**Data**:

1. We mostly use the OS\_Basic dataset. It has folders named as numbers each representing a rat. You can find the link to it in the #sleep\_scoring\_automated channel in the pinned messages.
2. Rats: 1, 3, 4, 6, 9, 11, 13
3. The OS stands for **O**bject **S**pace task. This task is to study cumulative memory formation. Novel tasks are thought to be first stored in the Hippocampus and then over time integrated in the prefrontal cortex. NREM sleep has been shown to be very important for this.  
   [Object Space Task — Genzel Lab](https://www.genzellab.com/object-space-task-1)  
   Link to paper: <https://doi.org/10.1371/journal.pbio.3000322>
4. Each Rat folder has condition folders, we work with OR\_N folders. OR\_N stands for Overlapping Novelty.
5. We have 5 post-trials for each dataset. The fifth is usually longer than others. These folders have recordings for Hippocampus and Prefrontal Cortex and also the sleep states.

**Sleep Scoring values**: The scoring files are arrays of numbers

* 1 - Awake
* 2 - Intermediate
* 3 - Non-REM
* 4 - Intermediate
* 5 - REM

**Filtering Considerations**:

1. FIR filter vs IIR Filter
2. Delta band = (0.1, 4) Hz

**Steps**:

1. Load all 5 post-trials for a Rat-Condition-Region (like Rat-4 OR\_N PFC)
2. Z-scoring: Separately for each post-trial (hence each recording)
3. Filtering: Bandpass on 0.1-4 Hz, using FIR or IIR filter
4. Get IP, IF and IA using EMD
5. Extract cycles from IP using EMD
6. Define metrics on each cycle, this can be used to further filter the data
7. Select cycles with is\_good=True and get metrics dataframe (also includes cycle timings)
8. Using cycle timings and the sleep\_scoring data, define the sleep state value for each cycle
9. Align all cycles to 128 points as UMAP requires all data to be of same size. There are different methods for alignment
10. Add all the metrics and cycles to a single dataframe and save in an HDF5 file. We can filter the cycles according to any metric.
11. Get Intrinsic dimension, it is almost always 4
12. Get a 3D UMAP embedding with default parameter settings

**Metrics on cycles**:

1. Cycle duration: Length of cycles
2. Peak2trough ratio: P/(P+T) where P represents the time in the peak region and T the trough region. Hence this represents the fraction of the whole signal length at which the signal changes sign.
3. asc2desc ratio: A/(A+D) where A is the amount of time the signal is rising (before peak and after trough) and D is the time it is falling.
4. Amplitude of the cycle
5. Peak value
6. Trough Value
7. mean IF
8. max IF
9. range IF

EMD Library: [Empirical Mode Decomposition in Python — emd 0.0.1.dev127 documentation](https://emd.readthedocs.io/en/stable/)

UMAP Library: [UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction — umap 0.5 documentation](https://umap-learn.readthedocs.io/en/latest/)

# Folder Structure for Data

Main Folder

----> data

----> OS\_basic\_separated

----> 1

----> study\_day\_5\_OR\_N

----> post\_trial1\_2017-10-03\_11-05-10

----> post\_trial2\_2017-10-03\_12-00-20

----> … (other post-trials)

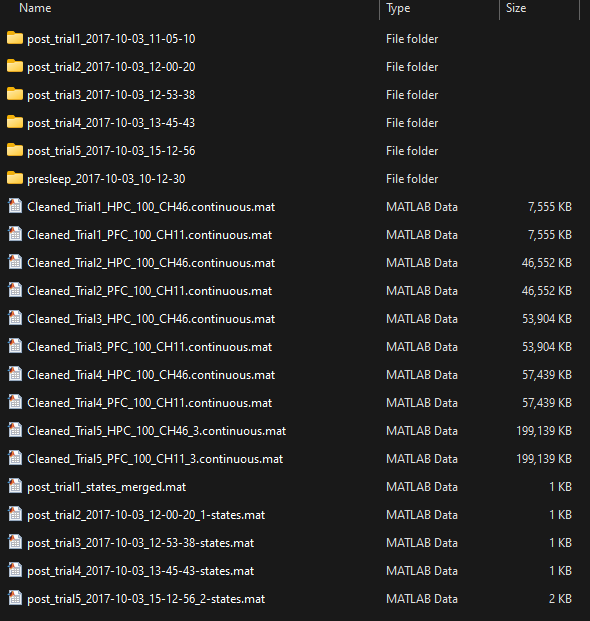
----> study\_day\_2\_OR

----> … (other condition folders)

----> 3

----> …. (other rat folders)

Copy the states.mat files from the post-trial folder to the conditions folder. Then run ‘1 - Cleaning\_script.ipynb’, this will put the cleaned data in the conditions folder. It will look like:



