassung anderer Matrixteile benötigen. Es ist schwer vorherzusagen, ob sich auch die 2. Variante aus Kapitel 7 für einen parallelen Ansatz eignet.

Literaturverzeichnis

[1] M. Bebendorf. Hierarchical LU decomposition based preconditioners for BEM. *Computing*, 74:225–247, 2005.

[2] M. Bebendorf. Why finite element discretizations can be factored by triangular hierarchical matrices. *SIAM J. of Numer. Anal.*, 45(4):1472–1494, 2007.

[3] M. Bebendorf and W. Hackbusch. A existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients. *Numer. Math.*, 95, no. 1:1–28, 2003.

[4] P. Benner, S. Börm, T. Mach, and K. Reimer. Computing the eigenvalues of symmetric $\mathcal{H}^2$-matrices by slicing the spectrum. *Computing and Visualization in Science*, 16(6):271–282, 2013.

[5] G. Beylkin, R. Coifman, and V. Rokhlin. The fast wavelet transform and numerical algorithms. *Comm. Pure and Appl. Math.*, 44:141–183, 1991.

[6] S. Bosch. *Lineare Algebra*. Springer Verlag, Berlin, Heidelberg, 5. edition, 2014.

[7] S. Börm. Approximation of integral operators by $\mathcal{H}^2$-matrices with adaptive bases. *Computing*, 74(3):249–271, 2005.

[8] S. Börm. $\mathcal{H}^2$-matrix arithmetics in linear complexity. *Computing*, 77:1–28, 2006.

[9] S. Börm. Data-sparse approximation of non-local operators by $\mathcal{H}^2$-matrices. *Linear Algebra Appl.*, 422:380–403, 2007.

[10] S. Börm. Construction of data-sparse $\mathcal{H}^2$-matrices by hierarchical compression. 31(3):1820–1839, 2009.

[11] S. Börm. Approximation of solution operators of elliptic partial differential equations by $\mathcal{H}$- and $\mathcal{H}^2$-matrices. *Numer. Math.*, 115(2):165–193, 2010.

[12] S. Börm. *Efficient Numerical Methods for Non-local Operators: $\mathcal{H}^2$-Matrix Compression, Algorithms and Analysis*, volume 14 of *EMS Tracts in Mathematics*. EMS, 2010.

[13] S. Börm and S. Christophersen. Approximation of integral operators by Green quadrature and nested cross approximation. Technical report, Department of Computer Science, University of Kiel, 2014. `http://arxiv.org/abs/1404.2234`.

[14] S. Börm and L. Grasedyck. Hlib package. `http://www.hlib.org`.

[15] S. Börm and L. Grasedyck. Low-rank approximation of integral operators by interpolation. *Computing*, 72:325–332, 2004.

[16] S. Börm and W. Hackbusch. Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices. *Computing*, 69:1–35, 2002.

[17] S. Börm and K. Reimer. Efficient arithmetic operations for rank-structured matrices based on hierarchical low-rank updates. *Computing and Visualization in Science*, 16(6):247–258, 2013.

[18] S. Chandrasekaran, M. Gu, and T. Pals. A fast ULV decomposition solver for hierarchically semiseparable representations. *SIAM J. Matrix Anal. Appl.*, 28(3):603–622, 2006.

[19] W. Dahmen and R. Schneider. Wavelets on manifolds I: Construction and domain decomposition. *SIAM J. Math. Anal.*, 31:184–230, 1999.

[20] M. Faustmann, J. M. Melenk, and D. Praetorius. Existence of $\mathcal{H}$-matrix approximants to the inverse of BEM matrices: the simple-layer operator. *to appear in Math. Comp.*, 2014. preprint: `http://www.asc.tuwien.ac.at/preprint/2013/asc37x2013.pdf`.

[21] M. Faustmann, J. M. Melenk, and D. Praetorius. $\mathcal{H}$-matrix approximability of the inverses of FEM matrices. *Numer. Math.*, 2015. `http://dx.doi.org/10.1007/s00211-015-0706-9`.

[22] C.F. Van Loan G.H. Golub. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996.

[23] K. Giebermann. Multilevel approximation of boundary integral operators. *Computing*, 67:183–207, 2001.

