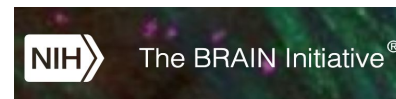


Searching the DANDI Archive



Support



1R24MH117295
AWS Public dataset program



BICCN

Dandiset. n.

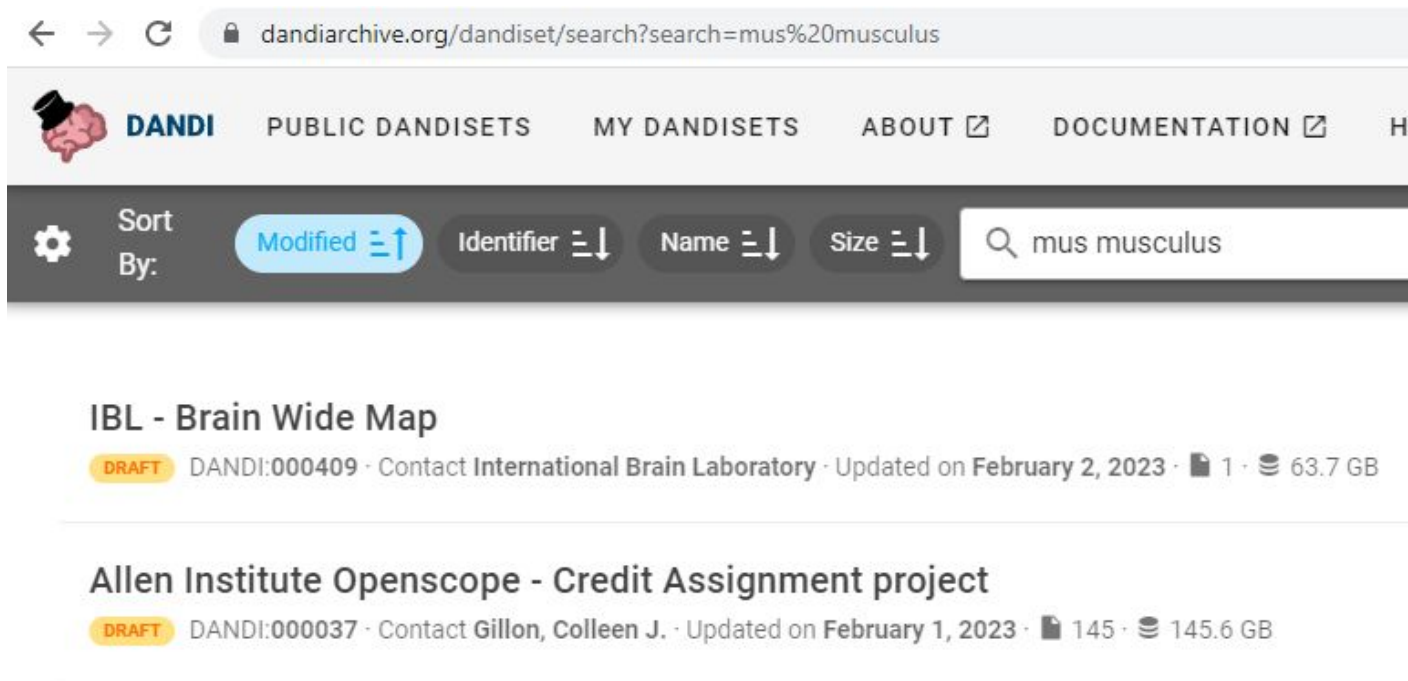
An organized collection of assets (files) with both file level and dataset level metadata generated from an experiment or a project.

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

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


From the Web Interface

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



The screenshot shows the DANDI Archive web interface. The browser address bar displays the URL `dandiarchive.org/dandiset/search?search=mus%20musculus`. The page header includes the DANDI logo (a brain with a top hat) and navigation links: PUBLIC DANDISETS, MY DANDISETS, ABOUT, and DOCUMENTATION. Below the header is a search bar containing the text "mus musculus". To the left of the search bar are sorting options: "Sort By:" with a gear icon, and buttons for "Modified" (selected), "Identifier", "Name", and "Size", each with a sort icon. The search results are listed below the search bar. The first result is "IBL - Brain Wide Map", marked as a "DRAFT" (orange label), with DANDI ID 000409, contact "International Brain Laboratory", updated on "February 2, 2023", 1 file, and 63.7 GB. The second result is "Allen Institute Openscope - Credit Assignment project", also marked as a "DRAFT", with DANDI ID 000037, contact "Gillon, Colleen J.", updated on "February 1, 2023", 145 files, and 145.6 GB.

