Analysis and interpretation of single-cell RNA-seq data

Cold Spring Harbor
Advanced Sequencing Technologies
Nov 18, 2021

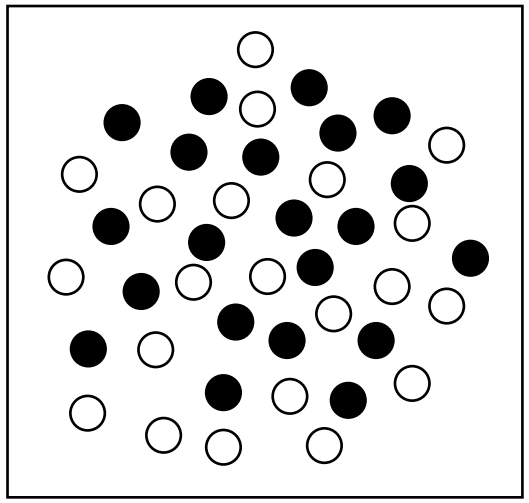
Chris Miller, PhD
Washington University in St. Louis
Division of Oncology
c.a.miller@wustl.edu

(adapted with permission from Allegra Petti, PhD)

Single-cell RNA-seq captures expression heterogeneity

Identifies and counts unique transcripts in each cell

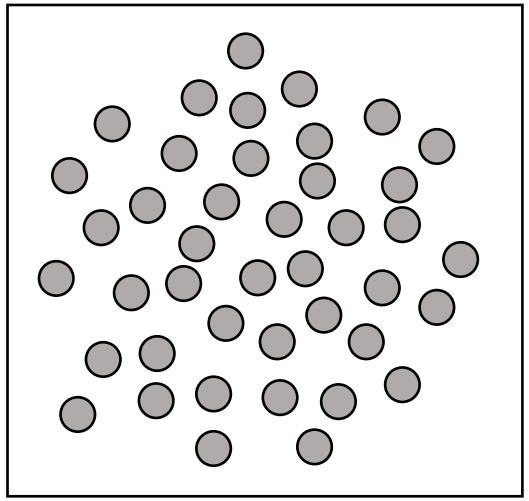
For one gene:

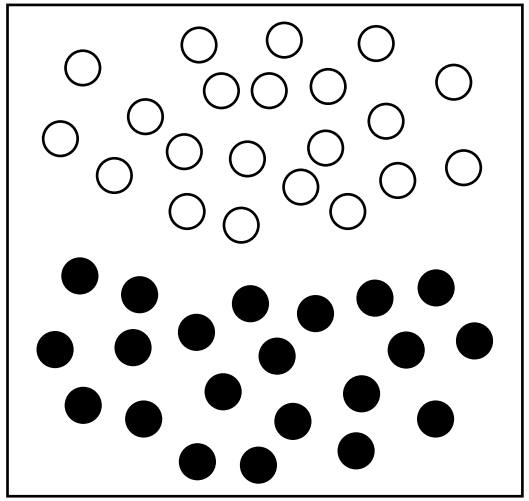


Gene X ON
Gene X OFF

Bulk

Single-cell



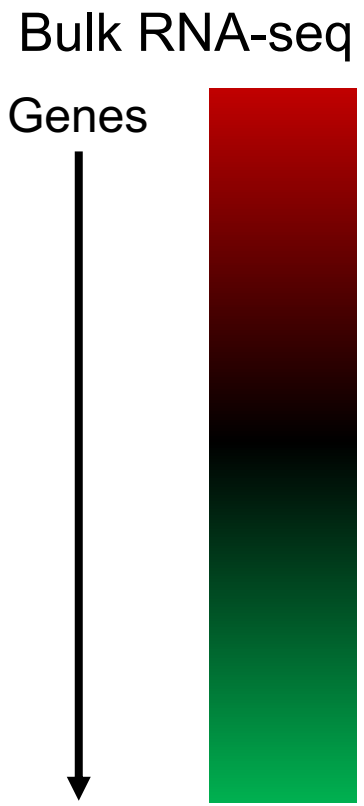


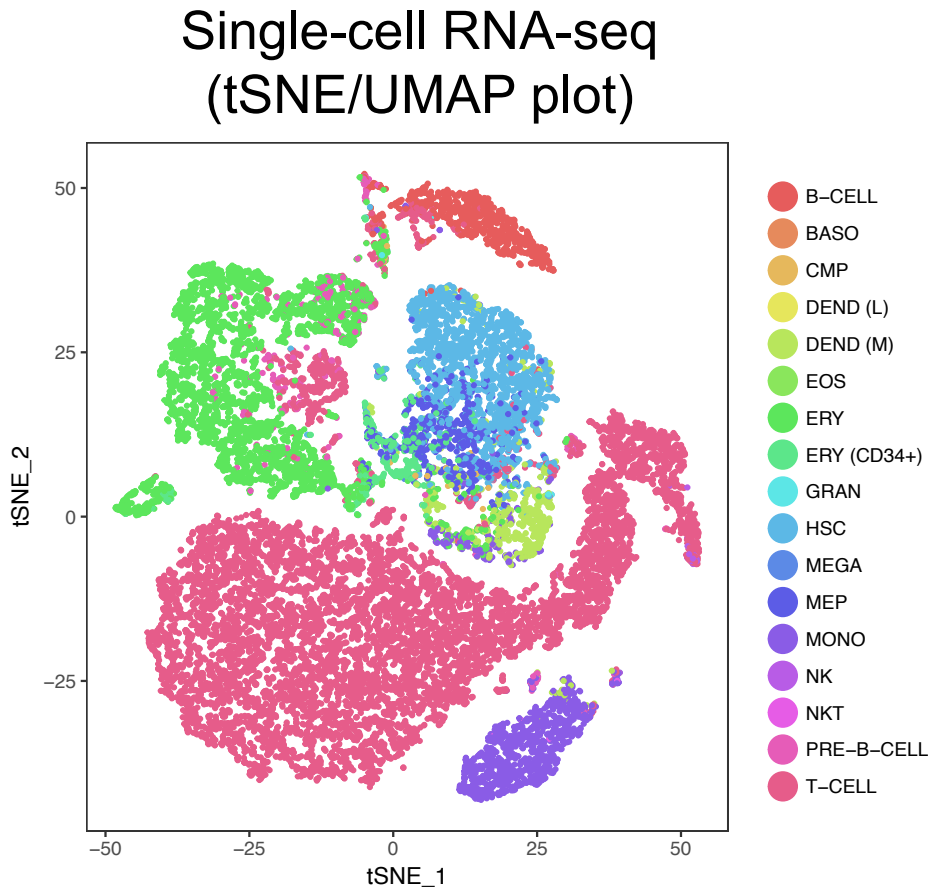
Bulk RNA-seq averages across the population

scRNA-seq reports per-cell expression and enables computational "sorting"

