In [1]:
```
import numpy as np                                      # scientific computing toolk
import pandas as pd                                     # data analysis toolkit
import scanpy as sc                                     # scanpy is referred to with
import matplotlib.pyplot as plt                         # Matplotlib is referred to
from scipy import stats                                 # for linear regressions
import seaborn as sns                                   # for easy heatmaps
import scirpy as ir                                     # TCR analysis
import scvelo as scv

sc.settings.verbosity = 3                               # verbosity: errors (0), war
sc.logging.print_header()                               # check if all needed versio
results_directory = './Analysis_output/'
results_file = results_directory + '/results_file/write/results.h5ad'
```

```
scanpy==1.8.2 anndata==0.7.6 umap==0.5.1 numpy==1.20.3 scipy==1.6.3 pandas==
1.2.4 scikit-learn==0.24.2 statsmodels==0.12.2 python-igraph==0.9.1 pynndesce
nt==0.5.2
```

Directory variables

In [2]:
```
import glob
import os
#import subprocess
#import sys
from pathlib import Path

sample_directory = './Annotation_stim_unstim/'
```

In [3]:
```
plt.rcParams['figure.figsize']=(6,6) #rescale figures
sc.settings.set_figure_params(dpi=100, dpi_save=300, color_map='viridis')
### you can set a default figure directory here for saving output
sc.settings.figdir = "./analysis_output/figures_sc/"
## Figure directory for matplotlib figures/axes
save_figure_sc = "./analysis_output/figures_sc/"
#Use
#plt.savefig(save_figure + 'image.pdf')
save_figure_ir = "./analysis_output/figures_scirpy/"
#csv directory
read_csv = './analysis_output/results_file/'
save_csv = r'./analysis_output/results_file/'

#save figure for presentation
```

Loading the Data

In [4]:
```
# define sample metadata. Usually read from a file.
samples = {
    "mult_neg_st": {"group": "mouse_31_40_stim_multimer_negative"},
    "mult_neg_us": {"group": "mouse_31_40_unstim_multimer_negative"},
    "mult_pos_st": {"group": "mouse_31_40_stim_multimer_positive"},
    "mult_pos_us": {"group": "mouse_31_40_unstim_multimer_positive"},
    "1_CD44hi_st": {"group": "mouse_1_10_stim"},
    "1_CD44hi_us": {"group": "mouse_1_10_unstim"},
    "2_CD44hi_st": {"group": "mouse_11_20_stim"},
    "2_CD44hi_us": {"group": "mouse_11_20_unstim"},
    "3_CD44hi_st": {"group": "mouse_21_30_stim"},
    "3_CD44hi_us": {"group": "mouse_21_30_unstim"},
```

In [5]:
```
# Create a list of AnnData objects (one for each sample)
adatas = []
for sample, sample_meta in samples.items():
    gex_file = glob.glob(f'{sample_directory}/GEX/*{sample}*/outs/filtered_fea
    tcr_file = glob.glob(f'{sample_directory}/VDJ/*{sample}*/outs/*annotations
    adata = sc.read_10x_mtx(gex_file,
                            var_names='gene_symbols',      # use gene symbols
                            cache=True, gex_only=False)   # write a cache file
    adata_tcr = ir.io.read_10x_vdj(tcr_file)
    change_cat = adata_tcr.obs.columns.tolist()            #NaN values in VDJ d
    for i in change_cat:                                    #here all NaN (basi
        adata_tcr.obs[i]=adata_tcr.obs[i].astype(str)
        adata_tcr.obs[i]=adata_tcr.obs[i].fillna('None')
        adata_tcr.obs[i]=adata_tcr.obs[i].replace('nan' ,'None')
        adata_tcr.obs[i]=adata_tcr.obs[i].astype('category')
    ir.pp.merge_with_ir(adata, adata_tcr)
    adata.obs["experiment"] = sample
    adata.obs["group"] = sample_meta["group"]
    # concatenation only works with unique gene names
    adata.var_names_make_unique()
```

```
... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-Mult_neg_st_GEX_4-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-Mult_neg_us_GEX_2-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-Mult_pos_st_GEX_3-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-Mult_pos_us_GEX_1-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-1_CD44hi_st_GEX_1-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-1_CD44hi_us_GEX_2-outs-filtered_feature_bc
```

```
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None
... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-2_CD44hi_st_GEX_6-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-2_CD44hi_us_GEX_5-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-3_CD44hi_st_GEX_8-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical

... reading from cache file cache\C-Users-ga36xaw-10X_Analysis-repertoire_sat
uration-Annotation_stim_unstim-GEX-3_CD44hi_us_GEX_7-outs-filtered_feature_bc
_matrix-matrix.h5ad
WARNING: Non-standard locus name ignored: None

... storing 'feature_types' as categorical
```

Re-annotate correct mouse ID from gene expression annotation

One mouse ID was falsely annotated during GEX gene_id annotation - corrected in the following cell

In [6]:
```
annotate_mouse_id_adata_2_CD44_hi_correct = ['69797' if id=='69782' else id fo
adatas[6].var_names = annotate_mouse_id_adata_2_CD44_hi_correct
annotate_mouse_id_adata_2_CD44_hi_correct = ['69797' if id=='69782' else id fo
```

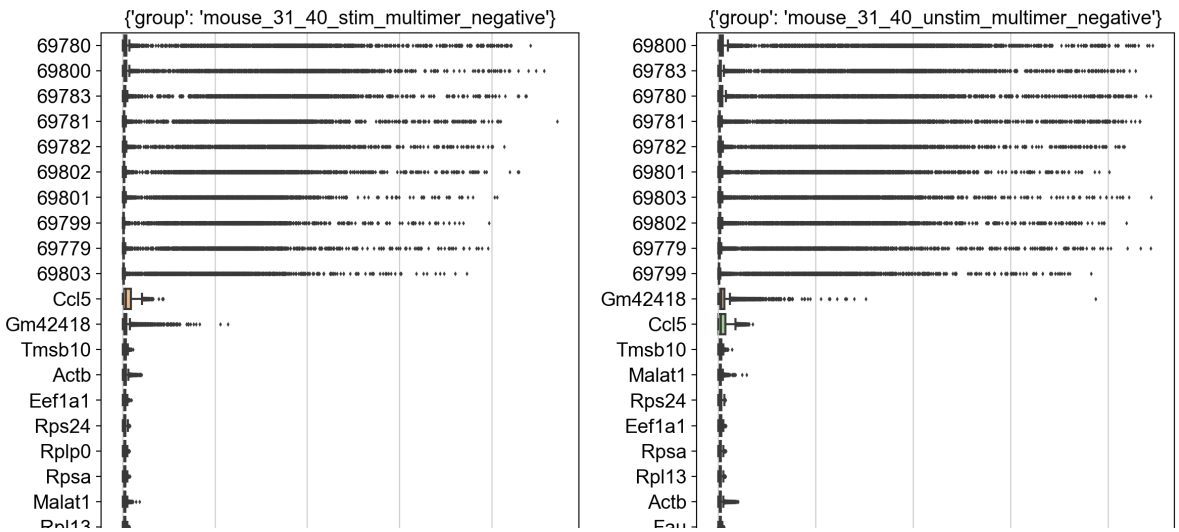
Preprocessing

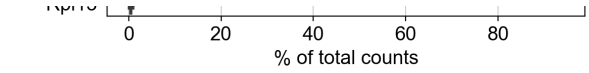
Hashtag Antibodies

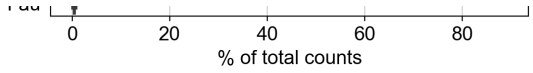
Highest expressed genes in all samples and quality check for hashtag antibody expression - Show those genes that yield the highest fraction of counts in each single cells, across all cells.

In [7]:
```
nrows = len(adatas)//2
ncols = 2
fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=(12,3*len(adatas)))
index = 0
axis = 0

for i, adata in enumerate(adatas):
    if index <ncols:
        sc.pl.highest_expr_genes(adata, n_top=20, ax=ax[axis,index], show=Fals
        ax[axis,index].set_title(samples[adata.obs.experiment[0]])
        index = index + 1
    else:
        axis = axis + 1
        sc.pl.highest_expr_genes(adata, n_top=20, ax=ax[axis,0], show=False)
        ax[axis,0].set_title(samples[adata.obs.experiment[0]])
        index = 1
fig.tight_layout()
plt.show()
```

```
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
normalizing counts per cell
    finished (0:00:00)
```

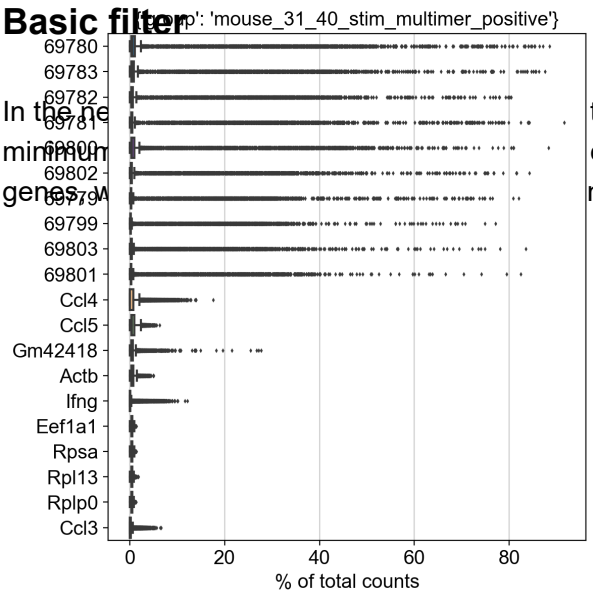


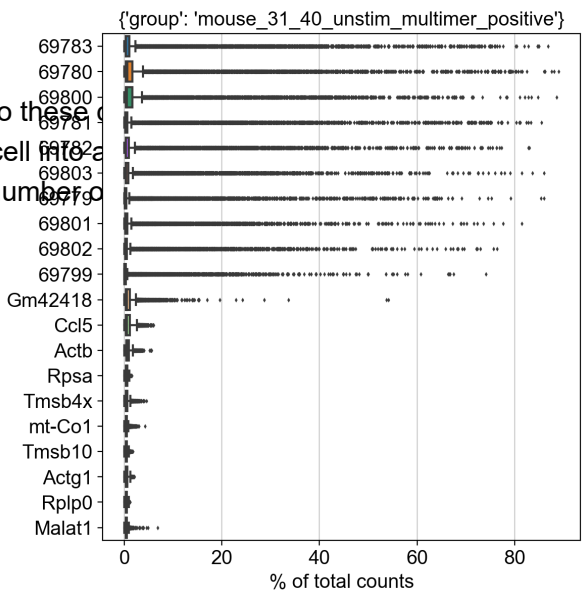


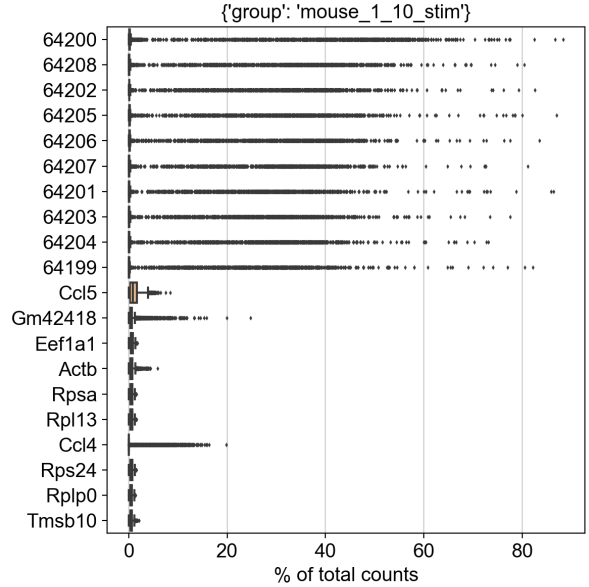


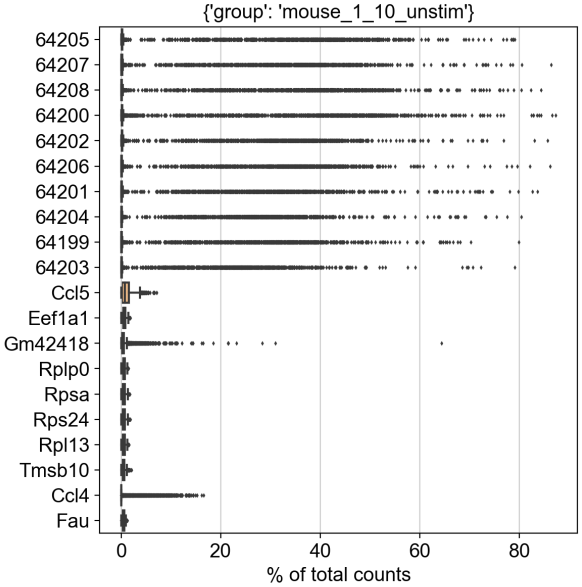
Basic filter

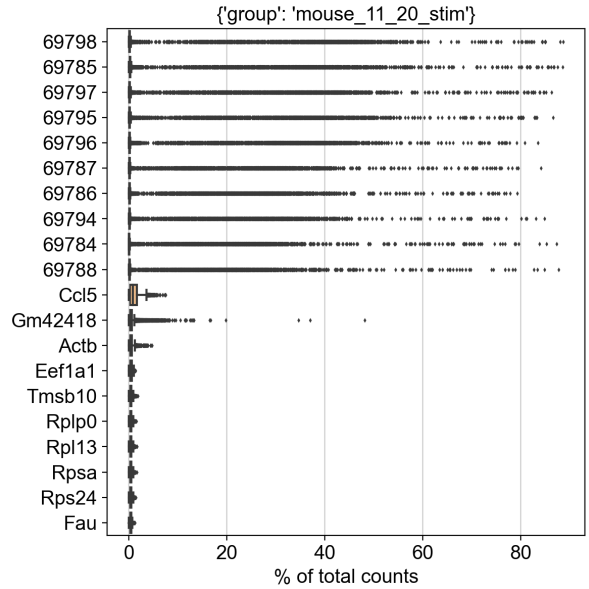
In the ne to these
minimum cell into a
genes w number o