← → ↻ dandiarchive.org/dandiset/search?search=mus%20musculus

 **DANDI** PUBLIC DANDISETS MY DANDISETS ABOUT DOCUMENTATION

Sort By: Modified Identifier Name Size

Q mus musculus

IBL - Brain Wide Map
DRAFT DANDI:000409 · Contact International Brain Laboratory · Updated on February 2, 2023 · 1 · 63.7 GB

Allen Institute Openscope - Credit Assignment project
DRAFT DANDI:000037 · Contact Gillon, Colleen J. · Updated on February 1, 2023 · 145 · 145.6 GB

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



Assets Summary

Species

Mus musculus - House mouse



Rattus norvegicus - Norway rat



Approach

electrophysiological approach

Variable Measured

ElectrodeGroup

ElectricalSeries

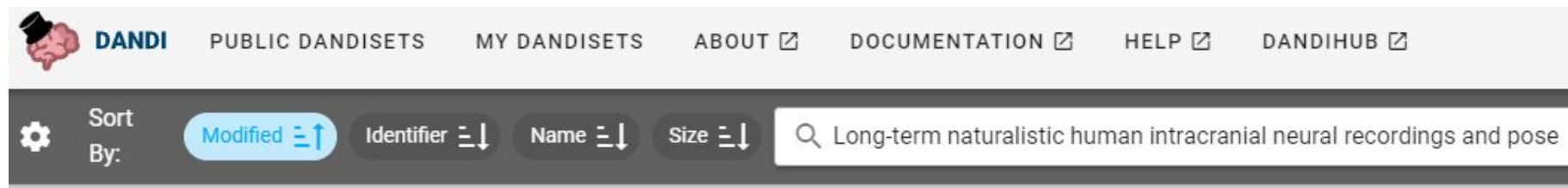
Measurement Technique

multi electrode extracellular electrophysiology
recording technique

surgical technique

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')



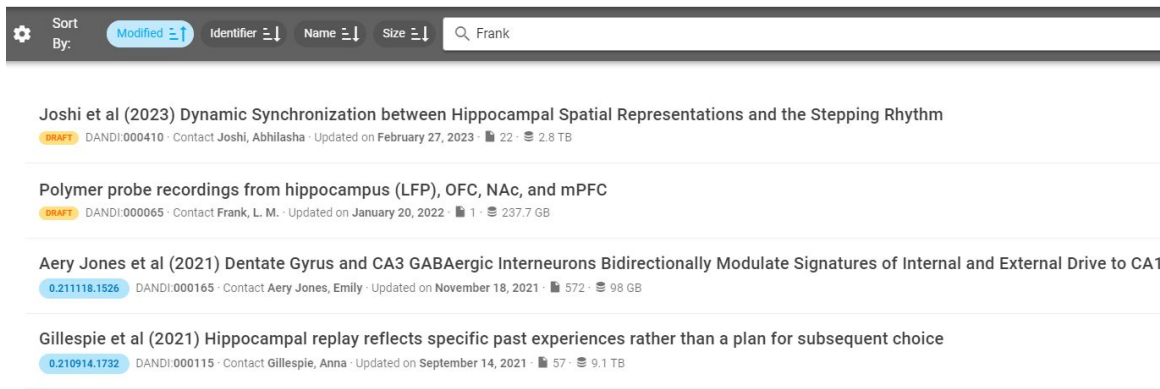
AJILE12: Long-term naturalistic human intracranial neural recordings and pose

0.220127.0436 DANDI:000055 · Contact Brunton, Bingni W. · Updated on January 26, 2022 · 55 · 845.9 GB

github.com/NeurodataWithoutBorders/nwb_hackathons/tree/main/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'



The screenshot shows the DANDI web interface. At the top, there is a navigation bar with a settings icon, a 'Sort By:' dropdown set to 'Modified', and several filter buttons: 'Identifier', 'Name', and 'Size'. A search bar on the right contains the text 'Frank'. Below the navigation bar, a list of datasets is displayed. Each entry includes a title, a draft status (yellow 'DRAFT' tag), a DANDI ID, contact information, an update date, and file statistics (number of files and size).

Sort By:	Modified	Identifier	Name	Size	Search
Joshi et al (2023) Dynamic Synchronization between Hippocampal Spatial Representations and the Stepping Rhythm					
DRAFT DANDI:000410 · Contact Joshi, Abhilasha · Updated on February 27, 2023 · 22 · 2.8 TB					
Polymer probe recordings from hippocampus (LFP), OFC, NAc, and mPFC					
DRAFT DANDI:000065 · Contact Frank, L. M. · Updated on January 20, 2022 · 1 · 237.7 GB					
Aery Jones et al (2021) Dentate Gyrus and CA3 GABAergic Interneurons Bidirectionally Modulate Signatures of Internal and External Drive to CA1					
0.211118.1526 DANDI:000165 · Contact Aery Jones, Emily · Updated on November 18, 2021 · 572 · 98 GB					
Gillespie et al (2021) Hippocampal replay reflects specific past experiences rather than a plan for subsequent choice					
0.210914.1732 DANDI:000115 · Contact Gillespie, Anna · Updated on September 14, 2021 · 57 · 9.1 TB					

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, pre-processed, or post-processed data
- Presence or absence of identified brain regions
 - unless the creator of the dandiset set these as keywords or 'subject matter'
- Presence or absence of exact brain coordinates (like Allen CCFv3)
- 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...

