nwbindexer

Release 0.1.0

Apr 22, 2020

Contents:

CHAPTER 1

About nwbindexer

nwbindexer is a python package that contains two tools for searching within *Neurodata Without Borders* files (abbreviation **NWB**, described at: https://nwb-schema.readthedocs.io/en/latest/) that are stored using the HDF5 (https://portal.hdfgroup.org/display/HDF5/HDF5) format. The two tools are:

- *nwbindexer* - builds an SQLite database (also called the 'index') containing metadata in a collection of NWB files and and allows searching the metadata in the database.
- *search_nwb.py* - searches within one or more NWB files directly (without building an index).

The source repository for nwbindexer is: https://github.com/jeffteeters/nwbindexer. A related third tool is the **NWB Query Engine**. It is at: https://github.com/jezekp/NwbQueryEngine. The two tools in nwbindexer use a *query format* similar to the one used in the NWB Query Engine. A web page allowing example searches using all three tools is at: http://eeg.kiv.zcu.cz:8080/nwb-query-engine-web/.

CHAPTER 2

Requirements

1. Python 3.7 (Tested with Anaconda Python 3.7; might also work with Python > 3.7. However, it will *not* work with Python < 3.7).
2. SQLite3. (Will normally be already installed).
3. pip. (Should normally be already installed).
4. pytest (only required for running tests). Can be installed using command:

   $ `pip install pytest`

CHAPTER 3

Installation

3.1 Install release from PyPI

The Python Package Index (PyPI) is a repository of software for the Python programming language. To install or update nwbindexer from PyPI run:

```
$ pip install -U nwbindexer
```

This will automatically install the following dependencies as well as *nwbindexer*:

- *parsimonious*
- *h5py*
- *numpy*

3.1.1 Testing the installation (optional)

To test the installation, *pytest* must be installed. It can be installed using:

```
$ pip install pytest
```

Once *pytest* is installed, the nwbindexer installation can be tested by running:

```
$ pytest --pyargs nwbindexer
```

Output should indicate all all tests (5) passed.

3.2 Install from Git repository

First clone the repository and cd into the created directory:

```
$ git clone https://github.com/jeffteeters/nwbindexer.git
$ cd nwbindexer
```

3.2.1 Test local files, not yet installed (optional)

To test the locally cloned files before installing, first packages *parsimonious*, *h5py* and *pytest* must be installed. They can be installed using:

```
$ pip install parsimonious
$ pip install h5py
$ pip install pytest
```

Then, the tests can be run when inside the top-level nwbindexer directory created from the clone (which contains file *setup.py*), by entering *pytest* with no arguments:

```
$ pytest
```

Output should indicate all tests (5) passed.

3.2.2 Completing the installation

To complete the installation, when directly inside the directory created by the clone (containing file *setup.py*) either enter the following for a normal installation (not used for development):

```
$ pip install .
```

OR enter the following to create an editable install which is recommended for development (include the *-e* option):

```
$ pip install -e .
```

3.2.3 Test the installation (optional)

To test the installation (separately from the cloned files downloaded), cd to a directory that does not contain the cloned nwbindexer directory, then type:

```
$ pytest --pyargs nwbindexer
```

Output should indicate all all tests (5) passed.

CHAPTER 4

Query Format

Queries are specified using the following format (BNF Grammar)::

```
query ::= subquery ( andor subquery )*
subquery ::= parent ':' ( <rhs> | '(' <rhs> ')' )
<rhs> ::= ( <child_list> <expression> | <child_list> | <expression> )
<child_list> ::= <child> ( [ ',' ] <child> )* [ ',' ]
<expression> ::= expression andor expression | '(' expression ')'
                 | child relop constant
                 | child 'LIKE' string | child
relop ::= '==' | '<=' | '<' | '>=' | '>' | '!='
constant ::= string | number
andor ::= '&' | '|'
```

In the grammar: square brackets *[ ]* indicate optional contents, *( )** indicates zero or more, *(x | y)* indicates *x* or *y* and:

<parent> is a path to an HDF5 group or dataset. The path can contain asterisk (*) characters which match zero or more characters (e.g. "*" functions as a wildcard).

<child> is the name of an HDF5 attribute or dataset within the parent.

<string> is a string constant enclosed in single or double quotes (with a backslash used to escape quotes). Any string constant used with LIKE must have wildcards ("%" or "_") explicitly included (if no wildcards are included, the query does an exact match).

<number> is a numeric constant.

Some example queries and a description of what they do is given below. (It may be necessary to scroll to the right to see the descriptions).

Query	Description
/general/subject: (species == “Mus musculus”)	Selects all files with the specified species
/general:(virus)	Selects all records with a virus dataset
/general:(virus LIKE “%infectionLocation: M2%”)	Selects all datasets virus with infectionLocation: M2
*:(neurodata_type == “RoiResponseSeries”)	Select all TimeSeries containing Calcium imaging data
*/data: (unit == “unknown”)	Selects all datasetes data which unit is unknown
/epochs/: (start_time > 500 & start_time < 550 & tags LIKE “%HitL%” & tags LIKE “%Lick-Early%”)	Select all epochs with the matching start_time and tags
/general/subject: (subject_id == “anm00210863”) & */epochs/*: (start_time > 500 & start_time < 550 & tags LIKE “%Lick-Early%”)	Select files with the specified subject_id and epochs
/units: id, location == “CA3” & quality > 0.8	Select unit id where location is CA3 and quality > 0.8

CHAPTER 5

nwbindexer usage

Searching using nwbindexer requires two steps: 1. build the index (a SQLite database) and 2. query the index.

