A perspective on neuroscience data standardization with Neurodata Without Borders

Andrea Pierré[1,2], Tuan Pham[1,2], Jonah Pearl[3], Sandeep Robert Datta[3], Jason T. Ritt[2,✉], and Alexander Fleischmann[1,2,✉]

[1]Department of Neuroscience, Division of Biology and Medicine, Brown University, Providence, USA
[2]The Robert J. and Nancy D. Carney Institute for Brain Science, Brown University, Providence, USA
[3]Department of Neurobiology, Harvard Medical School, Boston, USA

Neuroscience research has evolved to generate increasingly large and complex experimental data sets, and advanced data science tools are taking on central roles in neuroscience research. Neurodata Without Borders (NWB), a standard language for neurophysiology data, has recently emerged as a powerful solution for data management, analysis, and sharing. We here discuss our labs' efforts to implement NWB data science pipelines. We describe general principles and specific use cases that illustrate successes, challenges, and non-trivial decisions in software engineering. We hope that our experience can provide guidance for the neuroscience community and help bridge the gap between experimental neuroscience and data science.

Neurodata Without Borders | NWB | calcium imaging | behavior | standardization | data science

Correspondence:

- *jason_ritt@brown.edu*
- *alexander_fleischmann@brown.edu*

Contents

1. Introduction

1.1. Increasing complexity of neuroscience data Over the past 20 years, neuroscience research has been radically changed by two major trends in data production and analysis. First, neuroscience research now routinely generates large datasets of high complexity. Examples include recordings of activity across large populations of neurons, often with high resolution behavioral tracking [1–5], analyses of neural connectivity at high spatial resolution and across large brain areas [6, 7], and detailed molecular profiling of neural cells [8–11]. Such large, multi-modal data sets are essential for solving major questions about brain function [12–14].

Second, the collection and analysis of such datasets requires interdisciplinary teams, incorporating expertise in systems neuroscience, engineering, molecular biology, data science, and theory. These two trends are reflected in the increasing numbers of authors on scientific publications [15], and the creation of mechanisms to support team science by the NIH and similar research funding bodies [12, 16, 17].

There is also an increasing scope of research questions that can be addressed by aggregating "open data" from multiple studies across independent labs. Funding agencies and publishers have begun to aggressively promote data sharing and open data, with the goals of improving reproducibility and increasing data reuse [18–20]. However, open data may be unusable if scattered in a wide variety of naming conventions and file formats lacking machine-readable metadata.

Big data and team science necessitate new strategies for how to best organize data, with a key technical challenge being the development of standardized file formats for storing, sharing, and querying datasets. Prominent examples include the Brain Imaging Data Structure (BIDS) for neuroimaging, and Neurodata Without Borders (NWB) for neurophysiology data [21–24]. These initiatives provide technical tools for storing and accessing data in known formats, but more importantly provide conceptual frameworks with which to standardize data organization and description in an (ideally) universal, interoperable, and machine-readable way.

1.2. Our labs' perspective on implementing NWB data standards In 2019, the Fleischmann and Ritt labs initiated a collaboration to enhance the Fleischmann lab's data science and computational tooling and workflows. We expanded our team by hiring two research software engineers (RSE), and by extending collaborations with data scientists and computational biologists. Similar efforts were underway in the Datta lab. An early common goal was the standardization of neurophysiology and behavioral data using a framework such as NWB. In this manuscript, we provide our perspective on opportunities and challenges when adopting NWB data standardization.

Our labs investigate the functions of neural circuits for sensory processing and behavior in mice. Typical experiments include calcium imaging of neuronal activity in awake, head-fixed mice during odor presentation, with a number of behavioral readouts including sniffing, running, and facial movements (see Fig. 1). In other experiments, mice are freely moving, with implanted GRIN lenses for miniscope imaging, odor and reward delivery in nose ports, and behavioral readouts including videographic tracking. Our experimental designs, data generation, and analyses are similar to many other labs investigating neural circuit mechanisms for sensory-motor transformations, learning, and memory (Box 1), though each lab has its own idiosyncrasies impinging on data management.

In this manuscript, we first discuss our motivation and general considerations for implementing data standardization. We then describe the implementation of NWB data conversion pipelines, including domain-specific use cases and solutions for data sharing. We conclude by identifying opportunities for improving future user experience. We hope that by describing our experience, other labs planning to adopt NWB will benefit from comparisons with their own needs and capabilities. We also hope to provide a case study that may be informative for developers of NWB and similar data science toolboxes.

Box 1. *Fleischmann Lab workflow*

Data Acquisition – Experiments and Systems: We perform *in vivo* calcium imaging experiments in head-fixed (2-photon imaging) and freely moving (miniscope) mice. Experiments include multi-plane, multi-color, and/or multi-day recordings.

Data Acquisition – Tasks and Stimuli: In some experiments, animals receive pre-programmed odor stimuli independent of their behavior; in other experiments, sensory stimuli or an animal's behavior can trigger a reward. Behavior recording includes microcontroller-acquired time series (e.g. wheel speed, sniff rate, licks, rewards) and video recordings of the animal's face or body motion.

Preprocessing: Pipelines include conventional calcium imaging steps (e.g. motion correction, segmentation, deconvolution, multi-color or multi-day registration) using existing tools such as Suite2p [25] and Inscopix [26]. Experiments with behavioral videos may also be preprocessed with toolboxes such as DeepLabCut [3] for pose estimation and Facemap [27] for facial motion extraction.

Conversion to standard format: Raw and preprocessed data streams are integrated and stored in NWB files, using a custom tool, `calimag` [28], developed in the Fleischmann lab.

Analyses: Questions include stimulus or behavior tuning of single neuron or population activity, as well as how learning and experience shape neural activity.

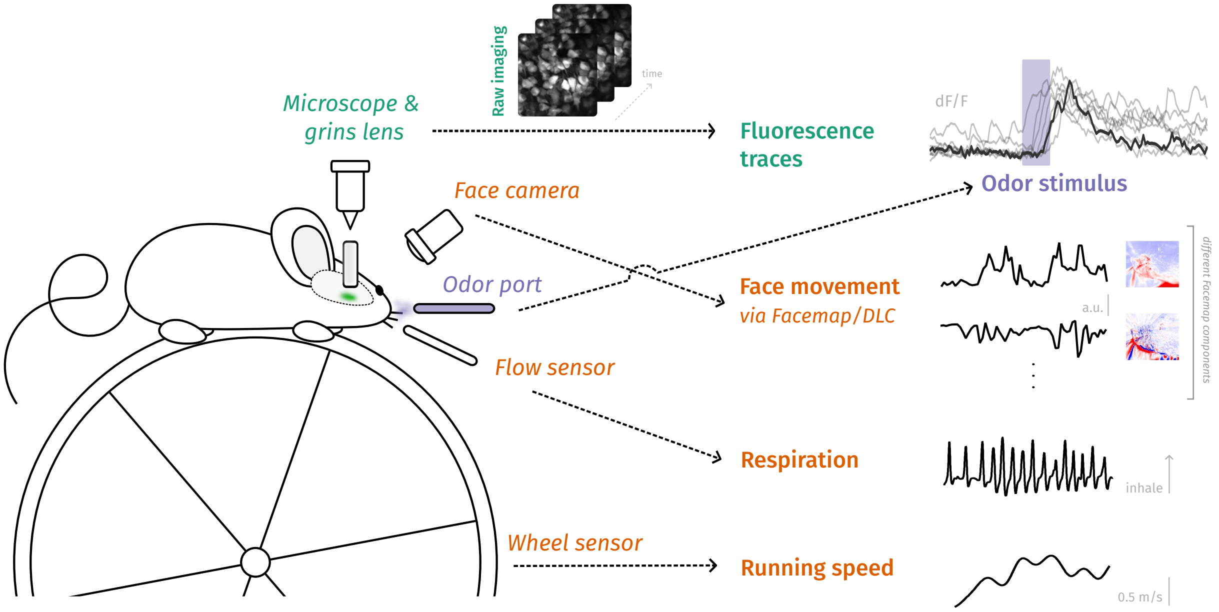


Fig. 1. Illustration of a typical experiment setup and the resulting data streams from the lab. This figure shows an example of an *in vivo* head-fixed two-photon calcium imaging recording in a deep brain area (e.g. piriform cortex) with grins lens. Throughout the paper, we use the following color scheme whenever possible: green for neural activity, orange for animal behaviors and purple for external variables (e.g. stimulus). Raw imaging recording from the microscope (top) is typically processed with `Suite2p` to obtain fluorescence time series data for each segmented neuron (top row, right). The animal receives odor stimuli with the odor port in a specific time window of each trial, illustrated with a light purple bar in the fluorescence time series plot. Various behaviors may be tracked. The high resolution face camera captures the animal's face movement, and processed with `facemap` to extract different facial motion principal components and the accompanying time-varying changes (middle). Alternatively, `DeepLabCut` can be used for pose estimation data. The data from flow and wheel sensors from the microcontroller (e.g. Arduino, Teensy, Raspberry Pi) are processed to obtain estimation of the animal's respiration and running speed, respectively.

2. Key stakeholders in adoption of a new lab standard

We first define, in high level terms, three distinct personnel roles in a typical research lab, each of whom has their own needs and incentives surrounding data standardization:

- **PIs** are principal investigators and senior researchers that manage research teams, labs, and projects.
- **Researchers** include research trainees (e.g. undergraduate and graduate students, postdoctoral associates), lab technicians, and data scientists, and more generally individuals collecting and/or analyzing data.
- **Research software engineers** (RSE) support researchers by developing and maintaining software, packages, and pipelines for data management, processing, and analysis.

2.1. PIs Key desired outcomes for the adoption of lab-wide standardized data formats include improved efficiency, rigor, reproducibility, and ease of collaboration. Efficiency could follow from using common tools for saving, retrieving, analyzing, and sharing data; technical improvements by one member can have knock-on value for others. Rigor and reproducibility similarly benefit from increased access and scrutiny brought by all lab members being able to see each other's work, instead of working in isolation; data already in standard formats could ease communication and usage. An additional value for PIs is meeting the norms of their field for data management and sharing, including mandates from funding agencies such as the NIH, without requiring extensive *ad hoc* effort at the time of grant submissions or publication.

However, there are several concerns when introducing standardized formats. PIs generally want to avoid major disruptions to scientific productivity in the lab. There is rarely a good time to slow or halt data collection and analysis in order to fully convert to new pipelines and workflows. On the other hand, a gradual transition can paradoxically lead to greater friction due to the simultaneous use of multiple incompatible systems. Adoption of a data standard can be much more than a point-and-click operation, requiring many decisions about the structure and use of the data not just as it is now, but also what the PI expects it to be in the future. One of the first decisions is the standard itself: it can be difficult to pick a "winner", as standards may quickly become incompatible with the lab's evolving methods.

It is also uncommon to have institutional support, in the form of grant funding or university staffing allocated to the "low level" task of revising data formats, or incentives such as promotion criteria that reward best practices in data management. While research software engineers (RSE) are increasingly recognized as valuable contributors to the research enterprise [29], most labs still do not have access to an RSE. This places the burden on students and postdocs, who are often enthusiastic to adopt new practices but are constrained by a need to make continual progress in their own careers. Moreover, lab members, including PIs, generally lack advanced training to know how to build automated systems that integrate multiple data streams into a single format with appropriate metadata, provide that data for analysis, and share data following community norms such as FAIR guidelines [30]. Without support, adopting a standard is often a shared aspiration with little personal buy-in to do the needed work.

2.2. Researchers The main motivation for researchers to adopt standardized data formats is to improve data analysis and shareability. Standardized data formats may support efficient and reproducible data processing and flexible, comprehensive data exploration and analysis. Efficient data analysis can, in turn, provide critical information for optimizing experimental design. Furthermore, standardized formats facilitate data sharing, which can yield new perspectives on datasets and increase their impact.

A main concern is that data standardization requires a significant increase in workload, whether researchers tackle it on their own or in collaboration with an RSE. The increased workload can happen at the experiment and data conversion stages, if data management standardization comes at the expense of experimental flexibility. At the stage of analysis, researchers may need to spend time to learn and adapt to the new standard in order to use the data. Researchers' diverse backgrounds, the availability/support of tools for standardized data, and the maturity of their project further complicate tradeoffs between making consistent experimental progress and standardizing experimental outputs. Additionally, researchers who decide to embrace standardization, open data, and reproducible workflows often lack recognition for the added work.

2.3. Research software engineers RSEs directly support researchers in data management, analysis, sharing, and publication. Adopting standardized formats establishes predictability in the data that the researchers produce. This facilitates communication and makes it easier for RSEs to efficiently provide support in finding, using, and building appropriate systems to interact with the data. RSEs can also take advantage of such predictability to provide sufficient documentation and usable examples of the data for analysis, sharing and re-use.