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**ppled **P**rogramming Interface (API) for DANDI to scan the `metadata.yml` files to programmatically obtain information
- Installation - preferably in a new conda environment

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



Notebook



Or use the DANDI Hub

How to use the **FREE** DANDI Hub computing resources

- Step 1: Make an account on the DANDI Archive
 - ▶ 1a: Requires a GitHub account
 - ▶ 1b: Wait for account approval (~1 day, faster during special events)
- Step 2: Go to <https://hub.dandiarchive.org/>
 - ▶ 2a: Sign in

Start My Server

How to use the **FREE** DANDI Hub computing resources

■ Step 1: Make an account on the DANDI Archive

- ▶ 1a: Requires a GitHub account
- ▶ 1b: Wait for account approval (~1 day, faster during special events)

■ Step 2: Go to <https://hub.dandiarchive.org/>

- ▶ 2a: Sign in
- ▶ 2b: Select instance size (amount of resources)

Server Options

<input type="radio"/>	Tiny. Useful for many quick things 0.5 CPU / 1 GB
<input checked="" type="radio"/>	Base 6 CPU / 16 GB upto 12C/32G. May take up to 15 mins to start.
<input type="radio"/>	Large 12C/32G upto 24C/64G. May take up to 15 mins to start.
<input type="radio"/>	Extra large 24C/64G upto 48C/96G. May take up to 15 mins to start.
<input type="radio"/>	T4 GPU for inference 4 CPU / 15 GB / 1 T4 GPU. May take up to 15 mins to start.
<input type="radio"/>	Beta (MATLAB - Bring your own license) 6 CPU / 16 GB upto 12C/32G. May take up to 15 mins to start. This requires your own license.

Start

How to use the **FREE** DANDI Hub computing resources

- Step 1: Make an account on the DANDI Archive
 - ▶ 1a: Requires a GitHub account
 - ▶ 1b: Wait for account approval (~1 day, faster during special events)
- Step 2: Go to <https://hub.dandiarchive.org/>
 - ▶ 2a: Sign in
 - ▶ 2b: Select instance size (amount of resources)
 - ▶ 2c: Wait a few minutes
- Step 3: No more steps

You're in!



Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**ppled **P**rogramming Interface (API) for DANDI to scan the `metadata.yml` files 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**ppled **P**rogramming Interface (API) for DANDI to scan the `metadata.yml` files 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**ppplied **P**rogramming Interface (API) for DANDI to scan the `metadata.yml` files 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**ppled **P**rogramming Interface (API) for DANDI to scan the `metadata.yml` files 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 >
```

```
}
```


Using the DANDI API in Python

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

A diagram consisting of a black arrow originates from the 'assetsSummary' field in the list on the left and points to a detailed JSON object structure on the right.

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

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\_10068"
  },
  "genotype": "Emx1-Cre[tg/wt];Ai32[tg/wt]",
  "schemaKey": "Participant",
  "identifier": "San4"
}
```

"assetsSummary": [

- "species",
- "approach",
- "schemaKey",
- "dataStandard",
- "numberOfBytes",
- "numberOfFiles",
- "numberOfSubjects",
- "variableMeasured",
- "measurementTechnique"



A black arrow originates from the "species" field within the main JSON object and points to the "species" field within the "assetsSummary" array, indicating a relationship or reference between the two.

Using the DANDI API in Python

- For finer-grain searchability, we can use the **A**ppplied **P**rogramming Interface (API) for DANDI to scan the `metadata.yml` files 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 >
}
```

Using the DANDI API in Python

- 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

Investigating an Individual NWB File on DANDI

- The dandi-api metadata aggregates information over the contents of the dandiset
- 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()
```

- ☐ Local dir
☐ Local file
☒ DANDI
☐ S3

Dandiset: 000409 - IBL - Brain Wide Map

File: sub-PL015/sub-PL015_ses-1d4a7bd6-296a-48b9-b20e-bd0ac80750a5

✓ Load file

The International Brain lab (IBL) aims to understand the neural basis of decision-making in the mouse by gathering a whole-brain activity map composed of electrophysiological recordings pooled from multiple laboratories. We have systematically recorded from nearly all major brain areas with Neuropixels probes, using a grid system for unbiased sampling and replicating each recording site in at least two laboratories. These data

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

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)...

```
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: [ros3](#) and [fsspec](#)

```
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