5.1 Build the index

The index is built by the *build_index.py* program which can be run by entering either:

$ `build_nwbindex` <*nwb_directory*> [<*index_path*>]

or

$ `python -m nwbindexer.build_index` <*nwb_directory*> [<*index_path*>]

The first form (*build_nwbindex*) uses a command-line utility installed by the nwbindexer package. The second runs the command by specifying the python module directly. If no arguments are entered, the usage information is displayed. The arguments are:

<nwb_directory> Name of directory to scan for NWB files (extension ".nwb")

<index_path> Path to index file. If nothing is specified, uses *nwb_index.db* in the current directory. If only a directory is specified, uses *nwb_index.db* in the specified directory.

The command scans all the nwb files in <*nwb_directory*> (and subdirectories) and saves the information about small datasets and attributes in the index file (a SQLite database) specified by <*index_path*>. The default name of the index file is *nwb_index.db*. The program can be run multiple times with a different <*nwb_directory*> and the same <*index_path*> to add information about additional NWB files to the specified index.

5.2 Query the index

Once the index file is built, queries can be run by entering either:

$ `query_nwbindex` <*index_path*> [<*query*>]

or

$ `python -m nwbindexer.query_index.py` <*index_path*> [<*query*>]

Where:

<index_path> Path to SQLite database file or a directory or '-' for the default database (nwb_index.db). If is path to a directory, then uses default database (nwb_index.db) in that directory.

<query> Query to execute (optional). If present, must be quoted. If not present, interactive mode is used which allows entering queries interactively.

CHAPTER 6

Example output

The output presented in this sections was generated using the three sample NWB files included in the nwbindexer package in the "test" directory. (The commands were run inside directory *nwbindexer/test* which contains three nwb files).

6.1 Building the index

Command entered:

```
python -m nwbindexer.build_index ./
```

Output is:

```
Creating database 'nwb_index.db'
scanning directory ./
Scanning file 1: ./ecephys_example.nwb
Scanning file 2: ./example_file_path.nwb
Scanning file 3: ./ophys_example.nwb
```

(results in creating file *nwb_index.db*).

6.2 Output from queries

Query entered:

```
python -m nwbindexer.query_index - "general/optophysiology/*:
excitation_lambda == 600.0"
```

Output:

```
Using index_path: 'nwb_index.db'
Opening 'nwb_index.db'
Found 1 matching files:
[   {   'file': './ophys_example.nwb',
        'subqueries': [   [   {   'node': '/general/optophysiology/my_imgpln',
                                  'vind': {'excitation_lambda': 600.0},
                                  'vtbl': {}}]]}]
```

Query:

```
python -m nwbindexer.query_index - "general/extracellular_ephys/tetrode1:
location LIKE '%hippocampus'"
```

Output:

```
Using index_path: 'nwb_index.db'
Opening 'nwb_index.db'
Found 1 matching files:
[   {   'file': './ecephys_example.nwb',
        'subqueries': [   [   {   'node': '/general/extracellular_ephys/tetrode1',
                                  'vind': {   'location': 'somewhere in the '
                                                         'hippocampus'},
                                  'vtbl': {}}]]}]
```

Query:

```
python -m nwbindexer.query_index - "units:  id, location == 'CA3' & quality >
0.8"
```

Output:

```
Using index_path: 'nwb_index.db'
Opening 'nwb_index.db'
Found 1 matching files:
[   {   'file': './example_file_path.nwb',
        'subqueries': [   [   {   'node': '/units',
                                  'vind': {},
                                  'vtbl': {   'child_names': [   'id',
                                                                 'location',
                                                                 'quality'],
                                              'combined': [   {   'id': 2,
                                                                  'location': 'CA3',
                                                                  'quality': 0.85}],
                                              'row_values': [   (   2,
                                                                    'CA3',
                                                                    0.85)]}}]]}]
```

6.3 Format of query output

The output of the *query_index.py* utility (and also the *search_nwb.py* utility described in the next section) is in JSON (https://www.json.org/) with the following structure:

[<*file 1 results*>, <*file 2 results*>, . . .]

Where each <*file N results*> is a JSON object (similar to a python dictionary) with keys *file* and *subqueries*.

The value associate with the *file* key is the full path to the NWB file. The value of the *subqueries* key is an array of subquery results:

[<*subquery 1 result*>, <*subquery 2 result*>, ...]

Each <*subquery N result*> is a list of <*node results*> for that subquery:

[<*node 1 result*>, <*node 2 result*>, ...]

Each <*node N result*> is a dictionary giving information about the parent node (location in the HDF5 / NWB file, and child nodes (groups, attributes or datasets directly within the parent) that are referenced in the subquery. The dictionary has keys:

node The path to the parent node (group or dataset) withing the HDF5 file.

vind Values for 'individual' children of the node, that is, children that are not part of a NWB DynamicTable (described below). The values are provided in a JSON object (Python dictionary) where the keys are the name of each child and the values are the values stored in the child.

vtbl Values for children that are part of a NWB DynamicTable. An NWB DynamicTable is a method used within the NWB format to store tabular data that are aligned along the rows, like a spreadsheet. It is described at: https://nwb-schema.readthedocs.io/en/latest/format.html#sec-dynamictable. The value of *vtbl* is described in the next section.

The value of *vtbl* is a JSON Object (Python dictionary) with keys: *child_names*, *row_values* and *combined*. They have the following meaning:

child_names A tuple listing all of the children. This is equivalent to the header row in a spreadsheet which lists in order, the columns in the spreadsheet.

row_values Contains a list of tuples, each tuple contains aligned values associated with the names in *child_names*. In other words, each tuple has values for one row of the spreadsheet in order of the header *child_names*.

combined Contains a list of JSON Objects (Python dictionaries), where each dictionary has data for one row in the returned results. That is, in each dictionary, the keys are the *child_names* (spreadsheet header column names) and the value for each key is the value of that child in the row. This is another way of representing the data that are in *child_names* and *row_values*.

