

Accessing NMDB data using NEST and pandas

Christian T. Steigies ¹, Nicolas Fuller ²

Correspondence

1 Extraterrestrial Physics, Institute of Experimental and Applied Physics, Kiel University, Germany,
steigies@physik.uni-kiel.de

2 Observatoire de Paris, LESIA, Université PSL, CNRS, Sorbonne Université, Université Paris Cité, France,
nicolas.fuller@obspm.fr

Keywords

cosmic rays; neutron monitor; database; python; data science

Abstract


The Neutron Monitor database (NMDB) was created by teams from 12 different countries in 2008. Data from neutron monitors worldwide is pooled and made available, for many stations in real-time. The NMDB Event Search Tool (NEST) started as a quick-look interface to the data in NMDB, but by now has become the main interface to all NMDB data. NEST does not only enable you to plot data from one or several NMDB stations in a very customizable way, it also allows you to retrieve the data in ASCII format for further processing or creating your own plots. Downloading data can be scripted using 'wget' or 'curl' as documented in '3 ways 2 use NEST'. Here we are presenting python functions to read in data from one or several stations directly from NEST into a pandas dataframe. Once your data is in a 'dataframe', you can easily sort, modify or plot data with python.

1. Introduction

The main goal of the Neutron Monitor Database (NMDB) is to make neutron monitor data easily accessible, in a common format. This includes real-time data (for example available at <http://rt.nmdb.eu>, last accessed July 4, 2023) as well as historical data. The data in NMDB is typically available at 1-minute resolution, but also in 1-hour resolution to study long-term variations. For intermediate resolutions, the database can provide averaged data as well.

While NMDB stores all data in an SQL database, only few users have (and need) direct access to the database: Neutron Monitor stations sending real-time data to NMDB, and applications that work with real-time data, for example GLE Alert tools. Regular users do not want to work with SQL commands to access the data, so a graphical user interface, the NMDB event search tool NEST (shown in [Fig. 1](#)), has been developed that allows users to interactively select stations and time ranges, datatypes and additional data like sunspot numbers and provides the results as configurable plot, or in ASCII format to download for further processing.

3 ways 2 use NEST





? Quick Plots
New | Last Data
GLE 73
GLE 72


conditions & information to use data

top


[jul 2022] In memoriam: Pr. Lev I. Dorman (1929-2022)

[jul 2022] Warning: All lines of the ascii files.








Last 3 days



Basic Plot Example



GLE&FD Plot Examples



Kp Ri p+ Plot Examples

? Stations

(When selecting multiple stations, note that only one variable can be plotted)

<input type="checkbox"/> AATA	<input type="checkbox"/> AATB	<input type="checkbox"/> APTY	<input type="checkbox"/> ARNM	<input type="checkbox"/> ATHN
<input type="checkbox"/> BKSJ	<input type="checkbox"/> CALG	<input type="checkbox"/> CALM	<input type="checkbox"/> DJON	<input type="checkbox"/> DOMB
<input type="checkbox"/> DOMC	<input type="checkbox"/> DRBS	<input type="checkbox"/> ESOI	<input type="checkbox"/> FSMT	<input type="checkbox"/> HRMS
<input type="checkbox"/> INVK	<input type="checkbox"/> IRK2	<input type="checkbox"/> IRK3	<input type="checkbox"/> IRKT	<input type="checkbox"/> JBGO
<input type="checkbox"/> JUNG	<input type="checkbox"/> JUNG1	<input type="checkbox"/> KERG	<input type="checkbox"/> KIEL	<input type="checkbox"/> KIEL2
<input type="checkbox"/> LMKS	<input type="checkbox"/> MCRL	<input type="checkbox"/> MGDN	<input type="checkbox"/> MOSC	<input type="checkbox"/> MRNY
<input type="checkbox"/> MWSN	<input type="checkbox"/> MXCO	<input type="checkbox"/> NAIN	<input type="checkbox"/> NANM	<input type="checkbox"/> NEU3
<input type="checkbox"/> NEWK	<input type="checkbox"/> NRLK	<input type="checkbox"/> NVBK	<input type="checkbox"/> OULU	<input type="checkbox"/> PSNM
<input type="checkbox"/> PTFM	<input type="checkbox"/> PWNK	<input type="checkbox"/> ROME	<input type="checkbox"/> SANB	<input type="checkbox"/> SNAE
<input type="checkbox"/> SOPB	<input type="checkbox"/> SOPO	<input type="checkbox"/> TERA	<input type="checkbox"/> THUL	<input type="checkbox"/> TSMB
<input type="checkbox"/> TXBY	<input type="checkbox"/> YKTK			