A core challenge is developing stable software implementations and workflows that are robust to small variations in experimental data, while still allowing flexibility to be useful to researchers engaged in rapid evolution of diverse experimental designs. Furthermore, choosing a new technology carries an elevated risk of bugs and missing features. Open source tools can be particularly unpredictable, and extensive in-house workarounds may be unsustainable and defeat the original purpose of standardization.

In addition, researchers and RSEs often come from different backgrounds. RSEs may not be familiar with scientific priorities and experimental constraints, and the expectations and timeline of research projects. Thus, diverging expectations and miscommunication between researchers and RSEs can lead to friction and delay in adopting the standards.

3. Social scales of working with the NWB standard

3.1. Within a lab It is often desirable for members of a lab to share and use common technology, including analysis code, data conversion pipelines, and/or acquisition systems. This commonality allows members to jointly address technical problems, and build on top of known solutions with some degree of prior validation, creating consistency across "generations" of graduate students and postdocs. For example, in our lab, researchers performing head fixed two-photon calcium imaging share the same acquisition systems and data conversion pipeline, which allows them to get advice from their peers and to contribute their own solutions to common pain-points.

A potential pitfall of sharing a common set of technologies may arise when the technology is not well maintained or kept up-to-date, forcing new projects to build on shaky ground. Another pitfall may come from the complexity of supporting a diverse enough set of use cases, and trying to make them all fit into the same technology.

On-boarding is key to encourage this economy of scale and self-regeneration of benefits, especially if a standard is not yet established. For example, rather than introduce NWB to researchers in new analysis notebooks, we tried to work backwards from the analysis pipeline they already used. That is, we refactored researchers' existing code by replacing only file load operations and converting to whatever variable names and data type conventions they already used (which often embedded excess structure from the original raw data file formats). Further experience with NWB might motivate changes to those conventions, but in this approach, initial learning is focused on practical steps whose value is innately recognized by the researcher, rather than on the generic NWB software interface.

The Fleischmann lab uses lab-wide Git hosting (on GitLab), facilitating internal sharing and collaborative development of code. Combined with regular lab meeting discussion of data management and analysis topics, this culture of open communication and sharing helps disseminate technical progress across all lab members.

3.2. Collaboration Our experience using NWB to send data to collaborators in other labs has been more mixed than for internal adoption. While standardization aims to establish a universal language for data, there can still be friction for recipients who have not already installed and used the necessary software, especially in the absence of good documentation and relevant working examples. We describe two cases with two different labs performing additional analyses on data we collected.

In the first case, we provided our collaborators with raw microscope images as TIFF stacks and pre-processed calcium activity time series in NWB format. However, as they were unfamiliar with NWB, it was challenging for them to learn how to use the files. With hindsight, we should have included working example code that loaded and displayed data, which they could use as a starting template for their own work. However, there would still have been some friction, as their lab works primarily in Matlab, while we work almost entirely in python. NWB provides APIs for both environments, but we would have needed to generate example code from scratch, and the two labs would have maintained two separate code bases. In the end our collaborators used only the TIFF stacks, though partly in order to also work on novel pre-processing algorithms.

In the second case, our collaborators had previous experience with NWB. However, we were still refining our NWB conversion of that data, and were regularly making code breaking changes. Hence, we chose to create and send python "pickle" files that contained only a subset of the data, organized to simplify usage on their end and make it easier for us to create example code and documentation. As we continued to develop our internal pipelines, this approach hampered code interoperability between our labs. However, it was the more expedient choice to get the collaborators up and running.

3.3. Public data sharing Researchers are increasingly asked to publish their data on public archives. Apart from publication and funding requirements and opportunities for collaboration, these public data repositories increase chances of data reuse, e.g. for education, benchmarking new tools, computational modelling, or meta-analysis. Popular repositories include Figshare [31], Zenodo [32], OSF [33, 34] and GIN G-Node [35]. These are more general repositories, with limited restrictions on data formats.

The Distributed Archives for Neurophysiology Data Integration (DANDI, [36]) is the recommended choice for public data sharing of NWB datasets, and is supported by both the BRAIN Initiative [37] and the AWS Public dataset programs. While it is more restrictive compared to other repositories (for example DANDI allows only standardized formats [38], while Zenodo allows all formats [39]), the resulting rigor and consistency from DANDI may better facilitate reproducibility, modelling, meta-analysis, and tool development [40, 41]. We discuss our experience contributing a demonstrative calcium imaging dataset [42] on DANDI in Section 6.

Apart from file format restrictions, researchers may need to take into account file size limits. DANDI has fairly generous limits, with 5 TB per file and no limit on dataset size, while some repositories have limits of less than 100 GB per file or dataset (some offer higher limits for a fee or other arrangement).

3.4. NWB community During the process of developing our NWB data conversion pipeline, we had several opportunities to interact with the NWB development team. Some of these ways were the NWB/DANDI Slack for quick questions, GitHub issues for a technical question or bug, GitHub discussions for entry level questions, remote meetings with the NWB team for more in-depth substantial guidance, and organized events (hackathons, user days, data re-hack) to meet others from the community and learn about the progress of the ecosystem. In general, our interactions with the NWB community were friendly, helpful, and responsive. For example, our questions on Slack usually received responses within the day. From our observation, this was also true for questions posed by other users.

As described in Section 5, we decided to design our own NWB extensions, which was technically challenging. Communication and assistance from the NWB team was very valuable in our design and implementation. Occasionally there were also helpful examples in GitHub issues or discussions on GitHub and Slack.

That said, many of these resources and communication channels are more familiar to computational scientists and software developers. The official documentation sometimes could be overwhelming to navigate (see, e.g. [43]), increasing a typical user's need to find and access these discussions scattered around many channels. It could have been helpful to have a centralized, searchable resource that aggregated and archived these different issues and discussions across different forums, as a complement to the official documentation.

3.5. Neuroscience community The advent of the open science movement, in parallel with standards development, has increased access to software tools and data that until recently was generally limited to high resource institutions. For example, the Allen Brain Institute released an SDK that simplifies retrieval of and interaction with extensive collections of NWB standardized data recorded with cutting edge electrophysiology and imaging tools. Such initiatives greatly expand opportunities to reuse data in education [44–46], basic research [47], and bench-marking of new computational models [48].

However, given differences in cultures, priorities, resources, and incentives across different labs and institutions, adoption of NWB, and of open science practices more generally, remains challenging. Institutional policies like the recently updated NIH Data Management Policy [49] add new expectations for researchers, but without creating meaningful recognition and training to support and encourage changes in their practice. Individual institutions also have historically provided minimal support for adoption of data management best practices. We advocate for better funding for standardization as an essential practice in science in general, and particularly for NWB adoption. Some of this support could include partnerships with public resources such as `nwb4edu` [44].

4. Building our NWB-based data conversion pipeline: Experiences, Challenges, and Lessons Learned

4.1. How to organize data into a standard format There have been many efforts at standardization of neuro-

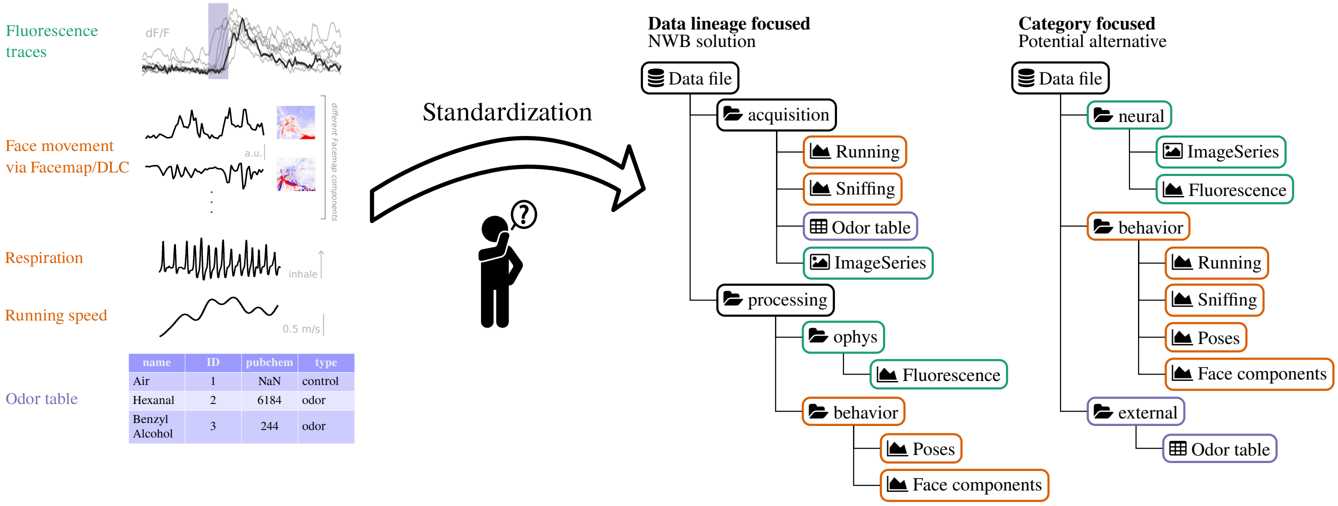


Fig. 2. The issue of data standardization. This figure illustrates the issue faced by researchers, who may record different types of data. The data may be multimodal, e.g. timeseries recorded from sensors, ROIs (Regions Of Interest) segmented from images or videos, behavioral markers tracking limbs on videos, tables tracking events, etc. This data may also be scattered in different files and formats. Researchers may then want to organize it in a unified way to make analysis and sharing easier. Two possible strategies to organize this data are shown: the first one is the strategy chosen by the NWB team, which organizes the data from a data lineage standpoint, the second one is an alternative strategy which organizes the data from a categorical standpoint. We use the following color scheme: green for neural activity, orange for animal behavior and purple for external variables (e.g. stimulus).

science data. Neurodata Without Borders (NWB) started as a pilot project to standardize neurophysiology data [21], which then matured into NWB:N version 2.0 (NWB:N 2.0) [50].

However, NWB is not really a file format. The substantive outcome of the NWB development effort was an "ontology" that encapsulates the logical structure of neuroscience data at a high level, and schemas to translate these conceptual objects into precise computational objects. Unlike saving an image in JPEG or a document in PDF, to use NWB researchers must make a number of choices specific to their data, with both technical and conceptual implications.

Fig. 2 illustrates questions faced by researchers who may record multi-modal data scattered across different files and formats. The resulting data need to be organized, unified, and aligned in order to support analysis and collaboration. There can be different strategies to standardize this data, for example from a data lineage standpoint (the choice of the NWB team, Fig. 2, middle) or from a categorical standpoint (Fig. 2, right).

Our files mostly follow the default NWB internal structure for optical physiology, though we made our own extension to handle odor data (see Section 5.3), and argue researchers could benefit from alternative structures, perhaps using aliases or tags, that allow them to interact with their data files following categorical or other organization (see Section 7.2.4).

4.2. When to create and use the standardized format

Few experimental acquisition systems produce NWB files natively, so use of the standard requires researchers to choose a process and time to convert to NWB from some mixture of other data files. One strategy is to convert at the end of a project, perhaps to upload to a repository for sharing. This choice minimizes disruption to existing research workflows and preserves flexibility for intermediate analyses. However, this strategy may reduce reproducibility, as analysis is done on different files than are eventually shared. Also, shared code needs to be refactored at time of publication to account for these file differences.

Alternatively, conversion could occur prior to internal use. In the pipeline illustrated in Fig. 3, conversion happens between preprocessing (using Suite2p and DeepLabCut) and analysis. Regardless of standardization, researchers typically reformat data before analysis, for example to compile information from "raw" files into a convenient single data array or table. The key aspect of standardization is that the output format carries restrictions that generalize the particular dataset with common practice in the field. If data is converted early, then archival repositories can be used also as backups, possibly including data version control. Moreover, shared code does not need substantial rewriting at time of publication. However, if there is not already a robust conversion pipeline in place, this strategy introduces additional effort prior to progress of the scientific aims.

4.3. Our experience with metadata capture Metadata can be defined as "data about data", for example, information about animal subjects (e.g. weight, sex, genetic line, age, whether naive or trained), recording sessions (e.g. date, task type, experimenter name, manufacturer and model of hardware), stimuli (e.g. chemical names, concentrations, frequency of audio tones), supplemental text descriptions, and/or parameters used in data processing. Generally, metadata can aid in quality control, communicate contextual information to future users, and support cross-analyses of multiple data sets. Its use can extend beyond the lifetime of a project, including archiving, sharing, and re-use.

