BASS demo

Faustine Ginoux, Gautam Sridhar

1. Dataset

Ensure that the data set is in the right format.
dataname_dataset_condition{n}.npy

e.g. phtaxis_dataset_condition0.npy
phtaxis_dataset_condition1.npy

Matrix shape: n_bouts x n_features

phtaxis_tailangles_condition → n_bouts x n_frames x n_points

n_points: number of points on the fish tail that are tracked

Features of the tail angle you find interesting: e.g. delta heading, speed...
Can also use the principal components.

In phtaxis: features = [delta_heading, speed, PC1, PC2, PC3, PC4]
/!\ The more PCs you keep, the longer the computation.

If the experiment was repeated several times, n_bouts is the total sum of bouts of all experiments. **dataname_lengths_condition{n}.npy** is an array describing the length of each experiment (i.e. *sum(dataname_lengths_condition{n}) = n_bouts*)

e.g. phtaxis_lengths_condition0.npy
phtaxis_lengths_condition1.npy

2. Learn GMM on full phtaxis dataset

python learn_gmm.py --help [you can do that for all scripts to find the arguments]

‘val’ mode - to find the best number of clusters (it takes a long time). n_clust = 7
python learn_gmm.py *-t train* -c 0 1 *-n 7* -dN phtaxis -gN phtaxis
-t: type = train (default)
-c: conditions = 0 and 1
-n: number of clusters = 7 (default)
-dN: data name = phtaxis (use full dataset to learn gmm)
-gN: GMM name (choose what you want e.g. phtaxis)

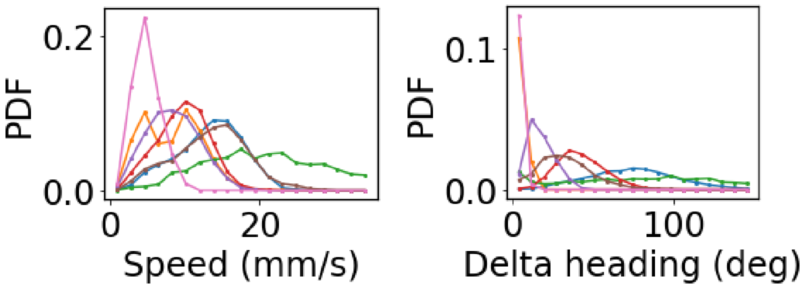
OUTPUT: GMM that will be used with BASS.

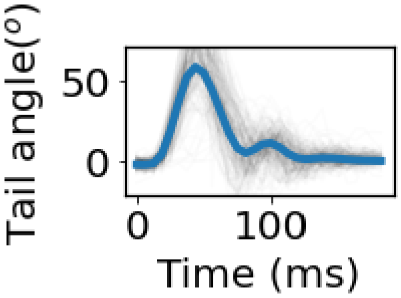
3. **Analyze GMM** (jupyter notebook)

See clusters, in 2D e.g. with PCA.
See figures of tail angles / speed / heading to determine classes of bouts and update **phtaxis_class_names** file (in Data folder).

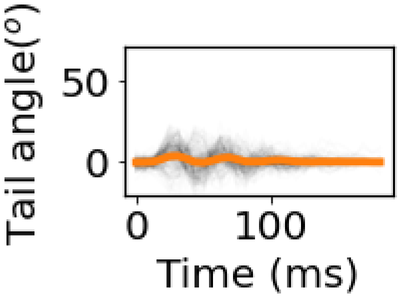
For n_clust = 7 on phtaxis:

- two forward classes (fast **F** & slow **f**)
- three turn classes (small **t**, mid **T**, large **L**)
- one burst class **b**
- and the class "other" **O**.