Online* stations in green

Closed Stations*
 Bonner Spheres*
 Smart Selection

? Date Selection (UTC)

Last 1 days
 hours mins

From 21 Oct 2022 0h 0mn

To 21 Oct 2022 23h 59mn

GLE number/date 73 (2021-10-28) detailed list

FD number/date 53 (2011-02-18)

? Resolution

Time resolution: best

Force**

Smooth window: 0

? Data variables

Pressure & efficiency corr.
 Pressure corrected
 Uncorrected
 Pressure

? Scale

Relative scale
 Counts/s*
 Log scale
(* mbar for pressure)

? Output

Plot
 Ascii
 Plot & ascii

Submit
Reset

read.nmdb.eu via TCP/IP

* online means some data (realtime or not) have been uploaded during the last 15mn

** force parameter: Read note 1 and note 3 of the help file

Contact: questions@nmdb.eu

? NMDB tables

? Overplot main

? Overplot Ri

? Proton / Kp plots

? Env. & meta data

? Scaling Options

? Event Options

? Ascii Options

? Style Options

NEST is provided by




Fig. 1: NEST web interface to NMDB data.

While this is excellent for occasional browsing the data, it is a bit cumbersome for serious data analysis. To enable automated downloads of NMDB data the NEST manual ‘3 ways 2 use NEST’ (Fuller 2022) provides also instructions on retrieving data in ASCII format without having to fill in a webform. A query string can be created and the data can be accessed with a tool like ‘wget’ or ‘curl’. When the data has been downloaded, it can be further processed by the user.

The python programming language has become the de-facto standard for Data Science (VanderPlas 2016) at most universities, even some proprietary software can use functions written in python (Origin 2022). So instead of downloading NMDB data with hand-created queries, or with simple shell scripts, it makes sense to download the data directly in python, where the data will be analyzed. Here we present some python functions to easily access user selected data from NMDB, and some examples on how to work with this data using the pandas library, a popular tool among data scientists (The pandas development team 2022).

2. NEST queries

NMDB data can be downloaded with a special URL, as explained in the NEST manual. This URL defines the Neutron Monitor station, the datatable, the datatype, and the timerange of the requested data. One query can either combine data for several stations (and one datatype), or one station with several datatypes (like uncorrected, pressure, and corrected data). We have created separate python functions for these use cases: `nest.multi` for querying multiple stations, and `nest.single` for querying a single station. Both functions use the same parameters:

- `station`: station shortname as used in NMDB (four or five letters, list of strings for multi, string for single)
- `table`: data table used in NMDB with the following abbreviations:
 - `e`: `corr_for_efficiency`
 - `c`: `corr_for_pressure`
 - `u`: `uncorrected`
 - `p`: `pressure_mbar`
- `data`: `ori`, `revori` (revised and original data merged), or `1h`
- `start`: datetime of the start of the requested data
- `end`: datetime of the end of the requested data

The functions include no error checking for valid station names or datetimes or the amount of data that will be returned by NEST. The data will be returned in the highest available resolution (1-min for `ori` and `revori`), or, for longer time periods, averaged to longer durations by NEST. For details, see the NEST manual (Fuller 2022).

2.1 `nmdb.multi`

The `nmdb.multi` function allows to access one datatype for multiple stations. The required arguments are a list with the station names, the datatable which can be one of `revori`, `ori`, or `1h`, the datatype, which can be one of `e`, `c`, `u`, `p`, and the datetimes for the start and the end of the requested data. See Fig. 2 for an example using the `nest.multi` function.