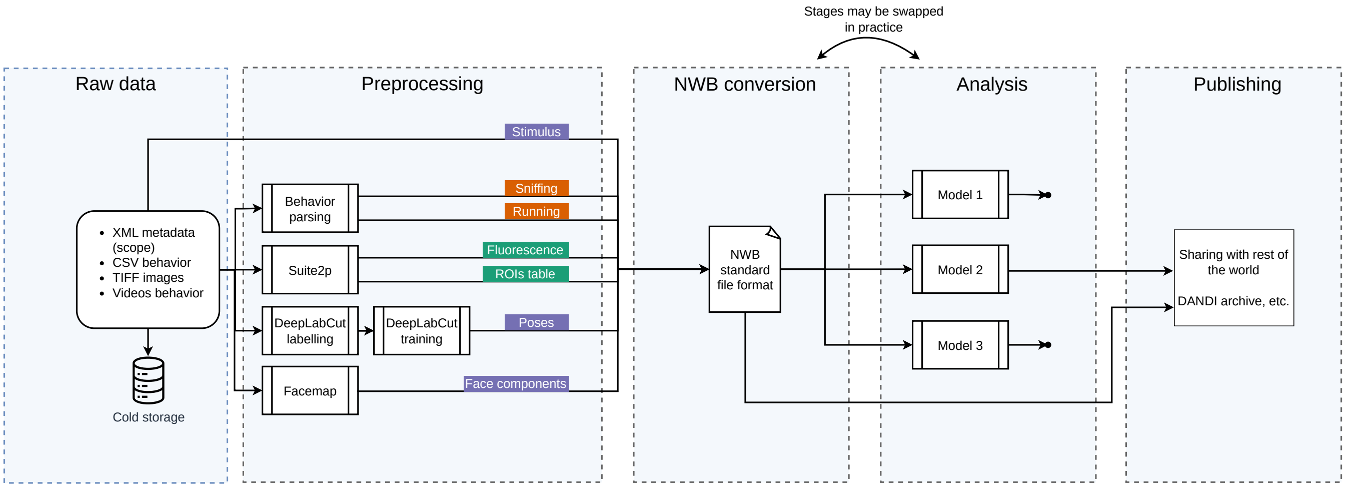


Fig. 3. Our data pipeline. In this pipeline, the data flows in fives different stages: It starts from the raw data acquired during the experiment, which is rarely used and get saved in cold storage.In the preprocessing stage, information is usually extracted from the raw data so that it is directly useable, this scattered data is then converted and standardized to NWB, the analysis happens on the standardized data, finally, the data is published with the rest of the scientific community. Ideally, standardization would happen before analysis for reproducibility purposes, but it often happens that they are swapped in practice as standardization may be thought only at publishing stage.

4.3.1. Quality of metadata capture A benefit of moving data to NWB is that it encourages systematic handling of metadata. To convert into NWB format, some types of metadata are required by the standard, while some are encouraged. Before moving to NWB, our metadata was scattered in several places. Now, all the relevant metadata is included in the NWB file, allowing consistent and easy access. This may help answer questions such as "What was the sex of animal X?", "What imaging frame rate was used in experiment Y?", or, when using our neurodata extension described in Section 5.3, "Which odor stimulus was used in trial Z?", without having to go back to the raw data or to the experiment notebook.

4.3.2. Challenges to metadata capture An obvious challenge to incorporating correct metadata in standardized files is that experimentalists do not always record metadata effectively. They may rapidly iterate an experimental design while piloting, and record only "core" data for preliminary analyses, with a fuzzy boundary between these initial pilots and subsequent "real" data collection. Moreover, metadata often takes unusual effort to document. Acquisition software may not support metadata capture at all. For example, mouse dates of birth or ages are often not included in data files produced during an experiment, yet at least one of these values is needed for NWB creation. Sometimes tools set incorrect metadata as a default; for example, we found the NWB conversion function within Suite2p defaulted to setting area of recording to be "V1" (see [51]). Also, there is not always a clear purpose to recording metadata that goes beyond the key variables in the original study design. Under the time pressure of the experiment, researchers may be induced either to use non-informative defaults or to enter random metadata to get underway.

This issue is exacerbated by a lack of accepted community standards of how to document for some types of metadata. For instance, in olfaction research, there is not yet consensus on how to document odor stimuli (though see [52], and Sections 4.10 and 5.3).

More generally, metadata capture is needed not only during acquisition conversion but also during preprocessing, analysis, and file conversion stages. For example, fluorescence is typically normalized ("dF/F"), but there is wide variation in how that normalization is performed. The choices of normalization parameters should also be captured as metadata.

4.3.3. Working with acquisition devices and software In our labs, research software engineers assist data conversion in part by working with researchers, equipment vendors, and others to determine what metadata is needed and how best to capture it.

Some commercial vendors put metadata in dedicated files (e.g., Bruker Microscope XML or ENV files) while others integrate metadata into the same files as core data (e.g., Inscopix Miniscope). However, some proprietary vendor files are poorly documented (and questions stayed unresolved after contacting support), such that we have had to reverse-engineer files and make educated guesses as to the information in them. For example, some things we had to independently infer from Bruker XML files were where frame rates are recorded, what physical units different fields have, and what the reference frame coordinates are. Our inferences relied on field names, and were incomplete and possibly in error. More importantly, certain metadata can change the algorithm used to parse a file; for example, a flag indicating whether an experiment has multi-plane imaging affects the correct way to extract timestamps from the XML file. NeuroConv, the conversion tool from NWB developers (see Section 4.12), is working to integrate Bruker metadata [53, 54], and we hope this support improves over time.

Open source tools typically fill a space between commercial vendors and in-lab custom development. Some of these tools lack an ability to input metadata. An example is ArControl [55], which is an experiment control platform used with general purpose microcontrollers to present stimuli and record behaviors. There is a project to convert its output into NWB format [56], but (as of this writing) still requiring *post* *hoc* metadata injection [57].

We also develop custom scripts ourselves that generate CSV-like files on microcontrollers. This approach would ideally include informative headers, for example to give each data column an informative name, a plain text description, physical units, a data type, and possibly other metadata. We find this step introduces friction and an increased chance of errors, especially as experimental designs change and researchers or software engineers need to keep code updated and documented. For now, metadata is often documented after acquisition. In an alternative approach, we implemented custom widgets in Jupyter notebooks used for data acquisition, that allow experimenters to write in odor names. The notebook then saves the names in a YAML file along with separate core data files, and all files are integrated into an NWB file in a later conversion process. The widget was tedious to develop, but substantially improved the quality of metadata capture for odors at the time of the experiment.

4.4. Where should raw data and supplemental information be stored? Researchers may want to store raw data in their NWB dataset. In our case, the raw data may contain calcium imaging TIFF stacks or behavior video recordings, both of which tend to be large. For example, a typical calcium imaging session in our lab generates a video of size around 40 GB, with associated behavioral videos around 3 GB. There has long been a question of what to do with videos [58, 59], contrasted with the much smaller data derived from them in pre-processing. Should raw videos be included in NWB files? If yes, how? If not, how should videos be handled when publishing to a repository [36]?

The NWB team discourages writing videos in lossy compressed formats within NWB files. The main reason is an inability to decode the video without first copying the data to a standard file type (e.g. MP3) on the user's computer; moreover, if the appropriate codec is not available, even a copied video would be unreadable. The preferred solution is to include videos in NWB files as an `ImageSeries` that has an external file reference (a relative path to, say, an MP4 file), see [60] as an example. This solution also allows adding videos in published datasets on DANDI [61].

Often researchers may want to share explanatory content such as videos of experimental setups or down-sampled videos of calcium imaging registration aligned to behavior recording. Only a subset of recording sessions may have such associated content. A solution could be similar storing raw data as external file references as described above, clearly labelled for demonstrative purposes to avoid confusion.

4.5. How should different data types be stored? In NWB, neurodata types refer to different modalities of data and metadata, for example `DfOverF`, `PupilTracking`, or `SpikeEventSeries`. Each type has specific rules to fit different use cases. If data belongs to a standard neurodata type, there are usually clear examples and guidelines about where and how to store it in an NWB file. When it does not, non-trivial choices may be required, and variation across labs, each implementing their own conventions, may impact general reusability.

For each data source to be integrated into an NWB file, users must answer a number of questions. Can the data be fit in a standard neurodata type? What metadata should be associated with it? Would an extension (see Section 5) add a more appropriate datatype? Does such extension exist? If not, is it worth the effort to develop one? Where in the file hierarchy should the data be stored (see Fig. 2)? Should it be saved in a separate container or combined with other "similar" data?

4.5.1. Cell type tagging As an example of how small experimental variations can lead to non-trivial design choices in NWB files, we describe an experiment in the Fleischmann lab involving two color imaging of red (tdTomato) labelled cells in parallel with green (gCaMP) functional imaging. After using Suite2p for cell segmentation, the researcher classified each cell as expressing or not expressing the red fluorophore, producing a table of ROI (cell) indices, boolean values for whether a cell is red, and auxiliary data about the classification (average pixel intensity and a quality metric).

There are three levels of detail one might choose to keep in an NWB file (in addition to the functional imaging contained in a standard datatype): as the full table, as only the boolean array, or as an array of indices of red cells. The last choice is the most compact, but does not preserve the auxillary information that might be useful for quality control and reproducibility. Similarly, parameters of the classifier itself (e.g. intensity thresholds) should likely be saved as well. The choice of what information to retain both suggests and is constrained by what datatypes are available, or whether we would need to develop an extension (see Section 5). And a further decision is where to save the data in the file hierarcy (Fig. 2): as pre-processed data or an analysis result?

There is obvious value to saving the classification in the same place that stored the segmentation table from Suite2p output, essentially by adding more columns to that table. However, since the classification is not available at the time of Suite2p segmentation, and updating existing objects in the Suite2p NWB file was problematic (see Section 4.7), we resorted to placing the classification table in another module called `cell_tag`. Given that the table came from Suite2p, whose outputs are in `processing`, we were unsure whether `cell_tag` should be considered `processing` or `analysis` in terms of lineage. However, in terms of usage, the tagging is not a useful result by itself, but is combined with the calcium dependent activity. Hence, we decided to consider the table as processed data needed for analysis, and save it in `processing`.

4.5.2. Breathing As a second example, the Datta lab records breathing signals with a temperature sensor implanted in the nose. An Arduino captures the signal, which is written into a CSV file in real-time. We developed a processing pipeline to clean and parse the breathing signal into individual breaths, and store the resulting data in an NWB file. There were a number of challenges along the way that highlight some limitations of the current NWB implementation.

Scipy's `signal.find_peaks` function was the core of the breath processing pipeline; good results relied on choosing correct parameters to find true breaths while ignoring noise in the data. Sometimes we would update the defaults of those parameters based on new analyses, and it would have been helpful to traverse old files programmatically and update them. As it was, many key parameters ended up stored in the "description" section of the relevant `TimeSeries`, which may not be an obvious location to those looking at the data for the first time.

Also, there were a number of options for how to store information about each breath, which were difficult to differentiate ahead of time. It would have been ideal to choose one rationally (e.g., based on efficiency of storage or common practice), but in the end our decision was purely pragmatic. We first considered a tabular format like the `TimeIntervals` table, but adding data to the `TimeIntervals` table proved to be buggy [62]. Then we considered an `IntervalSeries`, which would allow labeling onsets and offsets of inhales and exhales and convey the "interval" aspect of the data, but this did not lend itself to storing scalar descriptors for each breath, since the datatype stores only timestamps and not values. Finally, we settled on a simple solution: a `BehavioralTimeSeries`, containing many `TimeSeries` of length `number_of_breaths`. For example, inhale onset times, amplitudes, and peak flow rates each got their own `TimeSeries`. Inhales and exhales were paired in the pre-processing stage, and the `TimeSeries` that describe the inhales and exhales have the same length, thus implicitly pairing each inhale/exhale pair. We chose to save the `BehavioralTimeSeries` interface, called "breaths", in the "processing" section of the NWB file.

4.6. Should one standardize data from intermediate analysis stages? Research analysis pipelines typically have multiple stages, such as pre-processing, statistical modeling, simulation, or any computation whose inputs are the outputs of a previous stage. Those stages may also branch out to test a family of models, or vary analysis parameters. The NWB standard is limited in its handling of analysis parameters, for example as tables of metadata. Should intermediate results be appended to a single NWB file containing the entire history of analysis, each as their own "data source"? Should each analysis be stored in its own NWB file? Should all but the final published analysis be discarded?

It seems to us that the stages where NWB is most useful are to integrate relatively stable pre-processed data, and to archive finalized data and analysis for publication. It can be challenging to store raw data, as discussed above, and iterative analyses quickly become unwieldy without a strong effort to programmatically integrate with existing workflows and data version control (such as DataLad [63]). NWB was not designed to compactly represent collections of results such as arise from parameter sweeping in an analysis. Similarly, NWB does not natively support tracking the partitioning of data (such as into "training" and "testing" subsets for cross validation) common to many analysis pipelines.

4.7. Editing and merging of NWB files Early in our transition to NWB adoption, we needed to combine an NWB file exported from Suite2p with another NWB file produced by our own data pipeline. This turned out to be surprisingly difficult. Indeed, according to the PyNWB documentation, adding to files is supported, but removal and modifying of existing data is not allowed. We therefore tried two approaches to do this. In the first, we read the existing NWB file produced by Suite2p, added the missing data, and exported to a new NWB file. In the second, we looped over containers, i.e. HDF5 groups, in the existing NWB file, and copied each of them into a new NWB file, together with the new data.