CHAPTER 7

search_nwb.py usage

The search_nwb.py tool searches directly within NWB files for data matching a query. It does not use an index file (unlike the nwbindexer tool).

The search_nwb tool is run using either:

$ `search_nwb` <*path*> [<*query*>]

or

$ `python -m nwbindexer.search_nwb.py` <*path*> [<*query*>]

The first form (*search_nwb*) uses a command-line utility installed by the nwbindexer package. The second runs the command by specifying the python module directly. If no arguments are entered, the usage information is displayed. The arguments are:

<path> path to an NWB file or a directory containing nwb files.

<query> query to execute (optional). If present, must be quoted.

The <*query*> is described in Section *Query Format*. The output format of *search_nwb.py* is described in *Format of query output*. Is the same as for *query_index.py* described in *nwbindexer usage*.

CHAPTER 8

Multiple subquery examples

Queries can be created by combining multiple subqueries. Some examples along with the output are shown in this section. The datasets (nwb files) used in the examples are:

- The first 16 files in the alm-1 data set at CRCNS.org (http://crcns.org/data-sets/motor-cortex/alm-1), which contains anterior motor cortex recordings from the Svoboda Lab at Janelia Farm. The total size of these data files is (2.2 GB).
- NWB-formatted dataset from Steinmetz et al. Nature 2019. Available at: https://figshare.com/articles/Datasets_from_Steinmetz_et_al_2019_in_NWB_format/11274968 File: Steinmetz2019_Forssmann_2017-11-05.nwb (267.96 MB)

In the sample queries (done with the query_index.py utility) the index file (*nwb_index.db*) is located in the *../../sample_data'* directory). The output would be the same if the *search_nwb.py* utility is used if a directory containing the nwb files was specified as the <*path*> in the *search_nwb* command as described in section *search_nwb.py usage*. The format of the output is as described in section *Format of query output*.

Query:

```
$ python -m nwbindexer.query_index ../../sample_data '/general/subject: (age LIKE "%3␣
↪months 16 days%" & species LIKE "%Mus musculu%") & /:file_create_date LIKE "%2017-04
↪%" & /epochs/* : start_time < 15'
```

Output is:

```
Using index_path: '../../sample_data/nwb_index.db'
Opening '../../sample_data/nwb_index.db'
Found 2 matching files:
[   {   'file': './nwb1/data_structure_ANM210862_20130627.nwb',
        'subqueries': [   [   {   'node': '/general/subject',
                                  'vind': {   'age': '3 months 16 days  weeks',
                                              'species': 'Mus musculus'},
                                  'vtbl': {}}],
                          [   {   'node': '/',
                                  'vind': {   'file_create_date': '2017-04-
↪24T11:32:54.215883'},
```

(continues on next page)

(continued from previous page)

```
                                'vtbl': {}}],
                        [   {   'node': '/epochs/trial_001',
                                'vind': {'start_time': 2.284463},
                                'vtbl': {}}]]},
    {   'file': './nwb1/data_structure_ANM210863_20130627.nwb',
        'subqueries': [   [   {   'node': '/general/subject',
                                  'vind': {   'age': '3 months 16 days  weeks',
                                              'species': 'Mus musculus'},
                                  'vtbl': {}}],
                          [   {   'node': '/',
                                  'vind': {   'file_create_date': '2017-04-
↪24T11:32:54.076284'},
                                  'vtbl': {}}],
                          [   {   'node': '/epochs/trial_001',
                                  'vind': {'start_time': 2.222392},
                                  'vtbl': {}}]]}]
```

Query:

```
python -m nwbindexer.query_index ../../sample_data 'intervals/trials: id, visual_
↪stimulus_time, visual_stimulus_left_contrast == 0.25 & visual_stimulus_right_
↪contrast == 0.25'
```

Output:

```
Using index_path: '../../sample_data/nwb_index.db'
Opening '../../sample_data/nwb_index.db'
Found 1 matching files:
[   {   'file': './steinmentz2019/Steinmetz2019_Forssmann_2017-11-05.nwb',
        'subqueries': [   [   {   'node': '/intervals/trials',
                                  'vind': {},
                                  'vtbl': {   'child_names': [   'id',
                                                                 'visual_stimulus_time
↪',
                                                                 'visual_stimulus_
↪left_contrast',
                                                                 'visual_stimulus_
↪right_contrast'],
                                              'combined': [   {   'id': 95,
                                                                  'visual_stimulus_
↪left_contrast': 0.25,
                                                                  'visual_stimulus_
↪right_contrast': 0.25,
                                                                  'visual_stimulus_
↪time': 468.198},
                                                              {   'id': 150,
                                                                  'visual_stimulus_
↪left_contrast': 0.25,
                                                                  'visual_stimulus_
↪right_contrast': 0.25,
                                                                  'visual_stimulus_
↪time': 697.717}],
                                              'row_values': [   (   95,
                                                                    468.198,
                                                                    0.25,
                                                                    0.25),
                                                                (   150,
```

(continues on next page)

(continued from previous page)

```
                                                                    697.717,
                                                                    0.25,
                                                                    0.25)]}}]]}]
```

CHAPTER 9

speed_check.py

Also included in the package is a program named *speed_check.py*, which can be used to compare the speed of different queries performed using the two query tools in this package (the nwbindexer *query_index.py* and *search_nwb.py*) and also the Java tool (the *NWB Query Engine*, available at: https://github.com/jezekp/NwbQueryEngine).