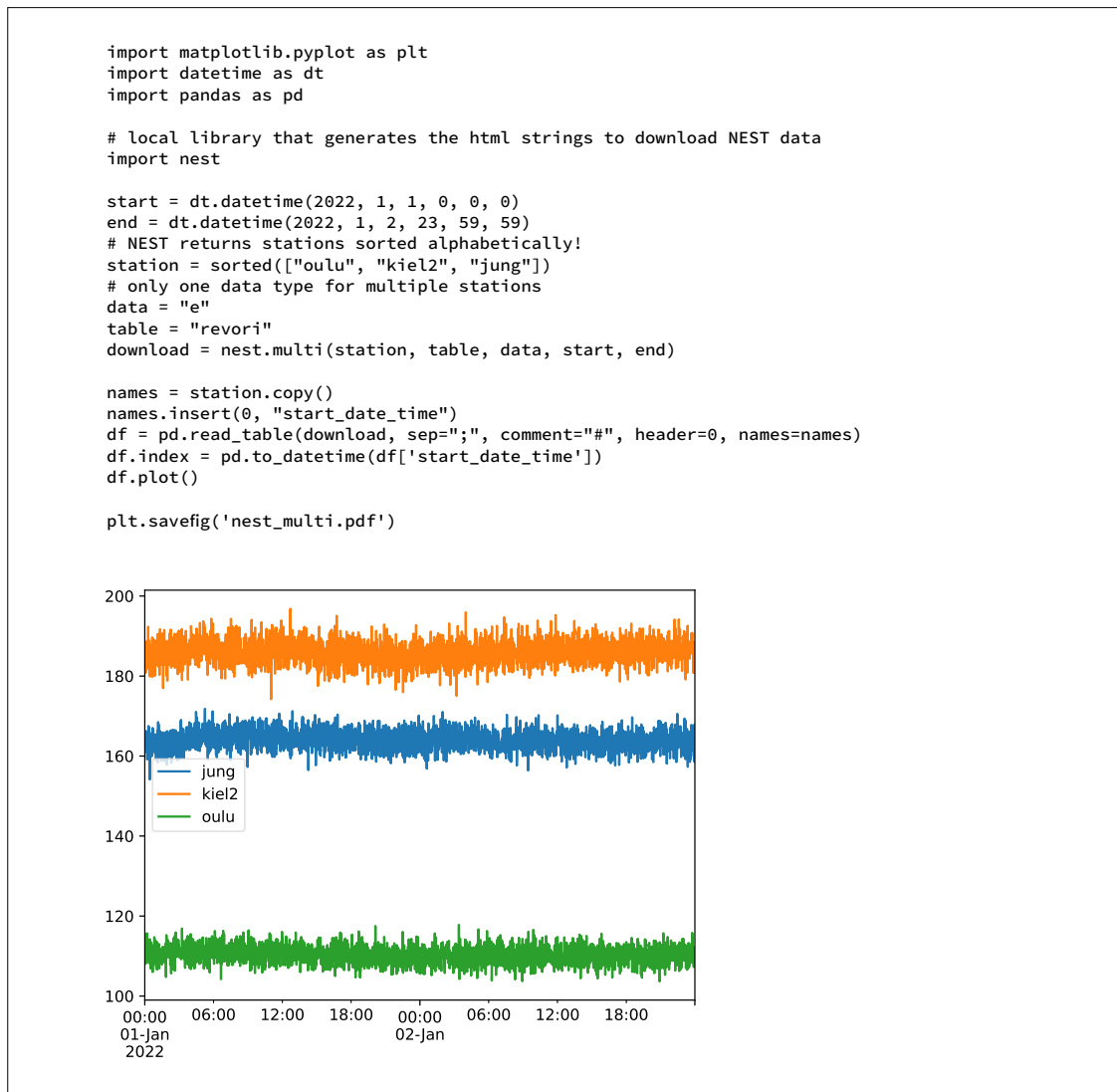


Fig. 2: Example using the `nest.multi` function.

2.2 `nmdb.single`

The `nmdb.single` function allows to access several datatypes for a single station. The required arguments are a string with the station name, the datatable, which can be one of `revori`, `ori`, or `1h`, a list with the datatype (which can be one of `e`, `c`, `u`, `p`), and the datetimes for the start and the end of the requested data. See Fig. 3 for an example using the `nest.single` function.

