Searching the DANDI Archive

Support

NIH The BRAIN Initiative®

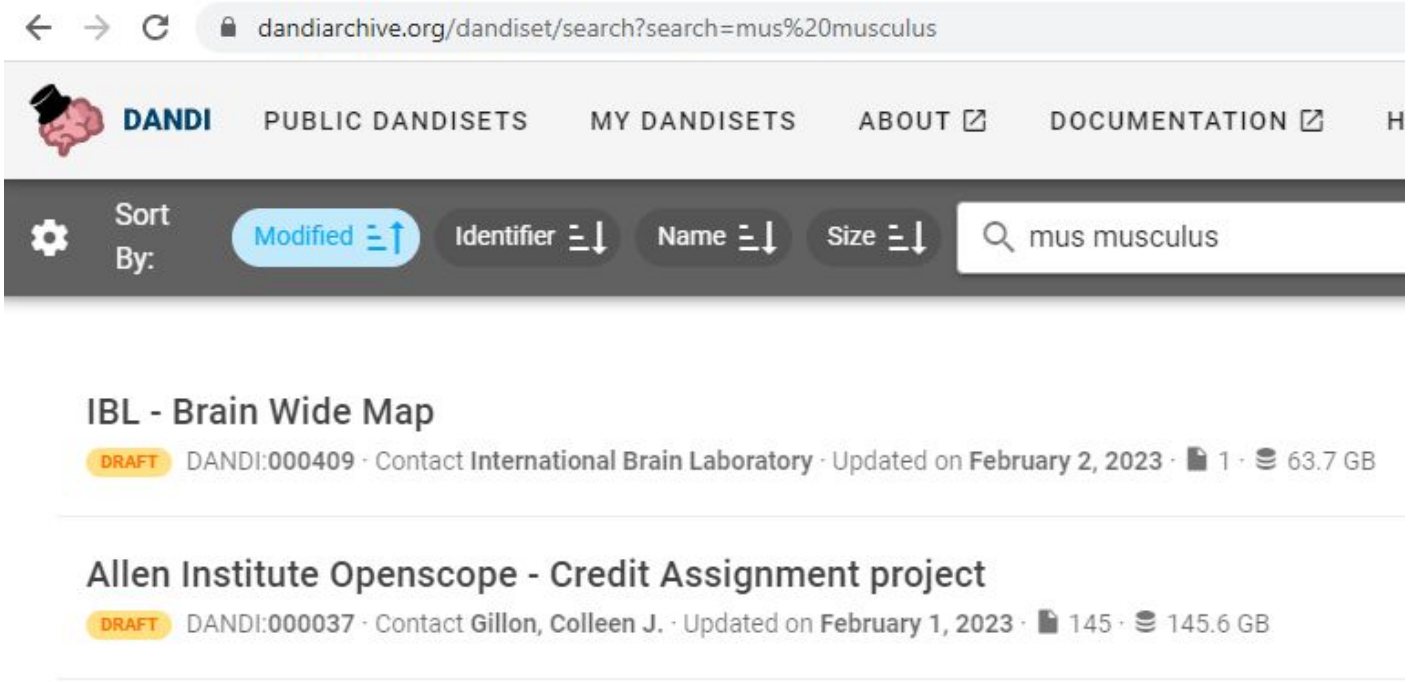
NIH National Institute of Mental Health

1R24MH117295
AWS Public dataset program

BICCN

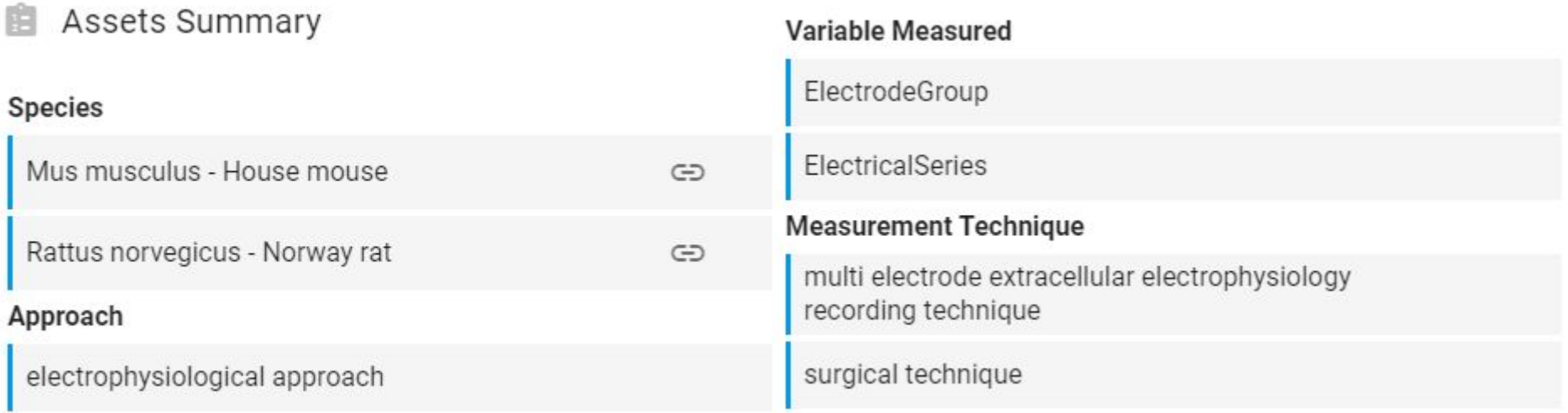
From the Web Interface

- From the main page https://dandiarchive.org/dandiset, type a query in the search bar



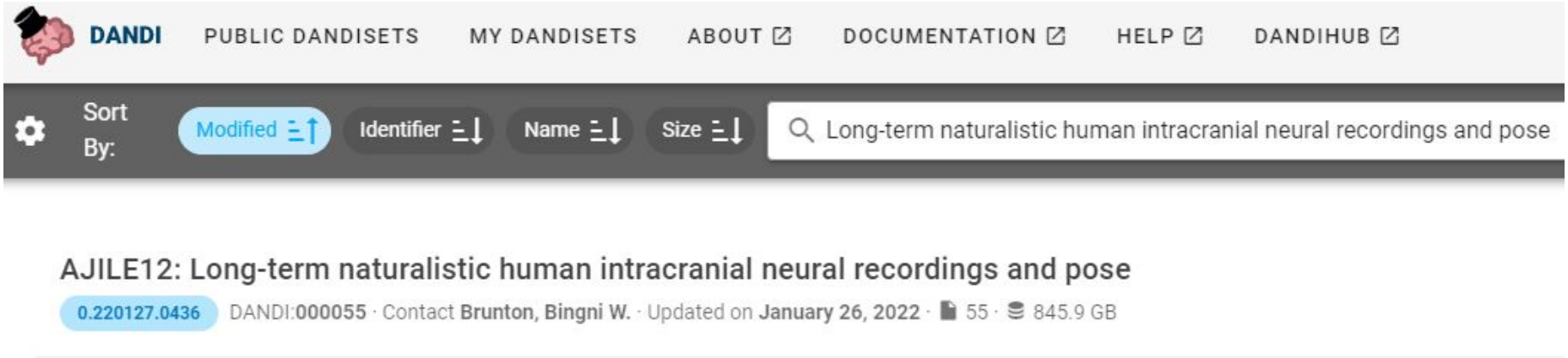
Web Interface

- These top-level queries look at the highest level of dandiset metadata
