Open Neurophysiology Environment Filename Convention

Kenneth D Harris
University College London

6 April 2022

Open Neurophysiology Environment Filename Convention

- These slides describe a standardized way to name and structure data files for neurophysiology or other scientific fields

- Organizing files this way will make it easy to share data when you need to do that

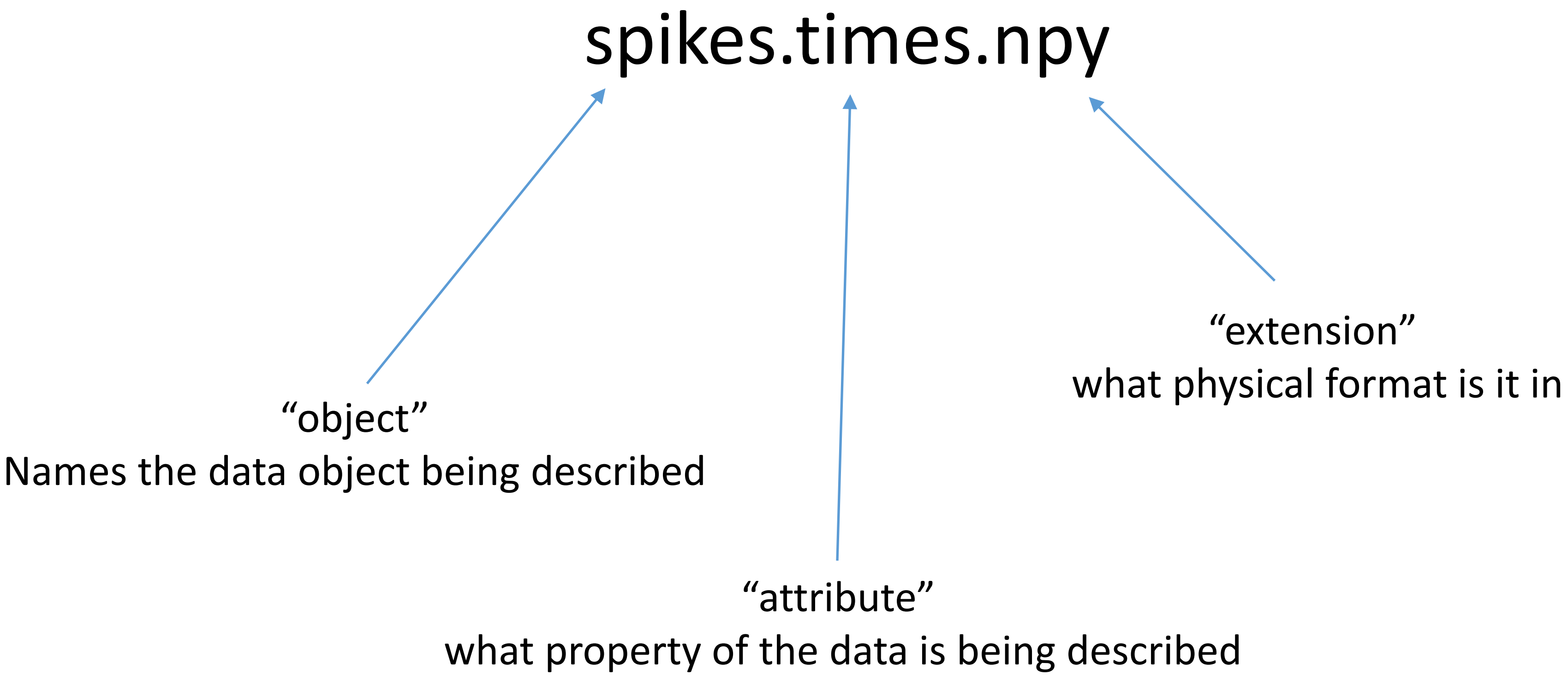
- It also helps keep track of your own data within your own lab

- Dveloped by the International Brain Lab but can be used without any other infrastructure

A filename convention – not a file format

- One directory per experiment, containing files that can be in multiple formats

- All files have 3-part names: object.attribute.extension



spikes.times.npy

“object”
Names the data object being described

“attribute”
what property of the data is being described

“extension”
what physical format is it in

Example filenames

Filename	Size	Contents
spikes.times.npy	nSpikes	Time of each spike
spikes.amps.npy	nSpikes	Amplitude of each spike
spikes.clusters.npy	nSpikes	Cluster assignment of each spike
clusters.waveforms.npy	nClusters x nChannels x nTimepoints	Mean waveform of each cluster
clusters.mlapv.npy	nClusters x 3	Allen CCF position of each cluster (ML, AD, DV coordinates)
clusters. brainLocationAcronyms_ccf_2017.txt	nClusters	Allen CCG brain location acronym assigned to each cluster

- All files with the same object name must have the same number of rows
- Any file format can be used provided it has a clear definition of "row" or leading dimension

Recommended file formats

- You can use anything where:
  - It is clear from the file extension alone how to load the file
  - It is clear what shape the array is
  - It is clear what the primary dimension (“rows”) corresponds to
- Recommended:
  - .npy: basic file format for storing numerical arrays.
    - Native to numpy/python; MATLAB access using https://github.com/kwikteam/npy-matlab
  - .tsv: tab-delimited string arrays
    - Better than .csv, which gets confused by commas in the strings
  - .mj2: movies
    - Each frame defines a “row”
- Not recommended:
  - Flat binary files: not clear what the array size or number of rows is
  - .csv: gets confused over commas in strings

