scanpy.tl.paga

`scanpy.tl.paga`*(adata, groups=None, use_rna_velocity=False, model='v1.2', neighbors_key=None, copy=False)*

Mapping out the coarse-grained connectivity structures of complex manifolds [Wolf19].

By quantifying the connectivity of partitions (groups, clusters) of the single-cell graph, partition-based graph abstraction (PAGA) generates a much simpler abstracted graph (*PAGA graph*) of partitions, in which edge weights represent confidence in the presence of connections. By tresholding this confidence in `paga()`, a much simpler representation of the manifold data is obtained, which is nonetheless faithful to the topology of the manifold.

The confidence should be interpreted as the ratio of the actual versus the expected value of connetions under the null model of randomly connecting partitions. We do not provide a p-value as this null model does not precisely capture what one would consider "connected" in real data, hence it strongly overestimates the expected value. See an extensive discussion of this in [Wolf19].

Note

Note that you can use the result of `paga()` in `umap()` and `draw_graph()` via `init_pos='paga'` to get single-cell embeddings that are typically more faithful to the global topology.

Parameters:

adata : `AnnData`

An annotated data matrix.

groups : `Optional`[`str`] (default: `None`)

Key for categorical in `adata.obs`. You can pass your predefined groups by choosing any categorical annotation of observations. Default: The first present key of `'leiden'` or `'louvain'`.

use_rna_velocity : `bool` (default: `False`)

Use RNA velocity to orient edges in the abstracted graph and estimate transitions. Requires that `adata.uns` contains a directed single-cell graph with key `['velocity_graph']`. This feature might be subject to change in the future.

model : `Literal`[`'v1.2'`, `'v1.0'`] (default: `'v1.2'`)

The PAGA connectivity model.

neighbors_key : `Optional`[`str`] (default: `None`)

If not specified, paga looks `.uns['neighbors']` for neighbors settings and `.obsp['connectivities']`, `.obsp['distances']` for connectivities and distances respectively (default storage places for `pp.neighbors`). If specified, paga looks `.uns[neighbors_key]` for neighbors settings and `.obsp[.uns[neighbors_key]['connectivities_key']]`, `.obsp[.uns[neighbors_key]['distances_key']]` for connectivities and distances respectively.

copy : `bool` (default: `False`)

Copy `adata` before computation and return a copy. Otherwise, perform computation inplace and return `None`.

Returns:

: **connectivities** : `numpy.ndarray` (adata.uns['connectivities'])

The full adjacency matrix of the abstracted graph, weights correspond to confidence in the connectivities of partitions.

connectivities_tree : `scipy.sparse.csr_matrix` (adata.uns['connectivities_tree'])

The adjacency matrix of the tree-like subgraph that best explains the topology.

Notes

Together with a random walk-based distance measure (e.g. `scanpy.tl.dpt()`) this generates a partial coordinatization of data useful for exploring and explaining its variation.

See also

`pl.paga`, `pl.paga_path`, `pl.paga_compare`