2.3 `nmdb.header`

NEST returns meta data together with the requested data as comments, however this is currently not stored in the pandas dataframe. To access the meta data a separate `nmdb.header` function is provided. This function takes the url created by either `nmdb.multi` or `nmdb.single` and extracts the meta data into a string. The meta data contains information such as if the data has been averaged or revised and how to cite the data. See Fig. 4 for an example using the `nest.header` function.

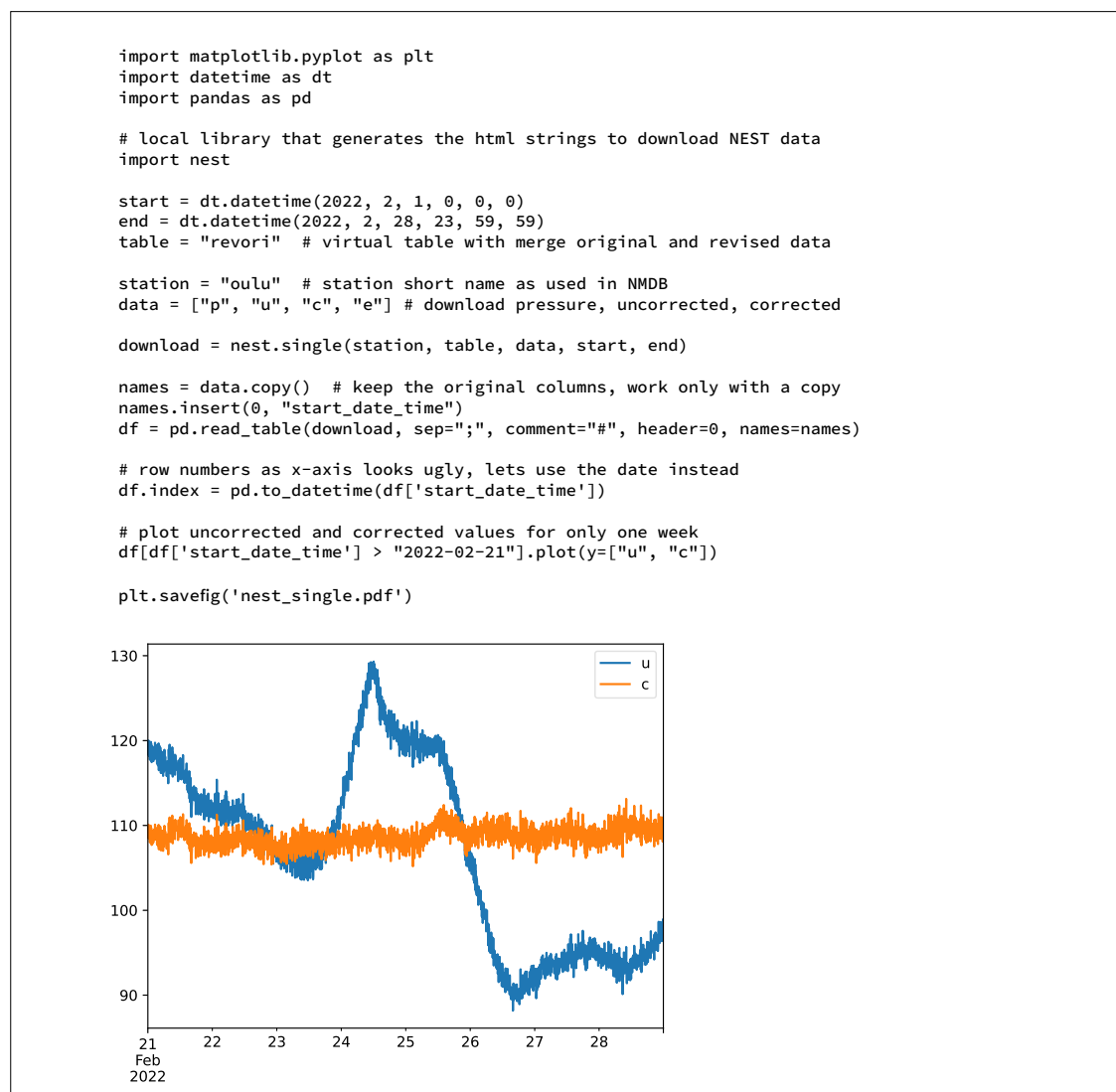


Fig. 3: Example using the nest.single function.