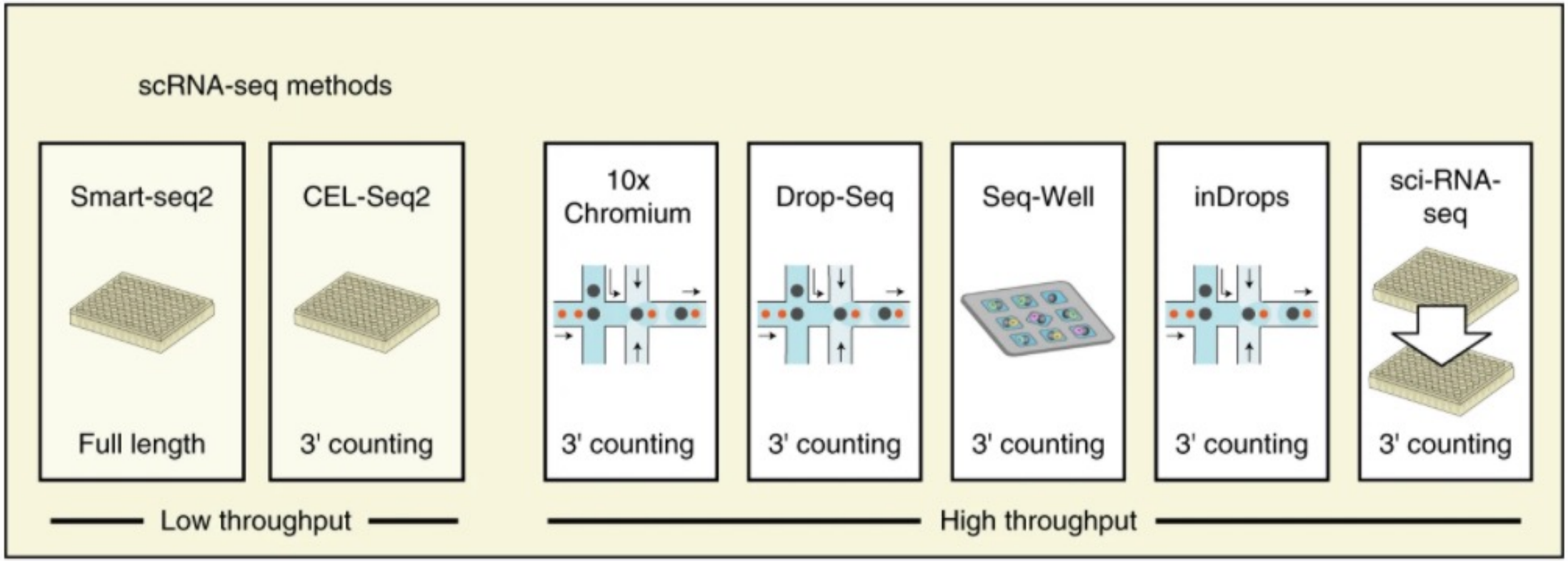
Single-cell RNA-seq captures expression (and genetic) heterogeneity

Identifies and counts unique transcripts in each cell





Technology comparison



Ding J, et al. (2020) Nature Biotech. 38:737-746

Popular commercial platforms: 10x Genomics vs Fluidigm SMART-Seq

- **Drop-Seq: 10x Genomics**
  - 3’ V3.1 GEM Gene expression
  - 5’ V1 Gene expression (more sequence; less-biased coverage; TCR/BCR sequencing)

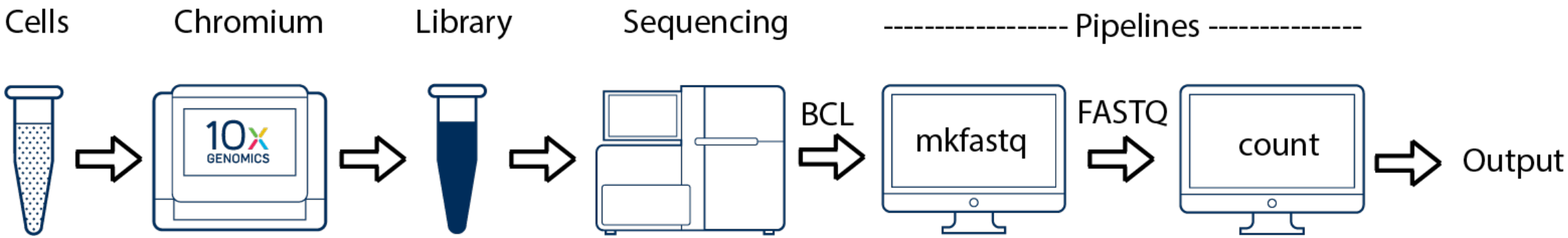
- Plate-based: Fluidigm
  - C1 SMART-Seq2: lower-throughput, more genes/cell, longer cDNAs, no UMIs
  - SMART-Seq3*: lower-throughput, more genes/cell, longer cDNAs (uses UMIs)

- All have limitations: must choose technology best suited to application
  - Lafzi et al, *Nature Protocols* 13:2742-2757