Running *speed_check.py* without any command-line arguments displays the instructions:

$ `python -m nwbindexer.speed_check`

Output is:

```
Usage: ../nwbindexer/speed_check.py ( i | <ndq> | <query> ) [ <data_dir> [ <java_tool_
↪dir> ] ]
 First parameter required, either:
    'i' - interactive mode (user enters queries interactively).
    <ndq> - an integer that specifies number of times to run default queries.  Times␣
↪for runs are averaged.
            It's good to use a multiple of six so all possible orders of the three␣
↪tools are used.
    <query> - a single query to execute; must be quoted.
 After the first parameter, optionally specify:
    <data_dir> - directory containing NWB files AND index file ('nwb_index.db' built␣
↪by build_index.py)
    <java_tool_dir> - directory containing NWB Query Engine (Java tool)
    If <data_dir> not specified, uses: ../sample_data
    If <java_tool_dir> not specified, uses: /Users/jt/crcns/projects/petr_nwbqe_paper/
↪NwbQueryEngine5
```

The source of the script (file *speed_check.py*) can be edited to change the defaults for <*data_dir*> and <*java_tool_dir*>.

An example run of the tool using the <ndq> option and output is shown below.

$ `time python -m nwbindexer.speed_check 12 ../../sample_data_dec2019/ > speed_check_dec1019_x12.txt`

Terminal output:

```
real  28m59.996s
user  29m52.901s
sys   1m35.288s
```

The terminal output (above) shows the time required to run the command. The output of the command (the speed_check utility) is stored in file *speed_check_dec1019_x12.txt*. The program also creates a bar chart showing the average, minimum and maximum time for each tool on each query.

Partial contents of the output (in file *speed_check_dec1019_x12.txt*) is shown below. A line with three dots (...) indicates lines that were removed from the output to reduce the length. The actual output is over 47,800 lines. The execution times found can vary between runs due to variations in the system operation (threads running, memory usage).

```
# A

------- query A -------
epochs*:(start_time>200 & stop_time<250 | stop_time>4850)

** Starting run 0, NWB Query Engine with: epochs*:(start_time>200 & stop_time<250 |␣
↪stop_time>4850)
dir:/Users/jt/crcns/projects/petr_nwbqe_paper/NwbQueryEngine5
cmd:java -Djava.library.path=src/main/resources/ -jar target/nwbqueryengine-1.0-
↪SNAPSHOT-jar-with-dependencies.jar /Users/jt/crcns/projects/petr_nwbqe_paper/sample_
↪data_dec2019 'epochs*:(start_time>200 & stop_time<250 | stop_time>4850)'
Dataset: epochs/trial_011/start_time, Value: 214.274403, DataStorageName: /Users/jt/
↪crcns/projects/petr_nwbqe_paper/sample_data_dec2019/alm-1/data_structure_ANM210861_
↪20130701.nwb
Dataset: epochs/trial_010/start_time, Value: 202.908281, DataStorageName: /Users/jt/
↪crcns/projects/petr_nwbqe_paper/sample_data_dec2019/alm-1/data_structure_ANM210861_
↪20130701.nwb
...

Time, user=11.6178, sys=0.7059, total=12.3236

** Starting run 0, search_nwb with: epochs*:(start_time>200 & stop_time<250 | stop_
↪time>4850)
dir:/Users/jt/crcns/projects/petr_nwbqe_paper/NwbQueryEngine5
cmd:python -m nwbindexer.search_nwb /Users/jt/crcns/projects/petr_nwbqe_paper/sample_
↪data_dec2019 'epochs*:(start_time>200 & stop_time<250 | stop_time>4850)'
Found 12 matching files:
[   {   'file': '/Users/jt/crcns/projects/petr_nwbqe_paper/sample_data_dec2019/alm-1/
↪data_structure_ANM210861_20130701.nwb',
        'subqueries': [   [   {   'node': '/epochs/trial_010',
                                  'vind': {   'start_time': 202.908281,
                                              'stop_time': 214.274403},
                                  'vtbl': {}},
                              {   'node': '/epochs/trial_011',
                                  'vind': {   'start_time': 214.274403,
                                              'stop_time': 223.430533},
                                  'vtbl': {}},

...

Time, user=17.2793, sys=0.5086, total=17.7879

** Starting run 0, nwbindexer with: epochs*:(start_time>200 & stop_time<250 | stop_
↪time>4850)
```

(continues on next page)

(continued from previous page)

```
dir:/Users/jt/crcns/projects/petr_nwbqe_paper/NwbQueryEngine5
cmd:python -m nwbindexer.query_index /Users/jt/crcns/projects/petr_nwbqe_paper/sample_
↪data_dec2019/nwb_index.db 'epochs*:(start_time>200 & stop_time<250 | stop_time>4850)
↪'
Opening '/Users/jt/crcns/projects/petr_nwbqe_paper/sample_data_dec2019/nwb_index.db'
Found 12 matching files:
[   {   'file': './alm-1/data_structure_ANM210861_20130701.nwb',
        'subqueries': [   [   {   'node': '/epochs/trial_010',
                                  'vind': {   'start_time': 202.908281,
                                              'stop_time': 214.274403},
                                  'vtbl': {}},
                              {   'node': '/epochs/trial_011',
                                  'vind': {   'start_time': 214.274403,
                                              'stop_time': 223.430533},
                                  'vtbl': {}},
...

Time, user=0.4064, sys=0.0699, total=0.4763
# B

------- query B -------
*/data: (unit == "unknown")

** Starting run 0, NWB Query Engine with: */data: (unit == "unknown")
dir:/Users/jt/crcns/projects/petr_nwbqe_paper/NwbQueryEngine5
cmd:java -Djava.library.path=src/main/resources/ -jar target/nwbqueryengine-1.0-
↪SNAPSHOT-jar-with-dependencies.jar /Users/jt/crcns/projects/petr_nwbqe_paper/sample_
↪data_dec2019 '*/data: (unit == "unknown")'
Dataset: acquisition/timeseries/lick_trace/data/unit, Value: unknown,␣
↪DataStorageName: /Users/jt/crcns/projects/petr_nwbqe_paper/sample_data_dec2019/alm-
↪1/data_structure_ANM210861_20130701.nwb
...

Time, user=30.6447, sys=1.8166, total=32.4613

** Starting run 0, search_nwb with: */data: (unit == "unknown")
dir:/Users/jt/crcns/projects/petr_nwbqe_paper/NwbQueryEngine5
cmd:python -m nwbindexer.search_nwb /Users/jt/crcns/projects/petr_nwbqe_paper/sample_
↪data_dec2019 '*/data: (unit == "unknown")'
Found 16 matching files:
[   {   'file': '/Users/jt/crcns/projects/petr_nwbqe_paper/sample_data_dec2019/alm-1/
↪data_structure_ANM210861_20130701.nwb',
        'subqueries': [   [   {   'node': '/acquisition/timeseries/lick_trace/data',
                                  'vind': {'unit': 'unknown'},
                                  'vtbl': {}},
                              {   'node': '/stimulus/presentation/pole_in/data',
                                  'vind': {'unit': 'unknown'},
                                  'vtbl': {}},
                              {   'node': '/stimulus/presentation/pole_out/data',
                                  'vind': {'unit': 'unknown'},
                                  'vtbl': {}}]]},
...

Time, user=42.6341, sys=1.6830, total=44.3171


...
```