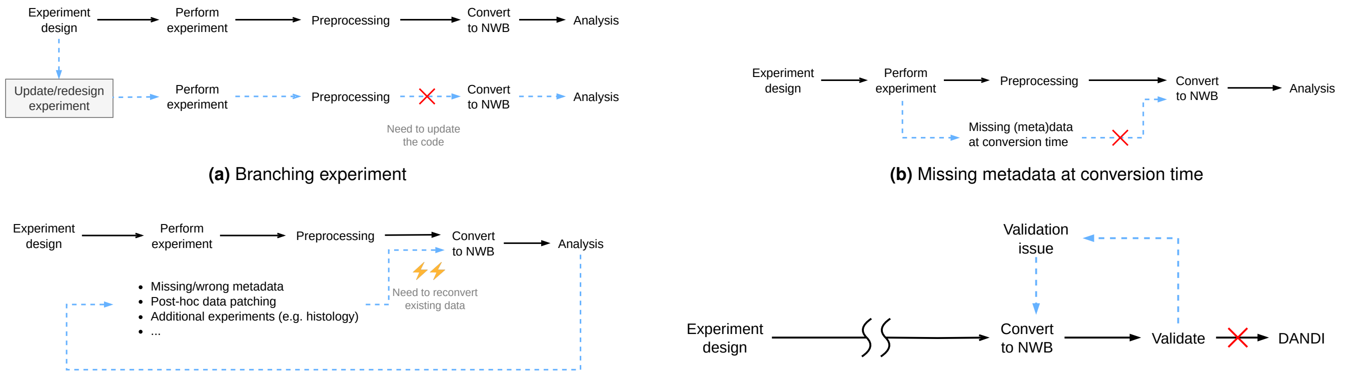
The first approach produced an NWB file that, due to a bug in the underlying packages (which has since been fixed), caused crashes while reading with PyNWB [64]. Because of a different bug, the second approach failed to create a new NWB file with the new containers [65]. These unexpected errors in what seemed like intuitive workflows were frustrating both for the delay in switching over to NWB, and the additional effort needed to diagnose the bugs and find workarounds.

There are still limitations in copying containers from one NWB file to another. But compared to when we started working on this project, it is now more straightforward to copy datasets, i.e. a data array and its timestamps, from one file to another, and to read an existing NWB file, modify it, and export the modified file to a new file. It is also possible to append data to a file, in the sense of creating new datasets. However, to our knowledge, the only way to update metadata in an NWB file is to read the content of the existing file, use the NWB API to create an object with the correct metadata, and then export to a new file. In general, we have found that users, especially those used to CSV and other general data formats, can be unpleasantly surprised by the rough spots in NWB file editing.

4.8. Pain points in the conversion workflow We encountered several pain points in our data conversion pipeline. One of the main pain points happens with branching experimental designs (Fig. 4a). Each time a design is updated, NWB conversion code may break and need to be updated. This is an issue especially early in project development, when many experimental details are undecided, but can continue far into a project's lifetime as researchers adjust their approach based on prior results.

Another pain point may arise when metadata is missing at conversion time (Fig. 4b). Researchers may be tempted to input nonsense values that need to be updated later, or the conversion may be blocked until the missing metadata is captured.

Sometimes, data in NWB files may need to be updated, e.g. to correct a previous entry, or to add data that becomes avail-



(a) Branching experiment

(b) Missing metadata at conversion time

(c) Data update/patching

(d) Validation issue before publishing

Fig. 4. Scenarios of pain points in the conversion workflow. This figure describes different scenarios adding burden to the research workflow. The red crosses represent a situation that breaks the existing workflow. The electric current symbol represents the location of a pain point. Fig. 4a shows that branching from the main experiment, i.e. a redesign or update of the experiment, may break the current conversion code to NWB. Fig. 4b shows that if some metadata is missing at conversion time, it may force the researcher to come back to the experiment, to the original data, or to the conversion code. Fig. 4c shows a scenario where existing NWB files need to be updated, e.g. when data from additional experiments like histology experiments become available, or when the NWB files have missing/wrong metadata, or if the NWB file has been found to have some data issues which need to be updated. Fig. 4d shows a validation issue before publishing the data to DANDI which may force the researcher to update their conversion code to NWB and reprocess their NWB files.

able later, such as histology (Fig. 4c). In this case, the pain point happens when the data conversion pipeline has to be run again on multiple already existing files. As discussed more in Section 6.1, a related issue can arise when sharing data in an archive such as DANDI [36]. Validation to DANDI is stricter than requirements to build a file with the python API (PyNWB), requiring conversion code updates even after conversion was locally "successful" (Fig. 4d).

4.9. Confusion with workflow ontology

The organization of the NWB standard is structured with data workflow stages at the top of the hierarchy: `acquisition` (usually raw), `processing`, and `analysis` (see Fig. 2). While in theory preserving some element of data lineage, the semantics in practice are not always clear or observed, and can cause confusion when creating and using NWB files.

For example, should raw behavior time series acquired from microcontrollers be in `acquisition`, a module called `behavior` in `acquisition`, or in the same `behavior` module in `processing` that is often used to store post-experiment processing such as DeepLabCut pose estimation? From a data lineage point of view, it should be stored in `acquisition`. But from an analysis point of view, doing so spreads multiple fragments of behavior-related data across multiple hierarchical levels and modules.

Returning to the example in Section 4.5 of two-color classification of cells, one solution is to save cell type with the Suite2p segmentation table. However, since the classification is not available at the time of Suite2p segmentation, and updating HDF5 objects was buggy, we resorted to placing the classification table in another module called `cell_tag`. As the table was created with information from Suite2p, whose outputs are in `processing`, we were unclear whether it should be considered `processing` or `analysis` in terms of lineage. However, in terms of usage, the table alone does not constitute useful results, but must be combined with calcium dependent fluorescence time series. Hence, we decided to consider the table as processed data needed for analysis, and save it in `processing`.

4.10. Lack of standard language

In general, we have experienced a lack of common language to describe processing applied to data, which impacts how such processes are documented in the NWB format. For instance, there are currently many ways to normalize fluorescence data. Methods used to obtain so-called dF/F_0 can differ in parameter choices or the algorithm itself (e.g. global z-scoring, quantile normalization, or running normalization with additional filtering). Some methods may attempt to compute $dF/$noise instead (e.g. Inscopix CNMFE [66]). Often these choices are not apparent in publications and require careful inspection of code, if provided. Such nuances may affect how the data are used, the assumptions of tools that analyze such data, and efforts to replicate analyses.

As another example, we needed to save metadata about odor stimuli, which led us to create our own extension (discussed more in Section 5.3). We were not aware at the time that our choices for metadata representation in the extension overlaped with an emerging effort to describe odor stimuli called `pyrfume` [52]. Both efforts involve technical components such as designing the computational instantiation, while also needing substantial and ongoing input from researchers to decide how to capture odor information in a concise but scientifically useful manner.

4.11. Timeliness of code contribution acceptance

We discovered Suite2p was dropping data from a second microscope channel in its NWB file output. The issue was that the NWB export function had been developed for only one microscope channel. Fig. 5 shows the timeline of the issue until a fix was released. While fixing the issue internally took around two months, it took around five months (including time for us to complete a GitHub "pull request") for the solution to be available to the Suite2p community. This is a long turnaround for what we considered to be a critical error, impacting all multicolor imaging analysis. We stress that we appreciate the Suite2p team's review and acceptance of our code contribution. However, this experience illustrates a general problem for research software development in the open source community; researchers maintaining software may not have the bandwidth to address every issue or feature request in as timely a fashion as desired.

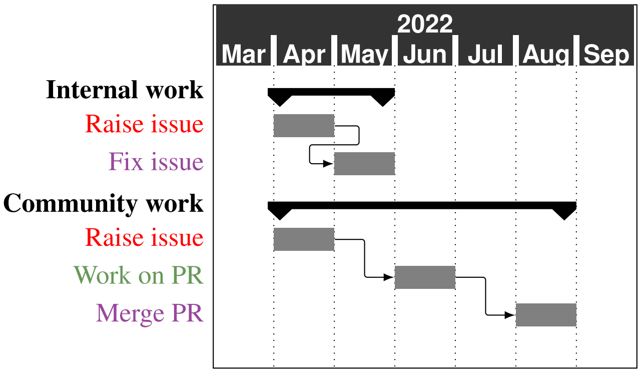


Fig. 5. Example of a broader community issue resolution timeline. This figure illustrates the time that it took to fix the issue internally (i.e. two months), compared to the time it took to fix the issue for the broader community (i.e. five months).

4.12. Off the shelf NWB conversion Some friction during adoption of NWB can arise from the level of technical skill needed to be able to convert one's data. When we started the process of adopting NWB, the options available were either to learn how to write our own data conversion pipeline, or hire a consultant to do the technical work. In the few years since, the NWB ecosystem has rapidly evolved. More recently introduced tools miss some areas of need (e.g. currently unsupported proprietary formats like Inscopix, or Suite2p output with multiple channels) , but they solve many popular use cases.

NeuroConv [53] is a rapidly advancing Python package from core NWB developers to make it easier to convert from a variety of common neuroscience data formats. It is a flexible low-code solution for use in one-off conversion or as part of a lab pipeline. One benefit of NeuroConv is that it includes utilities to get metadata from proprietary formats with minimal effort. Additionally, it can combine files from multiple data sources with functionality to align timestamps, and contains utilities for file path inference to aid batch-conversion based on user-defined data organization. Coupled with the development of the NWB Graphical User Interface for Data Entry (NWB GUIDE) [67], which uses NeuroConv as a back-end, NWB is considerably more accessible to newcomers than it was at the time we began our adoption.

These recent changes highlight a risk to early adopters of any standard, that one may build features from scratch that quickly become obsolete after further developments from the community. If we started this project today, we would leverage these community projects, developing less custom code and using existing features from more widely tested projects used by the entire NWB community.

4.13. An indirect benefit of using NWB is improved data awareness As a standard, NWB encourages good data practice. For example, each data array that is written in a file needs to have a timestamps vector attached to it, and ideally all the timestamps of the same NWB file would be on a common axis. This includes the acquisition timezone, meaning an NWB file can easily be analyzed in different parts of the world without risking timestamps collision.

In our case, standardization encouraged better timestamping with custom instruments and sensors like Arduino and Teensy boards. For example, before we developed our own data pipeline, one lab researcher manually specified intertrial intervals in their analysis code, as it was cumbersome to extract the (nearly constant) intervals from the recording system. Now they have access to the actual recorded timestamps for the inter-trial intervals and can catch and correct any system errors. Also, using NWB encouraged us to align timestamps across all data sources, simplifying downstream analysis work.

A general by-product of moving our lab to NWB is increased awareness regarding data management itself. Lab members have become more familiar with general principles such as FAIR [30] and emerging best practices. Although still harboring some skepticism of the direct usefulness to their research, lab members have become more welcoming to incorporating NWB into their workflows, and are supportive of the broader benefits, such as for data sharing.

5. Creating NWB extensions allows fitting domain specific use cases

An emerging standard with as broad a domain as NWB will naturally struggle to cover some applications, especially in less common experimental settings. Making the standard extensible creates a way for individual users or research groups to add functionality beyond what is created by the core developers. The NWB standard thus includes "neurodata extensions" to incorporate new data types. Extensions may be used individually, shared with the community, or, if the extension addresses a fundamental gap in NWB coverage, submitted for review to be added to the standard NWB data types. We have had some success using and creating NWB extensions to fit our specific research needs, though challenges and questions remain.

5.1. Existing Neurodata Extensions Before deciding to create an extension, researchers should check the Neurodata Extensions Catalog (NDX Catalog), a community led effort to create a central repository of contributions that, by design, arise from widely distributed effort [68]. The NDX catalog includes extensions that support diverse types of data such as TTL pulses [69], and popular acquisition systems such as miniscopes [70]. However, not all Neurodata Extensions are listed on the NDX Catalog, since anyone can create and post an extension on lab websites, GitHub, or other sites.

5.2. Lab-specific metadata One use case of NWB extensions is to record lab-specific metadata with greater flexibility than is supported in base NWB. We created `ndx-fleischmann-labmetadata` [71] to store additional detail on recorded brain areas, and descriptions of the experiment and animals. Within our general type of experiment we use many variations (Box 1), such as 1-photon or 2-photon calcium imaging, single or multicolor imaging, head-fixed or freely-moving animals, and passively presented or task-driven stimulation. NWB standard is missing fields to describe some of the complexity in these experiments; for example, we use multicolor imaging to retrograde label projections from the imaging site to distant brain regions, and there is no field to indicate this second (projection) area. Storing such additional experimental description as text in the top-level description field would be harder for quality control at time of entry, and less efficient to parse for queries at analysis time. With our extension, a subset of information ends up being repeated with standard locations in the NWB file; for example, imaging site is also stored under `ophys`, as suggested in the NWB documentation. However, we chose to centralize our metadata in one place, to make querying, analysis, and aggregation of multiple data files easier.