# Files

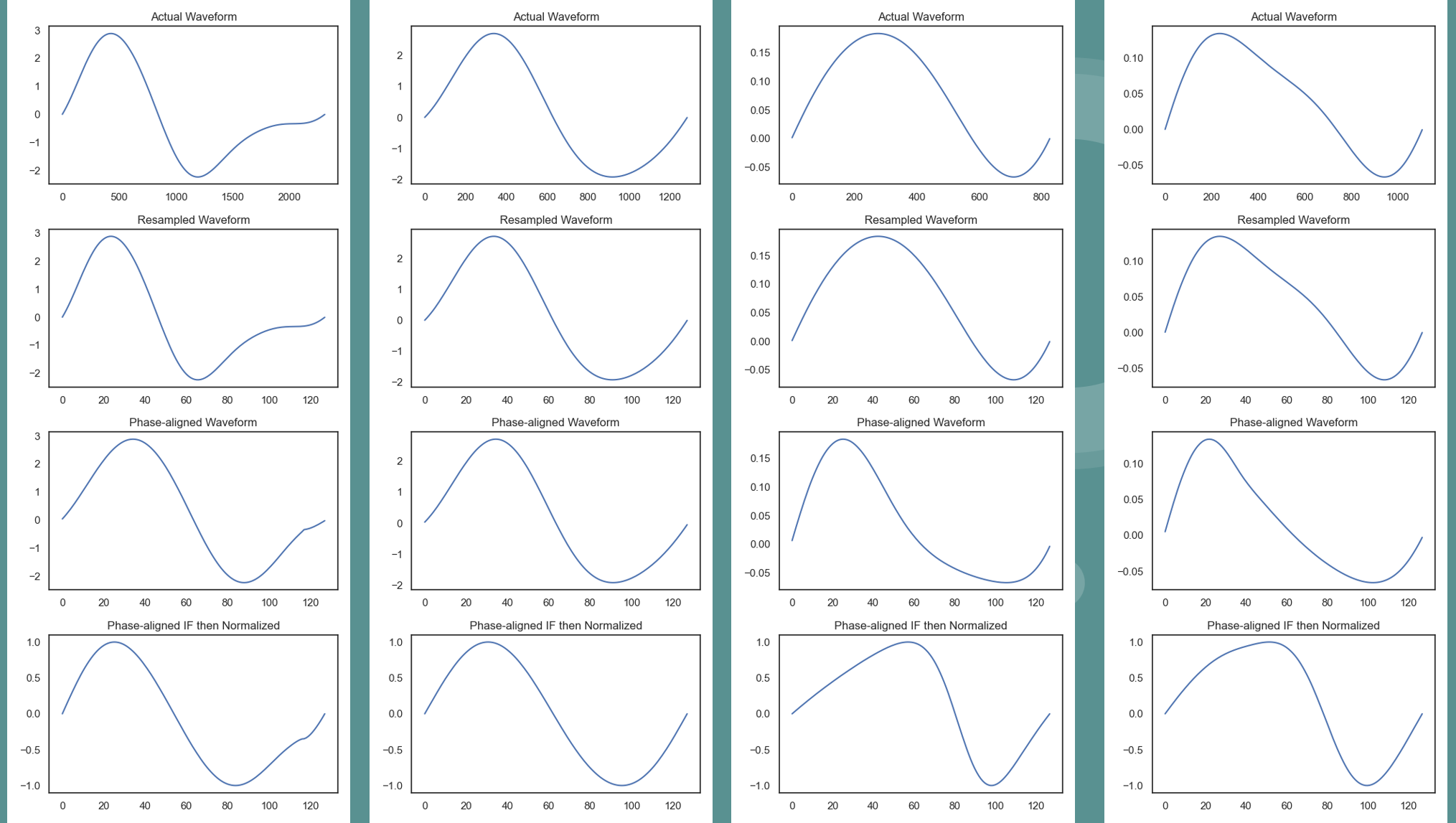
**File: 6 - Get\_Cycles\_All\_Posttrials.ipynb**

This loads data for a (rat-condition-region) and extracts good cycles using EMD from all 5 post-trials. We also define the above 9 metrics for all the cycles and save them in a dataframe.