(continues on next page)

(continued from previous page)

```
Dataset: general/optophysiology/imaging_plane/excitation_lambda, Value: 910.0,␣
↪DataStorageName: /Users/jt/crcns/projects/petr_nwbqe_paper/sample_data_dec2019/
↪tutorials/domain/brain_observatory.nwb
Dataset: general/optophysiology/my_imgpln/excitation_lambda, Value: 600.0,␣
↪DataStorageName: /Users/jt/crcns/projects/petr_nwbqe_paper/sample_data_dec2019/
↪tutorials/domain/ophys_example.nwb

Time, user=1.5345, sys=0.1322, total=1.6667

Queries in test:
A. epochs*:(start_time>200 & stop_time<250 | stop_time>4850)
B. */data: (unit == "unknown")
C. general/subject: (subject_id == "anm00210863") & epochs/*: (start_time > 500 &␣
↪start_time < 550 & tags LIKE "%LickEarly%")
D. units: (id > -1 & location == "CA3" & quality > 0.8)
E. general:(virus LIKE "%infectionLocation: M2%")
F. general/optophysiology/*: (excitation_lambda)
timing results are:
[   [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [12.323649999999999, 17.78795, 0.4762710000000021],
        [32.461324, 44.317103, 0.5755010000000045],
        [13.388241999999995, 18.805749000000002, 0.38864600000001115],
        [1.7776289999999744, 0.40620900000002624, 0.3828319999999854],
        [1.5525489999999929, 0.4276980000000199, 0.3758489999999952],
        [1.4789959999999809, 0.49392600000000186, 0.3884080000000002407]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [11.951703999999975, 17.684409000000002, 0.4850340000000246],
        [33.06460499999998, 44.871867000000016, 0.5884969999999967],
        [13.00898899999999, 18.879096999999987, 0.39565400000001283],
        [1.8240300000000254, 0.40825000000000244, 0.3842249999999545],
        [1.523098000000008, 0.4308119999999853, 0.3792180000000105],
        [1.4620449999999927, 0.48747699999997707, 0.3942920000000534]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [12.292634999999988, 17.71122100000002, 0.4931979999999925],
        [31.715346000000025, 44.269245999999974, 0.5827180000000176],
        [13.615158999999991, 18.62413699999996, 0.400113000000033],
        [1.77765100000002, 0.3981279999999856, 0.3793939999999729],
        [1.5594440000000311, 0.42230299999999943, 0.3798599999999581],
        [1.4684130000000266, 0.48533700000000124, 0.3832130000000012]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [12.56530700000004, 17.789719999999985, 0.4703429999999891],
        [31.675452000000043, 44.14379199999996, 0.5683930000000146],
        [13.648998000000073, 18.566490999999914, 0.38886699999999763],
        [1.7804190000000304, 0.40790099999992435, 0.38032800000005196],
        [1.55089499999999, 0.4261669999999782, 0.3787000000000198],
        [1.4687469999999614, 0.49429700000000665, 0.39292099999995145]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [12.537729000000027, 17.740118999999975, 0.48057100000004027],
        [30.873830999999957, 44.026783000000066, 0.5713640000000311],
        [13.659739000000002, 18.732270999999955, 0.3886049999999628],
        [1.732225999999919, 0.4136850000000223, 0.3754670000000644],
        [1.5275830000000497, 0.42749299999999124, 0.37764900000000523],
        [1.4371710000000277, 0.49300199999998995, 0.37664699999994866]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [12.065124000000033, 17.71671100000001, 0.473039],
        [34.59439099999999, 44.41940400000001, 0.5749450000000067],
```

(continues on next page)