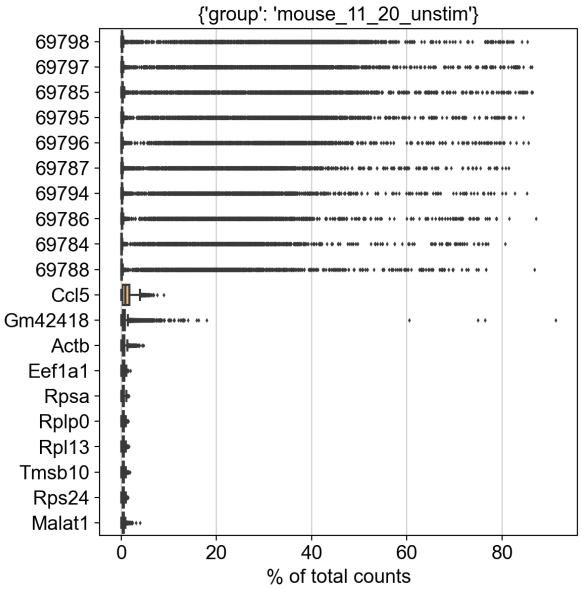


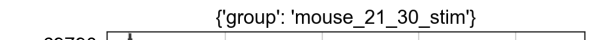


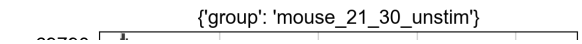




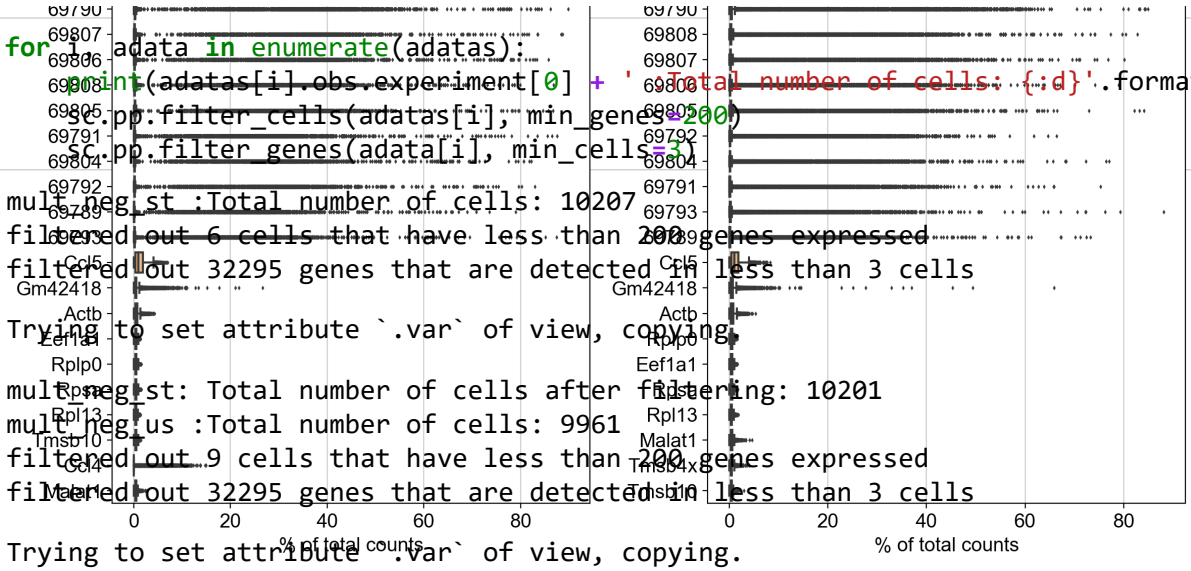








In [8]:
```
for i, adata in enumerate(adatas):
    print(adatas[i].obs.experiment[0] + ' :Total number of cells: {:d}'.format
    sc.pp.filter_cells(adatas[i], min_genes=200)
    sc.pp.filter_genes(adata[i], min_cells=3)
```




```
mult_neg_st :Total number of cells: 10207
filtered out 6 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

mult_neg_st: Total number of cells after filtering: 10201
mult_neg_us :Total number of cells: 9961
filtered out 9 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

mult_neg_us: Total number of cells after filtering: 9952
mult_pos_st :Total number of cells: 9199
filtered out 1 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

mult_pos_st: Total number of cells after filtering: 9198
mult_pos_us :Total number of cells: 9413
filtered out 6 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

mult_pos_us: Total number of cells after filtering: 9407
1_CD44hi_st :Total number of cells: 6820
filtered out 5 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

1_CD44hi_st: Total number of cells after filtering: 6815
1_CD44hi_us :Total number of cells: 7112
filtered out 11 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

1_CD44hi_us: Total number of cells after filtering: 7101
2_CD44hi_st :Total number of cells: 11065
filtered out 6 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

2_CD44hi_st: Total number of cells after filtering: 11059
2_CD44hi_us :Total number of cells: 10533
filtered out 8 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.
```

```
3_CD44hi_st: Total number of cells: 11743
2_CD44hi_us: Total number of cells after filtering: 10525
filtered out 5 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

3_CD44hi_st: Total number of cells after filtering: 11738
3_CD44hi_us :Total number of cells: 14354
filtered out 6 cells that have less than 200 genes expressed
filtered out 32295 genes that are detected in less than 3 cells

Trying to set attribute `.var` of view, copying.

3_CD44hi_us: Total number of cells after filtering: 14348
```

Annotation of data

In [9]:
```
m_list_1_CD44hi = ['64199', '64200', '64201', '64202', '64203', '64204', '6420
m_list_2_CD44hi = ['69784', '69785', '69786', '69787', '69788', '69794', '6979
m_list_3_CD44hi = ['69789', '69790', '69791', '69792', '69793', '69804', '6980
```

In [10]:
```
experiment_list = ['mult_neg_st','mult_neg_us','mult_pos_st', 'mult_pos_us', '
```

In [11]:
```
```

In [12]:
```
m_list_1_CD44hi_HTO = []
for i in m_list_1_CD44hi:
    m_list_1_CD44hi_HTO.append(i + '_HTO')

m_list_2_CD44hi_HTO = []
for i in m_list_2_CD44hi:
    m_list_2_CD44hi_HTO.append(i + '_HTO')

m_list_3_CD44hi_HTO = []
for i in m_list_3_CD44hi:
    m_list_3_CD44hi_HTO.append(i + '_HTO')

m_list_multimer_HTO = []
for i in m_list_multimer:
```

In [13]:
```
```

In [14]:
```
adatas_HTO_list = []
adatas_HTO_list.append(m_list_multimer_HTO)
adatas_HTO_list.append(m_list_multimer_HTO)
adatas_HTO_list.append(m_list_multimer_HTO)
adatas_HTO_list.append(m_list_multimer_HTO)
adatas_HTO_list.append(m_list_1_CD44hi_HTO)
adatas_HTO_list.append(m_list_1_CD44hi_HTO)
adatas_HTO_list.append(m_list_2_CD44hi_HTO)
adatas_HTO_list.append(m_list_2_CD44hi_HTO)
adatas_HTO_list.append(m_list_3_CD44hi_HTO)
adatas_HTO_list.append(m_list_3_CD44hi_HTO)
```

Generate adata object including only the hashtag antibodies

In [15]:
```
adatas_HTO_temp = [0]*len(adatas)
for i, adata in enumerate(adatas):
```

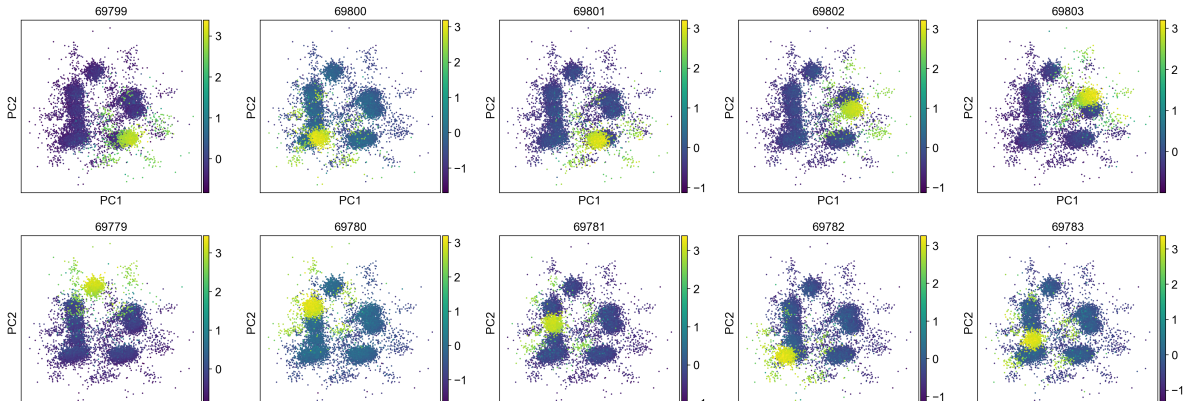
HashSolo

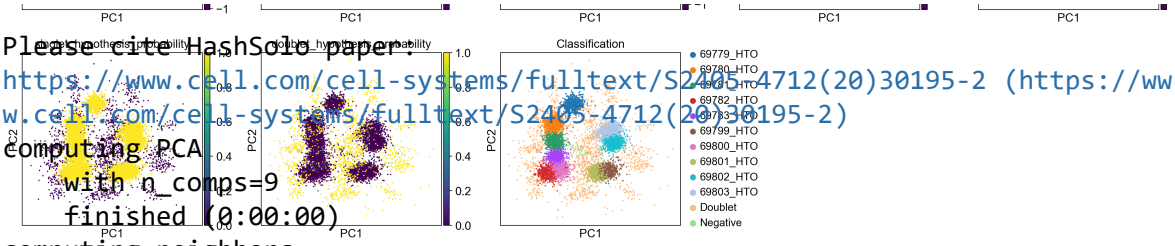
Normalize Barcode expression for plotting after running HashSolo

In [16]:
```
def clr_normalize_each_cell(adata, inplace=True):
    """Normalize count vector for each cell, i.e. for each row of .X"""

    import numpy as np
    import scipy

    def seurat_clr(x):
        # TODO: support sparseness
        s = np.sum(np.log1p(x[x > 0]))
        exp = np.exp(s / len(x))
        return np.log1p(x / exp)

    if not inplace:
        adata = adata.copy()

    # apply to dense or sparse matrix, along axis. returns dense matrix
    adata.X = np.apply_along_axis(
        seurat_clr, 1, (adata.X.A if scipy.sparse.issparse(adata.X) else adata
    )
```

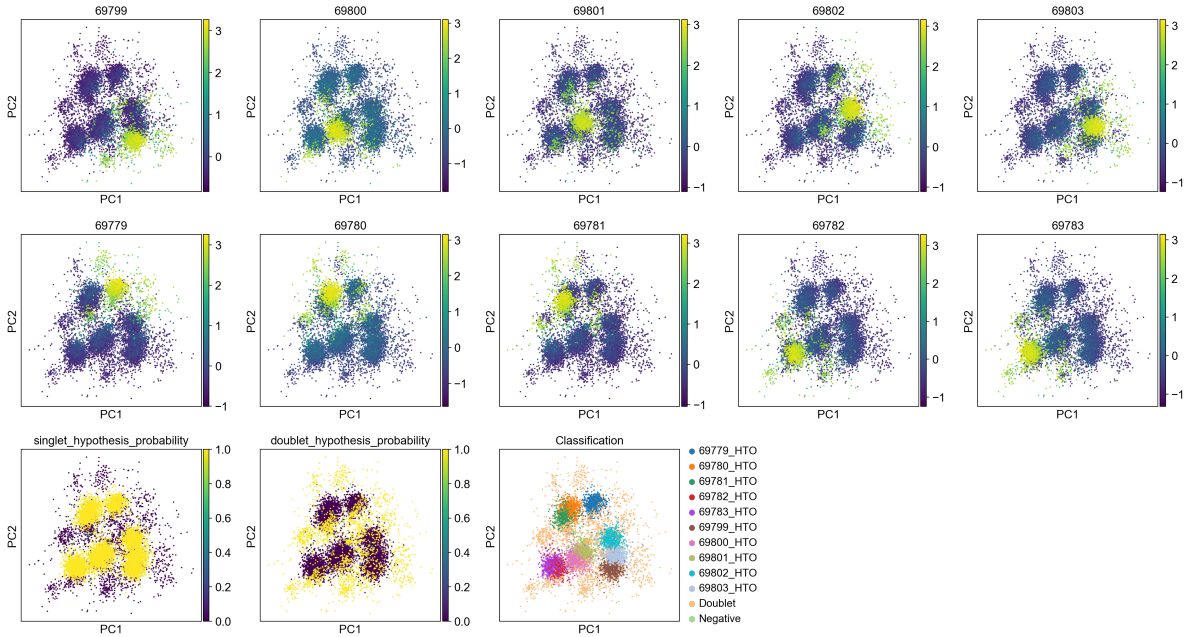
In [17]:
```
adatas_hashsolo = []

for i, adata in enumerate(adatas_HTO_temp):
    HTO_data=adata.copy()
    HTO_data.layers["counts"] = HTO_data.X.copy()
    HTO_counts=pd.DataFrame(HTO_data.X.todense(), columns=HTO_data.var_names+'
    HTO_data.obs=pd.concat([HTO_data.obs,HTO_counts], axis=1)
    sc.external.pp.hashsolo(HTO_data,cell_hashing_columns=adatas_HTO_list[i])

    adatas_hashsolo.append(HTO_data)

    clr_normalize_each_cell(HTO_data)
    sc.pp.log1p(HTO_data)
    sc.pp.scale(HTO_data)
    sc.pp.pca(HTO_data)
    sc.pp.neighbors(HTO_data, n_neighbors=50)
    #sc.tl.leiden(HTO_data, key_added="HTO_leiden", resolution=.1)
    #sc.tl.umap(HTO_data)
    #sc.tl.tsne(HTO_data)


    #sc.pl.umap(HTO_data, color=HTO_data.var_names.tolist()+['singlet_hypothes
    #sc.pl.tsne(HTO_data, color=HTO_data.var_names.tolist()+['singlet_hypothes
    sc.pl.pca(HTO_data, color=HTO_data.var_names.tolist()+['singlet_hypothesis
```

```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:13)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pcamult_neg_st_HTO.pd
f
```