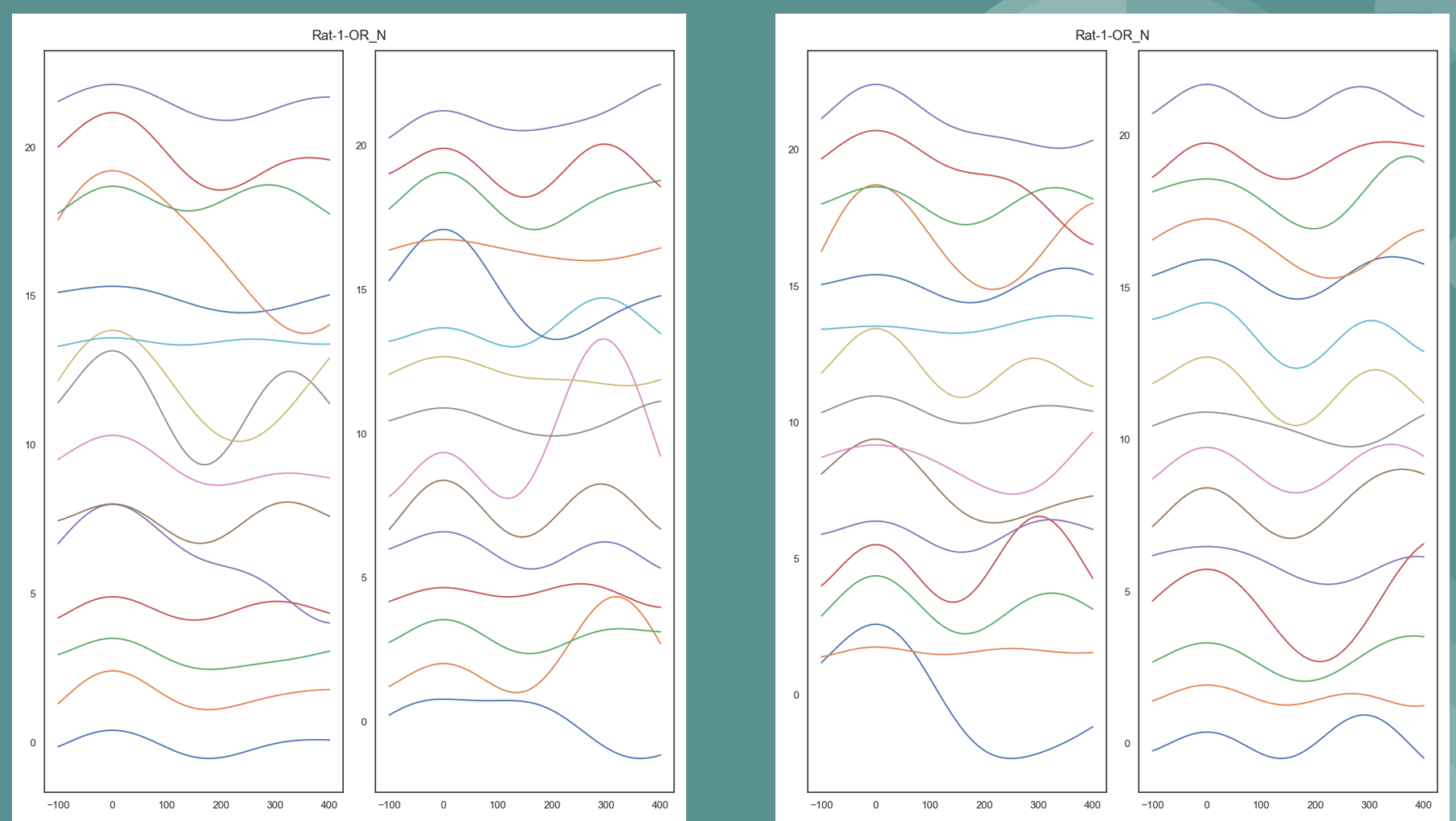
All the cycles will have different durations, we have to align them to same number of time points so we can put them through dimensionality reduction algorithms. We align the cycle waveforms to 128 time points in 3 different ways:

* **Normalized IF Waveforms**: This aligns the instantaneous frequency (IF) profiles using the Instantaneous Phase (IP). Internally, it finds a function that maps from IP to IF and then applies that function to a template instantaneous phase. As IP can be different for each cycle, so the mapping function will be different. Applying the mapping function to the template phase gives an IF waveform that is said to be aligned. Then, we get back the actual cycle waveform from the IF profile, but because only IF is used information about amplitude is lost, hence all waveforms are normalized to [-1,1]. This alignment results in cycles that are more slightly more sinusoidal.
* **Normalized Cycle Waveforms**: This does the same as above but directly to the cycle waveforms instead of IF profiles. Hence, the amplitude information is retained. This method results in the zero-point always being at the mid-point (time-point=64). Here, peak2trough information is lost.
* **Resampled Waveforms**: Simply resamples the cycle waveform to 128 points, so the overall shape is retained but the IF information is changed from the original waveform.

Examples:



We are currently checking the method of **Peak Alignment** where we align the waveforms by the taking the time window of -100ms to 400ms.