  - Titles, keywords, dandiset description
  - Automatically extracted subject species, modality, and techniques
  - You can see these fields listed at the bottom of the main page of any dandiset



Web Interface

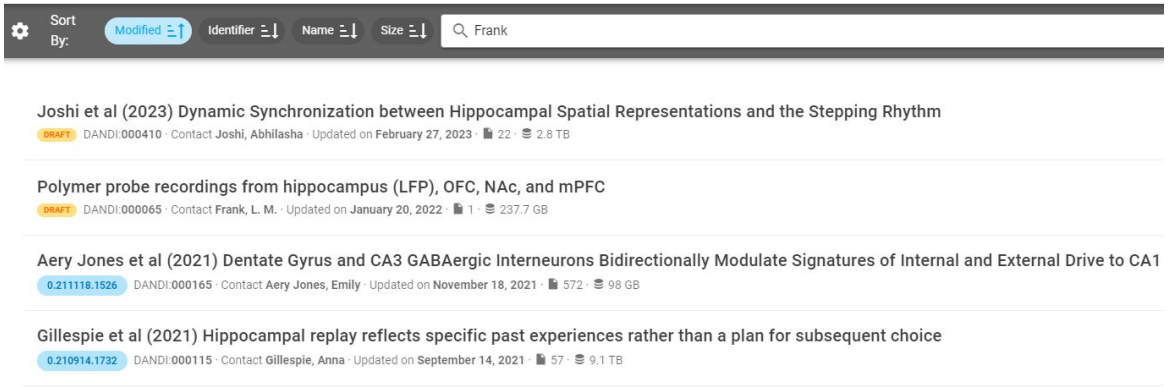
- Good for…
  - A quick glance or casual browsing of general content
  - Finding the dandiset corresponding to a publication
    - ➢ usually the title, or linked as a Related Resource (‘IsDescribedBy’)