5.3. Odor stimulus metadata Another use case for extensions is to describe stimuli that do not fit within base NWB types. Our calcium imaging experiments use primarily odor stimuli, and some non-chemical stimuli such as sound. We are not aware of an extension to adequately describe these stimuli, and hence a year ago developed `ndx-odor-metadata` [72]. We characterize odor stimulus with standardized information automatically obtained from PubChem [73] using a PubChem CID (chemical IUPAC names, molecular formulas, and weights); dilution details such as concentration and solvent; metadata that are useful for analysis such as stimulus category (e.g. control or conditioned stimulus) and common chemical names; and identifiers to cross-reference with associated time series. The extension also allows non-odor stimuli to be described in plain text.

A major challenge with such extension development, although not an issue specific to NWB, is that there may not be community consensus or documentation to be used as starting points for extension design. For odor stimuli, it was not obvious what type and level of description would be necessary for both in-lab analysis and general reproducibility. Fleischmann lab RSEs used existing spreadsheets as starting examples, and learned only later that outside collaborators had independently created a package, `pyrfume` [52], for documentation of odorants. Future work could better harmonize these two efforts at stimulus metadata capture. More generally, the technical development of metadata capture can grow only in concert with the research community's understanding of what the standards for metadata ought to be.

5.4. Documentation for extension development For most labs, we expect extension development will be out of reach unless the lab has access to personnel with strong coding experience. A general challenge for us was that the available documentation could be confusing, and information was scattered across multiple sources, including documentation pages for PyNWB [74], HDMF [75, 76], NWB Overview [77], and NWB Schema [78], and also in GitHub issues or examples on Slack. It would have been helpful in particular to have a larger set of use cases, examples, and/or tutorials. We stress that the NWB development team was highly responsive through GitHub, Slack, and emails, and their help was very valuable for our development work. In the future, we hope such support could be complemented by more comprehensive documentation.

5.5. Social challenges in extension development One lesson learned from our experience is that creating the extension is only a technical part of a solution. Sustained engagement with researchers to choose, document, and record key information is the more fundamental requirement, especially if metadata standards motivating the extension are unsettled.

As a lab, we continue to refine what metadata we should track and how we should capture it. Some changes arise from variation in experiments conducted by different lab members. Some changes reflect interest in adding further types of information, such as water restriction details for experiments with behavioral training, as inspired by an International Brain Lab extension [79]. An extension may lower the technical barrier to metadata capture, but only if the extension is aligned with researchers' goals and practices, including changes over time.

A closely related challenge is that many metadata records must be captured *post hoc* instead of automatically during acquisition or pre-processing. Some acquisition systems lack features to enter metadata in machine-readable formats (necessary for software to correctly place that information in NWB files) during the experiments. Even where real time capture is possible, the systems may be cumbersome to use, leading researchers to avoid comprehensive entry and checking of metadata. We usually need to work with researchers to collect metadata records in machine-readable formats after experiments and preprocessing are completed, leading to increased work and greater risk of errors and missing information.

We also have felt a tension between building minimal extensions that serve immediate needs versus investing in a longer development project that may have greater generalizability. For example, our odor stimulus extension provides for single odorant but not mixed odor stimuli. Though we generally do not use multi-component odors, they are used by some of our close collaborators [80]. We also designed our extension to build on PubChem standardization, which presents difficulties when studying custom-made or undocumented natural odors [81]. These limitations in our current implementation may become impediments as neuroscience tends towards more natural and ethologically relevant behaviors [82]. However, surmounting these challenges will require substantial engagement from a broad section of the olfaction research community, before any technical contributions such as extensions can have a substantial impact.

5.6. Framework extensions An extension is built on top of another NWB object. This object can be one of the four minimally structured objects (Groups, Attributes, Links, Datasets) of the base NWB specification [83], but it is often better for an extension to build on a previously developed high level data type that already captures much of the structure of the information being added. In addition to making it easier to develop the extension without starting from scratch, such inheritance can promote greater consistency by keeping almost all data organization the same as a "common" data type, except for the particular items added by the new extension. For example, a new fluorescence imaging data type might add beam path parameters to an existing fluorescence imaging type, to provide for a scope that uses non-uniform laser scanning but otherwise collects standard data.

In cases where NWB is missing a more basic category of data, there is motivation to develop extensions intended to be used specifically as building blocks for other extensions. We refer to these types of building blocks as "framework extensions". In addition to facilitating development and serving as illustrative examples, framework extensions could add technical precision to discussions if a research community is working to converge to a consensus data standard.

For example, DeepLabCut and Facemap output time series of spatial locations of points on an animal's body. While these outputs can be stored generically as simply behavior, they are both instances of a more specific concept of "pose", and can be stored using the `ndx-pose` extension [84] (the DLC developers offer the `DLC2NWB` utility to ease conversion using this extension, but we are not aware of an analogous tool for Facemap).

An example framework extension that could have broad utility would store results from principal component analysis (PCA) (one of the authors, TP, participated in discussing this idea at a 2023 NWB Hackathon, but it is not yet implemented as far as we know). PCA is used widely as a simple data dimensionality reduction technique. There are several variants of PCA, such as jPCA used to find low dimensional structure in the activity of large neural ensembles [85]. Moreover, many analysis applications, including Facemap and MoSeq [86, 87], use PCA as a preprocessing step. A general PCA extension could serve as a useful framework to incorporate these different uses within a consistent NWB format. The framework extension would define component eigenvalues, eigenvectors, and projections of the original time series.

As another example, BEADL [88] and ArControl [55] model behaviors in a finite state machine framework. The extension `ndx-beadl` [89] is available for BEADL outputs, and it is possible to adapt the extension to handle ArControl output [56]. However, as finite state machines are an important class of models for analysis, there could be value in establishing a more general framework extension, for example called `ndx-finite-state`, from which extensions for these specific analysis packages would inherit.

5.7. Wishlist for NWB extensions Development, cataloging [68, 90] and updating extensions could be more streamlined.

First, researchers may develop software using different repository hosting (e.g. GitLab instead of GitHub). It could be more inclusive for the `ndx-template` extension template [91] to not explicitly assume GitHub as the code repository. The template might also take into account both the Python Package Index (PyPI) and Anaconda as potential package repositories.

Second, currently, to be added to the NDX Catalog, new extensions are submitted via Pull Requests for review on GitHub. Some seem to be approved instantly while others are either stale (e.g. `ndx-pose`), or took around 2 months to be approved (see Fig. 6). While the timeline for open source development is often highly variable, researchers and RSEs have to balance many priorities, and usually cannot dedicate much time to the approval process.

To simplify the review process, a bot could check critical requirements before asking for intervention from an NWB maintainer (taking some inspiration from the Conda-Forge community). For example, the bot could check if the package is already published on PyPI, if all the metadata fields in the `ndx-meta.yaml` file are filled in, and if all tests pass. Also, the bot could help for updating the extensions, say if the extension template or if some dependency has changed. Also, publishing to PyPI could be streamlined, for example by having a CI job in the `ndx-template` extension template [91] that supports automatic publishing to PyPI.

Additionally, we suggest adding some metadata to improve quality checks, centralization, and organization of extensions. To maintain quality control, the catalog could allow entries to be tagged to indicate whether an extension has been reviewed, similar to the distinction of pre-print from peer-reviewed publications. To tackle fragmentation of extensions and tools, it might be helpful to also allow optional specification of the type and lineage of each entry, e.g. whether it is built upon another extension, and if it is a template extension for demonstration purposes. Additionally, it is unclear whether lab-specific extensions (e.g. `ndx-ibl` for the International Brain Laboratory (IBL) and ours `ndx-fleischmann-lab`) are encouraged to be submitted. It could still be useful for them to be deposited with an indication of lab specificity, as they can be a valuable source of examples that other labs can adopt or mirror.

We hope to see depositing on the community catalog become more flexible and timely. A disadvantage is a potential reduction of quality control. However, more engagement, contribution, feedback, and discussion from the community is in general more likely to accelerate development of the standard. Extensions may serve as a starting point for such dis-

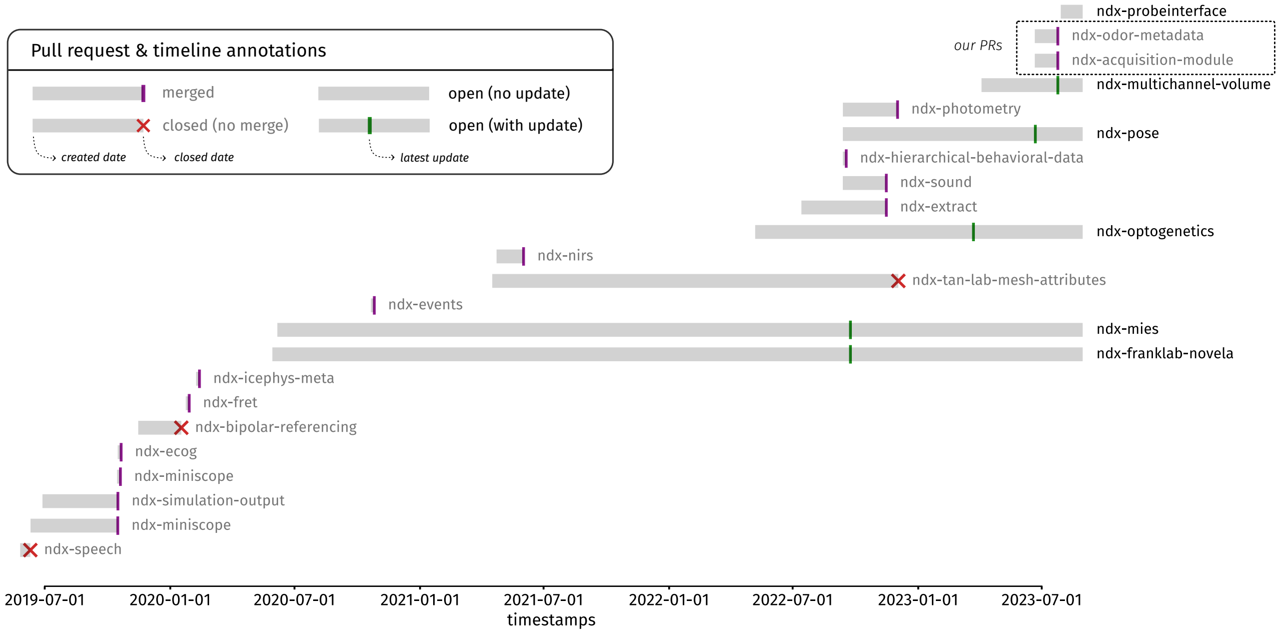


Fig. 6. Pull requests (PR) for publishing on the extension catalog may take a long time to be accepted. The data were obtained using GitHub API from `nwb-extensions/staged-extensions` repository, on 2023-07-30. Out of 23 extension requests, about 61 % (14/23) has been merged (bars ended with purple vertical sticks) and added to the catalog, while 13 % (3/23) is closed without being added to the catalog (bars ended with red crosses). The review times for finished PRs vary, ranging between within a day to less than 5 months for most of them, with the exception being 1.6 years for the closed request for `ndx-tan-lab-mesh-attributes`. About 26 % of the extension PRs (6/23) are still open, with 3 out of 6 being stale for more than a year. A notable one is `ndx-pose` for pose estimation extension (PR 31) is open for almost a year (Sep 2022). Note: any closed/merged PR finished within less than 5 days is artificially extended to be 5 days for visibility.

cussions, responding to community needs.

6. Considerations for sharing on DANDI

In this section, we look at the last step of the data conversion workflow: data has already been converted to NWB and the researcher wants to share the data on a public respoitory, for example to accompany a published paper. Here we look at DANDI [36], as the default solution recommended by the NWB team.

6.1. Potential surprises with data validation One possible source of friction is validating the data before being able to push to DANDI. DANDI enforces a set of rules that NWB files have to meet before upload and publication as a "dandiset" is allowed, intended to promote adherence to consistent metadata standards and ensure the FAIRness [30] of the archive. If files do not meet those requirements, researchers may need to (iteratively) redo their conversion with altered settings. This can be an unpleasant surprise, as one might have thought that having converted to NWB itself would be sufficient.

One solution could be to advertise and describe the NWBInspector tool [92], used to validate NWB files, in the documentation and tutorials on how to create NWB files. It would also be helpful to be able to run NWBInspector from PyNWB to check files and get feedback at the time of initial conversion. This solution may soon be implemented when using no-code tools like NWB-GUIDE [67] (see also Section 4.12), though it did not exist when we started our projects.