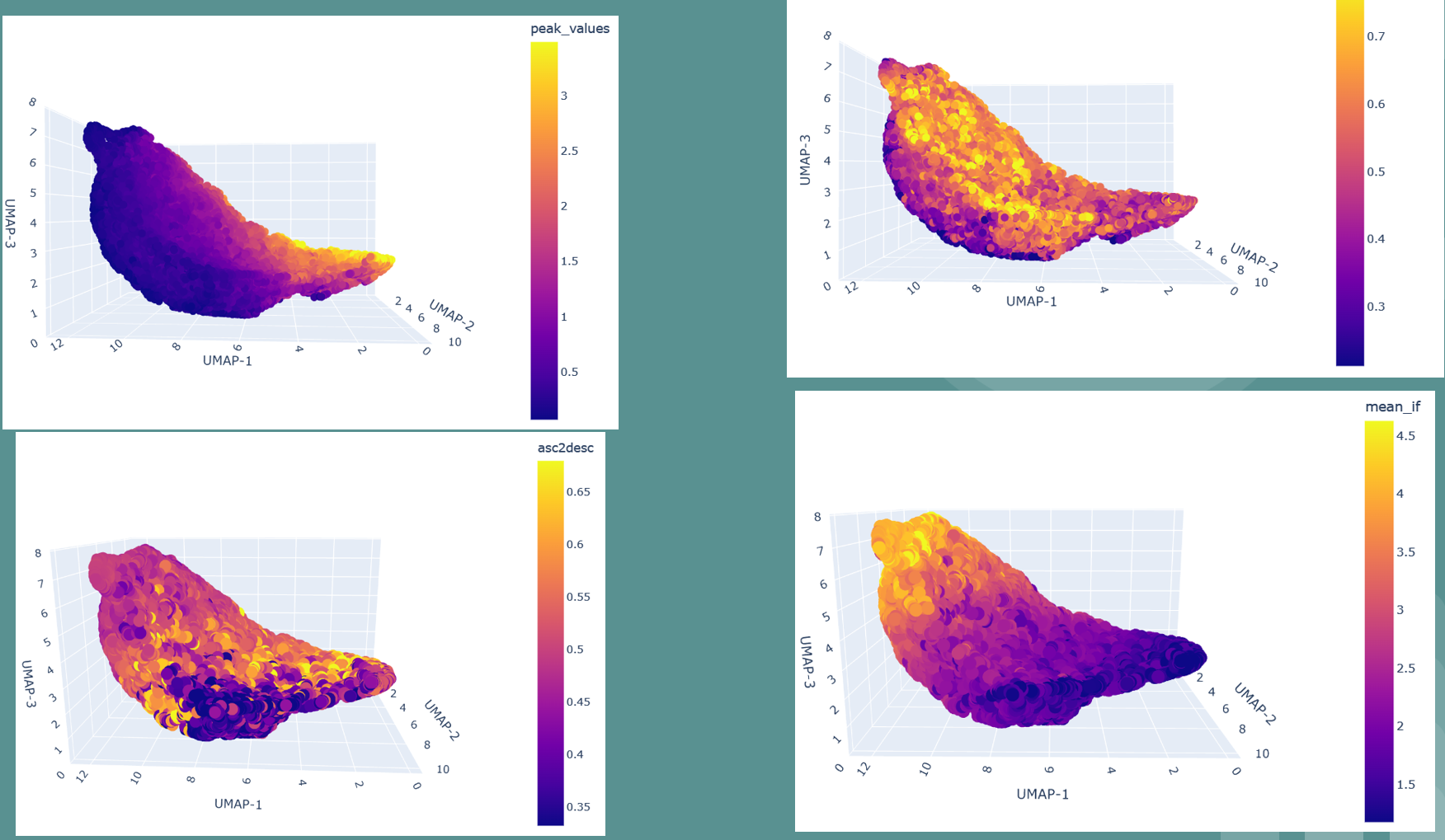


We save all the metrics and the aligned waveforms in HDF5 files for later use with UMAP.

**File: 8 - UMAP.ipynb**

This is the main file for looking at UMAPs for a (rat-condition-region) dataset.

* We generate a 3D UMAP from the aligned cycle waveforms data.
* We want to see if the axis of the UMAP corresponds to some metric, that is, we want to assign some meaning to the UMAP. For this, we project the defined metrics onto the UMAP which can give us smooth gradients. This can be done in a 2D or 3D view.



* We can also split the UMAP across time sections (of default 45 minutes) to see if some projected metric changes with time. We can also see if the sleep state projections change with time.
* If we want to see how sleep-state=3 (Non-REM) changes with time, we can quantify this either with trajectory of the centroid corresponding to the point-cloud for sleep-state=3 or looking at the structure index.