(continued from previous page)

```
        [14.824584000000094, 20.070431999999975, 0.4134939999999574],
        [2.0133959999999576, 0.49045400000002815, 0.4563129999999589],
        [1.7613109999999637, 0.5116970000000407, 0.45816899999998384],
        [1.6201010000000338, 0.5919250000000389, 0.4557319999999905]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [13.863780999999896, 20.10773000000004, 0.4887419999999736],
        [34.575612000000014, 48.08052500000007, 0.6126809999999523],
        [14.748455999999948, 21.717973000000015, 0.4133940000000962],
        [1.8125289999999268, 0.4475129999999936, 0.4192520000000499],
        [1.6164730000000205, 0.4934969999999623, 0.4398180000000451],
        [1.6412199999999615, 0.578479999999999, 0.46646600000001825]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [14.183431999999925, 19.463961999999952, 0.5685050000000729],
        [33.69680699999992, 46.44652799999985, 0.5893350000001405],
        [13.886630000000025, 19.663959999999953, 0.4755180000001431],
        [1.7565279999999603, 0.41089800000007415, 0.390275999999929],
        [1.5056289999999493, 0.4478090000000705, 0.40615499999995563],
        [1.4745139999999282, 0.5386419999998964, 0.41592400000001958]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [13.39936200000016, 18.57474999999993, 0.5622109999999694],
        [33.46206100000019, 46.73720199999987, 0.5941449999999762],
        [14.150975000000066, 19.905867999999963, 0.46484499999986184],
        [1.987123999999838, 0.48372400000012306, 0.4484090000000194],
        [1.7172100000001436, 0.5058690000000183, 0.4525939999999835],
        [1.6099890000000698, 0.573598999999831, 0.45447500000000448]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [13.135035000000023, 18.866990999999928, 0.5028779999999742],
        [34.27441100000003, 48.12650700000013, 0.5868599999999731],
        [14.724086999999983, 19.623490999999817, 0.40979000000018573],
        [1.9606950000000296, 0.4782629999999557, 0.44371199999993394],
        [1.705235000000016, 0.5093840000000824, 0.446410999999884],
        [1.648959000000076, 0.582082999999912, 0.4529080000000647]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [13.732621999999807, 19.141613000000206, 0.5681570000000562],
        [34.59157899999997, 46.49771199999998, 0.613060999999874],
        [14.279798999999983, 20.263692999999975, 0.40834800000018845],
        [1.6990480000000048, 0.42399300000012374, 0.3962589999999011],
        [1.5614639999999724, 0.46319500000002733, 0.399421999999916],
        [1.5296450000001016, 0.5506969999998432, 0.41318099999999447]],
    [   ['NWB Query Engine', 'search_nwb', 'nwbindexer'],
        [13.533899999999988, 19.061607000000024, 0.5465729999999951],
        [35.23903399999986, 47.30263100000009, 0.6926130000000512],
        [14.941060999999976, 19.97429000000018, 0.4123069999999416],
        [2.001859000000195, 0.47430499999980213, 0.44398300000004554],
        [1.7040489999999409, 0.507457000000116, 0.448450999999892],
        [1.6666550000001905, 0.5813069999999954, 0.452112999999855]]]
tool: NWB Query Engine
time_ave: [12.96535674999999, 33.352037749999994, 14.07305991666668, 1.84359449999999,
 ↪ 1.6070783333333398, 1.5422045833333626]
time_min: [11.951703999999975, 30.873830999999957, 13.00898899999999, 1.
↪6990480000000048, 1.5056289999999493, 1.4371710000000277]
time_max: [14.183431999999925, 35.23903399999986, 14.941060999999976, 2.
↪0133959999999576, 1.7613109999999637, 1.6666550000001905]
tool: search_nwb
time_ave: [18.470565250000007, 45.76994166666666, 19.56895433333331, 0.
↪43694358333333844, 0.46444841666669096, 0.537564333333291]
time_min: [17.684409000000002, 44.026783000000066, 18.566490999999914, 0.
↪3981279999999856, 0.42230299999999943, 0.48533700000000124]
```

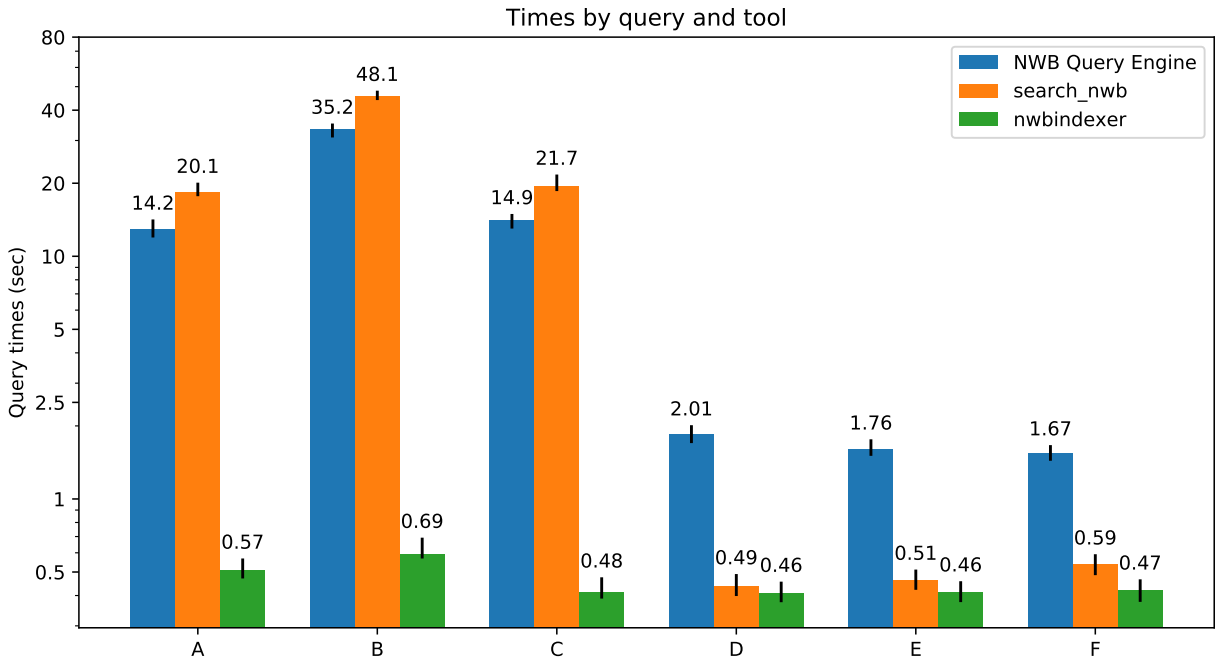
(continues on next page)