Another point of friction can arise if a dandiset has already been published but needs to be updated later, for example [93]. In our case, the validation rules changed after we first released the dandiset, and files that were already published became retroactively non-compliant. We had to go back to conversion from raw data. In general, if the cost to update a dandiset is too high, the risk is that researchers may decide not to correct stale or inaccurate information.

A potential solution would be to allow version-controlled inspection (Fig. 7). There could be at least two levels of NWBInspector passing. Files that pass the most recent NWBInspector can always be uploaded. But if some files already on DANDI get updated and fail the most recent inspection, they could still be uploadable given they passed the previous working version of NWBInspector. Similar to CI systems, logs of fail/pass versions could be attached to the archive for developers and others to inspect. This approach would allow for researchers to flexibly upload corrections and updates, while still being transparent about compliance status. Failures could be reported to the DANDI team, allowing them to work with researchers to follow up-to-date best practices.

6.2. Modification of file organization Another potential surprise is that the DANDI upload tool renames and reorganizes files into a "flatter" hierarchy. For example, one could have their NWB files organized by experiments with a nested directory structure organized by areas of recording, but DANDI refactors this structure to be organized only by subject directories, and moreover renames files by subject

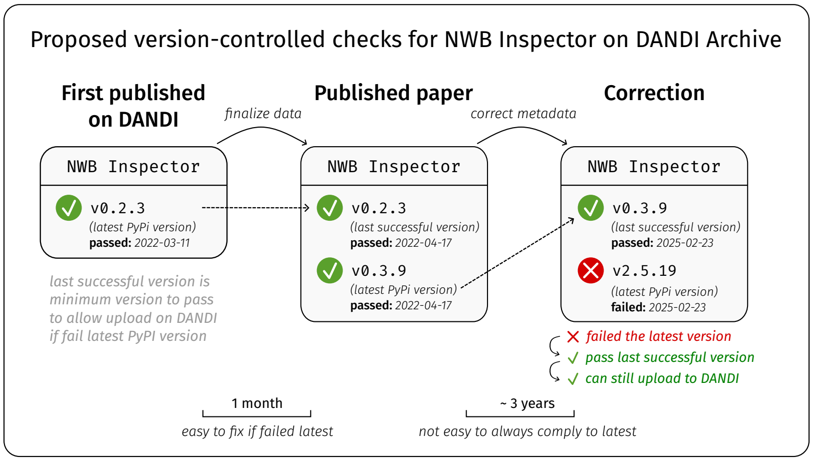

Proposed version-controlled checks for NWB Inspector on DANDI Archive
First published on DANDI
finalize data
Published paper
correct metadata
Correction
NWB Inspector
v0.2.3
(latest PyPi version)
passed: 2022-03-11
last successful version is minimum version to pass to allow upload on DANDI if fail latest PyPI version
NWB Inspector
v0.2.3
(last successful version)
passed: 2022-04-17
v0.3.9
(latest PyPi version)
passed: 2022-04-17
NWB Inspector
v0.3.9
(last successful version)
passed: 2025-02-23
v2.5.19
(latest PyPi version)
failed: 2025-02-23
✗ failed the latest version
✓ pass last successful version
✓ can still upload to DANDI
1 month
easy to fix if failed latest
~ 3 years
not easy to always comply to latest

Fig. 7. Illustration for proposed version-controlled checks for NWB Inspector when uploading to DANDI Archive. To be published on DANDI Archive, datasets should always be checked and pass the latest version of NWB Inspector (first and second boxes), to maintain compliance with best practices. When existing datasets need updating, they may fail the latest version, for example 3 years after publication, to correct metadata (third box on left). The proposed solution is to allow for checking against the *last working version* for existing datasets, in cases of non-compliance with the latest version. This still allows researchers to still disseminate updates and correction, while maintaining transparency for the community in terms of non-compliance. This can be allowed a limited number of times, and failures can also be reported to DANDI Archive maintainers.

name and data type. DANDI also modifies external file links stored inside each NWB file to stay consistent with these file changes.

Changing the file structure may break existing analysis pipelines based on the original paths. Thus, it may be useful to think about data archiving from the start of a project. In that case, publishing the data to DANDI from the beginning of the project, with occasional updates, would make the researcher aware of this reorganization and account for it in their own code. In addition to saving effort at publication time, such a workflow would enhance analysis reproducibility. However, the cost is some increased overhead while data collection is still occurring.

6.3. Alternatives to DANDI and general strategy with data repositories DANDI has strong restrictions on data file formats. While there is currently an exception on DANDI [60] that contains free-form source data (e.g. Python and NPY files), it is unclear whether this feature will officially be supported in the long run. Alternative repositories include Zenodo [32], Figshare [31], GIN G-Node [35], OSF [34], or university data storage, potentially with Globus endpoints [94]. These data archives can include NWB data and all related data such as raw data, pre-conversion data, analysis and summary data.

However, it may not always be feasible to centralize all data, and researchers might instead use distributed storage. Large source data, including raw and pre-conversion data, can be deposited on university storage solutions, with Globus endpoints if possible, to take advantage of university's less restrictive quotas, assuming these data would rarely be accessed, updated, or used after conversion. Converted NWB files can then be deposited on DANDI, on which researchers can benefit from specialized software tools, as well as DANDI Hub, a Jupyter Hub with free computing resources on Amazon Web Services (AWS). Lastly, along with code and documentation, researchers can continuously work on data with their analysis pipelines using solutions such as GIN G-Node, GitHub/GitLab with a DataLad [63] or DVC [95, 96] backend, to manage aggregated and analyzed data and code. This helps with version-controlled code and data, without the restrictions from DANDI Archive.

We note that if researchers decide to decentralize storage, they would need to manually link these different archives together, preferably with DOI numbers and in machine-readable metadata on these different providers. The outlined example strategy separate the three archives (e.g. university storage, DANDI Archive, and GIN G-Node) by an assumed increasing update frequency, i.e. raw data files are less frequently updated compared to NWB files, and NWB files less than files with analysis or modelling results. With distributed storage, especially if these assumptions do not apply, researchers would need to manually keep track and link the updates regularly.

7. Suggestions to streamline data reading and writing

7.1. Data exploration tool guidance The NWB ecosystem has many applications available for a researcher to quickly get a sense of what is inside an NWB file. As of writing, there are four general and 15 specialized data tools listed on the NWB Overview [77], and new tools continue to emerge. The number of active projects indicates a vibrant development community. However, new users may be overwhelmed by the choices, and not know how, except through brute force trials, to determine which tools are best for them. Moreover, consolidation around a few key applications could help channel valuable developer efforts into refining and improving existing tools, some of which still exhibit rough spots like freezing on large files or frequent crashes.

This situation is common in open source development ecosystems (for example, there are many partially redundant but not interchangeable python plotting packages). A difference here is that the NWB standard was created and continues to be maintained through a somewhat centralized development team, with an explicit agenda to be adopted as a ubiquitous standard for neurophysiology. There is thus a stronger case that innovation arising from widely dispersed development should be balanced by centralized advising over third party tools.

For example, primary NWB documentation could maintain a section with some (automatically scraped) metrics for each tool (e.g. number of GitHub stars, number of downloads on PyPI) next to accessible summaries of the features of each tool, and descriptions of who their target users are.

A more assertive approach would select recommended tools, on the basis of features, robustness (e.g. resolution of bugs, handling of large file sizes), and probable longevity. For data exploration, some natural candidates could be NWB-Widgets [97], which is also integrated with DANDI Hub, and NeuroSift [41], which is an interactive visualization tool that works directly in the user's browser. Both tools support streaming data from the DANDI Archive. Again, the goal would be to provide soft incentives that encourage contributors to focus on existing tool refinement, while still leaving space for new specialized projects in early development.

7.2. Data access pain points

7.2.1. Figuring out where data is We find new NWB users often struggle to find and access information, with confusion arising from where the information is in the internal hierarchy, or because the datatype of a particular object does not intuitively describe what it is. Many scientists look first for modules based on source of data (e.g. fluorescence, behavior, stimuli). But access under the NWB schema runs first through stage of processing (e.g. acquisition, pre-processing, analysis) and then descends through multiple levels of hierarchy to data source. That is, researchers may employ a mental sequence of *where is my behavior* (say) then *what processing has been applied*, which is the opposite ordering from what NWB currently uses (Fig. 2).

An outlier is that `stimulus` is at the top of the hierarchy, with `acquisition` and `processing`. However, stimulus time series sometimes need additional processing, for example, to transform raw digital outputs recorded by behavior control devices into a semantically useful tabular format. Should such stimuli be saved within `stimulus` (with processing stage indicated in `name` or `description` attributes), or in a module inside `processing`? Additionally, tables cannot be saved inside `stimulus`, and only limited metadata can be associated. It is recommended to use dedicated modules or objects designed to save metadata, for example `devices` for recording or `lab_metadata` for lab-specific metadata. This again runs into the potential issue of categorically similar objects being widely separated.

```
# Retrieve data with PyNWB API
# -----------------------------

# 1D array of timestamps
t = nwb_file.processing["behavior"]["interpd_500"
    ↪ ]["therm_highpassed"].timestamps[:]

# 1D array of data
therm = nwb_file.processing["behavior"]["
    ↪ interpd_500"]["therm_highpassed"].data[:]

# Retrieve data through a custom wrapper
# --------------------------------------

# Use wrapper to create an alias to the data
interpd = MyCustomWrapper(
    nwb_field="processing",
    path_to_interface=["behavior", "interpd_500"],
    nwb_file=nwb_file
)

# After one-time setup, simpler data retrieval
t, therm = interpd.get("therm_highpassed")
```

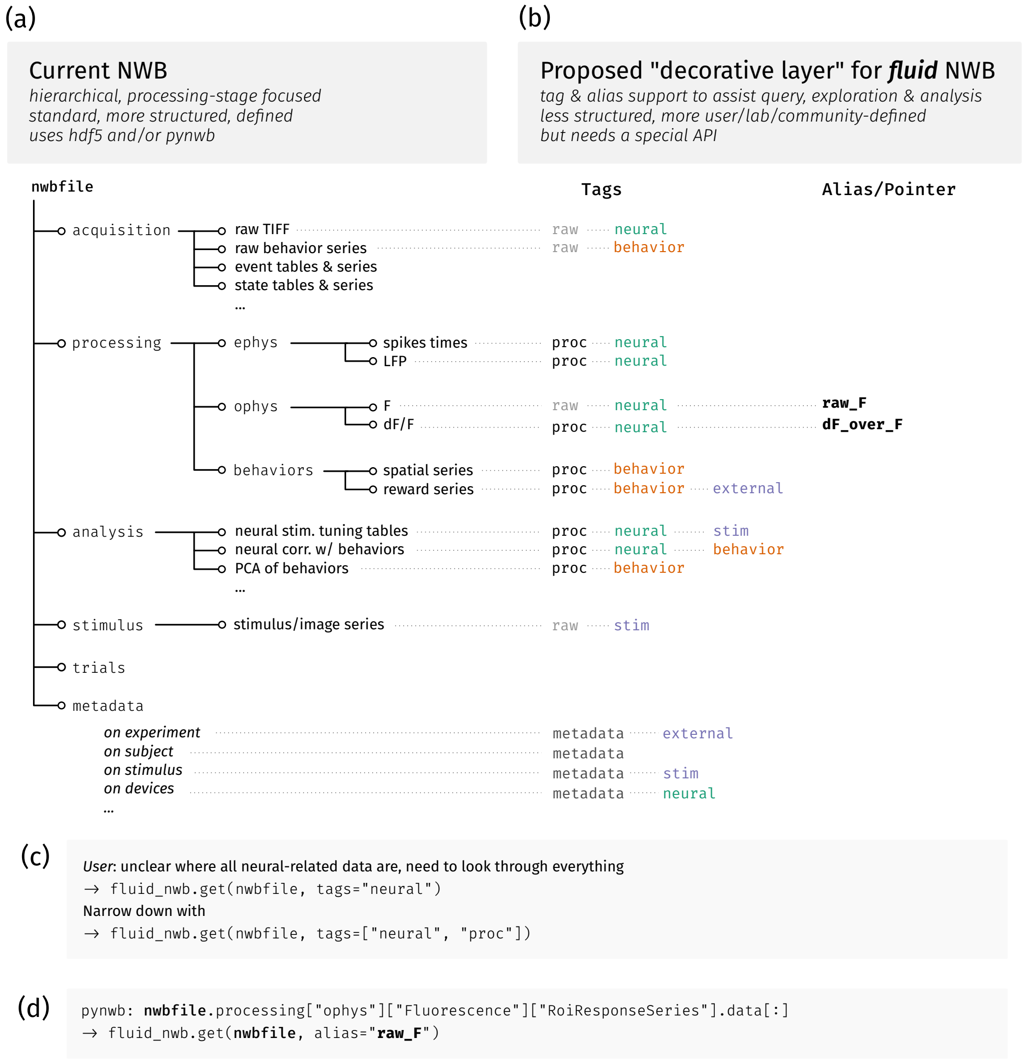
Listing 1. Data loading with NWB API and with a wrapper

7.2.2. Cumbersome syntax to extract data A challenge for new users that is parallel to understanding object locations is confusion over the addressing syntax, i.e. when to use dot syntax, `object1.object2`, or Python dictionary syntax, `object1["object2"]`. The syntactic variation derives from the structure of the HDF5 file specification and the NWB schema, both of which are generally unknown and opaque to users.