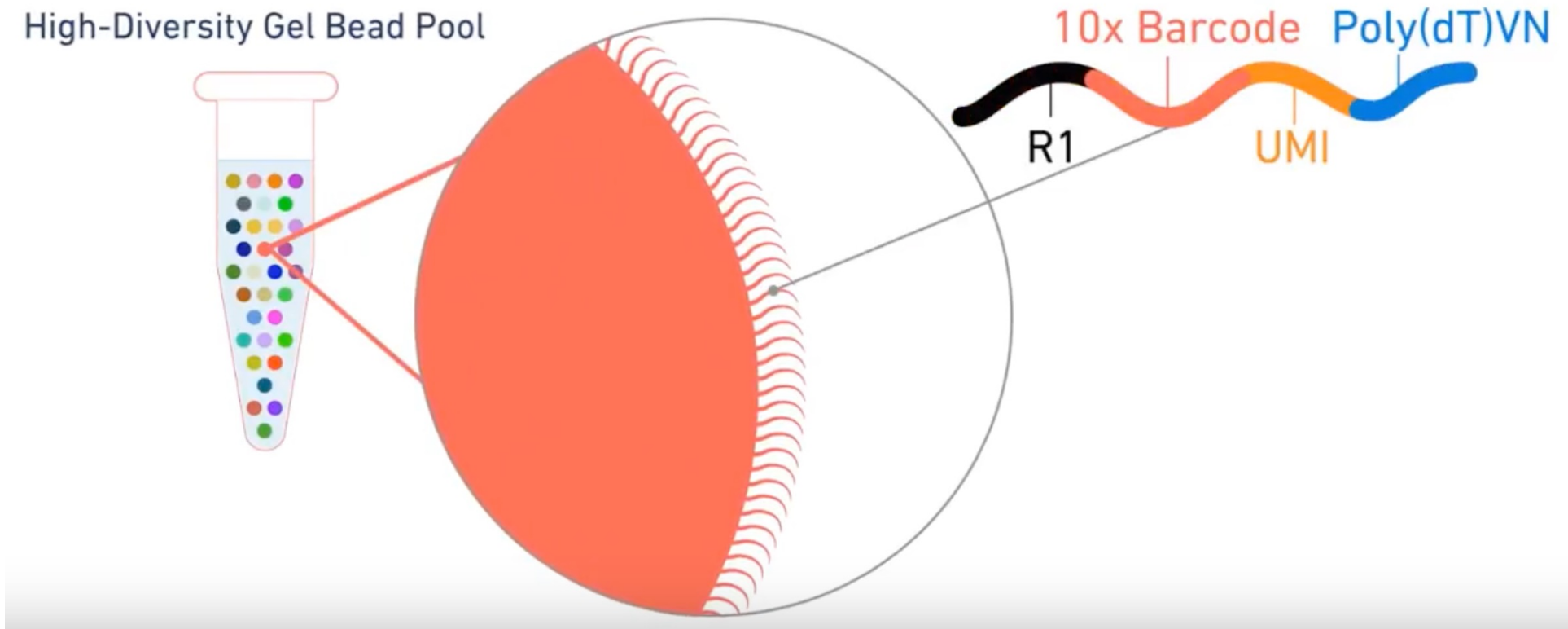
Extensions/Variations

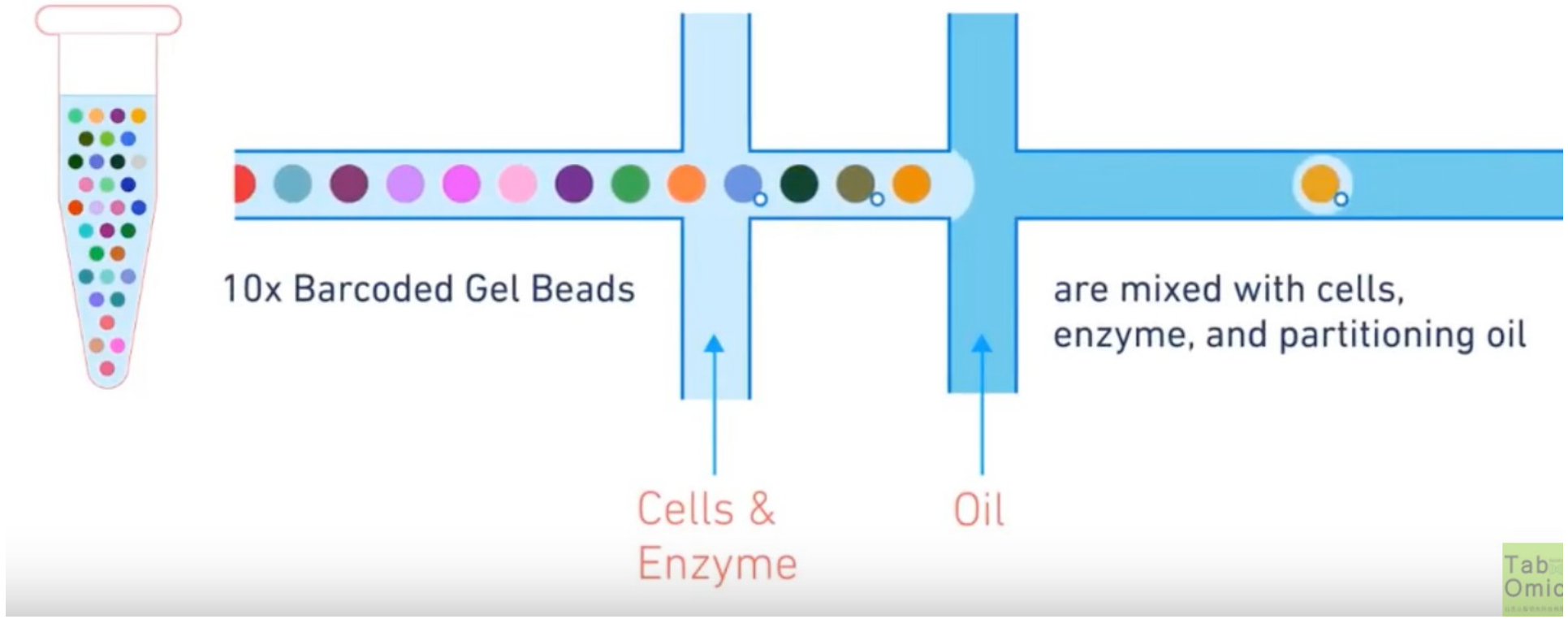
- Single-nucleus RNA-sequencing for frozen or hard-to-dissociate tissues
- CITE-seq (aka “feature barcoding”)
- scATAC-seq
- scMethyl-seq
- TARGET-seq, G&T-seq
- CRISPR screens