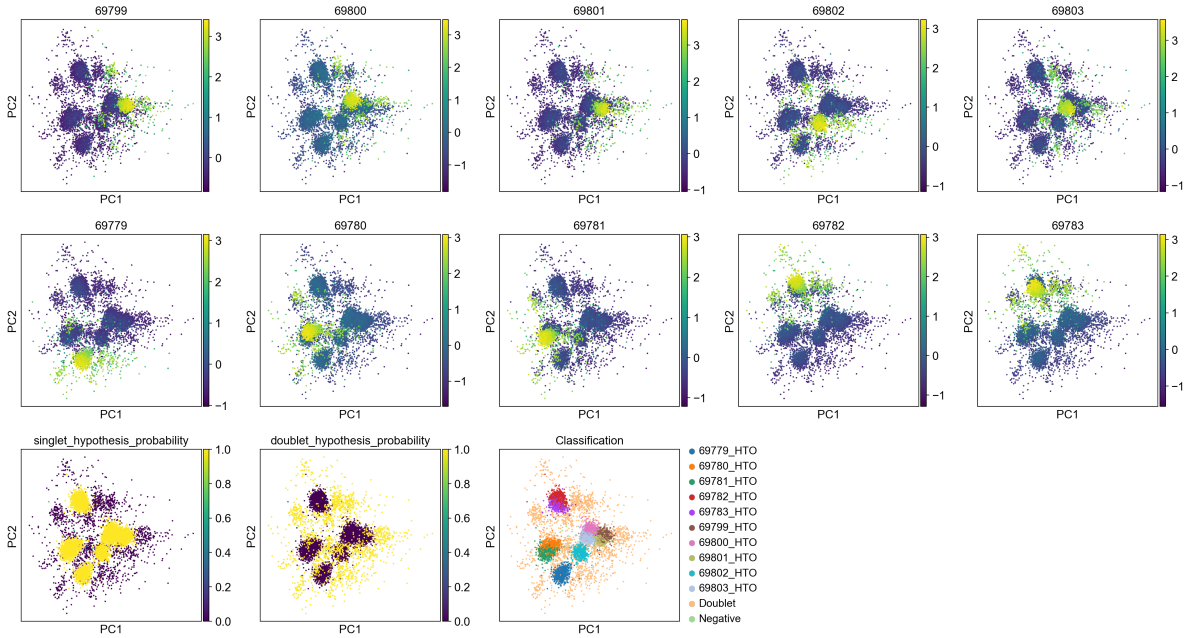



```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pcamult_neg_us_HTO.pd
f
```

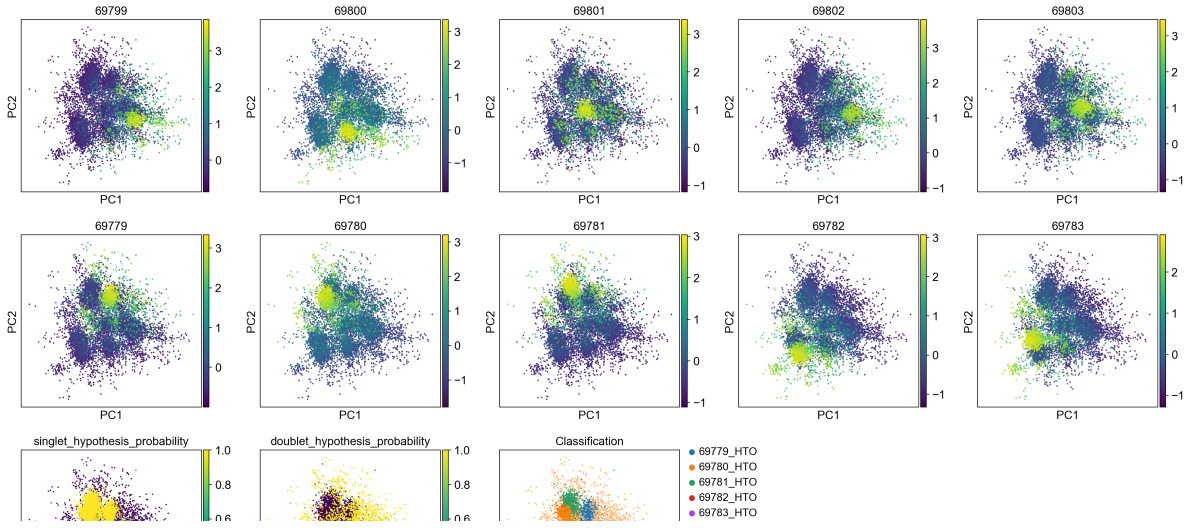


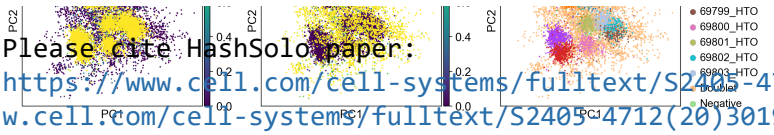

```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
```

```
ation\Annotation_stim_unstim\analysis_output\figures_sc\pcamult_pos_st_HTO.pd
f
```

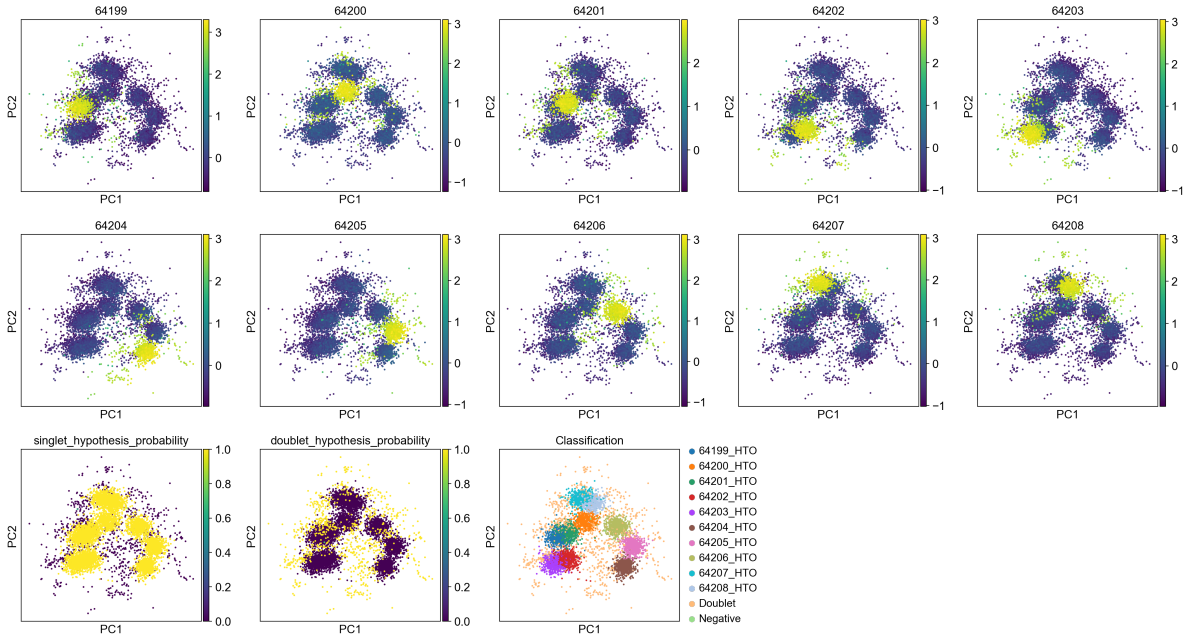



```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pcamult_pos_us_HTO.pd
f
```



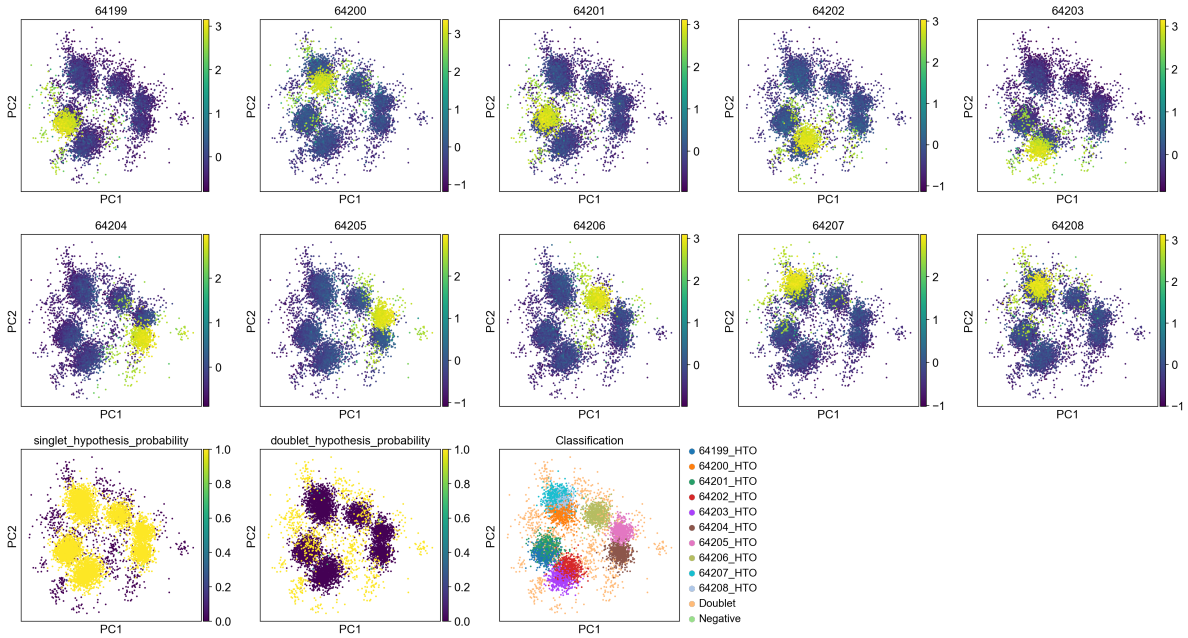



```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pca1_CD44hi_st_HTO.pd
f
```

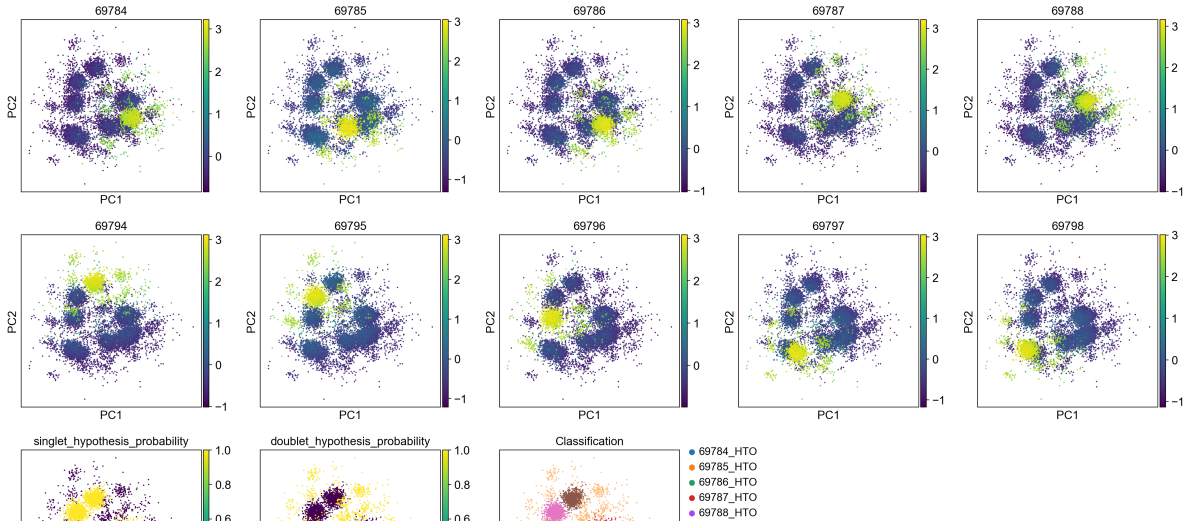


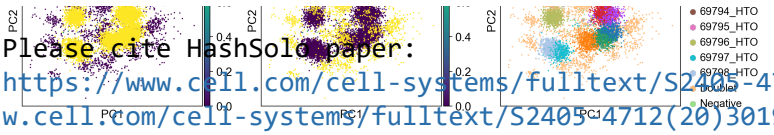

```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
```

```
ation\Annotation_stim_unstim\analysis_output\figures_sc\pca1_CD44hi_us_HTO.pd
f
```

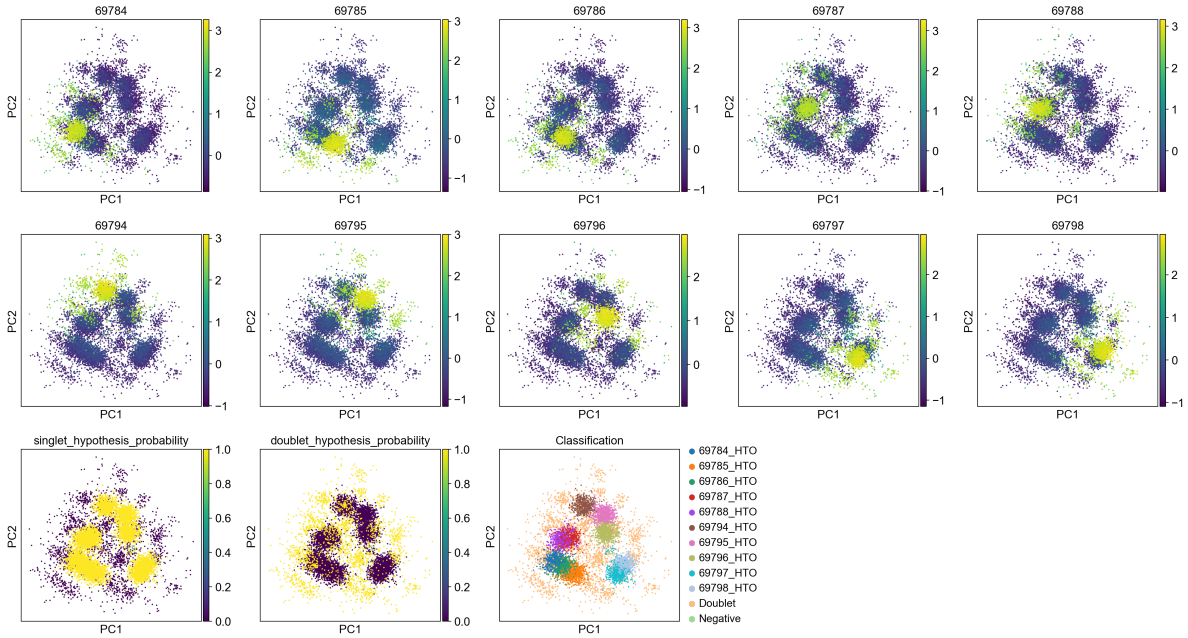



```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pca2_CD44hi_st_HTO.pd
f
```