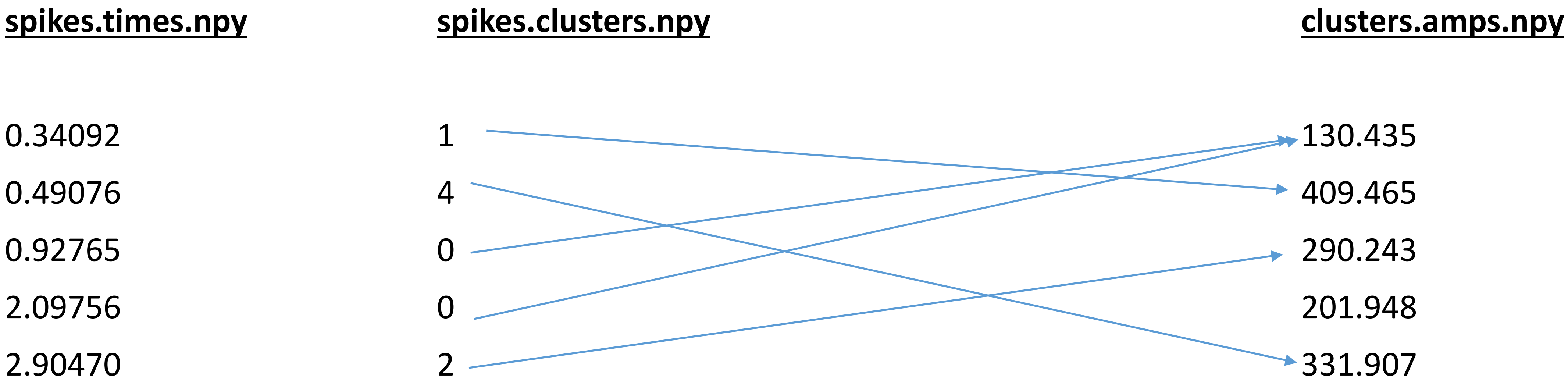
Standard file names

- Some file names can mean the same thing across projects
  - E.g. spikes.times, spikes.clusters, clusters.amps

- A list of standard dataset types defined by IBL are kept here. Use these for the object.attribute part of your file name if it matches your data. (Physical format and extension are up to you.)

- Many files you want to save won't match any of these, for example to describe your specific behavior task. When creating non-standard data sets, use a prefix with underscores:
  - _kdh_trials.stimulus_times.npy, _kdh_trials.movement_times.npy

Cross-references



spikes.times.npy	spikes.clusters.npy	clusters.amps.npy
0.34092	1	130.435
0.49076	4	409.465
0.92765	0	290.243
2.09756	0	201.948
2.90470	2	331.907

- When the attribute of one file matches the object of another, it defines a cross-reference
- Contains integers pointing to the row of the second file. **Always counting from 0**
- Lets you encode relationships like in a relational database

Times

- Any files whose attribute name is or ends with *times* define a time measured in seconds relative to experiment start.
  - spikes.times.npy, licks.times.npy, _kdh_trials.goCue_times.npy, _kdh_trials.response_times.npy, _kdh_trials.feedback_times.npy

- Any files whose attribute name is or ends with *intervals* defines a 2-column array giving the start and end of time intervals
  - _kdh_wheelMoves.intervals.npy, _kdh_trials.movement_intervals.npy, _kdh_trials.stimulus_intervals

- All timing information must be synchronized to a common timebase, in seconds relative to experiment start, before you save it.

Directory organization

- One directory per subject, containing one directory per experiment:

```
Hercules/2022-03-26/
Hercules/2022-03-27/
Hercules/2022-03-28/

Megara/2022-03-20/
Megara/2022-03-21/
Megara/2022-03-26/
```

Collections via subdirectories

- Sometimes you have multiple files containing the same data type

- For example recordings from multiple silicon probes

- Create subdirectory of experiment directory for each probe:
  - probe00/spikes.times.npy, probe00/spikes.clusters.npy, probe00/clusters.waveforms.npy, …
  - probe01/spikes.times.npy, probe01/spikes.clusters.npy, probe01/clusters.waveforms.npy, …

- Subdirectories are called “collections” – use them whenever you need to store the same file type multiple times
  - Cross-references refer to other files inside the collection

- Files in all collections must be synchronized to a common timebase: seconds relative to experiment start

Extra filename parts

- Sometimes you want to add additional information into the filename. You can have extra parts between the attribute and the extension. They don’t affect the naming conventions

spikes.times.XXXX.YYYY.ZZZZ.npy

- For example if you are worried about your files getting mixed up between directories, you could add the subject name and experiment date:

spikes.times.Hercules.2022-03-28.npy
spikes.clusters.Hercules.2022-03-28.npy
clusters.amps.Hercules.2022-03-28.npy

- Or you could add a UUID to make each file unique:

spikes.times.19232c05-946f-4ca6-a4cc-24c783fde3d2.npy

- The extra parts go between the attribute and the extension – that way you still see the object and attribute in a directory listing, and you can still double click on the file to open it.
- Cross-references, Collections, and other features work just as before

Avoid .csv files

- People should be able to load your files and understand what they contain just from the file names, without reading any documentation. Csv files make this hard.

- Csv files contain multiple comma-separated columns of strings or numbers. If the strings contain commas, the file becomes impossible to parse. Sometimes people put quotes around all strings to avoid this problem. Sometimes .csv files have header lines describing the columns. The downloader can't know what your conventions are without reading documentation.

- If you have numerical array data, better to use .npy files, which are also much smaller. If you want to make it human-readable, use a .tsv (tab separated) file with no header line. This is unambiguous.

- If you want to store a table of heterogeneous data (multiple attributes of multiple objects), split it into one file per attribute using the filename convention. If you really need to put a heterogenous data table in one file, use parquet (compressed, self-describing) or a .htsv file: tab-separated with one tab-separated header line that names each column.