3. Installation

The python sourcecode for the nest module, the examples for the multi, single and header function, as well as a jupyter notebook with further examples are available on the NMDB website at <https://www.nmdb.eu/software/python/nest> (last accessed July 4, 2023). Updates to the nest module as well as extensions, for example an option to select the resolution of the data, will be made available on that webpage.

To install the nest module, simply copy the module nest.py and any example scripts you want to use into your working directory. If you execute the example scripts in an Integrated Development Environment (IDE) like spyder, or use the jupyter notebook, you can interactively work with the dataframes and immediately display the plots without adding further commands that would be necessary to execute in a stand-alone python script.

```

In [1]: import datetime as dt
import pandas as pd

# local library that generates the html strings to download NEST data
import nest

start = dt.datetime(2022, 2, 1, 0, 0, 0)
end = dt.datetime(2022, 2, 28, 23, 59, 59)
table = "revori" # virtual table with merge original and revised data

station = "oulu" # station short name as used in NMDB
# download pressure, uncorrected, corrected (for pressure) and efficiency corrected data
data = ["p", "u", "c", "e"]

download = nest.single(station, table, data, start, end)
header = nest.header(download)
print(header)

-----QUERY RESULTS SUMMARY-----
STATION: OULU
START TIME: 2022-02-01 00:00:00 UTC
END TIME: 2022-02-28 23:55:00 UTC
NMDB TABLE: revised original
REV EQ. ORI: Yes, revised data are identical to the original data for this period
DATA TYPE: pressure_mbar(RPRESS)
OTHER DATA: uncorrected(RUNCORR)corr_for_pressure(RCORR_P) corr_for_efficiency(RCORR_E)
AVERAGING: Yes / 5 min
ORIGINAL RES: 1 min

-----
Timestamps always correspond to the beginning of the time interval
-----

Data retrieved via NMDB are the property of the individual data providers. These data are free for non commercial use to within the restriction imposed by the providers. If you use such data for your research or applications, please acknowledge the origin by a sentence like 'We acknowledge the NMDB Database (www.nmdb.eu) founded under the European Union's FP7 programme (contract no. 213 007), and the PIs of individual neutron monitors at: Oulu (Sodankyla Geophysical Observatory of the University of Oulu, Finland)

```

Fig. 4: Example using the nest.header function.

4. Conclusion

We have presented an open source python module that enables easy access to NMDB data via the NEST webinterface. Users can access all NMDB data simply by specifying stations, time-periods and datatypes to create a URL to the data and read in the data using this URL into a pandas dataframe. With the data in a dataframe users can plot the data or further analyse it using all the powers provided by pandas and the entire python ecosystem.

References

- Fuller, N. 2023, 3 ways 2 use NEST, <https://www.nmdb.eu/nest/help.php#howto> (last accessed January 16, 2023)
- NMDB 2008, the Neutron Monitor Database, <https://nmdb.eu> (last accessed January 16, 2023)
- Origin 2022, <https://www.originlab.com/doc/python/Run-Python-in-Origin> (last accessed January 16, 2023)
- The pandas development team 2022, pandas-dev/pandas: Pandas (v1.5.2), <https://doi.org/10.5281/zenodo.3509134>
- VanderPlas, J. 2016. The Python Data Science Handbook. O'Reilly.

Open Access

This paper is published under the Creative Commons Attribution 4.0 International license (<https://creativecommons.org/licenses/by/4.0/>). Please note that individual, appropriately marked parts of the paper may be excluded from the license mentioned or may be subject to other copyright conditions. If such third party material is not under the Creative Commons license, any copying, editing or public reproduction is only permitted with the prior consent of the respective copyright owner or on the basis of relevant legal authorization regulations.