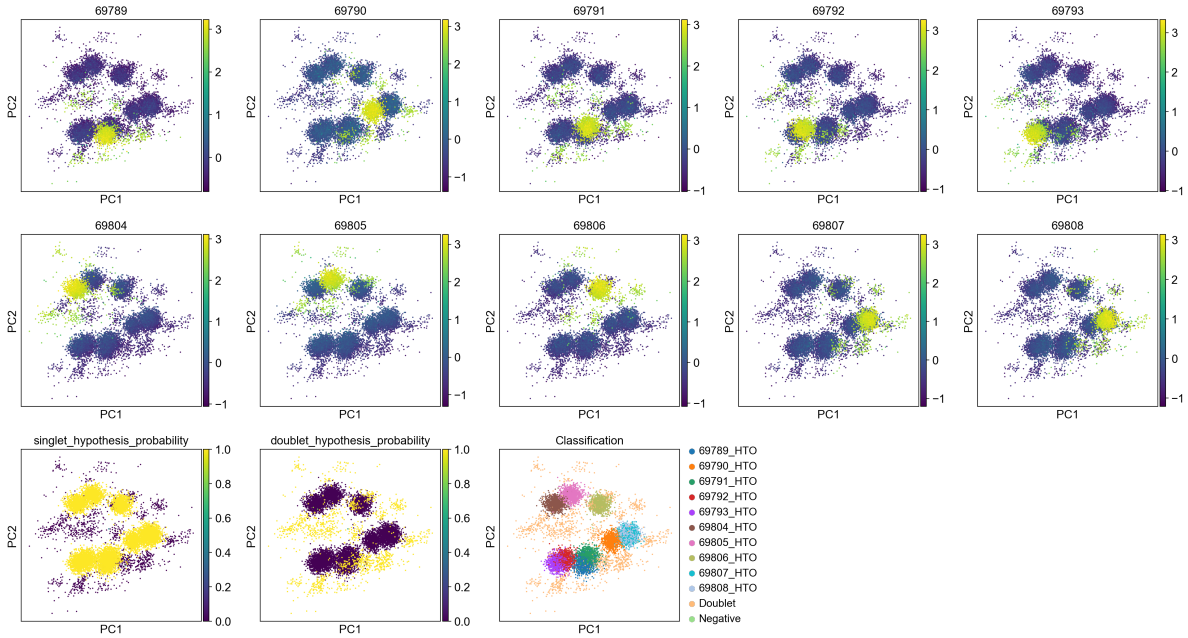



```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pca2_CD44hi_us_HTO.pd
f
```

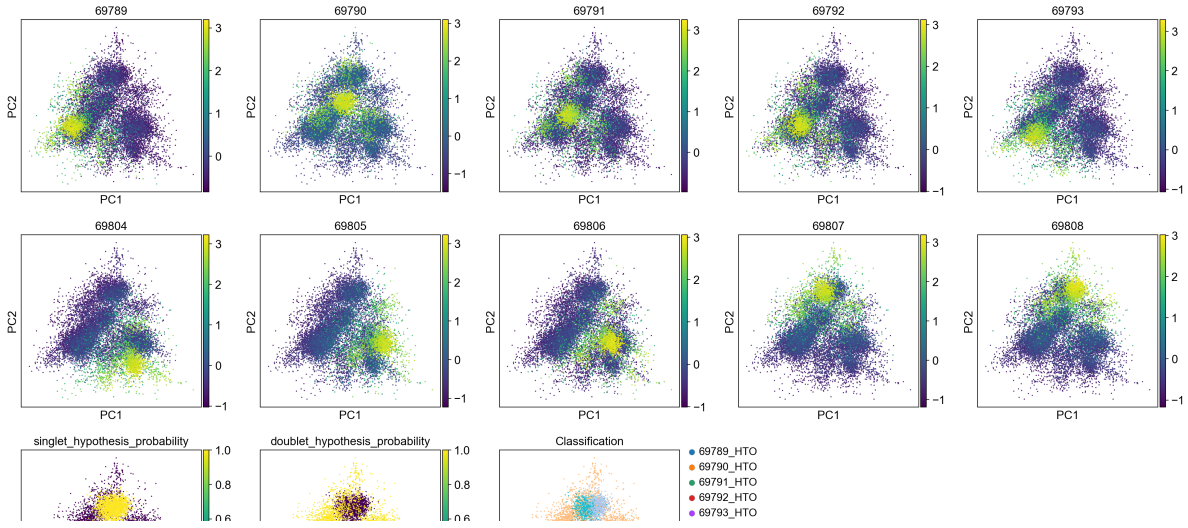


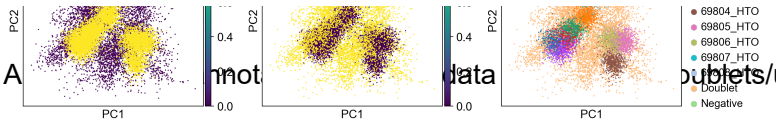

```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:01)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
```

```
ation\Annotation_stim_unstim\analysis_output\figures_sc\pca3_CD44hi_st_HTO.pd
f
```




```
Please cite HashSolo paper:
https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2 (https://ww
w.cell.com/cell-systems/fulltext/S2405-4712(20)30195-2)
computing PCA
    with n_comps=9
    finished (0:00:00)
computing neighbors
    using data matrix X directly
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:02)

... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'Classification' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\pca3_CD44hi_us_HTO.pd
f
```





A[illegible] not[illegible] data [illegible]ublets/unidentified cells

In [18]:
```
for i, adata in enumerate(adatas_hashsolo):
    adatas[i].obs['single_bc']='None'
```

In [19]:
```
adatas_filtered = adatas.copy()
for i, adata in enumerate(adatas_filtered):
```

In [20]:
```
for i, adata in enumerate(adatas_filtered):
```

```
mult_neg_st n_obs before bc:10201 || n_obs after bc:8619
mult_neg_us n_obs before bc:9952 || n_obs after bc:7779
mult_pos_st n_obs before bc:9198 || n_obs after bc:7330
mult_pos_us n_obs before bc:9407 || n_obs after bc:6052
1_CD44hi_st n_obs before bc:6815 || n_obs after bc:5940
1_CD44hi_us n_obs before bc:7101 || n_obs after bc:6172
2_CD44hi_st n_obs before bc:11059 || n_obs after bc:9069
2_CD44hi_us n_obs before bc:10525 || n_obs after bc:8315
3_CD44hi_st n_obs before bc:11738 || n_obs after bc:10290
3_CD44hi_us n_obs before bc:14348 || n_obs after bc:8947
```

Fraction of barcoded cells with singlet identification

In [21]:
```
for i, adata in enumerate(adatas_filtered):
```

```
mult_neg_st: 84.4917164983825
mult_neg_us: 78.16519292604501
mult_pos_st: 79.6912372254838
mult_pos_us: 64.33506962899969
1_CD44hi_st: 87.1606749816581
1_CD44hi_us: 86.9173355865371
2_CD44hi_st: 82.00560629351659
2_CD44hi_us: 79.0023752969121
3_CD44hi_st: 87.66399727381156
3_CD44hi_us: 62.35712294396431
```

Annotation of mouse samples

Here we are removing the hashtag expression from the adata sets

In [22]:
```
for i, adata in enumerate(adatas_filtered):
```

Annotation of experimental conditions

In [23]:
```
adatas_filtered[0].obs['multimer']='multimer_negative'
adatas_filtered[1].obs['multimer']='multimer_negative'
adatas_filtered[2].obs['multimer']='multimer_positive'
adatas_filtered[3].obs['multimer']='multimer_positive'
adatas_filtered[4].obs['multimer']='not_stained'
adatas_filtered[5].obs['multimer']='not_stained'
adatas_filtered[6].obs['multimer']='not_stained'
adatas_filtered[7].obs['multimer']='not_stained'
adatas_filtered[8].obs['multimer']='not_stained'
adatas_filtered[9].obs['multimer']='not_stained'

adatas_filtered[0].obs['batch']='stim'
adatas_filtered[1].obs['batch']='unstim'
adatas_filtered[2].obs['batch']='stim'
adatas_filtered[3].obs['batch']='unstim'
adatas_filtered[4].obs['batch']='stim'
adatas_filtered[5].obs['batch']='unstim'
adatas_filtered[6].obs['batch']='stim'
adatas_filtered[7].obs['batch']='unstim'
adatas_filtered[8].obs['batch']='stim'
adatas_filtered[9].obs['batch']='unstim'

adatas_filtered[0].obs['exp_group']='mult_neg'
adatas_filtered[1].obs['exp_group']='mult_neg'
adatas_filtered[2].obs['exp_group']='mult_pos'
adatas_filtered[3].obs['exp_group']='mult_pos'
adatas_filtered[4].obs['exp_group']='1_CD44'
adatas_filtered[5].obs['exp_group']='1_CD44'
adatas_filtered[6].obs['exp_group']='2_CD44'
adatas_filtered[7].obs['exp_group']='2_CD44'
adatas_filtered[8].obs['exp_group']='3_CD44'
```

Read-in barcode annotation

In [24]:
```
```

In [25]:
```
hto_annotation_stim_unstim['single_bc']= hto_annotation_stim_unstim['single_bc
```

In [26]:
```
for i, adata in enumerate(adatas_filtered):
    adatas_filtered[i].obs['mouse']='None'
    for barcode in (m_list_all_HTO):
        anot = adata.obs_names.values[adata.obs.single_bc==barcode]
        mouse = hto_annotation_stim_unstim.loc[hto_annotation_stim_unstim.sing
```

Excludes all cells without a hashtag

In [27]:
```
for i, adata in enumerate(adatas_filtered):
```

Concatenation of samples

In [28]:

In [29]:
```
key_list = ['mult_neg_st', 'mult_neg_us',
            'mult_pos_st', 'mult_pos_us',
            '1_CD44hi_st', '1_CD44hi_us',
            '2_CD44hi_st', '2_CD44hi_us',
            '3_CD44hi_st', '3_CD44hi_us'
```

In [30]:

Clear memory

In [31]:
```
#import gc

#del adatas
#del adatas_filtered
#del adatas_hashsolo
```

Preprocessing: Quality control

Data quality control can be split into cell QC and gene QC. Typical quality measures for assessing the quality of a cell include the number of molecule counts (UMIs), the number of expressed genes, and the fraction of counts that are mitochondrial.

You can also use the function `pp.calculate_qc_metrics()` to compute the fraction of mitochondrial genes and additional measures.

Fraction of mitochondrial genes

In [32]:
```
mito_genes = adata.var_names.str.startswith('mt-')
adata.obs['percent_mito'] = np.sum(
adata[:, mito_genes].X, axis=1).A1 / np.sum(adata.X, axis=1).A1
```

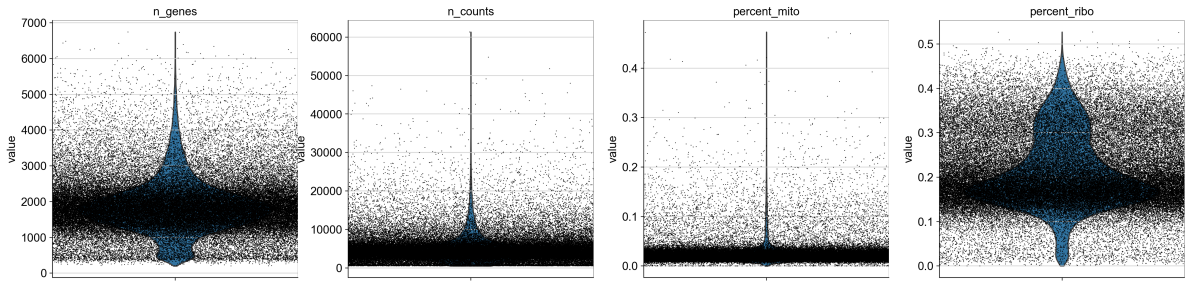
Fraction of ribosomal genes

In [33]:
```
ribo_genes = ((adata.var_names.str.startswith("Rpl")) | (adata.var_names.str.s
adata.obs['percent_ribo'] = np.sum(
adata[:, ribo_genes].X, axis=1) / np.sum(adata.X, axis=1)
adata.var['ribo'] = ribo_genes
```

A violin plot of the computed quality measures.