bit.ly/nwb-cosyne-2023

Web Interface

- Good for…
  - A quick glance or casual browsing of general content
  - Finding the dandiset corresponding to a publication
    - ➢ usually the title, or linked as a Related Resource (‘IsDescribedBy’)
  - Finding all the dandisets belonging to a particular lab
    - ➢ search by name of a ‘Contributor’



Web Interface

- Good for…
  - A quick glance or casual browsing of general content
  - Finding the dandiset corresponding to a publication
    - ➢ usually the title, or linked as a Related Resource (‘IsDescribedBy’)
  - Finding all the dandisets belonging to a particular lab
    - ➢ search by name of a ‘Contributor’
  - Finding all the dandisets that use a particular species
    - ➢ Latin binomial, *e.g.*: *Mus musculus*, *Rattus norvegicus*, *Danio rerio*, etc.

Web Interface

- Doesn’t help with…
  - Presence or absence of raw vs. processed data
  - Presence or absence of identified brain regions or coordinates
  - The huge variety of behavioral techniques
    - ➢ open exploration vs. maze task
    - ➢ virtual reality vs. simple stimulus presentation
    - ➢ trialized tasks or spontaneous events
    - ➢ and many, many more…

bit.ly/nwb-cosyne-2023

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**pplication **P**rogramming **I**nterface (API) for DANDI to scan the metadata to programmatically obtain information

- Installation - preferably in a new conda environment

```
pip install dandi jupyter
jupyter notebook
```

Or use the DANDI Hub

bit.ly/nwb-cosyne-2023

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**pplication **P**rogramming **I**nterface (API) for DANDI to scan the metadata to programmatically obtain information

- Relevant methods
  - The `client` initiates communication with the archive

```
from dandi.dandiapi import DandiAPIClient

client = DandiAPIClient()
```

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**pplication **P**rogramming **I**nterface (API) for DANDI to scan the metadata to programmatically obtain information

- Relevant methods
  - The `client` can be queried to return currently public dandisets

```
dandisets = list(client.get_dandisets())
dandiset = dandisets[0]
dandiset
> dandi.dandiapi.RemoteDandiset
```

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**pplication **P**rogramming **I**nterface (API) for DANDI to scan the metadata to programmatically obtain information

- Relevant methods
  - A `dandi.dandiapi.RemoteDandiset` can return its pre-parsed metadata

```
raw_metadata = dandiset.get_raw_metadata()
raw_metadata
> { < kind of messy > }
```

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**pplication **P**rogramming **I**nterface (API) for DANDI to scan the metadata to programmatically obtain information

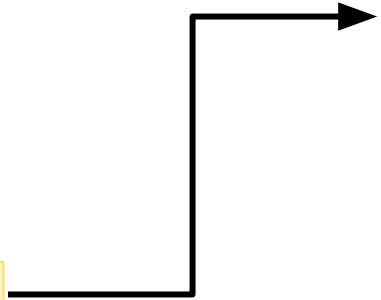
- Relevant methods
  - A `dandi.dandiapi.RemoteDandiset` can return its pre-parsed metadata

```
import json

print(json.dumps(raw_metadata, indent=4))
> {
        < A lot more readable >
  }
```

bit.ly/nwb-cosyne-2023

Using the DANDI API in Python

```
['id',
 'doi',
 'url',
 'name',
 'about',
 'access',
 'license',
 'version',
 '@context',
 'citation',
 'keywords',
 'protocol',
 'schemaKey',
 'identifier',
 'repository',
 'contributor',
 'description',
 'publishedBy',
 'studyTarget',
 'assetsSummary',
 'datePublished',
 'schemaVersion',
 'ethicsApproval',
 'wasGeneratedBy',
 'relatedResource',
 'manifestLocation']
```



```
"assetsSummary": [
    "species",
    "approach",
    "schemaKey",
    "dataStandard",
    "numberOfBytes",
    "numberOfFiles",
    "numberOfSubjects",
    "variableMeasured",
    "measurementTechnique"
]
```