(continued from previous page)

```
time_max: [20.10773000000004, 48.12650700000013, 21.717973000000015, 0.
↪49045400000002815, 0.5116970000000407, 0.5919250000000389]
tool: nwbindexer
time_ave: [0.5096268333333408, 0.5958427500000032, 0.4132984166666993, 0.
↪40837083333332225, 0.41185799999997075, 0.42052333333334607]
time_min: [0.4703429999999891, 0.5683930000000146, 0.3886049999999628, 0.
↪3754670000000644, 0.3758489999999952, 0.37664699999994866]
time_max: [0.5685050000000729, 0.6926130000000512, 0.4755180000001431, 0.
↪4563129999999589, 0.45816899999998384, 0.46646600000001825]
```

In addition to the above output, the speed_check.py program also generates a bar chart showing the average time required for each tool to perform each query. Superimposed on the top of each bar is a vertical line which shows the minimum and maximum times required for the tool to run the query.

The bar chart generated from the above run is shown below.



CHAPTER 10

NWB files used

This sections lists the NWB files used for the *speed_check.py* utility example run which has partial output given in the previous section. The files used are:

- File: *Steinmetz2019_Forssmann_2017-11-05.nwb* (267.96 MB). From an NWB-formatted dataset referenced in Steinmetz et al. Nature 2019. Available at: https://figshare.com/articles/Datasets_from_Steinmetz_et_al_2019_in_NWB_format/11274968 MD5: fc4052bffd3c2f4cf8d42f000696348c
- A data file from Buzsaki Lab. File: *YutaMouse41-150903.nwb* (10 GB). It is available from https://buzsakilab.nyumc.org/datasets/NWB/SenzaiNeuron2017/ MD5: 3dd2251ba2c61abb9edae1117bbaecf0
- The first 16 files from the alm-1 data set at CRCNS.org, which contains anterior motor cortex recordings from the Svoboda Lab at Janelia Farm. (Total 2.2 GB). Available from: http://crcns.org/data-sets/motor-cortex/alm-1 MD5 sums are listed with the files at CRCNS.org.
- Files in file *nwb_1_28.zip* from the Anne K Churchland lab.17 (28 files, 17 GB). Available from: http://labshare.cshl.edu/shares/library/repository/37693/ MD5: b9890ad36b7721a1b1c921289f19c698
- Files generated from the PyNWB tutorials. (22 files, 542 MB total). The tutorials (Python code to generate the files) are at: https://pynwb.readthedocs.io/en/latest/tutorials/index.html The version of the tutorials used here was from commit 4f054b87 (which was the latest on Dec 27, 2019). Steps for generating the same version of the files are:

  ```
  git clone --recurse-submodules https://github.com/NeurodataWithoutBorders/pynwb.git
  ```

  ```
  cd pynwb
  ```

  ```
  git log --oneline -1
  ```

  If output is NOT:

  > 4f054b87 (HEAD -> dev, tag: latest, origin/dev, origin/HEAD) Update legacy import of ObjectMapper (#1124)

  then do:

  ```
  git checkout 4f054b87
  ```

(To checkout the version of PyNWB used to generate the version of the NWB files used in the *speed_check.py* run). Once the proper version of PyNWB is checked-out, do:

```
pip install -r requirements.txt

pip install .

cd docs/gallery/general

python file.py
```

(The 'python' command generates file *example_file_path.nwb*). To generate the other NWB files, run the python command on the other .py files in both general and the "gallery/domain" directory. (In directory "general" files: *advanced_hdf5_io.py*, *extensions.py*, *iterative_write.py*, *linking_data.py*, *scratch.py*); in directory "domain" files: *ecephys.py*, *icephys.py*, *ophys.py*, and *brain_observatory.py*). A total of 24 NWB files should be generated. They are included in the list below.

The list of files (in the index file, e.g. *nwb_index.db*) used in the *speed_check.py example run* is given below. This list was displayed by running the *query_index.py* program in interactive mode in the directory containing file *nwb_index.db*.

```
$ query_nwbindex ./
Using index_path: './nwb_index.db'
Opening './nwb_index.db'
Searching 70 files:
1. ./Steinmetz2019_Forssmann_2017-11-05.nwb
2. ./YutaMouse41-150903.nwb
3. ./alm-1/data_structure_ANM210861_20130701.nwb
4. ./alm-1/data_structure_ANM210861_20130702.nwb
5. ./alm-1/data_structure_ANM210861_20130703.nwb
6. ./alm-1/data_structure_ANM210862_20130626.nwb
7. ./alm-1/data_structure_ANM210862_20130627.nwb
8. ./alm-1/data_structure_ANM210862_20130628.nwb
9. ./alm-1/data_structure_ANM210863_20130626.nwb
10. ./alm-1/data_structure_ANM210863_20130627.nwb
11. ./alm-1/data_structure_ANM210863_20130628.nwb
12. ./alm-1/data_structure_ANM214427_20130805.nwb
13. ./alm-1/data_structure_ANM214427_20130806.nwb
14. ./alm-1/data_structure_ANM214427_20130807.nwb
15. ./alm-1/data_structure_ANM214427_20130808.nwb
16. ./alm-1/data_structure_ANM214429_20130805.nwb
17. ./alm-1/data_structure_ANM214429_20130806.nwb
18. ./alm-1/data_structure_ANM214429_20130807.nwb
19. ./churchland/mouse1_fni16_150817_001_ch2-PnevPanResults-170808-190057.nwb
20. ./churchland/mouse1_fni16_150818_001_ch2-PnevPanResults-170808-180842.nwb
21. ./churchland/mouse1_fni16_150819_001_ch2-PnevPanResults-170815-163235.nwb
22. ./churchland/mouse1_fni16_150820_001_ch2-PnevPanResults-170808-185044.nwb
23. ./churchland/mouse1_fni16_150821_001-002_ch2-PnevPanResults-170808-184141.nwb
24. ./churchland/mouse1_fni16_150825_001-002-003_ch2-PnevPanResults-170814-191401.nwb
25. ./churchland/mouse1_fni16_150826_001_ch2-PnevPanResults-170808-002053.nwb
26. ./churchland/mouse1_fni16_150827_001_ch2-PnevPanResults-170807-171156.nwb
27. ./churchland/mouse1_fni16_150828_001-002_ch2-PnevPanResults-170807-204746.nwb
28. ./churchland/mouse1_fni16_150831_001-002_ch2-PnevPanResults-170807-193348.nwb
29. ./churchland/mouse1_fni16_150901_001_ch2-PnevPanResults-170807-072732.nwb
30. ./churchland/mouse1_fni16_150903_001_ch2-PnevPanResults-170809-075033.nwb
31. ./churchland/mouse1_fni16_150904_001_ch2-PnevPanResults-170809-073123.nwb
32. ./churchland/mouse1_fni16_150915_001_ch2-PnevPanResults-170806-163508.nwb
33. ./churchland/mouse1_fni16_150916_001-002_ch2-PnevPanResults-170806-114920.nwb
```