In [34]:
```
sc.pl.violin(adata, ['n_genes', 'n_counts', 'percent_mito', 'percent_ribo'],
```

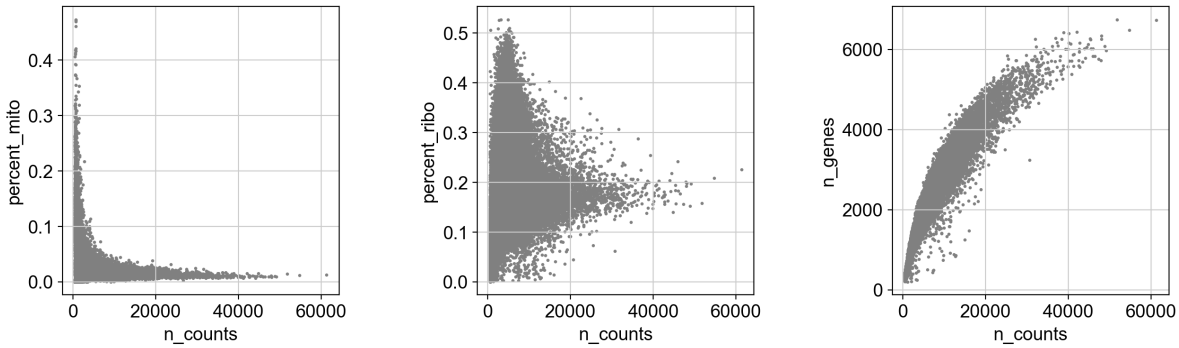
```
... storing 'extra_chains' as categorical
... storing 'IR_VJ_1_consensus_count' as categorical
... storing 'IR_VJ_2_consensus_count' as categorical
... storing 'IR_VDJ_1_consensus_count' as categorical
... storing 'IR_VDJ_2_consensus_count' as categorical
... storing 'IR_VJ_1_duplicate_count' as categorical
... storing 'IR_VJ_2_duplicate_count' as categorical
... storing 'IR_VDJ_1_duplicate_count' as categorical
... storing 'IR_VDJ_2_duplicate_count' as categorical
... storing 'IR_VJ_2_j_call' as categorical
... storing 'IR_VJ_1_junction' as categorical
... storing 'IR_VJ_2_junction' as categorical
... storing 'IR_VDJ_1_junction' as categorical
... storing 'IR_VDJ_2_junction' as categorical
... storing 'IR_VJ_1_junction_aa' as categorical
... storing 'IR_VJ_2_junction_aa' as categorical
... storing 'IR_VDJ_1_junction_aa' as categorical
... storing 'IR_VDJ_2_junction_aa' as categorical
... storing 'IR_VJ_1_v_call' as categorical
... storing 'IR_VJ_2_v_call' as categorical
... storing 'IR_VDJ_2_v_call' as categorical
... storing 'experiment' as categorical
... storing 'group' as categorical
... storing 'single_bc' as categorical
... storing 'multimer' as categorical
... storing 'batch' as categorical
... storing 'exp_group' as categorical
... storing 'mouse' as categorical
```



Remove cells that have too many mitochondrial genes expressed or too many total counts. First, plot the counts versus the percentage of mitochondrial genes and versus the number of genes to define a threshold for later filtering.

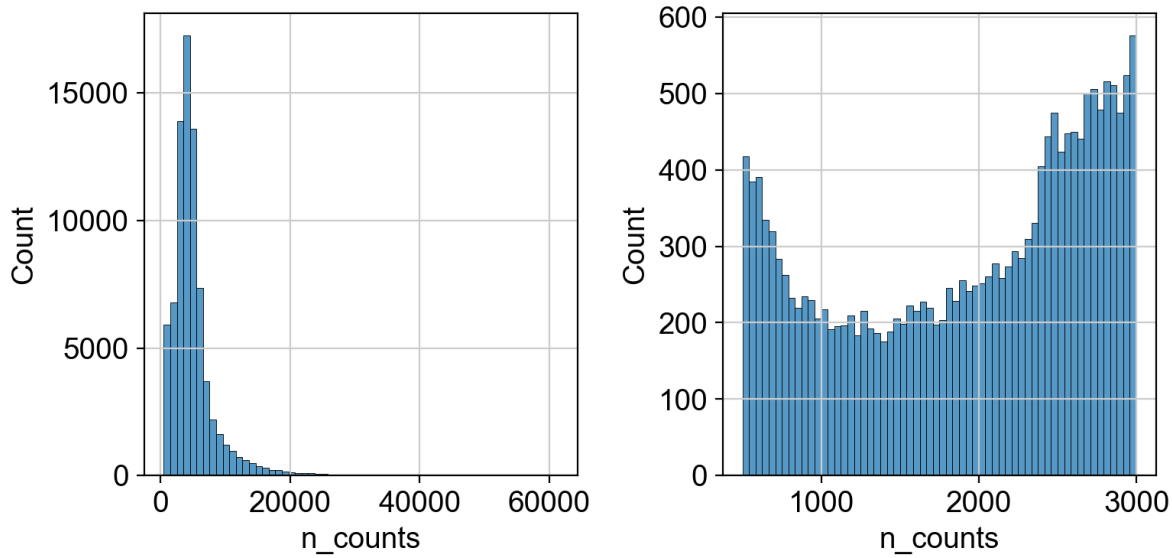
In [35]:
```
counts = adata.obs['n_counts']
mito = adata.obs['percent_mito']
genes = adata.obs['n_genes']
ribo = adata.obs['percent_ribo']
fig = plt.figure(figsize=(15, 4))
grid = plt.GridSpec(1, 3, hspace=0, wspace=0.5)
percent_mito = fig.add_subplot(grid[0, 0], xlabel='n_counts', ylabel='percent_
percent_ribo = fig.add_subplot(grid[0, 1], xlabel='n_counts', ylabel='percent_
n_genes = fig.add_subplot(grid[0, 2], xlabel='n_counts', ylabel='n_genes')
percent_mito.scatter(counts, mito, s=2, c='gray')
percent_ribo.scatter(counts, ribo, s=2, c='gray')
```

Out[35]:
```
<matplotlib.collections.PathCollection at 0x14cd952c370>
```



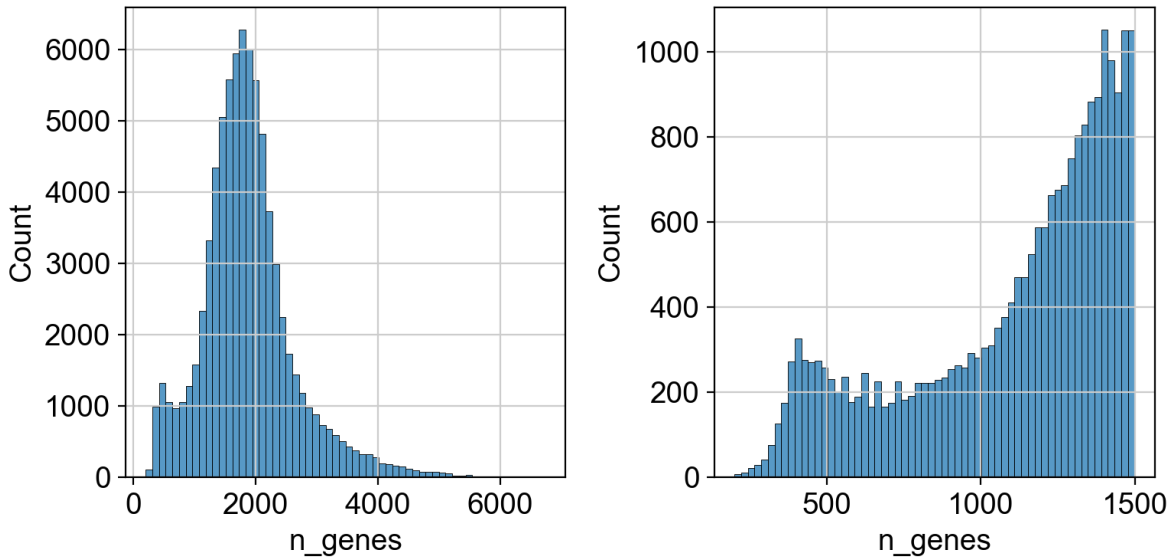
Histograms of the number of counts per cell

In [36]:
```
#Thresholding decision:  counts
fig, ax = plt.subplots(ncols=2, figsize=(8,4))

sns.histplot(adata.obs['n_counts'], kde=False, bins=60, ax=ax[0])
sns.histplot(adata.obs['n_counts'][adata.obs['n_counts']<3000], kde=False, bin

fig.tight_layout()
plt.show()
```



Histograms of the number of genes per cell

In [37]:
```
#Thresholding decision:  genes

fig, ax = plt.subplots(ncols=2, figsize=(8,4))

sns.histplot(adata.obs['n_genes'], kde=False, bins=60, ax=ax[0])
sns.histplot(adata.obs['n_genes'][adata.obs['n_genes']<1500], kde=False, bins=

fig.tight_layout()
```



Apply filter thresholds

In [38]:
```
print('Total number of cells: {:d}'.format(adata.n_obs))

sc.pp.filter_cells(adata, min_counts = 1000)
print('Number of cells after min count filter: {:d}'.format(adata.n_obs))

sc.pp.filter_cells(adata, max_counts = 40000)
print('Number of cells after max count filter: {:d}'.format(adata.n_obs))

adata = adata[adata.obs['percent_mito'] < 0.3]
print('Number of cells after MT filter: {:d}'.format(adata.n_obs))

sc.pp.filter_cells(adata, min_genes = 500)
```

```
Total number of cells: 78513
filtered out 3515 cells that have less than 1000 counts
Number of cells after min count filter: 74998
filtered out 36 cells that have more than 40000 counts
Number of cells after max count filter: 74962
Number of cells after MT filter: 74959
filtered out 44 cells that have less than 500 genes expressed

Trying to set attribute `.obs` of view, copying.

Number of cells after gene filter: 74915
```

In [39]:
```
#Filter genes:
print('Total number of genes: {:d}'.format(adata.n_vars))

# Min 5 cells - filters out 0 count genes
sc.pp.filter_genes(adata, min_cells=5)
```

```
Total number of genes: 32285
filtered out 14351 genes that are detected in less than 5 cells
Number of genes after cell filter: 17934
```

Normalization

Total-count normalize (library-size correct) the data matrix to 10,000 reads per cell, so that counts become comparable among cells.

In [40]:
```
sc.pp.normalize_per_cell(adata, counts_per_cell_after=1e4)
```

```
normalizing by total count per cell
    finished (0:00:01): normalized adata.X and added    'n_counts', counts pe
r cell before normalization (adata.obs)
```

In [ ]:
```

```

Cell cycle scoring

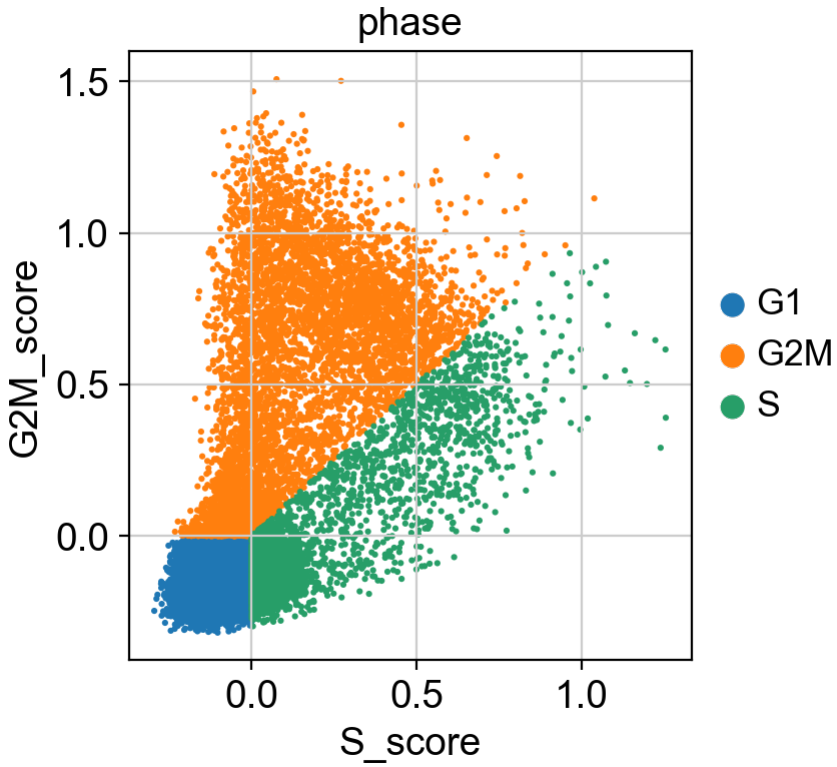
In [41]:
```
cc_genes=pd.read_csv(analysis_info+'regev_lab_cell_cycle_genes.txt', header=No
```

In [42]:
```
#Score cell cycle and visualize the effect:
s_genes=cc_genes[:46]
g2m_genes=cc_genes[47:]
s_genes.columns=['genes']
g2m_genes.columns=['genes']
```

```
calculating cell cycle phase
computing score 'S_score'
WARNING: genes are not in var_names and ignored: ['Mlf1ip']
    finished: added
    'S_score', score of gene set (adata.obs).
    641 total control genes are used. (0:00:02)
computing score 'G2M_score'
WARNING: genes are not in var_names and ignored: ['Fam64a', 'Hn1']
    finished: added
    'G2M_score', score of gene set (adata.obs).
    597 total control genes are used. (0:00:02)
-->     'phase', cell cycle phase (adata.obs)
```

In [43]:

```
... storing 'phase' as categorical
```



Set the .raw attribute of AnnData object to the logarithmized raw gene expression for later use in differential testing and visualizations of gene expression. This simply freezes the state of the AnnData object. While many people consider the normalized data matrix as the "relevant data" for visualization and differential testing, some would prefer to store the unnormalized data.