bit.ly/nwb-cosyne-2023

Using the DANDI API in Python


```
{
    "age": {
        "value": "P209DT55274S",
        "unitText": "ISO-8601 duration",
        "schemaKey": "PropertyValue",
        "valueReference": {
            "value": "dandi:BirthReference",
            "schemaKey": "PropertyValue"
        }
    },
    "sex": {
        "name": "Male",
        "schemaKey": "SexType",
        "identifier": "http://purl.obolibrary.org/obo/PATO_0000384"
    },
    "species": {
        "name": "Mus musculus - House mouse",
        "schemaKey": "SpeciesType",
        "identifier": "http://purl.obolibrary.org/obo/NCBITaxon_100
    },
    "genotype": "Emx1-Cre[tg/wt];Ai32[tg/wt]",
    "schemaKey": "Participant",
    "identifier": "San4"
}
```

```
"assetsSummary": [
    "species",
    "approach",
    "schemaKey",
    "dataStandard",
    "numberOfBytes",
    "numberOfFiles",
    "numberOfSubjects",
    "variableMeasured",
    "measurementTechnique"
]
```


bit.ly/nwb-cosyne-2023

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**pplication **P**rogramming **I**nterface (API) for DANDI to scan the metadata to programmatically obtain information

- Relevant methods
  - Each file from `dandiset.get_asset(...)` can return its pre-parsed metadata

```
all_assets = dandiset.get_assets()
first_asset_raw_metadata = all_assets[0].get_raw_metadata()

print(json.dumps(first_asset_raw_metadata, indent=4))
> {
        < A lot of information >
  }
```

bit.ly/nwb-cosyne-2023

Using the DANDI API in Python

- Two demo notebooks are available on the DANDI Hub under
  ~/dandi-notebooks/tutorials/cosyne_2023
- Can also be downloaded directly from the links below
- Simple dandiset-level examples
  https://github.com/NeurodataWithoutBorders/nwb_hackathons/tree/main/Cosyne_2023/tutorials/simple_dandiset_search.ipynb
- Advanced asset-level examples
  https://github.com/NeurodataWithoutBorders/nwb_hackathons/tree/main/Cosyne_2023/tutorials/advanced_asset_search.ipynb

bit.ly/nwb-cosyne-2023

Investigating an Individual NWB File on DANDI

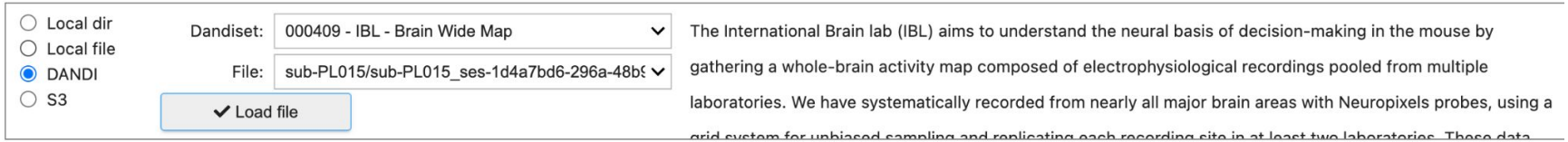
- So far we've only aggregated information over dandisets
- To investigate the contents of a single file, a good place to start is to try the NWB Widgets

```
pip install -U pynwb dandi jupyter nwbwidgets

jupyter notebook
```

And in the notebook…

```
from nwbwidgets import Panel

Panel()
```

bit.ly/nwb-cosyne-2023



session_description:	The full description of the session/task protocol can be found in Appendix 2 of Inte
identifier:	c33e2740-5475-463e-bd16-d1c38da37463
session_start_time:	2022-07-21 16:08:53.428769+01:00
timestamps_reference_time:	2022-07-21 16:08:53.428769+01:00
related_publications:	https://doi.org/10.6084/m9.figshare.21400815.v6, https://doi.org/10.1101/2020.01.17.909838
experiment_description:	IBL aims to understand the neural basis of decision-making in the mouse by gather
session_id:	1d4a7bd6-296a-48b9-b20e-bd0ac80750a5
lab:	Hausser
institution:	University College London
protocol:	_iblrig_tasks_ephysChoiceWorld6.6.1