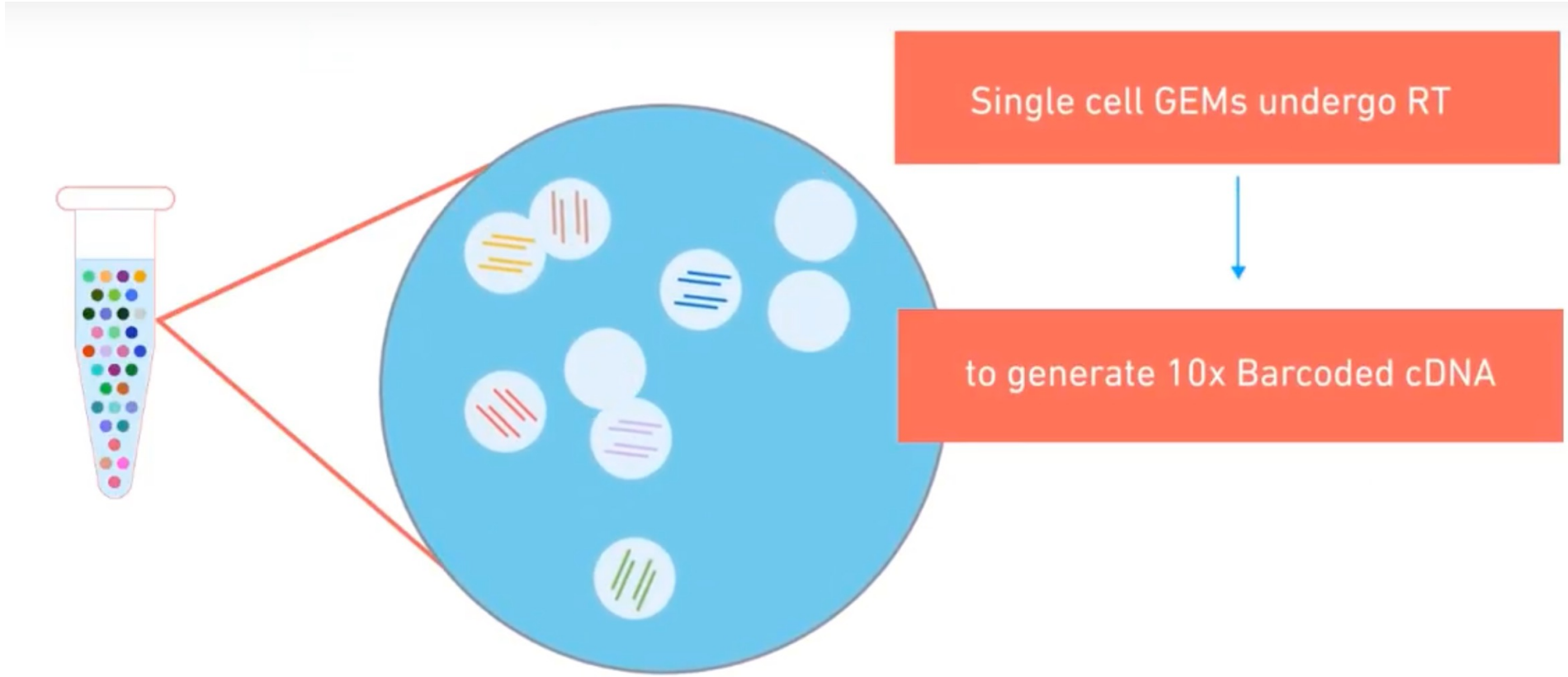
10xGenomics Technology, Pipeline, and Analysis



What happens in the Chromium instrument?


High-Diversity Gel Bead Pool
10x Barcode
Poly(dT)VN
R1
UMI


10x Barcoded Gel Beads
are mixed with cells,
enzyme, and partitioning oil
Cells &
Enzyme
Oil

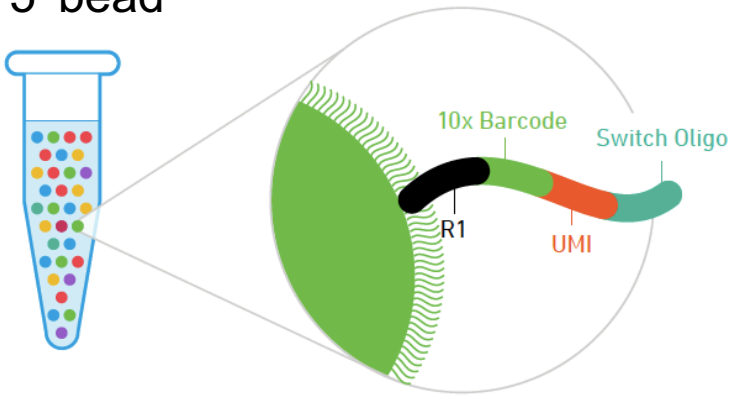

Single cell GEMs undergo RT
to generate 10x Barcoded cDNA

“Single Cell 3’ Solution” (10x Genomics)

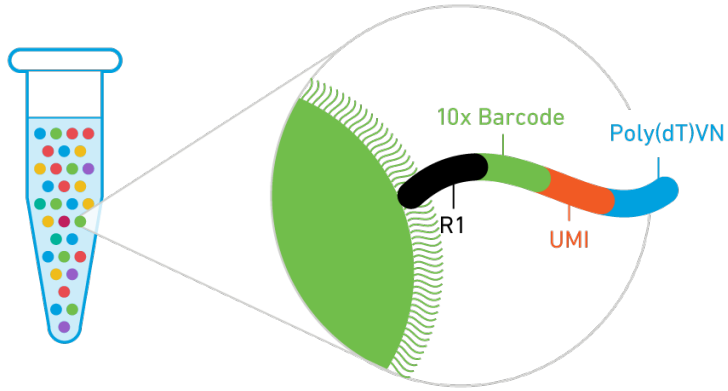
Barcoded bead + cell = bar-coded cDNA library