In [44]:

```
# Store the full data set in 'raw' as log-normalised data for statistical test
```

Highly variable genes

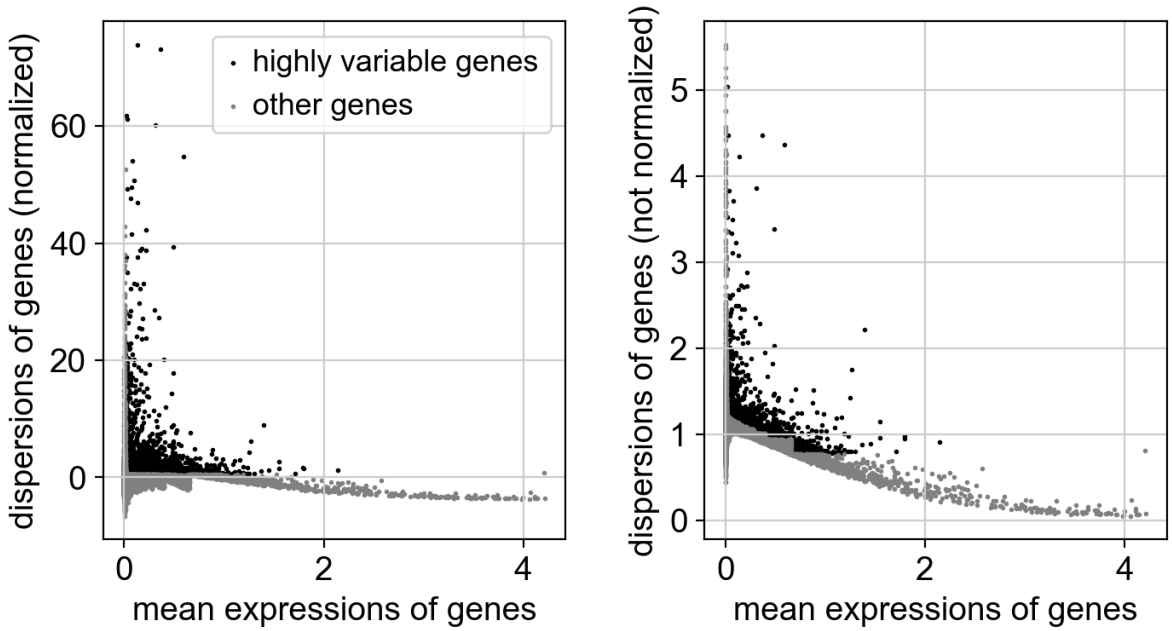
In [45]:

```
sc.pp.highly_variable_genes(adata, flavor='cell_ranger')#, n_top_genes=4000)
```

```
extracting highly variable genes
    finished (0:00:01)
--> added
    'highly_variable', boolean vector (adata.var)
    'means', float vector (adata.var)
    'dispersions', float vector (adata.var)
    'dispersions_norm', float vector (adata.var)

 Number of highly variable genes: 3083
```

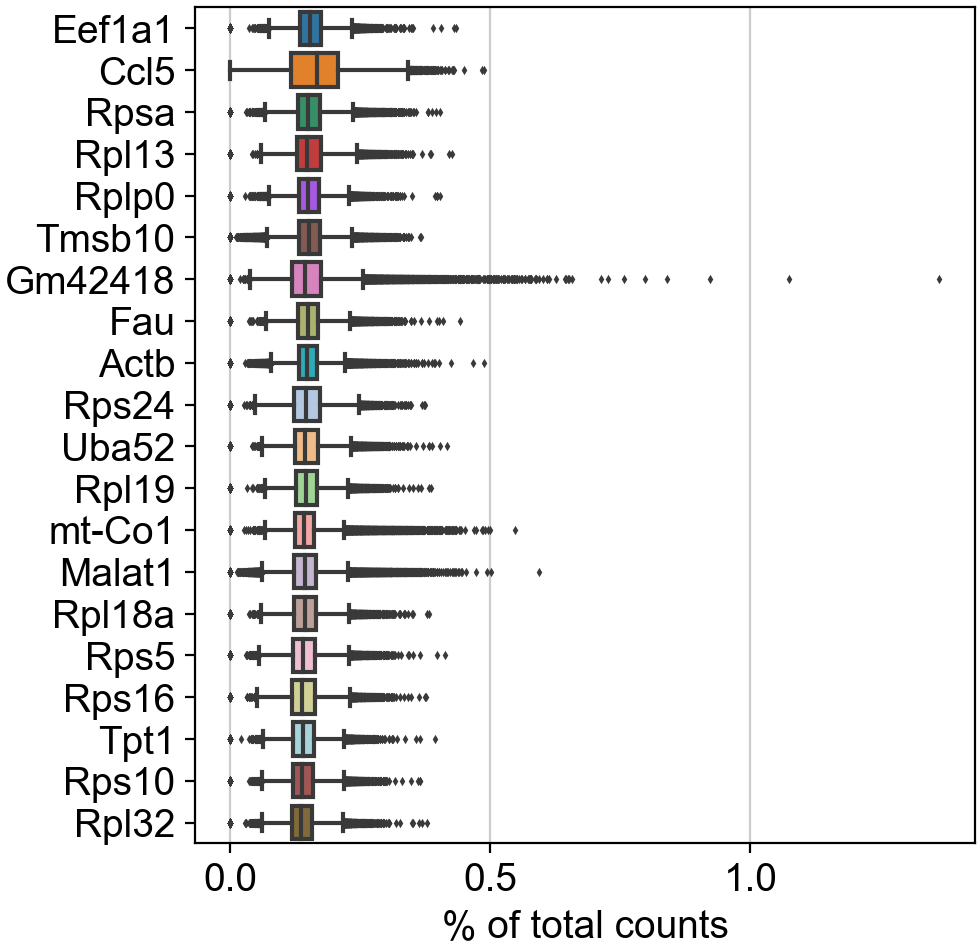
In [46]:

```
WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\filter_genes_dispersi
on_highly_variable.pdf
```



In [47]:

```
normalizing counts per cell
    finished (0:00:00)
WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\highest_expr_genes_to
p20_expressed.pdf
```



Filter out cell cycle and TCR chains

In [48]:

```
adata = adata[:, adata.var['highly_variable']]
sc.pp.regress_out(adata, ['n_counts', 'percent_mito', 'S_score', 'G2M_score'])
```

```
regressing out ['n_counts', 'percent_mito', 'S_score', 'G2M_score']
    sparse input is densified and may lead to high memory use
    finished (0:03:02)
```

In [49]:

```
# Remove TCR chains
adata=adata[:,adata.var_names[~np.logical_or(adata.var_names.str.contains('Trb
```

Principal component analysis

Reduce the dimensionality of the data by running principal component analysis (PCA), which

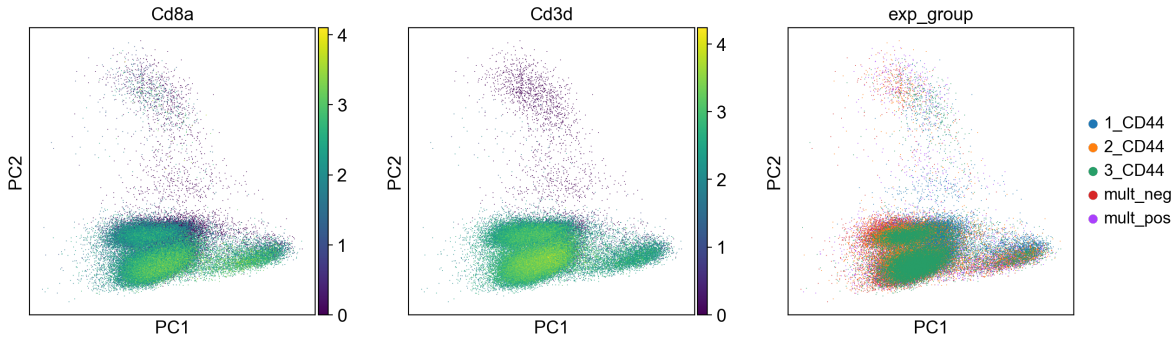
reveals the main axes of variation and denoises the data

In [50]:

```
computing PCA
    on highly variable genes
    with n_comps=50
    finished (0:00:18)
```

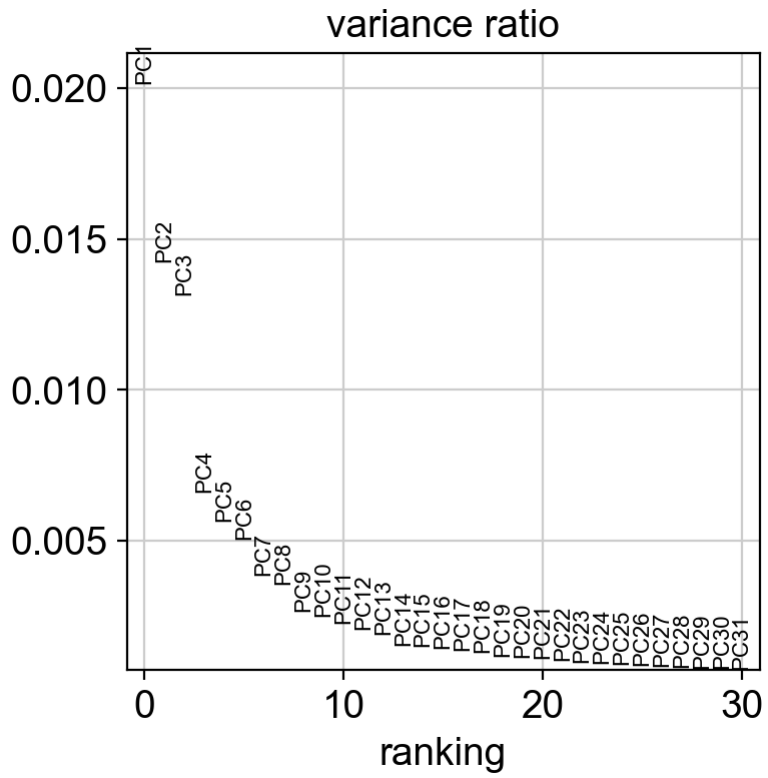
We can make a scatter plot in the PCA coordinates, but we will not use that later on.

In [51]:



Let us inspect the contribution of single PCs to the total variance in the data. This gives us information about how many PCs we should consider in order to compute the neighborhood relations of cells, e.g. used in the clustering function `sc.tl.louvain()` or tSNE `sc.tl.tsne()` . In our experience, often, a rough estimate of the number of PCs does fine.

In [52]:



Batch effect correction

In [53]:
```
#Batch balanced kNN method
import bbknn
```

```
computing batch balanced neighbors
        finished: added to `.uns['neighbors']`
        `.obsp['distances']`, distances for each pair of neighbors
        `.obsp['connectivities']`, weighted adjacency matrix (0:00:14)
```

Computing the neighborhood graph

Let us compute the neighborhood graph of cells using the PCA representation of the data matrix. You might simply use default values here.

In [54]:

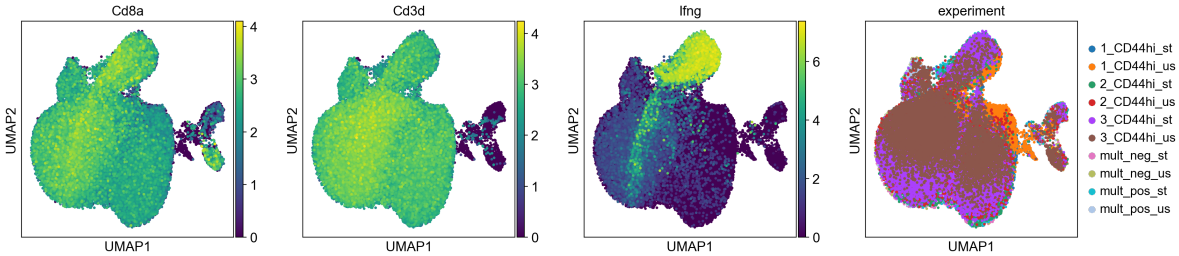
```
computing neighbors
    using 'X_pca' with n_pcs = 10
    finished: added to `.uns['neighbors']`
    `.obsp['distances']`, distances for each pair of neighbors
    `.obsp['connectivities']`, weighted adjacency matrix (0:00:03)
```

Embedding the neighborhood graph

In [55]:

```
computing UMAP
    finished: added
    'X_umap', UMAP coordinates (adata.obsm) (0:00:49)
```

In [56]:



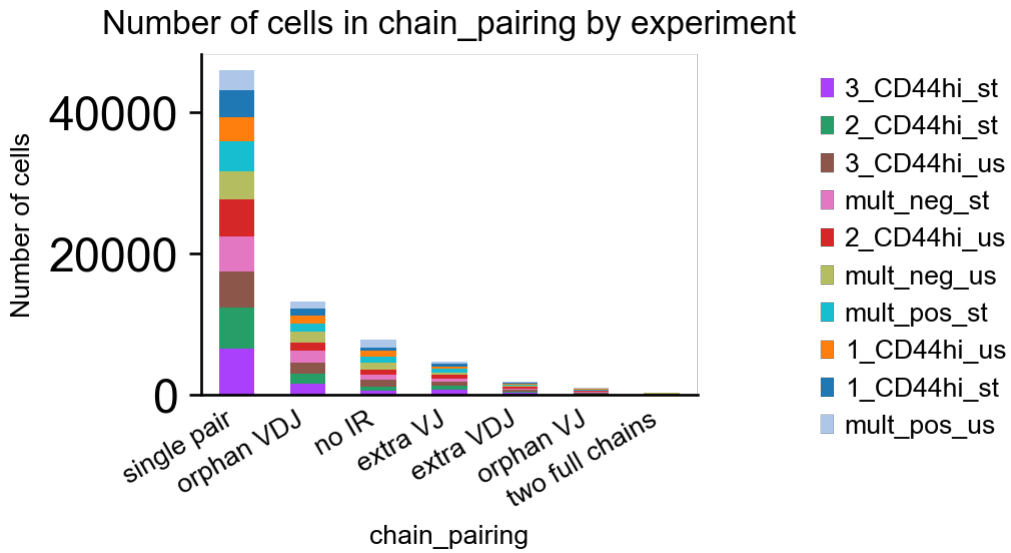
Integrate TCR expression

In [57]:

In [58]:

```
ax=ir.pl.group_abundance(adata, groupby='chain_pairing', target_col='experimen
#ax.figure.savefig(save_figure_ir + 'chain_pairing_leiden.png')
```

```
... storing 'receptor_type' as categorical
... storing 'receptor_subtype' as categorical
... storing 'chain_pairing' as categorical
```



