Co-occurrence Code Documentation

Jim Brunner

August 14, 2017

Contents

1 `co_occ_funs.py`

Module of python functions used in the rest of the project.

1.1 `both_occ`

Function that counts the number of times both `r1` and `r2` are within a range, and then returns the fraction of times this occurs. The range is half open: $(a, b]$.

- Input: array `r1` and array `r2`.
- Optional input: lower bound `lbd`, default 0. Upper bound `ubd` default 1.
- Output: int count of indices i such that $r1(i) \in (lbd, ubd]$ and $r2(i) \in (lbd, ubd]$.

1.2 `both_same_bin`

Function that counts the number of times the two occur in the same abundance range.

- Input: array `r1`, array `r2`, float `lthresh`, int `numthresh`.
- Optional input: bool `rel`, default True.
- Output: count of indices i such that such that $r1(i) \in (lbd_j, ubd_j]$ and $r2(i) \in (lbd_j, ubd_j]$ for $j = 0, .., numthresh$.

Divides the interval $[0, 1]$ into $numthresh$ intervals $(a_j, b_j]$ and calls `both_occ(r1,r2,lbd = a_j, ubd = b_j)` for each.

1.3 `color_picker`

Function that classifies nodes by which type of sample they have the highest abundance in. If it's a dataframe it will return the column head of the winner'

- Input:
- Optional input:
- Output:

1.4 `matchyn`

- Input:
- Optional input:
- Output:

1.5 `occ_probs`

- Input:
- Optional input:
- Output:

1.6 `random_coocc_prob`

- Input:
- Optional input:
- Output:

1.7 `approx_rand_prob`

- Input:
- Optional input:
- Output:

1.8 `mc_pearson`

- Input:
- Optional input:
- Output:

1.9 `make_null`

- Input:
- Optional input:
- Output:

1.10 `mc_pearson_thr`

- Input:
- Optional input:
- Output:

1.11 `min_nz`

- Input:
- Optional input:
- Output:

1.12 `build_network`

- Input:
- Optional input:
- Output:

1.13 `make_meta`

- Input:
- Optional input:
- Output:

1.14 `make_meta_from_file`

- Input:
- Optional input:
- Output:

1.15 `mc_network_stats`

- Input:
- Optional input:
- Output:

1.16 `sim_pears`

- Input:
- Optional input:
- Output:

1.17 `sim_pears_thr`

- Input:
- Optional input:
- Output:

1.18 `sim_bins`

- Input:
- Optional input:
- Output:

1.19 `edge_prob`

- Input:
- Optional input:
- Output:

1.20 `nodes_in_sub`

- Input:
- Optional input:
- Output:

1.21 `random_sub_graph`

- Input:
- Optional input:
- Output:

1.22 `exp_cut_edges`

- Input:
- Optional input:
- Output:

1.23 `cut_cond`

- Input:
- Optional input:
- Output:

1.24 `com_clust`

- Input:
- Optional input:
- Output:

1.25 `spectral_cluster`

- Input:
- Optional input:
- Output:

1.26 `clust_judge`

- Input:
- Optional input:
- Output:

1.27 `color_picker2`

- Input:
- Optional input:
- Output:

1.28 `est_prob`

- Input:
- Optional input:
- Output:

1.29 `find_cliques`

- Input:
- Optional input:
- Output:

1.30 `psi_over_psi`

- Input:
- Optional input:
- Output:

1.31 `diff_cliques`

- Input:
- Optional input:
- Output:

1.32 `diffusion_ivp`

- Input:
- Optional input:
- Output:

1.33 `diffusion_bvp`

- Input:
- Optional input:
- Output:

1.34 `diffusion_forced`

- Input:
- Optional input:
- Output:

1.35 `ivp_score`

- Input:
- Optional input:
- Output:

1.36 `make_sample`

- Input:
- Optional input:
- Output:

1.37 `get_sample`

- Input:
- Optional input:
- Output:

1.38 `flat_two_deep`

- Input:
- Optional input:
- Output:

1.39 `flat_one_deep`

- Input:
- Optional input:
- Output:

2 co_occurrence.py

3 cluster_net.py

4 network_stats.py

5 falsepm.py

6 sample_analysis.py

7 examples.py

8 adding_data.py

9 mc_speed.py

10 add_gender.py

11 cleanup.py

12 color_key.py

13 funct_tests.py

14 rand_samp.py

15 net_making.sh

16 sample_test.sh

References