5’ vs. 3’ Chemistry

5’ bead

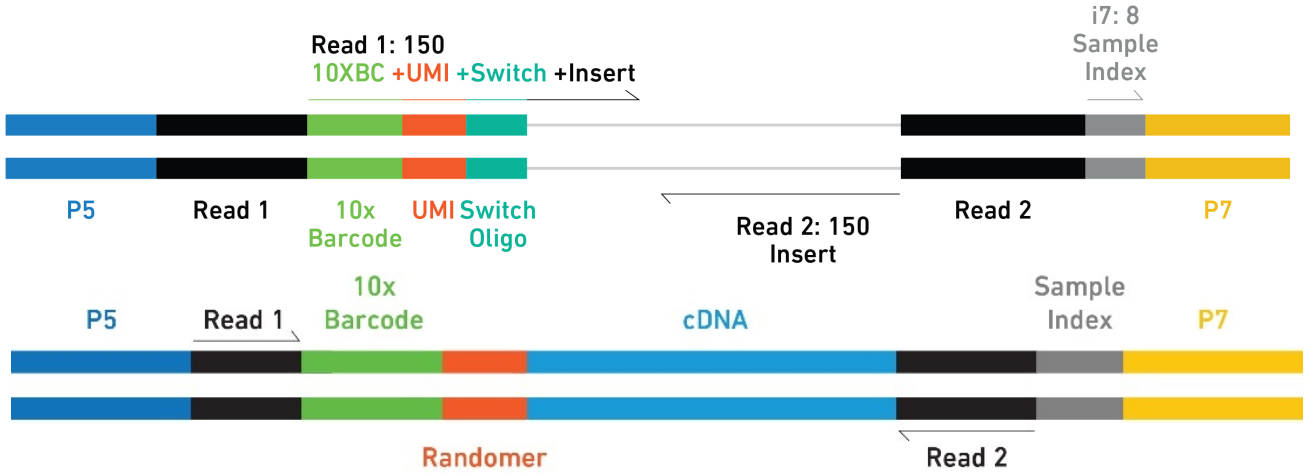


3’ bead



5’ library

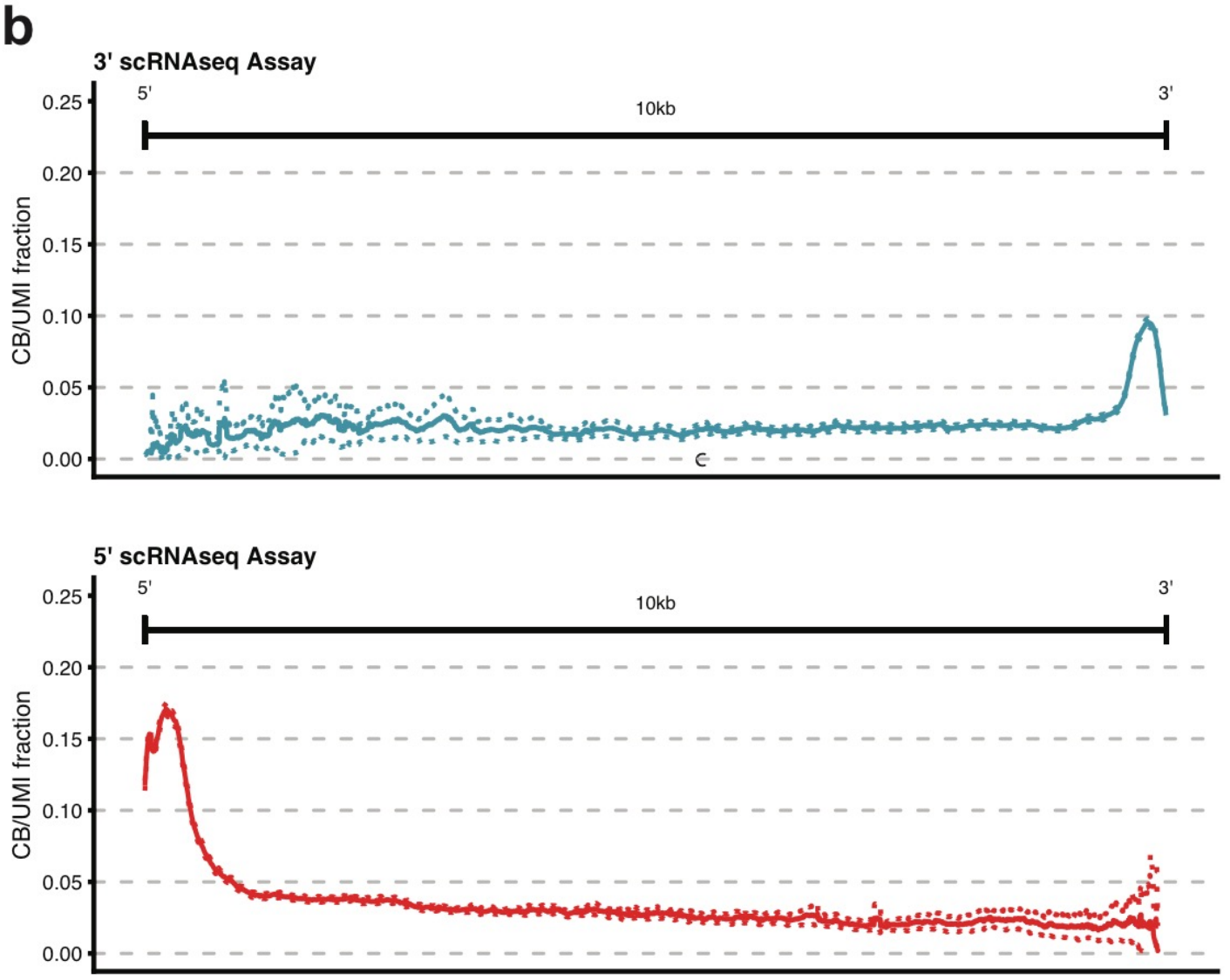
3’ library



And more! https://teichlab.github.io/scg_lib_structs/

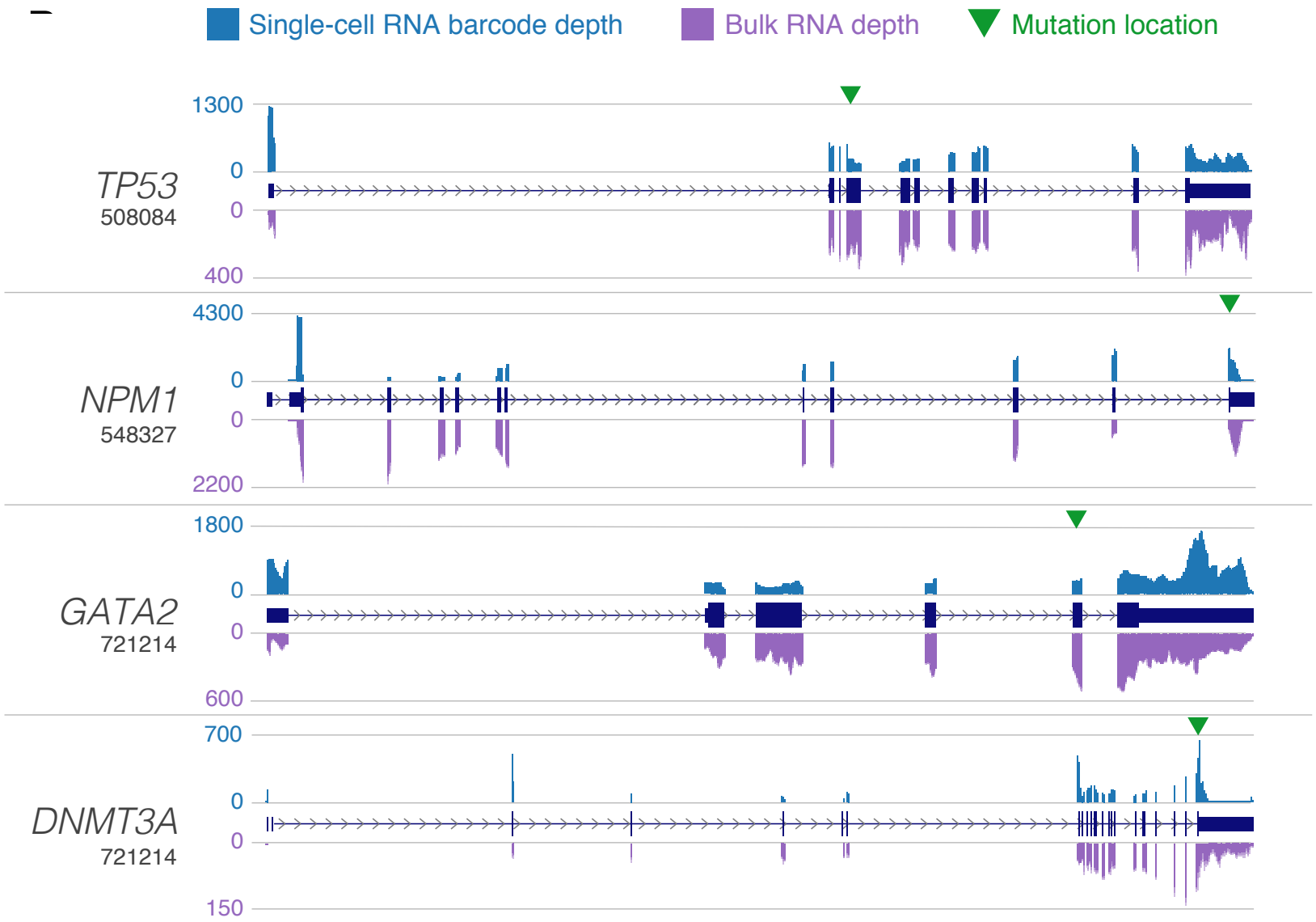