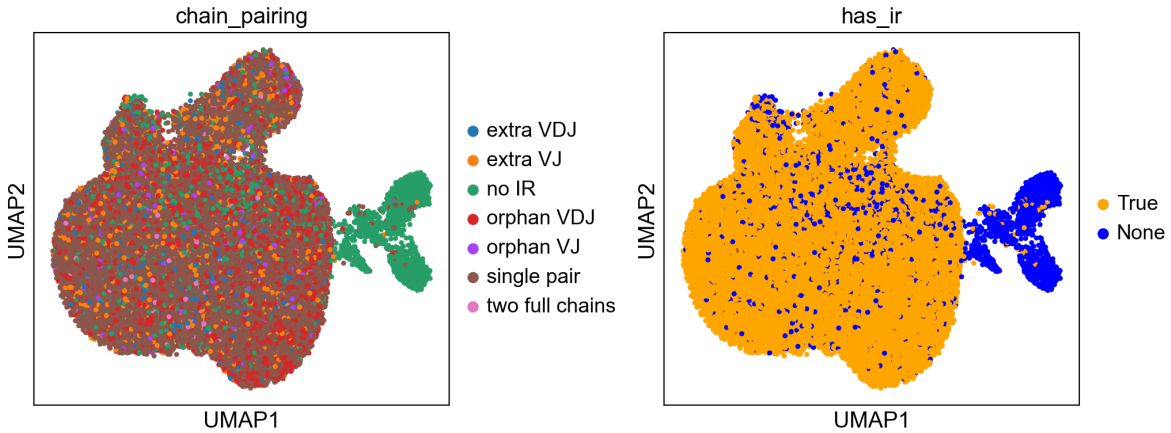
In [59]:
```
print('TCR sequencing efficiency: '+str(np.round(adata.obs.has_ir.value_counts
1))+'%')
print('Unique CDR3 alpha: '+str(adata.obs.IR_VJ_1_junction_aa.append(adata.obs
print('Unique CDR3 beta: '+str(adata.obs.IR_VDJ_1_junction_aa.append(adata.obs
```

```
TCR sequencing efficiency: 89.6%
Unique CDR3 alpha: 23999
Unique CDR3 beta: 36457
```

In [60]:
```
print("Fraction of Orphan VJ and VDJ: {:.2f}".format(
        np.sum(
            adata.obs["chain_pairing"].isin(["orphan VDJ", "orphan VJ"]
                                            )
                                )
                                /adata.n_obs * 100
            ) + '%'
```

```
Fraction of Orphan VJ and VDJ: 19.02%
```

In [61]:
```
fig, ax = plt.subplots(ncols=2, figsize=(10,4))
fig.tight_layout(w_pad=7)
sc.pl.umap(adata, color='chain_pairing', ax=ax[0], show=False, s=40)
sc.pl.umap(adata, color='has_ir', palette=['orange', 'blue'], ax=ax[1], show=F
```



Remove no TCR reads and orphan sequences

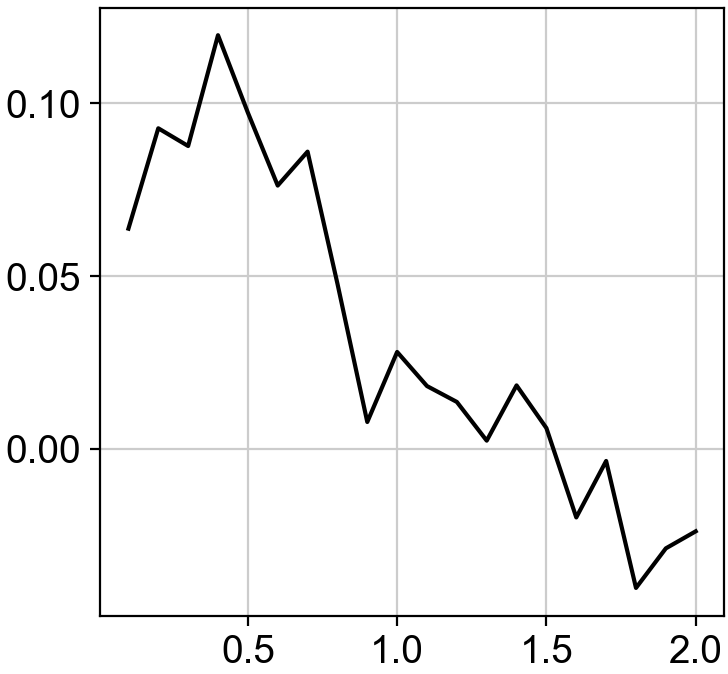
In [62]:
```
adata = adata[adata.obs["has_ir"] != 'None',:].copy()
```

Clustering the neighborhood graph

In [63]:
```
from sklearn.metrics import silhouette_score
def titrate_leiden_resolution(adata):
    res_array=list(np.round(np.linspace(.1, 2, 20),1))
    silhoutte_list=list()
    sc.settings.verbosity = 1
    for i,res in enumerate(res_array):
        sc.tl.leiden(adata, resolution=res)
        if adata.obs.leiden.unique().shape[0]>2:
            silhoutte_list.append(silhouette_score(adata.obsm['X_umap'],
                    adata.obs[f'leiden'],
                    metric='euclidean'))
        else:
            silhoutte_list.append(np.nan)
        print(str(np.round(((i+1)/len(res_array))*100,1))+"%", end="\r", flush
```

In [64]:

```
100.0%
```

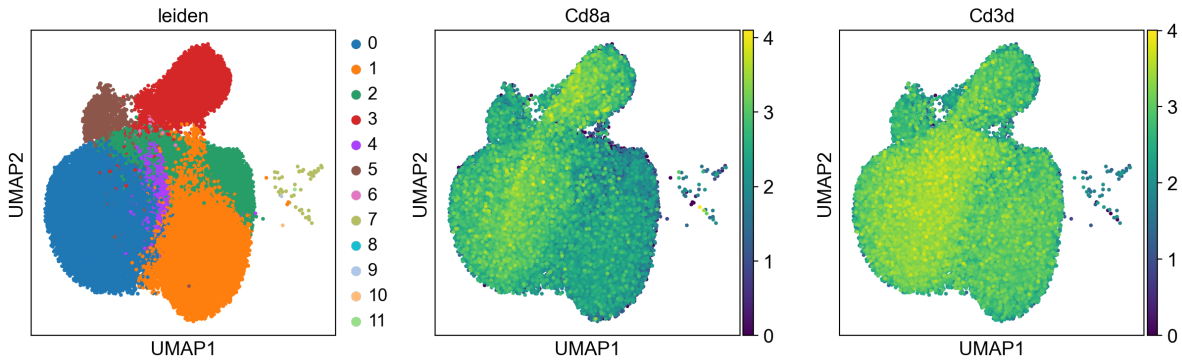


In [65]:
```
sc.tl.leiden(adata, resolution=0.3)
```

```
0.08768394
```

In [66]:

```
WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\umap_leiden.pdf
```



Remove impurities

In [67]:

Rerun clustering

In [68]:

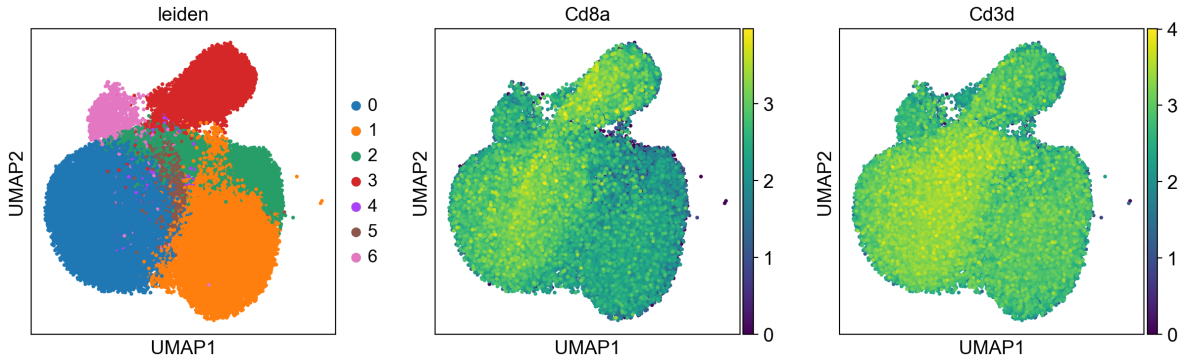
```
sc.tl.leiden(adata, resolution=0.3)
```

```
Trying to set attribute `.obs` of view, copying.
```

```
0.11444347
```

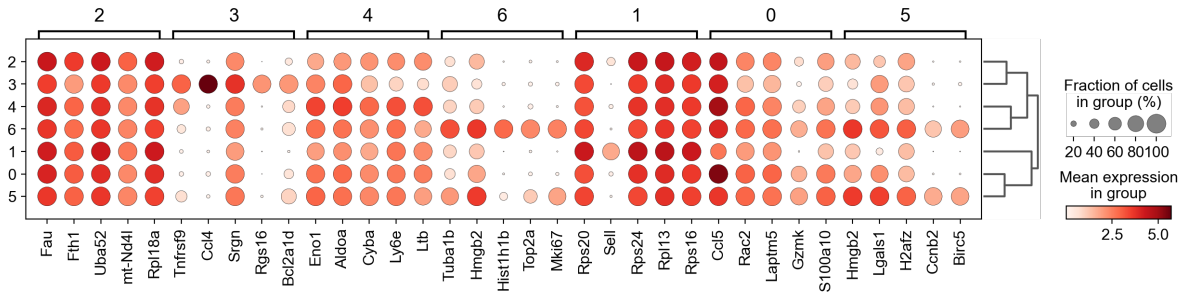
In [69]:

```
WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\umap_leiden.pdf
```



In [70]:
```
sc.tl.dendrogram(adata, groupby='leiden')
sc.tl.rank_genes_groups(adata, 'leiden', use_raw=True, method='t-test')
```

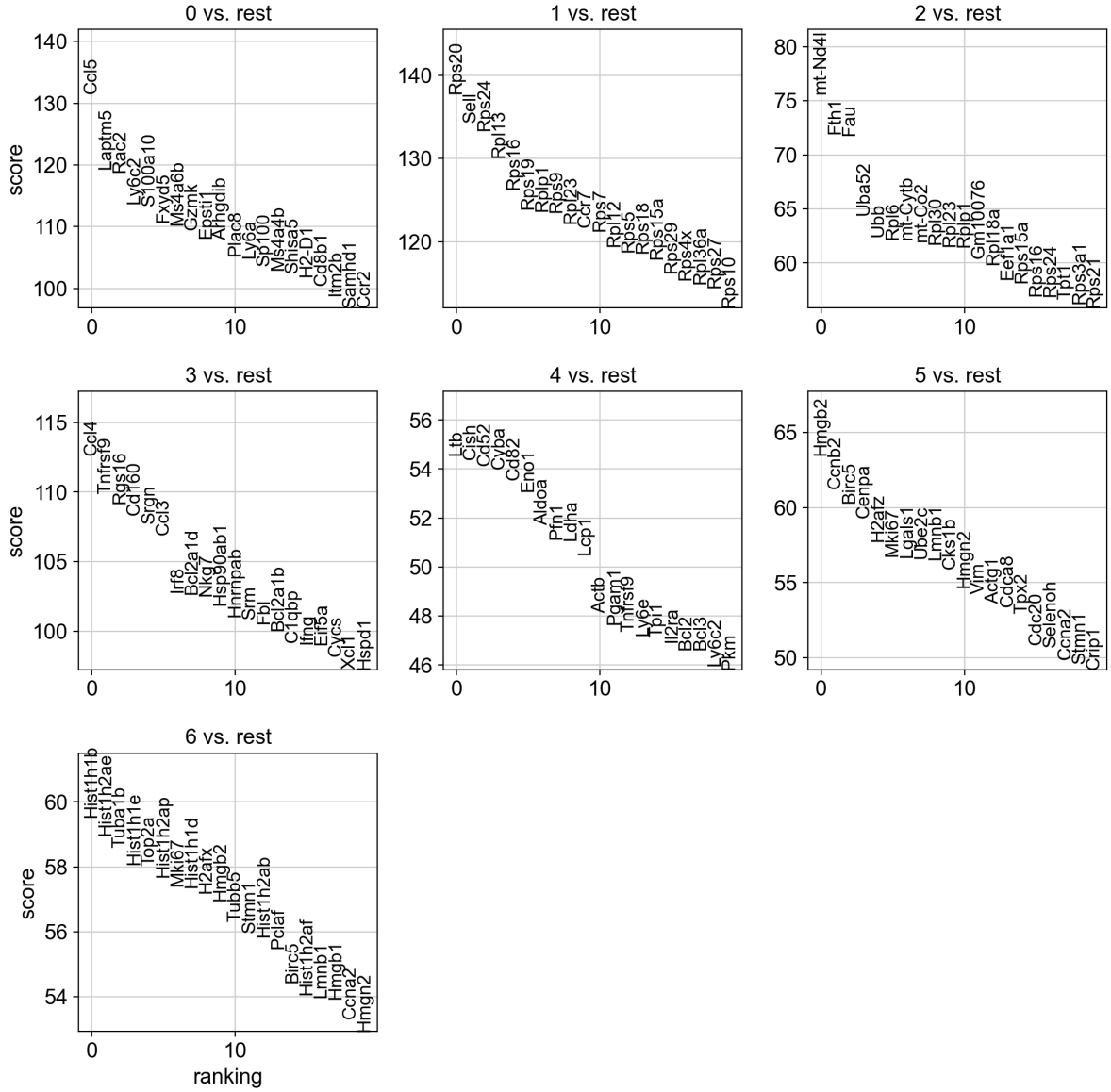
Out[70]:
```
{'mainplot_ax': <AxesSubplot:>,
 'group_extra_ax': <AxesSubplot:>,
 'gene_group_ax': <AxesSubplot:>,
 'size_legend_ax': <AxesSubplot:title={'center':'Fraction of cells\nin group
(%)'}>,
 'color_legend_ax': <AxesSubplot:title={'center':'Mean expression\nin group'}
>}
```



In [71]:
```
#Calculate marker genes
```

In [72]:
```
#Plot marker genes
sc.pl.rank_genes_groups(adata, n_genes=20, sharey=False, ncols=3, fontsize=12,
```

```
WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\rank_genes_groups_lei
den_genes_by_group.pdf
```



Clonotype analysis

Analysis based on nucleotide identity

In [73]:
```
# using default parameters, `ir_dist` will compute nucleotide sequence identit
```