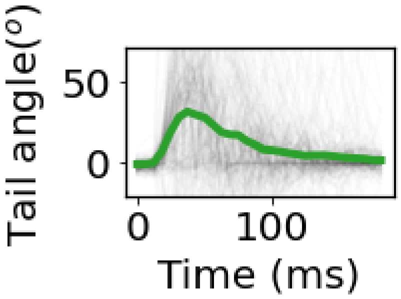




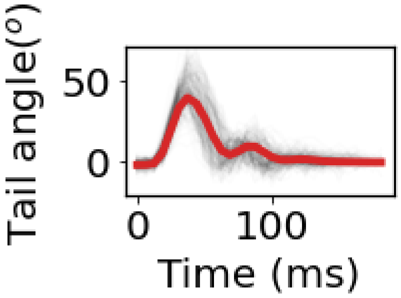
Cluster 1 → Large turn **L**



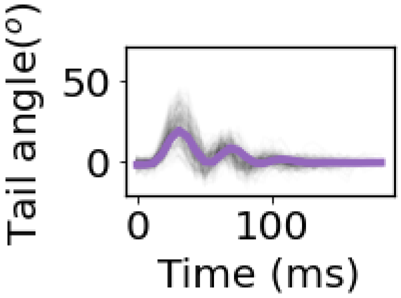
Cluster 2 → fast forward **F**



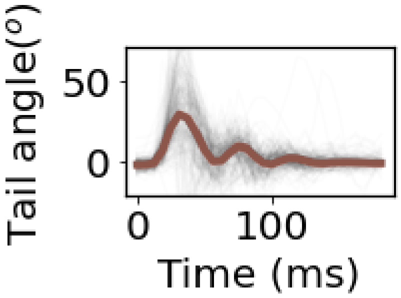
Cluster 3 → other **O**



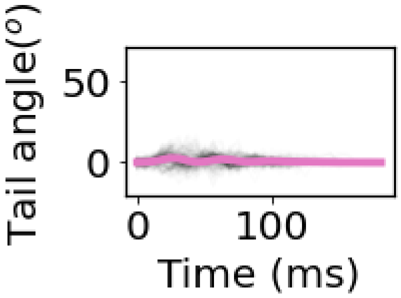
Cluster 4 → medium turn **T**



Cluster 5 → slow turn **t**



Cluster 6 → burst **b**



Cluster 7 → slow forward **f**

→ My **phtaxis_class_names:** L, F, O, T, t, b, f

/!\ The order of classes will change every time you run learn_gmm again.

How to find the classes: large first peak → large turn (same for medium turn but slightly smaller)
How to differentiate burst and slow turn? burst should have a higher frequency (and more peaks) → not very clear in this phtaxis dataset
O - the noisiest
Slow forward → the flattest
Fast forward → similar to **f** but larger amplitude

4. Run bass on phtaxis1 one condition at a time

Run on phtaxis1 because phtaxis is too large.
Each condition should take around 10 minutes.

python run_bass.py -c 0 -dN phtaxis1 -gN phtaxis -x phtaxis1
python run_bass.py -c 1 -dN phtaxis1 -gN phtaxis -x phtaxis1
-c: condition = 0 or 1 (unlike learn_gmm, **each condition must be run separately** with BASS)
-dN: data name = phtaxis1 (subset of phtaxis ~10 000 bouts)
-gN: GMM name = phtaxis (GMM learned on full dataset of ~80 000 bouts)
-x: name of result file (e.g. phtaxis1)

OUTPUT: BASS dictionary (csv) and comparison to HMM (csv).

5. Compare datasets

Compare the two conditions.
python compare_datasets.py -cn 0 -ch 1 *-t 0.1* -dN phtaxis1 -gN phtaxis -x phtaxis1
-cn: condition to use as null hypothesis (here: 0)
-ch: condition to use as the test case (here: 1)
-t: threshold on the - log p value to select relevant motifs (default = 0.1)
-dN: data name = phtaxis1 (subset of phtaxis ~10 000 bouts)
-gN: GMM name = phtaxis (GMM learned on full dataset of ~80 000 bouts)
-x: name of result file (e.g. phtaxis1)

OUTPUT: comparison - phtaxis1_Comparisons_cond0vcond1 (csv).

6. Decode

python decode.py -c 0 -dN phtaxis1 -gN phtaxis -x phtaxis1
python decode.py -c 1 -dN phtaxis1 -gN phtaxis -x phtaxis1

OUTPUT: npy files - segmented lengths and segmented words (and segmented bout types).

7. Analyze decoded