- **file_create_date**
- **acquisition**
- **processing**
- **electrodes: metadata about extracellular electrodes**
- **electrode_groups**
- **devices**

nwbwidgets - Jupyter Notebook

localhost:8888/notebooks/nwbwidgets.ipynb

Apps | Bookmarks | Google Scholar | Gmail | teach python | gifs | Facebook | Classes | flatironinstitute/Cal... | Other Bookmarks

jupyter **nwbwidgets** Last Checkpoint: Last Wednesday at 9:07 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code Voila

▸ epochs: experimental epochs

▸ trials: experimental trials

▾ **units: Autogenerated by NWBFile**

Session Raster | Grouped PSTH | Raster Grid | table

unit 0

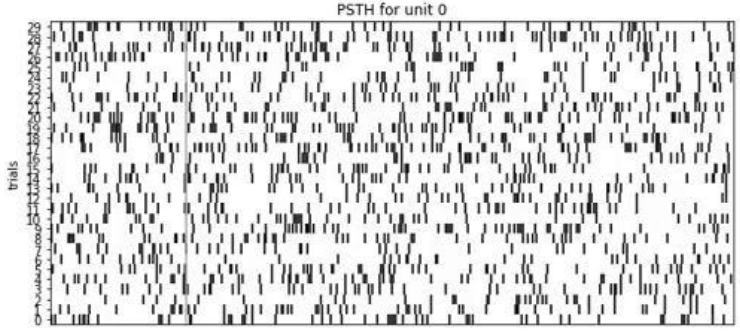
align to: start_time

order by start_time

group by None

before (s) 0.50

after (s) 2.00



Reading directly from an identified file

- Once you have concluded your investigation and found some NWB files of interest, you can either…
  - download them locally via command-line terminal

```
dandi download DANDI:<six-digit-ID>  # Will download all files
```

Or

```
dandi download <copy and paste individual file URL>
```

Then in Python (script or notebook)...

```python
from pynwb import NWBHDF5IO


io = NWBHDF5IO(path="../path_to_single_file.nwb", load_namespaces=True)

nwbfile = io.read()
```

Reading directly from an identified file

- Once you have concluded your investigation and found some NWB files of interest, you can either…
  - download them locally via command-line terminal

```
dandi download DANDI:<six-digit-ID>  # Will download all files
```

Or

```
dandi download <copy and paste individual file URL>
```

Then in MATLAB…

```
%% With MatNWB downloaded and added to your MATLAB session path…
nwbfile = nwbRead('.../path_to_single_file.nwb')
```

Streaming directly from an identified file

- Or stream from the cloud (most recommended for one-off analyses or quick calculations)

  In Python, there are two methods: fsspec and ros3

```
import h5py
import fsspec
from fsspec.implementations.cached import CachingFileSystem
from nwbinspector.tools import get_s3_urls_and_dandi_paths
from pynwb import NWBHDF5IO

S3_urls_to_dandi_paths = get_s3_urls_and_dandi_paths(dandiset_id="<sig-digit ID>")
dandi_paths_to_s3_urls = {dandi_path: s3_url for s3_url, dandi_path in s3_urls_to_dandi_paths.items()}
s3_url = dandi_paths_to_s3_urls["<file path on DANDI>.nwb"]

cache = CachingFileSystem(fs=fsspec.filesystem("http"), cache_storage="some/temporary/folder")
file_system = cache.open(s3_url, "rb")
file = h5py.File(file_system)

io = NWBHDF5IO(file=file, load_namespaces=True)
nwbfile = io.read()
```

Streaming directly from an identified file

- Or stream from the cloud (most recommended for one-off analyses or quick calculations)

  In MATLAB*…

```
%% The S3 path must be copy/pasted manually

s3_url = 's3://dandiarchive/blobs/7ee/415/7ee41580-9b0b-44ca-8675-6959ddd8dc33'

nwbfile = nwbRead(s3_url)
```

* streaming speeds are much slower than in Python

Dandiset. n.

An organized collection of assets (files) with both file level and dataset level metadata generated from an experiment or a project.

*A dandiset is a **FAIR** collection.*

***F**indable* ***A**ccessible* ***I**nteroperable* ***R**eusable*

FAIR is challenging but essential

Icons courtesy of Anita Bandrowski