scanpy.pp.normalize_total

scanpy.pp.normalize_total(*adata, target_sum=None, exclude_highly_expressed=False, max_fraction=0.05, key_added=None, layer=None, layers=None, layer_norm=None, inplace=True, copy=False*)

Normalize counts per cell.

Normalize each cell by total counts over all genes, so that every cell has the same total count after normalization. If choosing `target_sum=1e6`, this is CPM normalization.

If `exclude_highly_expressed=True`, very highly expressed genes are excluded from the computation of the normalization factor (size factor) for each cell. This is meaningful as these can strongly influence the resulting normalized values for all other genes [Weinreb17].

Similar functions are used, for example, by Seurat [Satija15], Cell Ranger [Zheng17] or SPRING [Weinreb17].

Parameters:

adata : `AnnData`

The annotated data matrix of shape `n_obs` × `n_vars`. Rows correspond to cells and columns to genes.

target_sum : `Optional` [`float`] (default: `None`)

If `None`, after normalization, each observation (cell) has a total count equal to the median of total counts for observations (cells) before normalization.

exclude_highly_expressed : `bool` (default: `False`)

Exclude (very) highly expressed genes for the computation of the normalization factor (size factor) for each cell. A gene is considered highly expressed, if it has more than `max_fraction` of the total counts in at least one cell. The not-excluded genes will sum up to `target_sum`.

max_fraction : `float` (default: `0.05`)

If `exclude_highly_expressed=True`, consider cells as highly expressed that have more counts than `max_fraction` of the original total counts in at least one cell.

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

Name of the field in `adata.obs` where the normalization factor is stored.

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

Layer to normalize instead of `X`. If `None`, `X` is normalized.

inplace : `bool` (default: `True`)

Whether to update `adata` or return dictionary with normalized copies of `adata.X` and `adata.layers`.

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

Whether to modify copied input object. Not compatible with inplace=False.

Return type: `Optional` [`Dict` [`str` , `ndarray`]]

Returns:

: Returns dictionary with normalized copies of `adata.X` and `adata.layers` or updates `adata` with normalized version of the original `adata.X` and `adata.layers`, depending on `inplace`.

Example

```
>>> from anndata import AnnData
>>> import scanpy as sc
>>> sc.settings.verbosity = 2
>>> np.set_printoptions(precision=2)
>>> adata = AnnData(np.array([
...    [3, 3, 3, 6, 6],
...    [1, 1, 1, 2, 2],
...    [1, 22, 1, 2, 2],
... ]))
>>> adata.X
array([[ 3.,  3.,  3.,  6.,  6.],
       [ 1.,  1.,  1.,  2.,  2.],
       [ 1., 22.,  1.,  2.,  2.]], dtype=float32)
>>> X_norm = sc.pp.normalize_total(adata, target_sum=1, inplace=False)['X']
>>> X_norm
array([[0.14, 0.14, 0.14, 0.29, 0.29],
       [0.14, 0.14, 0.14, 0.29, 0.29],
       [0.04, 0.79, 0.04, 0.07, 0.07]], dtype=float32)
>>> X_norm = sc.pp.normalize_total(
...     adata, target_sum=1, exclude_highly_expressed=True,
...     max_fraction=0.2, inplace=False
... )['X']
The following highly-expressed genes are not considered during normalization factor
computation:
['1', '3', '4']
>>> X_norm
array([[ 0.5,  0.5,  0.5,  1. ,  1. ],
       [ 0.5,  0.5,  0.5,  1. ,  1. ],
       [ 0.5, 11. ,  0.5,  1. ,  1. ]], dtype=float32)
```