(continues on next page)

(continued from previous page)

```
34. ./churchland/mouse1_fni16_150917_001_ch2-PnevPanResults-170806-110934.nwb
35. ./churchland/mouse1_fni16_150918_001-002-003-004_ch2-PnevPanResults-170715-124821.
↪nwb
36. ./churchland/mouse1_fni16_150921_001_ch2-PnevPanResults-170715-114212.nwb
37. ./churchland/mouse1_fni16_150922_001_ch2-PnevPanResults-170715-120548.nwb
38. ./churchland/mouse1_fni16_150923_001_ch2-PnevPanResults-170715-124558.nwb
39. ./churchland/mouse1_fni16_150924_001_ch2-PnevPanResults-170715-124619.nwb
40. ./churchland/mouse1_fni16_150925_001-002-003_ch2-PnevPanResults-170715-164713.nwb
41. ./churchland/mouse1_fni16_150928_001-002_ch2-PnevPanResults-170716-002540.nwb
42. ./churchland/mouse1_fni16_150929_001-002_ch2-PnevPanResults-170715-205011.nwb
43. ./churchland/mouse1_fni16_150930_001-002_ch2-PnevPanResults-161221-134921.nwb
44. ./churchland/mouse1_fni16_151001_001_ch2-PnevPanResults-161220-141515.nwb
45. ./churchland/mouse1_fni16_151002_001-002_ch2-PnevPanResults-161221-152112.nwb
46. ./churchland/mouse1_fni16_151005_001-002-003-004_ch2-PnevPanResults-161221-150439.
↪nwb
47. ./tutorials/domain/brain_observatory.nwb
48. ./tutorials/domain/ecephys_example.nwb
49. ./tutorials/domain/icephys_example.nwb
50. ./tutorials/domain/ophys_example.nwb
51. ./tutorials/general/advanced_io_example.nwb
52. ./tutorials/general/basic_alternative_custom_write.nwb
53. ./tutorials/general/basic_iterwrite_example.nwb
54. ./tutorials/general/basic_sparse_iterwrite_compressed_example.nwb
55. ./tutorials/general/basic_sparse_iterwrite_example.nwb
56. ./tutorials/general/basic_sparse_iterwrite_largearray.nwb
57. ./tutorials/general/basic_sparse_iterwrite_largechunks_compressed_example.nwb
58. ./tutorials/general/basic_sparse_iterwrite_largechunks_example.nwb
59. ./tutorials/general/basic_sparse_iterwrite_multifile.nwb
60. ./tutorials/general/cache_spec_example.nwb
61. ./tutorials/general/example_file_path.nwb
62. ./tutorials/general/external1_example.nwb
63. ./tutorials/general/external2_example.nwb
64. ./tutorials/general/external_linkcontainer_example.nwb
65. ./tutorials/general/external_linkdataset_example.nwb
66. ./tutorials/general/processed_data.nwb
67. ./tutorials/general/raw_data.nwb
68. ./tutorials/general/scratch_analysis.nwb
69. ./tutorials/general/test_cortical_surface.nwb
70. ./tutorials/general/test_multicontainerinterface.nwb
Enter query, control-d to quit
```

CHAPTER 11

License

The license for this package is given in file license.txt, which is included below.

```
Copyright ©2019. The Regents of the University of California (Regents).  All Rights␣
↪Reserved.

Permission to use, copy, modify, and distribute this software
and its documentation for educational, research, and not-for-profit purposes, without
fee and without a signed licensing agreement, is hereby granted, provided that the
above copyright notice, this paragraph and the following two paragraphs appear in all
copies, modifications, and distributions.  Contact The Office of Technology Licensing,
UC Berkeley, 2150 Shattuck Avenue, Suite 510, Berkeley, CA 94720-1620,
(510) 643-7201, otl@berkeley.edu, http://ipira.berkeley.edu/industry-info
for commercial licensing opportunities.

IN NO EVENT SHALL REGENTS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL,
INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE
OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF REGENTS HAS BEEN ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

REGENTS SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
THE SOFTWARE AND ACCOMPANYING DOCUMENTATION, IF ANY, PROVIDED HEREUNDER IS PROVIDED
"AS IS". REGENTS HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES,
ENHANCEMENTS, OR MODIFICATIONS.
```

CHAPTER 12

Indices and tables

- genindex
- search

Index