5’ vs. 3’ Transcript Coverage

Fraction of UMIs with coverage at indicated position

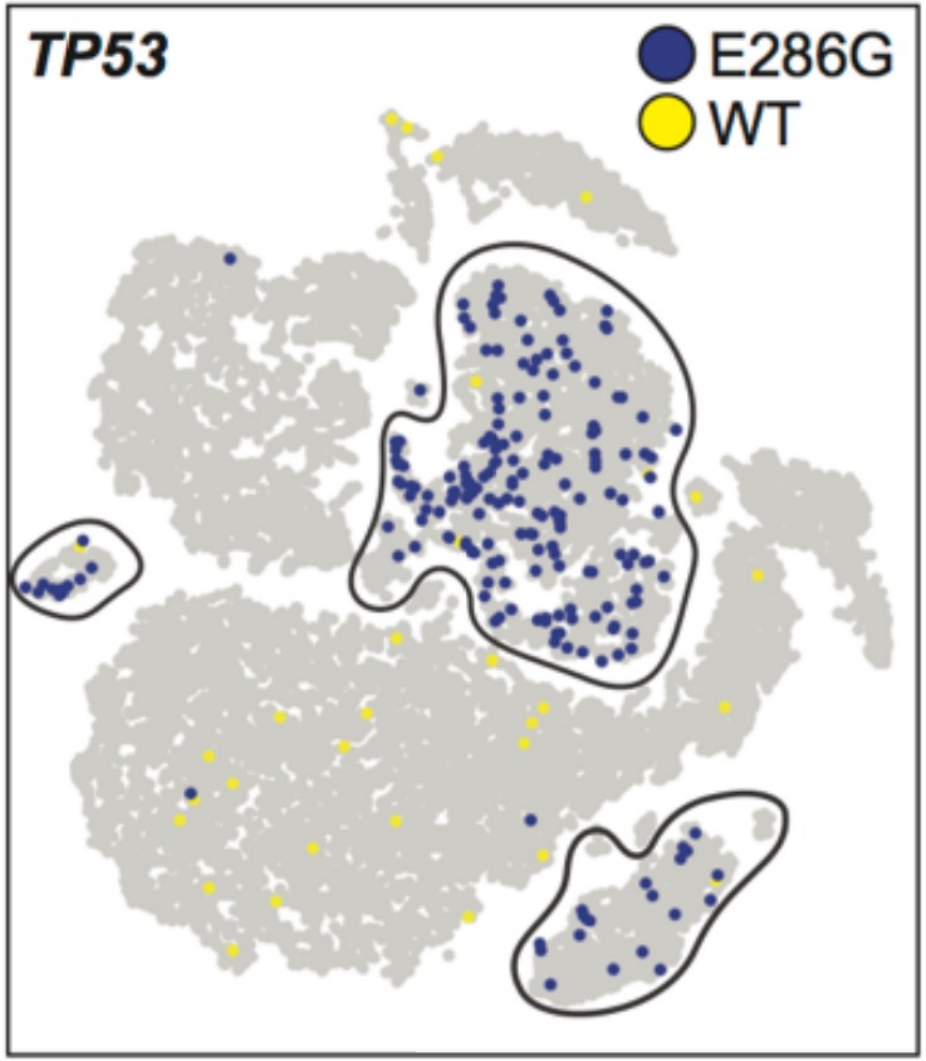


Petti A, et al. (2019) Nat. Comm. 10:3660

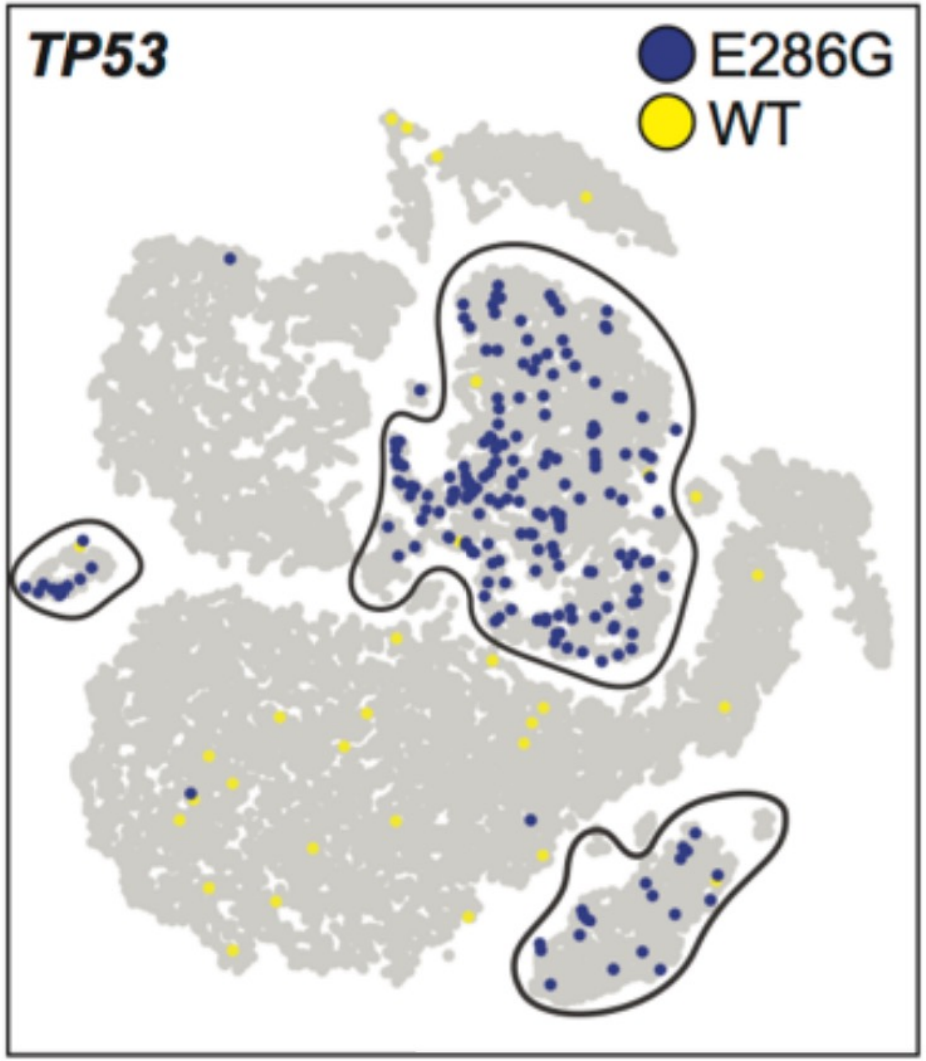
scRNA-seq recapitulates bulk transcript coverage