[24] Z. Gimbutas and V. Rokhlin. A generalized fast multipole method for nonoscillatory kernels. *SIAM J. Sci. Comput.*, 24(3):796–817, 2002.

[25] L. Grasedyck. *Theorie und Anwendungen Hierarchischer Matrizen*. Dissertation, Mathematisch-Naturwissenschaftliche Fakultät der Christian-Albrechts-Universität zu Kiel, July 2001.

[26] L. Grasedyck and W. Hackbusch. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing*, 70:295–334, 2003.

[27] L. Grasedyck, R. Kriemann, and S. Le Borne. Parallel black box $\mathcal{H}$-LU preconditioning for elliptic boundary value problems. *Comput. Visual. Sci.*, 11:273–291, 2008.

[28] L. Grasedyck, R. Kriemann, and S. Le Borne. Domain decomposition based $\mathcal{H}$-LU preconditioning. *Numer. Math.*, 112, no. 4:565–600, 2009.

[29] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comp. Phys.*, 73:325–348, 1987.

[30] W. Hackbusch. *Iterative solution of large sparse systems of equations*, volume 95 of *Applied mathematical science*. Springer, New York, 1994.

[31] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. part I: Introduction to $\mathcal{H}$-matrices. *Computing*, 62:89–108, 1999.

[32] W. Hackbusch. *Hierarchische Matrizen — Algorithmen und Analysis*. Springer, Berlin, Heidelberg, 2009.

[33] W. Hackbusch and B. N. Khoromskij. A sparse $\mathcal{H}$-matrix arithmetic. part II: Application to multi-dimensional problems. *Computing*, 64:21–47, 2000.

[34] W. Hackbusch, B. N. Khoromskij, and S. A. Sauter. On $\mathcal{H}^2$-matrices. *Lectures on applied mathematics*, pages 9–29, 2000.

[35] W. Hackbusch and Z. P. Nowak. On the fast matrix multiplication in the boundary element method by panel clustering. *Numer. Math.*, 54:463–491, 1989.

[36] H. Harbrecht and R. Schneider. Wavelet Galerkin schemes for boundary integral equations – Implementation and quadrature. *SIAM J. Sci. Comput.*, 27:1347–1370, 2006.

[37] B. Huppert. *Angewandte Lineare Algebra*. de Gruyter, Berlin, 1990.

[38] R. Kriemann. $\mathcal{H}$-LU factorization on many-core systems. *Comput. Visual. Sci.*, 16:105–117, 2014.

[39] Scientific Computing Group of Kiel University. H2lib package. `http://www.h2lib.org`.

[40] V. Rokhlin. Rapid solution of integral equations of classical potential theory. *J. Comp. Phys.*, 60:187–207, 1985.

[41] S. A. Sauter. Variable order panel clustering. *Computing*, 64:223–261, 2000.

[42] G. W. Stewart. *Matrix Algorithms, Volume I: Basic Decompositions*. SIAM, Philadelphia, PA, 1998.

[43] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li. Superfast multifrontal method for large structured linear systems of equations. *SIAM J. Matrix Anal. Appl.*, 31(3):1382–1411, 2009.

Erklärung

Hiermit erkläre ich,

- dass die vorgelegte Arbeit, abgesehen von der Beratung durch meinen Betreuer Steffen Börm, nach Form und Inhalt meine eigene ist,
- dass Teile dieser Arbeit bereits in den folgenden Veröffentlichungen enthalten sind:
  (i) *Efficient arithmetic operations for rank-structured matrices based on hierarchical low-rank updates* zusammen mit Steffen Börm, Computing and Visualization in Science, zur Veröffentlichung angenommen
  (ii) *Computing the eigenvalues of symmetrix $\mathcal{H}^2$-matrices by slicing the spectrum* zusammen mit Peter Benner, Steffen Börm und Thomas Mach, zur Veröffentlichung angenommen

  Diese Artikel weisen einen Eigenanteil gemäß dem Schreiben meines Betreuers vor.
- und dass diese Arbeit unter Einhaltung der Regeln guter wissenschaftlicher Praxis der Deutschen Forschungsgemeinschaft entstanden ist.

Kiel, 26. Februar 2015