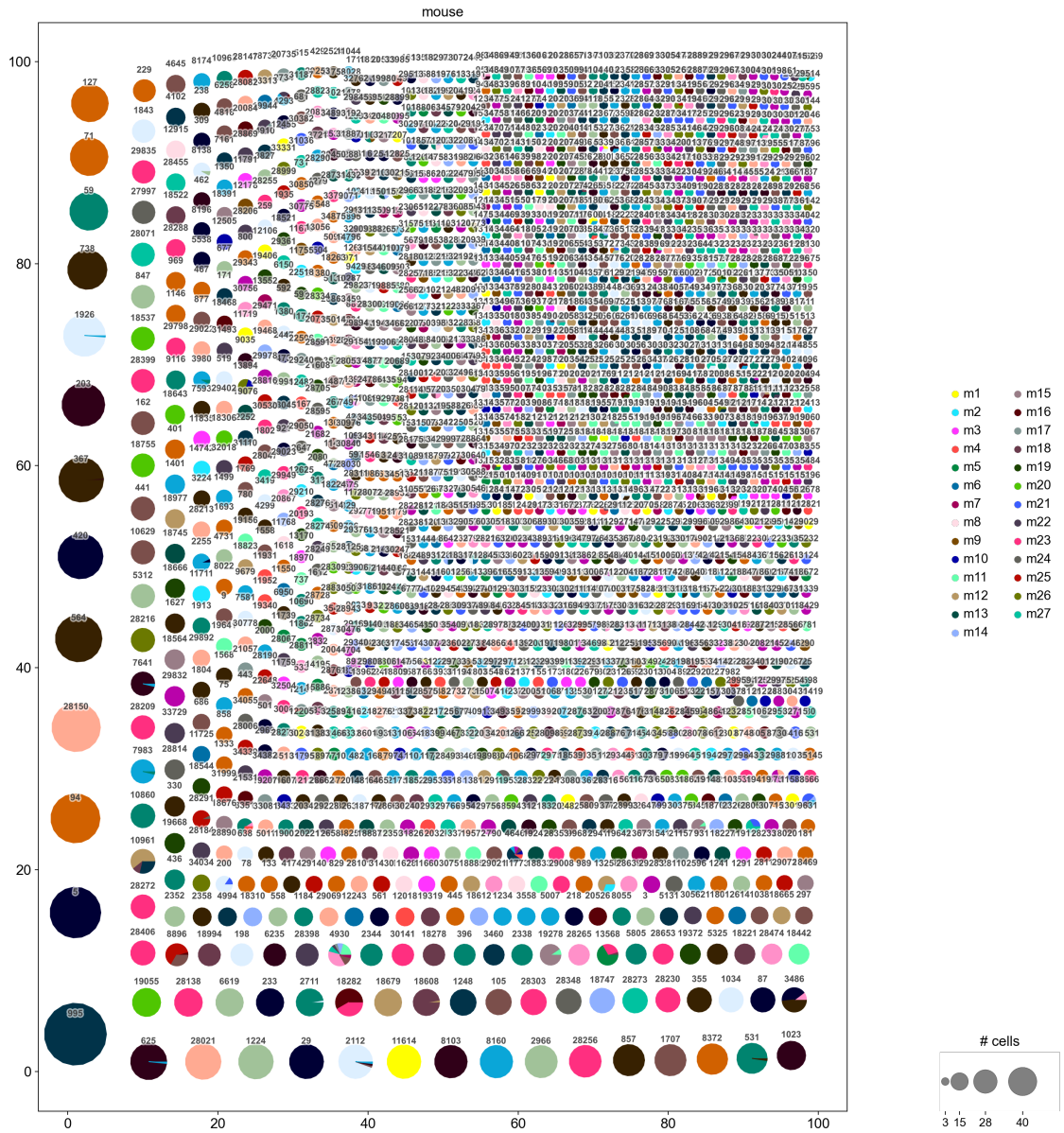
In [74]:

```
  0%|          | 0/37344 [00:00<?, ?it/s]
```

In [75]:

In [76]:
```
fig, ax = plt.subplots(figsize=(16,16))
ir.pl.clonotype_network(adata, color='mouse', base_size=20, label_fontsize=9,

fig.tight_layout()
```

```
... storing 'clone_id' as categorical
```



Analysis based on amino acid identity

In [77]:
```
ir.pp.ir_dist(
    adata,
    metric="alignment",
    sequence="aa",
    cutoff=15,
```

```
  0%|          | 0/113050 [00:00<?, ?it/s]

  0%|          | 0/182106 [00:00<?, ?it/s]
```

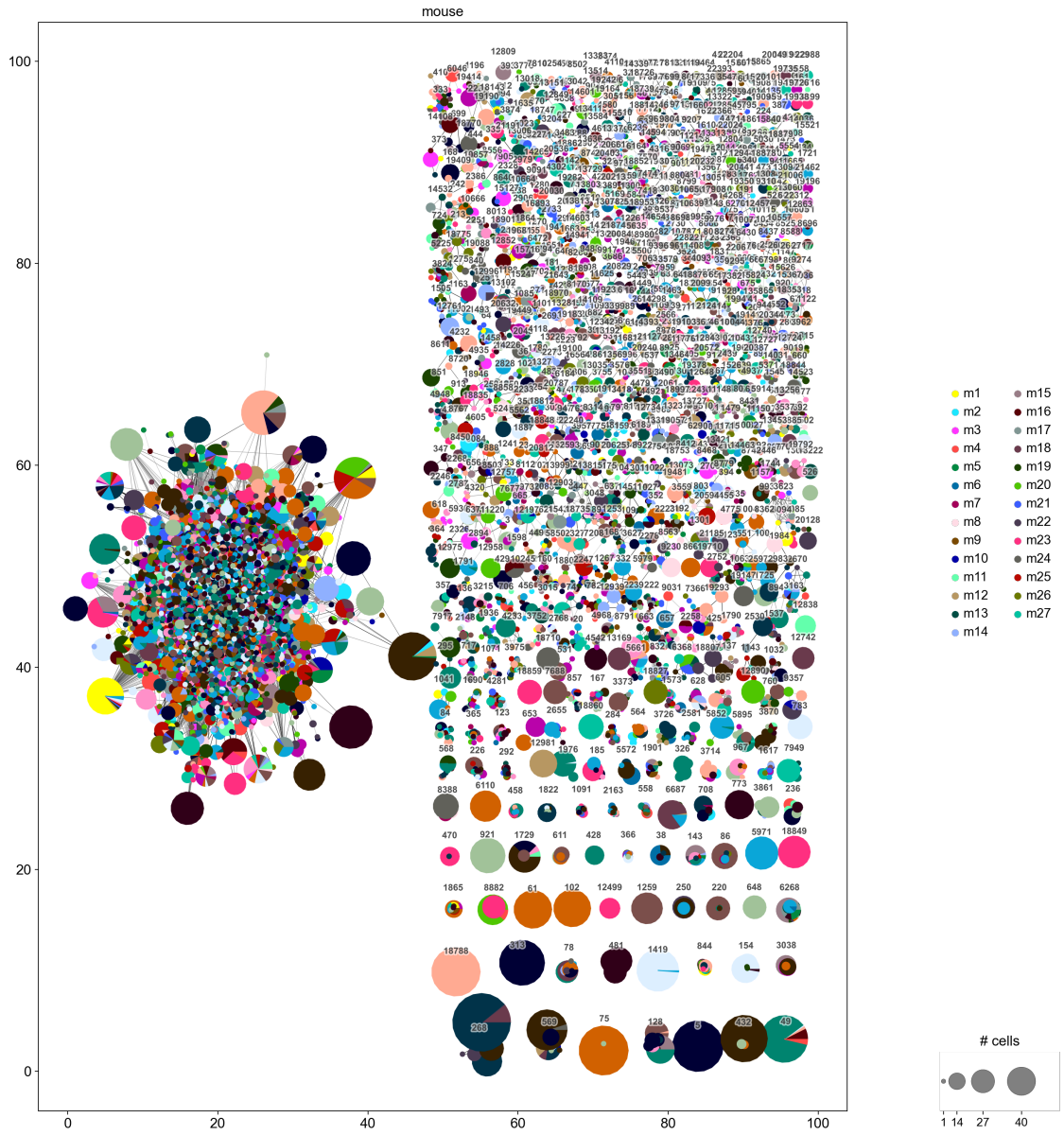
In [78]:
```
ir.tl.define_clonotype_clusters(
    adata, sequence="aa", metric="alignment", receptor_arms="all", dual_ir="an
```

```
  0%|          | 0/37207 [00:00<?, ?it/s]
```

In [79]:

In [80]:
```
fig, ax = plt.subplots(figsize=(16,16))
ir.pl.clonotype_network(adata, color="mouse", label_fontsize=9, panel_size=(10

fig.tight_layout()
```

```
... storing 'cc_aa_alignment' as categorical
```



Clonotype Cluster 9 with large overlaps between mice

Include V-gene in clonotype clustering

In [81]:
```
ir.tl.define_clonotype_clusters(
    adata,
    sequence="aa",
    metric="alignment",
    receptor_arms="all",
    dual_ir="any",
    same_v_gene=True,
    key_added="cc_aa_alignment_same_v",
```

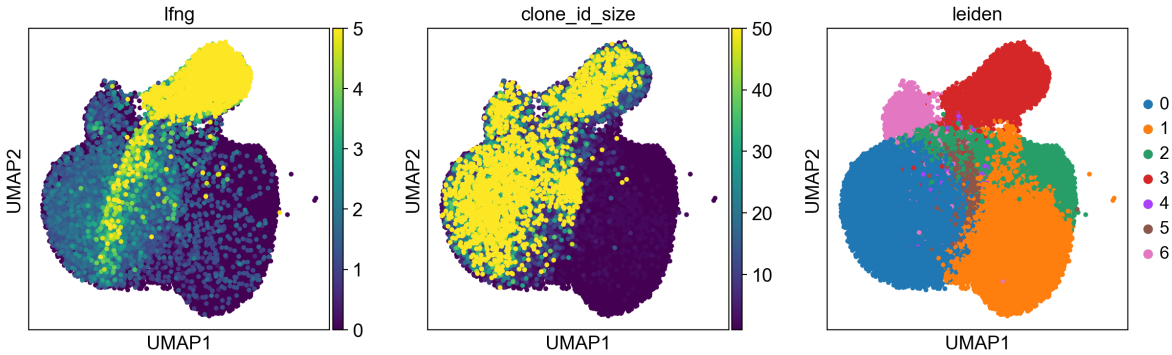
```
  0%|          | 0/37548 [00:00<?, ?it/s]
```

Clonal expansion

In [82]:

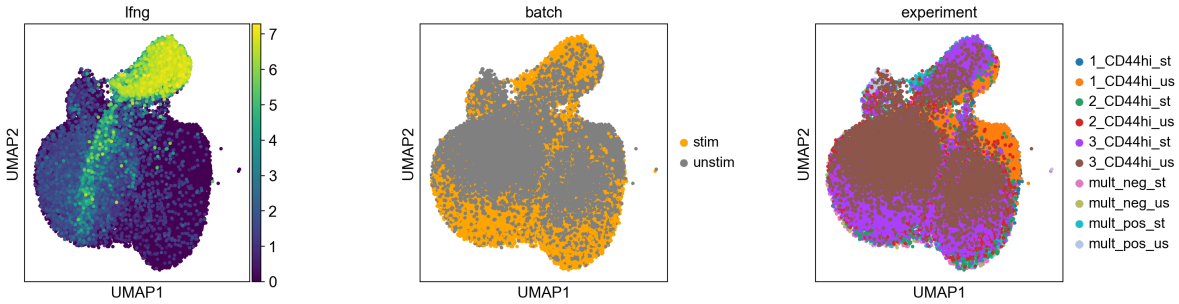
In [83]:

```
... storing 'cc_aa_alignment_same_v' as categorical
... storing 'clonal_expansion' as categorical

WARNING: saving figure to file C:\Users\ga36xaw\10X_Analysis\repertoire_satur
ation\Annotation_stim_unstim\analysis_output\figures_sc\umap_clonal_expansio
n.pdf
```

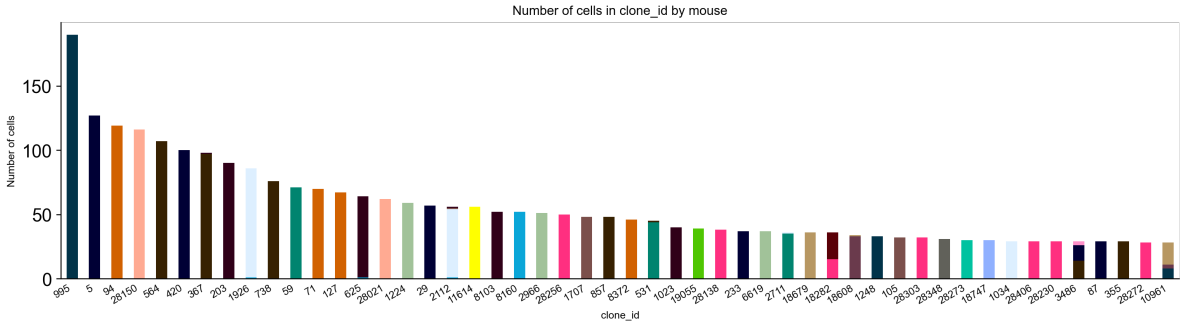


In [84]:
```
#sc.pl.umap(adata, color=["Ifng", "batch", 'experiment'], s=30, save='_batch.p

fig, ax = plt.subplots(ncols=3, figsize=(14,4))
fig.tight_layout(w_pad=7)
sc.pl.umap(adata, color="Ifng", ax=ax[0], show=False, s=40)
sc.pl.umap(adata, color="batch", palette=['orange','grey'], ax=ax[1], show=Fal
sc.pl.umap(adata, color="experiment", ax=ax[2], show=False, s=40)
```

Out[84]: <AxesSubplot:title={'center':'experiment'}, xlabel='UMAP1', ylabel='UMAP2'>



In [85]:
```
ax = ir.pl.group_abundance(
    adata, groupby="clone_id", target_col="mouse", max_cols=50, figsize=(20, 5
ax.legend([], frameon=False)
```



Store adata file for annotated clonotypes

In [86]:

In [ ]:

In [ ]:

In [ ]: