In [1]:
```
# Autoreaload
%load_ext autoreload
%autoreload 2
```

In [2]:
```
# Imports
import pandas as pd
import re
import scanpy as sc
from scipy.sparse import csr_matrix
from IPython.display import Markdown as md, display
from helper import add_top_column
```

Paper

CRISPR activation and interference screens decode stimulation responses in primary human T cells

https://www.science.org/doi/10.1126/science.abj4008#body-ref-R56-1

Data

- https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE190604
- https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE174292
- https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE190846
- https://zenodo.org/records/5784651 (R Repository with results)

We performed CRISPRa Perturb-seq characterization of regulators of stimulation responses in ~56,000 primary human T cells, targeting 70 hits and controls from our genome-wide CRISPRa cytokine screens (Fig. 4, A and B, and fig. S17, A to C). First, we confirmed that sgRNAs led to significant increases in the expression of their target genes (fig. S17D). Next, uniform manifold approximation and projection (UMAP) dimensionality reduction revealed discrete separation of the resting and restimulated cells (fig. S17E) and showed relatively even distribution of cells from two donors (Fig. 4C and fig. S17F). Gene signatures allowed us to resolve most T cells as either CD4+ or CD8+ (Fig. 4D and fig. S17, G and H). Thus, we generated a high-quality CRISPRa Perturb-seq dataset.

Data experimento:

- ~150 sgRNA
- ~ 70 Target Gene
- ~56,000 primary human T cells

Data Loading and preprocess

Raw Data (scRNA-seq)

In [3]:
```
# Open data file
adata = sc.read_10x_mtx(
    '../data/GSE190604/',
    prefix='GSE190604_',
    cache=True,
    gex_only=False, # Only gene expression
    # make_unique=True
)
```

In [4]:
```
# Cell Barcodes
adata.obs
```

Out[4]:

AAACCCACAACAAGAT-1
AAACCCACAACGGCTC-1
AAACCCACACAGAAGC-1
AAACCCACACCCTGTT-1
AAACCCACACTATGTG-1
...
TTTGTTGGTCCAGCCA-8
TTTGTTGGTCCCACGA-8
TTTGTTGGTGAGTGAC-8
TTTGTTGGTGCAAGAC-8
TTTGTTGTCTTCGCTG-8

103805 rows × 0 columns

In [5]:
```
display(md(f"""#### Raw data has {adata.shape[1]} genes and {adata.shape[0]}
display(md(f"""#### Cells are identified by barcode (e.g.: {', '.join(adata.
display(md(f"""#### The number -1, -2, etc, represent the chromium well"""))
display(md(f"""#### A chromium well refers to the microfluidic chambers in t
```

Raw data has 36755 genes and 103805 cells

Cells are identified by barcode (e.g.: AAACCCACAACAAGAT-1, AAACCCACAACGGCTC-1, AAACCCACACAGAAGC-1, etc)

The number -1, -2, etc, represent the chromium well

A chromium well refers to the microfluidic chambers in the 10x Genomics Chromium Controller that are used to encapsulate single cells and barcoded beads into individual droplets, enabling high-throughput single-cell genomics. These wells play a key role in isolating single cells, capturing their RNA, and associating it with unique barcodes for sequencing.

In [6]:
```
display(md("##### Gene Expression: Represent the transcriptomic profile of t
display(md("##### CRISPR Guide Capture: Which sgRNAs (and therefore which ge
display(adata.var)
display(adata.var.feature_types.value_counts().reset_index())
```

Gene Expression: Represent the transcriptomic profile of the cells (the genes that are being expressed)

CRISPR Guide Capture: Which sgRNAs (and therefore which genes) were targeted in each cell.

	gene_ids	feature_types
MIR1302-2HG	ENSG00000243485	Gene Expression
FAM138A	ENSG00000237613	Gene Expression
OR4F5	ENSG00000186092	Gene Expression
AL627309.1	ENSG00000238009	Gene Expression
AL627309.3	ENSG00000239945	Gene Expression
...	...	...
TRIM21-2	TRIM21-2	CRISPR Guide Capture
VAV1-1	VAV1-1	CRISPR Guide Capture
VAV1-2	VAV1-2	CRISPR Guide Capture
WT1-1	WT1-1	CRISPR Guide Capture
WT1-2	WT1-2	CRISPR Guide Capture

36755 rows × 2 columns

	feature_types	count
0	Gene Expression	36601
1	CRISPR Guide Capture	154

Cell Metadata

From the CRISPR Guide Capture columns we can get the cell metada

reference: 01_build_metadata_table_for_guide_calls

In [7]:
```
df_cell_metadata = pd.read_csv('../data/cell_metadata.txt', sep='\t')
df_cell_metadata['well'] = df_cell_metadata['cell_barcode'].apply(lambda x:
df_cell_metadata = df_cell_metadata.set_index('cell_barcode')
df_cell_metadata
```

Out[7]:

cell_barcode	condition	crispr	guide_id	gene	well
GGGAGATAGACCGTTT-1	Nostim	perturbed	ABCB10-1	ABCB10	1
GACGCTGCATTGTCGA-1	Nostim	perturbed	ABCB10-1	ABCB10	1
TTAATCCTCGTGCACG-1	Nostim	perturbed	ABCB10-1	ABCB10	1
ACACGCGTCGACCTAA-1	Nostim	perturbed	ABCB10-1	ABCB10	1
CATCCACCATCGATGT-1	Nostim	perturbed	ABCB10-1	ABCB10	1
...	...	...	...	...	...
GTTGTCCGTGGTTTAC-8	Stim	perturbed	WT1-2	WT1	8
GTCTAGAAGGCACTCC-8	Stim	perturbed	WT1-2	WT1	8
TCCTAATCATACACCA-8	Stim	perturbed	WT1-2	WT1	8
AGACCCGGTATTGACC-8	Stim	perturbed	WT1-2	WT1	8
AGTGTTGTCATTTACC-8	Stim	perturbed	WT1-2	WT1	8

60657 rows × 5 columns

In [8]:
```
display(md("#### Perturbed vs No TARGET"))
display(df_cell_metadata['crispr'].value_counts().reset_index())
display(md('# ~56,000 cels'))
```

Perturbed vs No TARGET

	crispr	count
0	perturbed	56774
1	NT	3883

~56,000 cels

In [9]:
```
display(md("#### Total cells per guide id"))
display(df_cell_metadata[['guide_id']].value_counts().reset_index())
```

```
display(md('# ~150,000 sgRNAs'))
# Un sgRNA activa solo una porción del gen
```

Total cells per guide id

	guide_id	count
0	TRAF3IP2-1	1056
1	LAT2-2	983
2	EMP3-1	931
3	CD27-1	849
4	TNFRSF1B-2	797
...	...	...
149	PRDM1-2	20
150	IRX4-2	15
151	DEF6-2	11
152	IRX4-1	2
153	TCF7-1	1

154 rows × 2 columns

~150,000 sgRNAs

In [10]:
```
display(md("#### Total cells per gene"))
display(df_cell_metadata['gene'].value_counts().reset_index())
display(md('# ~70 genes'))
```

Total cells per gene

	gene	count
0	NO-TARGET	3883
1	EMP3	1588
2	TRAF3IP2	1564
3	CD27	1455
4	TNFRSF1B	1344
...	...	...
69	EOMES	262
70	HELZ2	128
71	TCF7	115
72	PRDM1	40
73	IRX4	17

74 rows × 2 columns

~70 genes

In [11]:
```
display(md("#### Total cells per gene per guide_id"))
genes = df_cell_metadata['gene'].unique()[:4]
'WT1'
all_genes_count = []
for gene in genes:
    gene_count_df = df_cell_metadata[df_cell_metadata['gene'] == gene].value
    gene_count_df = add_top_column(gene_count_df, gene)
    all_genes_count.append(gene_count_df)
pd.concat(all_genes_count, axis=1)
```

Total cells per gene per guide_id

Out[11]:

	ABCB10		AKAP12		ALX4		APOBEC3C	
	guide_id	count	guide_id	count	guide_id	count	guide_id	count
0	ABCB10-2	208	AKAP12-1	255	ALX4-1	380	APOBEC3C-1	486
1	ABCB10-1	189	AKAP12-2	104	ALX4-2	132	APOBEC3C-2	168

