scanpy.pp.filter_cells

`scanpy.pp.filter_cells`*(data, min_counts=None, min_genes=None, max_counts=None, max_genes=None, inplace=True, copy=False)*

Filter cell outliers based on counts and numbers of genes expressed.

For instance, only keep cells with at least `min_counts` counts or `min_genes` genes expressed. This is to filter measurement outliers, i.e. “unreliable” observations.

Only provide one of the optional parameters `min_counts`, `min_genes`, `max_counts`, `max_genes` per call.

Parameters:

data : `AnnData`

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

min_counts : `Optional`[`int`] (default: `None`)

Minimum number of counts required for a cell to pass filtering.

min_genes : `Optional`[`int`] (default: `None`)

Minimum number of genes expressed required for a cell to pass filtering.

max_counts : `Optional`[`int`] (default: `None`)

Maximum number of counts required for a cell to pass filtering.

max_genes : `Optional`[`int`] (default: `None`)

Maximum number of genes expressed required for a cell to pass filtering.

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

Perform computation inplace or return result.

Return type:

`Optional`[`Tuple`[`ndarray`, `ndarray`]]

Returns:

: Depending on `inplace`, returns the following arrays or directly subsets and annotates the data matrix:

cells_subset

Boolean index mask that does filtering. `True` means that the cell is kept. `False` means the cell is removed.

number_per_cell

Depending on what was tresholded (`counts` or `genes`), the array stores `n_counts` or `n_cells` per gene.

Examples

```
>>> import scanpy as sc
>>> adata = sc.datasets.krumsiek11()
>>> adata.n_obs
640
>>> adata.var_names
['Gata2' 'Gata1' 'Fog1' 'EKLF' 'Fli1' 'SCL' 'Cebpa'
 'Pu.1' 'cJun' 'EgrNab' 'Gfi1']
>>> # add some true zeros
>>> adata.X[adata.X < 0.3] = 0
>>> # simply compute the number of genes per cell
>>> sc.pp.filter_cells(adata, min_genes=0)
>>> adata.n_obs
640
>>> adata.obs['n_genes'].min()
1
>>> # filter manually
>>> adata_copy = adata[adata.obs['n_genes'] >= 3]
>>> adata_copy.obs['n_genes'].min()
>>> adata.n_obs
554
>>> adata.obs['n_genes'].min()
3
>>> # actually do some filtering
>>> sc.pp.filter_cells(adata, min_genes=3)
>>> adata.n_obs
554
>>> adata.obs['n_genes'].min()
3
```