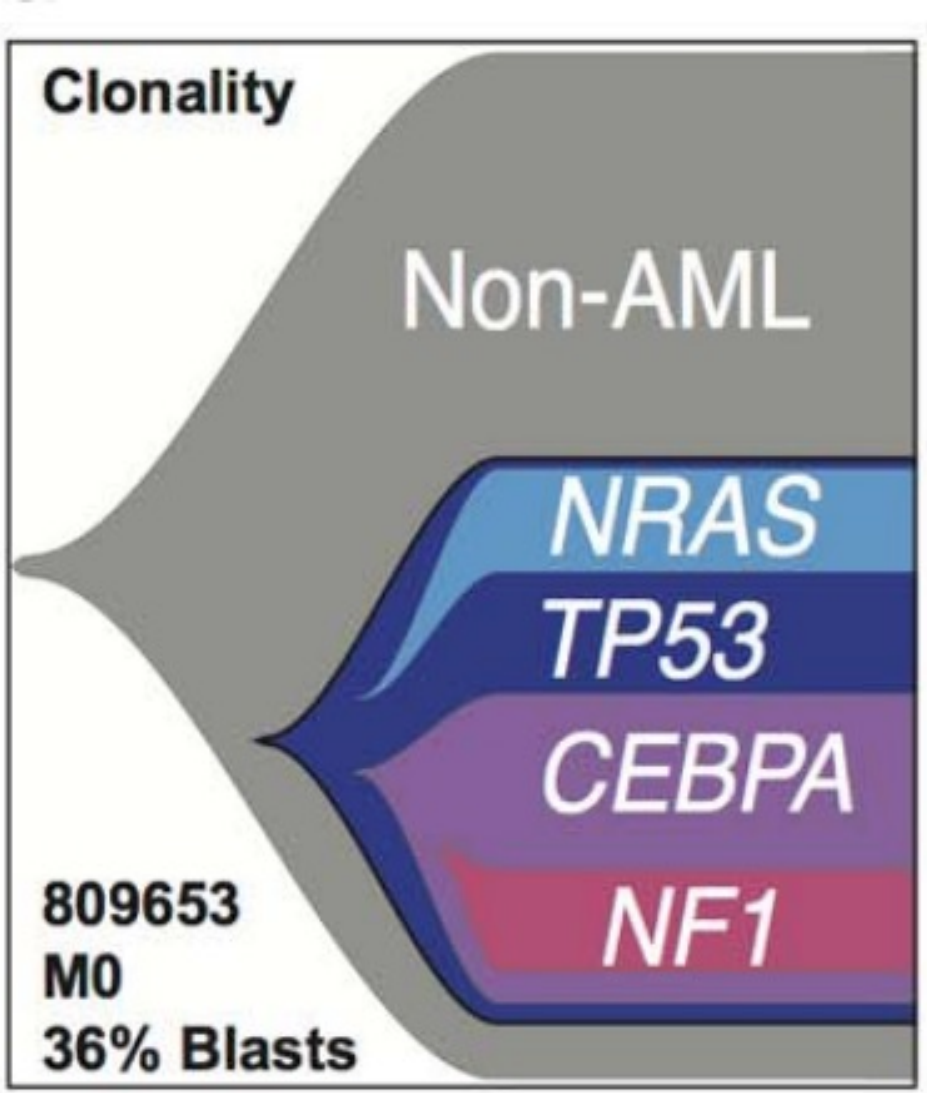


Mutation detection

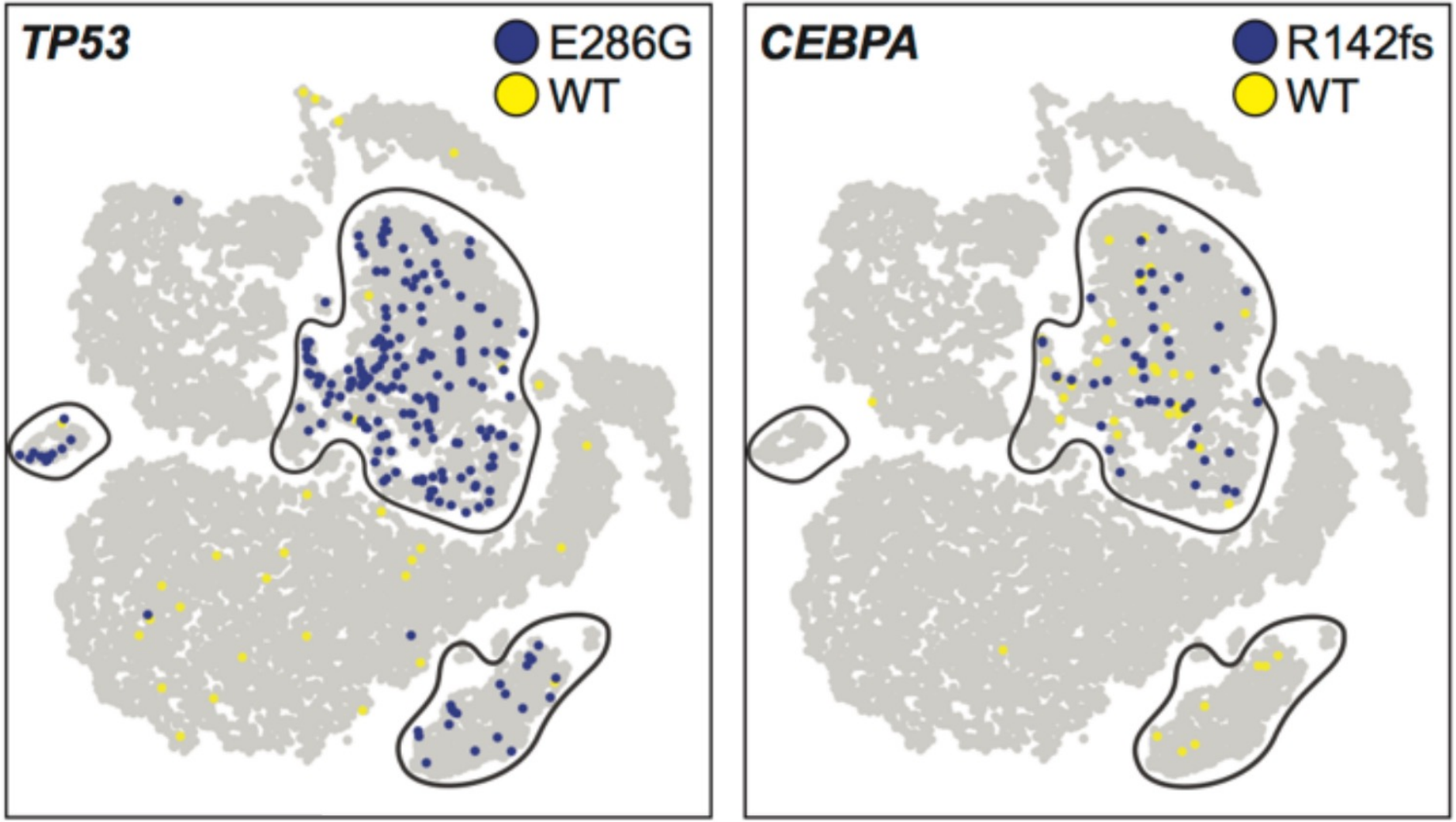