In [12]:
```
display(md("#### Total cells per well"))
df_cell_metadata.value_counts('well').reset_index()
```

Total cells per well

Out[12]:

	well	count
0	2	7838
1	3	7750
2	1	7742
3	5	7642
4	7	7502
5	4	7491
6	6	7430
7	8	7262

In [13]:
```
# sgRNAs are uniformly distributed per chromium well
sgRNAs = df_cell_metadata['guide_id'].unique()[:10]
sgRNAcounts = pd.DataFrame([{'well': i} for i in range(8)])
for sgRNA in sgRNAs:
    sgRNAcounts[sgRNA] = df_cell_metadata[df_cell_metadata['guide_id'] == sg
sgRNAcounts.set_index('well')
```

Out[13]:

	ABCB10-1	ABCB10-2	AKAP12-1	AKAP12-2	ALX4-1	ALX4-2	APOBEC3C-1	APOBEC3C-2	AP
well									
0	29	37	45	19	62	26	71	27	
1	28	34	39	18	51	20	70	26	
2	24	33	34	14	48	19	62	25	
3	24	27	33	12	47	16	62	21	
4	24	26	30	12	47	15	59	19	
5	20	22	26	12	44	14	57	18	
6	20	15	24	10	41	12	54	18	
7	20	14	24	7	40	10	51	14	

Merge and clean data

In [14]:
```
# Merge cell metadata with matrix
adata.obs = adata.obs.merge(df_cell_metadata, left_index=True, right_index=T
adata.obs
```

Out[14]:

	condition	crispr	guide_id	gene	well
AAACCCACAACAAGAT-1	Nostim	perturbed	PLCG2-2	PLCG2	1
AAACCCACAACGGCTC-1	Nostim	perturbed	HELZ2-1	HELZ2	1
AAACCCACACAGAAGC-1	NaN	NaN	NaN	NaN	NaN
AAACCCACACCCTGTT-1	Nostim	perturbed	OTUD7B-1	OTUD7B	1
AAACCCACACTATGTG-1	Nostim	perturbed	CD247-1	CD247	1
...	...	...	...	...	...
TTTGTTGGTCCAGCCA-8	NaN	NaN	NaN	NaN	NaN
TTTGTTGGTCCCACGA-8	NaN	NaN	NaN	NaN	NaN
TTTGTTGGTGAGTGAC-8	Stim	perturbed	IL2RB-1	IL2RB	8
TTTGTTGGTGCAAGAC-8	NaN	NaN	NaN	NaN	NaN
TTTGTTGTCTTCGCTG-8	NaN	NaN	NaN	NaN	NaN

103805 rows × 5 columns

In [32]:
```
genes = df_cell_metadata['gene'].unique()
set(genes) - set(adata.var.index)
```

Out[32]:
```
{'NO-TARGET'}
```

In [41]:
```
[g for g in adata.var.index if 'targ' in g.lower()]
```

Out[41]:
```
['NO-TARGET-1',
 'NO-TARGET-2',
 'NO-TARGET-3',
 'NO-TARGET-4',
 'NO-TARGET-5',
 'NO-TARGET-6',
 'NO-TARGET-7',
 'NO-TARGET-8']
```

In [52]:
```
display(md("#### Data merged"))
adata_filtered = adata[~adata.obs['guide_id'].isna(), adata.var.index.isin(g
display(adata_filtered)
display(md("#### Around 4000 cells have crispr NT"))
```

Data merged

```
View of AnnData object with n_obs × n_vars = 60657 × 73
    obs: 'condition', 'crispr', 'guide_id', 'gene', 'well'
    var: 'gene_ids', 'feature_types'
```

Around 4000 cells have crispr NT

In [45]:
```
sc.write('../data_out/matrix_filtered.h5ad', adata_filtered)
```

In [54]:
```
# Around 4000 cells have crispr NT
(adata_filtered.obs['crispr'] == 'NT').sum()
```

Out[54]: 3883