Two obvious alternative possibilities for API syntax would simply make one or the other access method universal (e.g. through a Python DataClass). Either choice would obscure the real differences between types of objects in the NWB implementation (e.g. a fluorescence object including metadata attributes, vs a numpy array just of the dF/F_0 values), but we are not convinced that most users benefit from having these differences encoded in syntax.

Another possibility that is both general and convenient for programmatic access would support universal reference via "path strings", such as `nwbfile[pathstr]` where `pathstr='object1/object2/object3'`.

7.2.3. Lab specific wrapper workaround In its current state, long hierarchies in NWB files (e.g. processing → behavior → interpolated → position → data) are slow to type and hard to remember, and tend to clutter code. A common method to hide complexity in an individual user's analysis code is to first create "wrappers" (Listing 1). For example, a wrapper may define simple "`get()`" methods that automatically skip parts of the object path, e.g. `data=nwb_wrapper.get('dFF0')`. Wrappers can also add convenience features, such as aggregating different time series into a single data frame, and, wrappers can be stored in dictionaries for easy looping over multiple files.



User: unclear where all neural-related data are, need to look through everything

```
→ fluid_nwb.get(nwbfile, tags="neural")
```

Narrow down with

```
→ fluid_nwb.get(nwbfile, tags=["neural", "proc"])
```

```
pynwb: nwbfile.processing["ophys"]["Fluorescence"]["RoiResponseSeries"].data[:]
→ fluid_nwb.get(nwbfile, alias="raw_F")
```

Fig. 8. A proposed design layer for the NWB standard to assist with the usage of data. The nature of the current NWB structure is hierarchical and tends to be organized by processing stages; panel (**a**) shows an example of this structure. Accessing relevant data requires knowledge of where they are, which may be multiple levels deep, for example, bottom box (**d**) with `PyNWB` to access raw fluorescence data. The proposed "decorative layer" allows for more "fluid" interaction with NWB via additional specifications in NWB objects, to assist querying, exploration and analysis with more user/lab/community's control and customization, without breaking the existing hierarchical NWB structure. Panel (**b**) illustrates examples of adding tags and aliases. Tags can be more specific, multi-faceted and customized to concepts of recording/analysis perspective oriented that users tend to look for (e.g. *neural, behavior, stim, external*), as well as higher level details such as processing stages (e.g. *raw, proc*). Aliases and/or pointers allow users to add names for objects that are most frequently accessed, or expected to be so. Taking advantage of this "decorative layer", users and developers may design a `fluid_nwb` API to interact with NWB files in a more flexible, less verbose, manner, for example with tags in box (**c**) and aliases in box (**d**).

On the other hand, wrappers may be complex to design and may introduce a maintenance burden if they aim to work across the usually wide range of experiments and data streams that arise even within a single lab. In practice, then, individual researchers often end up partially or completely rewriting similar helper code with each new project.

7.2.4. Suggestion for better data access: tags and aliases

A potential solution for better data access is a feature we call "fluid NWB" (Fig. 8), allowing for a list of tags for each object, including "flat" objects such as timeseries, tables, and modules. Users could add annotations and categories as they see fit, and specialized communities could evolve their own norms for "virtual" file organization, without confounding the underlying standard. Aliases, to our knowledge, are currently not possible, but the integration of such a feature may allow for users to have easier and quicker access, and could also aid documentation. Supporting custom tags for neurodata types is currently an open GitHub issue [98].

Tags and aliases would be a "decorative layer" on top of the NWB standard, allowing for more "fluid" data structures, which researchers and developers could exploit for usability and discoverability. However, in the absence of convergence on naming norms within a given research area, overlapping tags, complex tag formatting, and tag relations could proliferate to the point of no longer being useful. For example, should cardiac recordings (EKG), saccades, and arena locations all carry a common *behavior* tag? Should muscle recordings (EMG) be tagged both as *neural* and *behavior* in a brain-machine-interface (BMI) study? The added flexibility of an alias or tag system would produce the greatest benefit if complemented by a process to secure community consensus around tagging conventions.

8. Conclusion

Standardization is an essential component of modern data management, analysis, and sharing, and NWB has introduced a comprehensive and versatile data science ecosystem for neuroscience research. However, our experience suggests that implementation of NWB workflows at the level of individual labs or research collaborations still requires significant effort and commitment. Furthermore, given the rapid pace of technology development in neuroscience research, we expect that the development and implementation of adequate data science tools will continue to pose new challenges for some time. Solutions to these challenges will likely require a reorganization of neuroscience research to facilitate interdisciplinary collaborations, including additional institutional support not just for the creation of new tools, but also for their adoption by research labs at all levels of technical capability.

ACKNOWLEDGEMENTS

We would like to thank Max Seppo and Simon Daste for their input and experimental data they provided. We thank the Osmonauts (U19NS112953), Emilya Ventriglia and Rebecca Tripp for helpful comments on earlier drafts. Work in the Fleischmann and Datta labs was supported by NIH award U19NS112953. Work in the AF lab was also supported by NIH award R01DC017437, and the Robert J and Nancy D Carney Institute for Brain Science. Carney Institute computational resources used in this work were supported by the NIH Office of the Director award S10OD025181.

9. References

1. Steinmetz, N.A. et al. **Distributed coding of choice, action and engagement across the mouse brain**. *Nature*, 576(7786):266–273, December 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1787-x. Number: 7786 Publisher: Nature Publishing Group.
2. Stringer, C. et al. **Spontaneous behaviors drive multidimensional, brainwide activity**. *Science*, 364(6437):eaav7893, April 2019. doi: 10.1126/science.aav7893. Publisher: American Association for the Advancement of Science.
3. Mathis, A. et al. **DeepLabCut: markerless pose estimation of user-defined body parts with deep learning**. *Nature Neuroscience*, 21(9):1281–1289, September 2018. ISSN 1546-1726. doi: 10.1038/s41593-018-0209-y. Number: 9 Publisher: Nature Publishing Group.
4. Siegle, J.H. et al. **Survey of spiking in the mouse visual system reveals functional hierarchy**. *Nature*, 592(7852):86–92, April 2021. ISSN 1476-4687. doi: 10.1038/s41586-020-03171-x. Number: 7852 Publisher: Nature Publishing Group.
5. Koch, C. et al. **Next-generation brain observatories**. *Neuron*, 110(22):3661–3666, November 2022. ISSN 0896-6273. doi: 10.1016/j.neuron.2022.09.033.
6. Scheffer, L.K. et al. **A connectome and analysis of the adult Drosophila central brain**. *eLife*, 9:e57443, September 2020. ISSN 2050-084X. doi: 10.7554/eLife.57443. Publisher: eLife Sciences Publications, Ltd.
7. Loomba, S. et al. **Connectomic comparison of mouse and human cortex**. *Science*, 377(6602):eabo0924, June 2022. doi: 10.1126/science.abo0924. Publisher: American Association for the Advancement of Science.
8. Yao, Z. et al. **A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain**, March 2023. Pages: 2023.03.06.531121 Section: New Results, Accessed at https://www.biorxiv.org/content/10.1101/2023.03.06.531121v1, on 2023-07-09.
9. Langlieb, J. et al. **The cell type composition of the adult mouse brain revealed by single cell and spatial genomics**, March 2023. Pages: 2023.03.06.531307 Section: New Results, Accessed at https://www.biorxiv.org/content/10.1101/2023.03.06.531307v2, on 2023-07-09.
10. Braun, E. et al. **Comprehensive cell atlas of the first-trimester developing human brain**, October 2022. Pages: 2022.10.24.513487 Section: New Results, Accessed at https://www.biorxiv.org/content/10.1101/2022.10.24.513487v1, on 2023-07-09.
11. Callaway, E.M. et al. **A multimodal cell census and atlas of the mammalian primary motor cortex**. *Nature*, 598(7879):86–102, October 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03950-0. Number: 7879 Publisher: Nature Publishing Group.
12. Brose, K. **Global Neuroscience**. *Neuron*, 92(3):557–558, November 2016. ISSN 0896-6273. doi: 10.1016/j.neuron.2016.10.047.
13. Jorgenson, L.A. et al. **The BRAIN Initiative: developing technology to catalyse neuroscience discovery**. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1668):20140164, May 2015. doi: 10.1098/rstb.2014.0164. Publisher: Royal Society.
14. Koch, C. and Jones, A. **Big Science, Team Science, and Open Science for Neuroscience**. *Neuron*, 92(3):612–616, November 2016. ISSN 0896-6273. doi: 10.1016/j.neuron.2016.10.019.
15. Wareham, A. **New investigation reveals the number of authors named on research papers is increasing**, December 2016. Accessed at https://thepublicationplan.com/2016/12/13/new-investigation-reveals-the-number-of-authors-named-on-research-papers-is-increasing/, on 2023-07-09.
16. Cooke, N.J. and Hilton, M.L., editors. *Enhancing the Effectiveness of Team Science*. National Academies Press, Washington, D.C., July 2015. ISBN 978-0-309-31682-8. doi: 10.17226/19007.
17. Volkow, N.D. **Enhancing the Effectiveness of Team Science**, July 2022. Accessed at https://directorsblog.nih.gov/tag/enhancing-the-effectiveness-of-team-science/, on 2023-07-09.
18. Dallmeier-Tiessen, S. et al. **Enabling Sharing and Reuse of Scientific Data**. *New Review of Information Networking*, 19(1):16–43, January 2014. ISSN 1361-4576. doi: 10.1080/13614576.2014.883936. Publisher: Routledge _eprint: https://doi.org/10.1080/13614576.2014.883936.
19. Tenopir, C. et al. **Changes in Data Sharing and Data Reuse Practices and Perceptions among Scientists Worldwide**. *PLOS ONE*, 10(8):e0134826, August 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0134826. Publisher: Public Library of Science.
20. Pasquetto, I.V., Randles, B.M. and Borgman, C.L. **On the Reuse of Scientific Data**. *CODATA Data Science Journal*, 16(0):8, March 2017. ISSN 1683-1470. doi: 10.5334/dsj-2017-008. Number: 0 Publisher: Ubiquity Press.
21. Teeters, J.L. et al. **Neurodata Without Borders: Creating a Common Data Format for Neurophysiology**. *Neuron*, 88(4):629–634, November 2015. ISSN 0896-6273. doi: 10.1016/j.neuron.2015.10.025.
22. Gorgolewski, K.J. et al. **The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments**. *Scientific Data*, 3(1): 160044, June 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.44. Number: 1 Publisher: Nature Publishing Group.
23. Rübel, O. et al. **The Neurodata Without Borders ecosystem for neurophysiological data science**. *eLife*, 11:e78362, October 2022. ISSN 2050-084X. doi: 10.7554/eLife.78362. Publisher: eLife Sciences Publications, Ltd.
24. Holdgraf, C. et al. **iEEG-BIDS, extending the Brain Imaging Data Structure specification to human intracranial electrophysiology**. *Scientific Data*, 6(1):102, June 2019. ISSN 2052-4463. doi: 10.1038/s41597-019-0105-7.
25. Pachitariu, M. et al. **Suite2p: beyond 10,000 neurons with standard two-photon microscopy**. preprint, Neuroscience, June 2016. doi: 10.1101/061507.
26. **Inscopix**. Accessed at https://www.inscopix.com/, on 2023-08-23.
27. Syeda, A. et al. **Facemap: a framework for modeling neural activity based on orofacial tracking**, November 2022. Pages: 2022.11.03.515121 Section: New Results, Accessed at https://www.biorxiv.org/content/10.1101/2022.11.03.515121v1, on 2023-05-16.
28. Pierré, A. and Pham, T. **calimag**, October 2023. Language: eng, Accessed at https://zenodo.org/record/8411296, on 2023-10-05.
29. Carver, J.C. et al. **A survey of the state of the practice for research software in the United States**. *PeerJ Computer Science*, 8:e963, May 2022. ISSN 2376-5992. doi: 10.7717/peerj-cs.963. Publisher: PeerJ Inc.
30. Wilkinson, M.D. et al. **The FAIR Guiding Principles for scientific data management and stewardship**. *Scientific Data*, 3(1):160018, March 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.18. Number: 1 Publisher: Nature Publishing Group.
31. **Figshare**. Accessed at https://figshare.com/, on 2023-08-21.
32. **Zenodo**. Accessed at https://zenodo.org/, on 2023-08-21.
33. Foster, E.D. and Deardorff, A. **Open Science Framework (OSF)**. *Journal of the Medical Library Association*, 105(2), April 2017. ISSN 1558-9439, 1536-5050. doi: 10.5195/jmla.2017.88.
34. **Open Science Framework**. Accessed at https://osf.io/, on 2023-08-21.
35. **GIN: Modern Research Data Management for Neuroscience**. Accessed at https://gin.g-node.org/, on 2023-08-21.
36. Halchenko, Y. et al. **dandi/dandi-cli: 0.46.2**, September 2022. Accessed at https://zenodo.org/record/7041535, on 2023-06-21.
37. Kaiser, J. **NIH's BRAIN Initiative puts $500 million into creating most detailed ever human brain atlas**, September 2022. Accessed at https://www.science.org/content/article/nihs-brain-initiative-puts-dollar500-million-creating-detailed-ever-human-brain-atlas, on 2023-08-12.
38. **DANDI Archive General Policies v1.0.1**. Accessed at https://www.dandiarchive.org/handbook/about/policies/, on 2023-08-23.
39. **Zenodo General Policies v1.0**. Accessed at https://about.zenodo.org/policies/, on 2023-08-23.
40. Dichter, B. and Tauffer, L. **DANDI LLMs**. Accessed at https://github.com/catalystneuro/dandi_llms/, on 2023-08-21.
41. Dichter, B. and Magland, J. **neurosift**. Accessed at https://github.com/flatironinstitute/neurosift, on 2023-08-22.
42. Daste, S. **Two photon calcium imaging of mice piriform cortex under passive odor presentation**, September 2022. Accessed at https://dandiarchive.org/dandiset/000167/0.220928.1306, on 2023-06-19.
43. Saunders, J. **[Documentation]: Structure of docs considered harmful to a project that i love! · Issue #1482 · NeurodataWithoutBorders/pynwb**, May 2022. Accessed at https://github.com/NeurodataWithoutBorders/pynwb/issues/1482, on 2023-05-22.

44. Voytek, B., Juavinett, A. and Magdaleno-Garcia, V. **Teaching & Learning with NWB Datasets**, September 2020. Accessed at https://nwb4edu.github.io/, on 2023-08-16.
45. Van Viegen, T. et al. **Neuromatch Academy: Teaching Computational Neuroscience with Global Accessibility**. *Trends in Cognitive Sciences*, 25(7):535–538, July 2021. ISSN 13646613. doi: 10.1016/j.tics.2021.03.018.
46. **INCF Training space - Neurodata Without Borders: Neurophysiology (NWB:N)**. Accessed at https://training.incf.org/collection/neurodata-without-borders-neurophysiology-nwbn, on 2023-08-23.
47. Deitch, D., Rubin, A. and Ziv, Y. **Representational drift in the mouse visual cortex**. *Current Biology*, 31(19):4327–4339.e6, October 2021. ISSN 09609822. doi: 10.1016/j.cub.2021.07.062.
48. Schneider, A. et al. **Transcriptomic cell type structures in vivo neuronal activity across multiple timescales**. *Cell Reports*, 42(4):112318, April 2023. ISSN 22111247. doi: 10.1016/j.celrep.2023.112318.
49. of Intramural Research, N.O. **2023 NIH Data Management and Sharing Policy**, February 2023. Accessed at https://oir.nih.gov/sourcebook/intramural-program-oversight/intramural-data-sharing/2023-nih-data-management-sharing-policy, on 2023-05-22.
50. Rübel, O. et al. **NWB:N 2.0: An Accessible Data Standard for Neurophysiology**, January 2019. Pages: 523035 Section: New Results, Accessed at https://www.biorxiv.org/content/10.1101/523035v1, on 2023-07-06.
51. **Suite2p NWB function**. Accessed at https://github.com/MouseLand/suite2p/blob/118901ac15c6881502c65e011a46fbca16e7a52d/suite2p/io/nwb.py#L346C27-L346C27, on 2023-08-23.
52. Castro, J.B. et al. **Pyrfume: A Window to the World's Olfactory Data**. preprint, Neuroscience, September 2022. doi: 10.1101/2022.09.08.507170.
53. Baker, C. et al. **NeuroConv**, June 2023. original-date: 2022-07-19T16:49:38Z, Accessed at https://github.com/catalystneuro/neuroconv.git, on 2023-06-16.
54. Weigl, S. **add 'BrukerTiffImagingInterface' by weiglszonja · Pull Request #390 · catalystneuro/neuroconv**, May 2023. Accessed at https://github.com/catalystneuro/neuroconv/pull/390, on 2023-08-04.
55. Chen, X. and Li, H. **ArControl: An Arduino-Based Comprehensive Behavioral Platform with Real-Time Performance**. *Frontiers in Behavioral Neuroscience*, 11: 244, December 2017. ISSN 1662-5153. doi: 10.3389/fnbeh.2017.00244.
56. Chen, X. and Rubel, O. **ArControl-convert2-nwb**, 2023. Publication Title: GitHub, Accessed at https://github.com/chenxinfeng4/ArControl-convert2-nwb, on 2023-06-30.
57. Chen, X. **Issue comment 1416018374 of [Feature request] Conversion to NWB · Issue #2 · chenxinfeng4/ArControl**, February 2023. Publication Title: GitHub, Accessed at https://github.com/chenxinfeng4/ArControl/issues/2#issuecomment-1416018374, on 2023-06-30.
58. Alkan, G. **Need to store video · Issue #1647 · NeurodataWithoutBorders/pynwb**, February 2023. Accessed at https://github.com/NeurodataWithoutBorders/pynwb/issues/1647, on 2023-05-22.
59. Baker, C. **FAQ: Why shouldn't I write video data to an NWB file? · Issue #78 · NeurodataWithoutBorders/nwb-overview**, February 2023. Accessed at https://github.com/NeurodataWithoutBorders/nwb-overview/issues/78, on 2023-08-02.
60. Rodgers, C.C. **A detailed behavioral, videographic, and neural dataset on object recognition in mice**. *Scientific Data*, 9(1):620, October 2022. ISSN 2052-4463. doi: 10.1038/s41597-022-01728-1. Number: 1 Publisher: Nature Publishing Group.
61. Sharda, S. **External Links in NWB and DANDI**, March 2022. Accessed at https://www.dandiarchive.org/2022/03/03/external-links-organize.html, on 2023-08-02.
62. Pearl, J. **pandas to HDMF dynamic table · NeurodataWithoutBorders/helpdesk · Discussion #30**, May 2022. Accessed at https://github.com/NeurodataWithoutBorders/helpdesk/discussions/30, on 2023-10-02.
63. Halchenko, Y. et al. **DataLad: distributed system for joint management of code, data, and their relationship**. *Journal of Open Source Software*, 6(63):3262, July 2021. ISSN 2475-9066. doi: 10.21105/joss.03262.
64. Pierré, A. **RecursionError when reading an exported NWB file · Issue #1301 · NeurodataWithoutBorders/pynwb**, September 2020. Accessed at https://github.com/NeurodataWithoutBorders/pynwb/issues/1301, on 2023-06-12.
65. Pierré, A. **Unable to copy data containers from one NWB file to another already existing NWB file · Issue #1297 · NeurodataWithoutBorders/pynwb**, September 2020. Accessed at https://github.com/NeurodataWithoutBorders/pynwb/issues/1297, on 2023-06-12.
66. **Inscopix CNMF-E**, December 2021. Accessed at https://github.com/inscopix/inscopix-cnmfe, on 2023-06-30.
67. **NWB GUIDE**. Accessed at https://github.com/NeurodataWithoutBorders/nwb-guide/, on 2023-08-21.
68. Ruebel, O., Ly, R. and Dichter, B. **NDX Catalog**, May 2023. Accessed at https://nwb-extensions.github.io/, on 2023-06-19.
69. Ly, R. **ndx-events Extension for NWB**, March 2023. original-date: 2020-03-11T23:34:50Z, Accessed at https://github.com/rly/ndx-events, on 2023-06-20.
70. Dichter, B. **ndx-miniscope Extension for NWB**, January 2023. original-date: 2019-05-25T00:03:07Z, Accessed at https://github.com/catalystneuro/ndx-miniscope, on 2023-06-20.
71. Pham, T. **ndx-fleischmann-labmetadata Extension for NWB**, March 2023. Accessed at https://gitlab.com/fleischmann-lab/ndx/ndx-fleischmann-labmetadata.
72. Pham, T. **ndx-odor-metadata Extension for NWB**, March 2023. Accessed at https://gitlab.com/fleischmann-lab/ndx/ndx-odor-metadata.
73. Kim, S. et al. **PubChem 2023 update**. *Nucleic Acids Research*, 51(D1):D1373–D1380, January 2023. ISSN 0305-1048. doi: 10.1093/nar/gkac956.
74. **pynwb documentation**. Accessed at https://pynwb.readthedocs.io/en/stable/.
75. **hdmf documentation**. Accessed at https://hdmf.readthedocs.io/en/stable/.
76. Tritt, A.J. et al. **HDMF: Hierarchical Data Modeling Framework for Modern Science Data Standards**. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 165–179, December 2019. doi: 10.1109/BigData47090.2019.9005648.
77. **nwb-overview documentation**. Accessed at https://nwb-overview.readthedocs.io/en/latest/.
78. **nwb-schema documentation**. Accessed at https://nwb-schema.readthedocs.io/en/latest/.
79. Baker, C. and Sharda, S. **ndx-ibl Extension for NWB**, February 2023. Accessed at https://github.com/catalystneuro/ndx-ibl.
80. Wilson, C.D. et al. **A primacy code for odor identity**. *Nature Communications*, 8(1): 1477, November 2017. ISSN 2041-1723. doi: 10.1038/s41467-017-01432-4.
81. Li, B. et al. **From musk to body odor: Decoding olfaction through genetic variation**. *PLOS Genetics*, 18(2):e1009564, February 2022. ISSN 1553-7404. doi: 10.1371/journal.pgen.1009564.
82. Krakauer, J.W. et al. **Neuroscience Needs Behavior: Correcting a Reductionist Bias**. *Neuron*, 93(3):480–490, February 2017. ISSN 08966273. doi: 10.1016/j.neuron.2016.12.041.
83. Ruebel, O. et al. **The NWB Specification Language — Neurodata Without Borders Specification Language v2.1.0-beta documentation**, March 2020. Accessed at https://schema-language.readthedocs.io/en/latest/index.html, on 2023-09-18.
84. Ly, R. **ndx-pose Extension for NWB**. Accessed at https://github.com/rly/ndx-pose.
85. Churchland, M.M. et al. **Neural population dynamics during reaching**. *Nature*, 487 (7405):51–56, July 2012. ISSN 1476-4687. doi: 10.1038/nature11129. Number: 7405 Publisher: Nature Publishing Group.
86. **Moseq2-app**. Accessed at https://github.com/dattalab/moseq2-app, on 2023-08-23.
87. Wiltschko, A.B. et al. **Mapping Sub-Second Structure in Mouse Behavior**. *Neuron*, 88(6):1121–1135, December 2015. ISSN 08966273. doi: 10.1016/j.neuron.2015.11.031.
88. **BEADL**. Accessed at https://beadl.org/, on 2023-05-22.
89. Ly, R. **ndx-beadl Extension for NWB**. Accessed at https://github.com/rly/ndx-beadl/.
90. **NWB staged extensions**. Accessed at https://github.com/nwb-extensions/staged-extensions/, on 2023-06-30.
91. Ly, R. **ndx-template: A place to submit NWB Extensions for registration in the official NDX Catalog**, April 2019. Accessed at https://github.com/nwb-extensions/ndx-template.
92. Baker, C. and Dichter, B. **NWB Inspector — NWBInspector documentation**, February 2020. Accessed at https://nwbinspector.readthedocs.io/en/dev/, on 2023-07-06.
93. Pierré, A. **Update an old dandiset · Issue #98 · dandi/helpdesk**, June 2023. Accessed at https://github.com/dandi/helpdesk/issues/98, on 2023-07-07.
94. **Globus**. Accessed at https://www.globus.org/.
95. Skshetry. **DVC: Data Version Control - Git for Data & Models**, June 2023. Accessed at https://zenodo.org/record/3677553, on 2023-06-16.
96. Barrak, A., Eghan, E.E. and Adams, B. **On the Co-evolution of ML Pipelines and Source Code - Empirical Study of DVC Projects**. In *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 422–433, Honolulu, HI, USA, March 2021. IEEE. ISBN 978-1-72819-630-5. doi: 10.1109/SANER50967.2021.00046.
97. Dichter, B. and McCormick, M. **NeurodataWithoutBorders/nwbwidgets: Explore the hierarchical structure of NWB 2.0 files and visualize data with Jupyter widgets.**, August 2019. Accessed at https://github.com/NeurodataWithoutBorders/nwbwidgets, on 2023-08-22.
98. Ly, R. **Support custom tags to neurodata types #531**, November 2022. Accessed at https://github.com/NeurodataWithoutBorders/nwb-schema/issues/531.