Mutation detection





Mutation detection

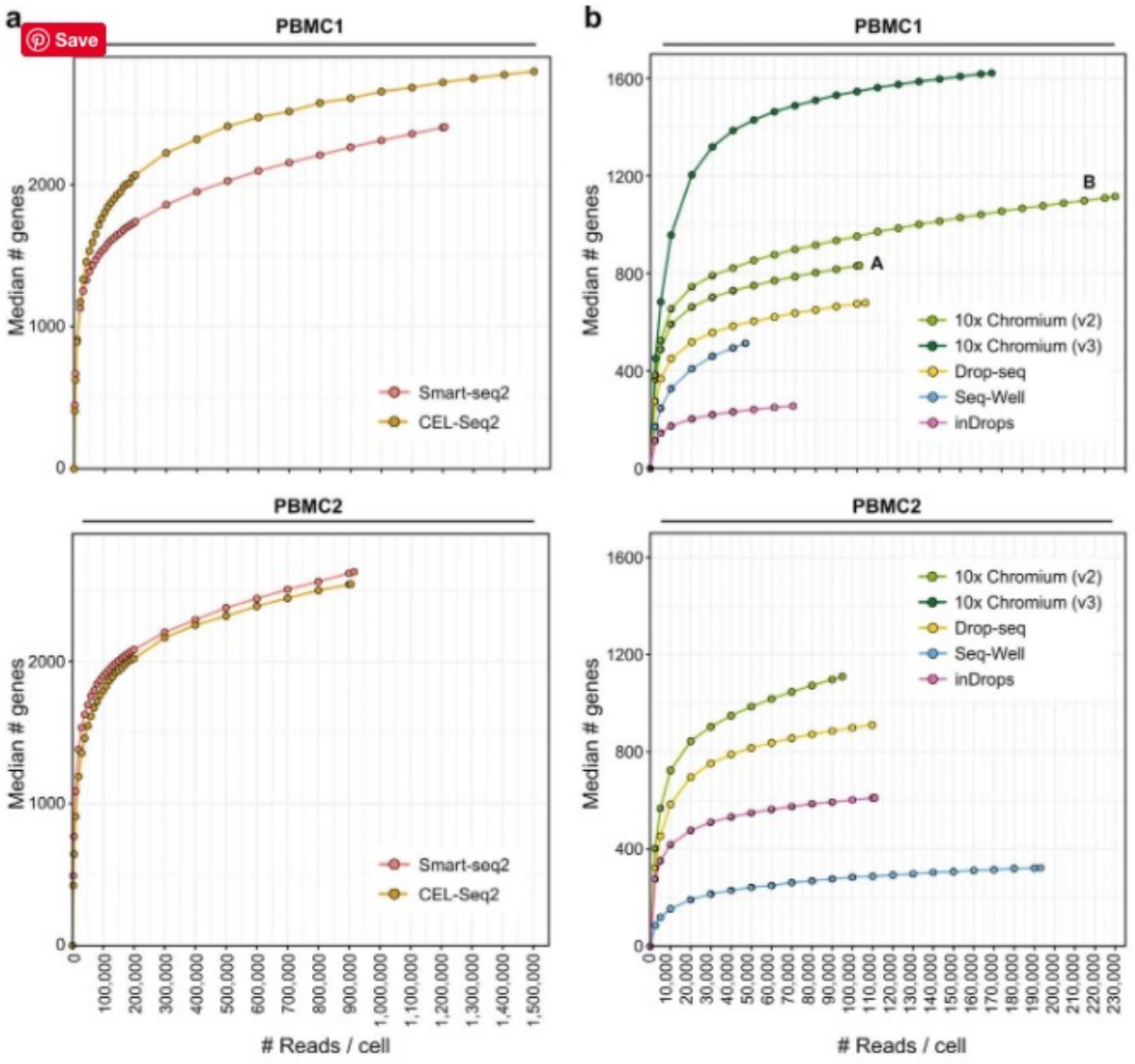


How deeply do you need to sequence?

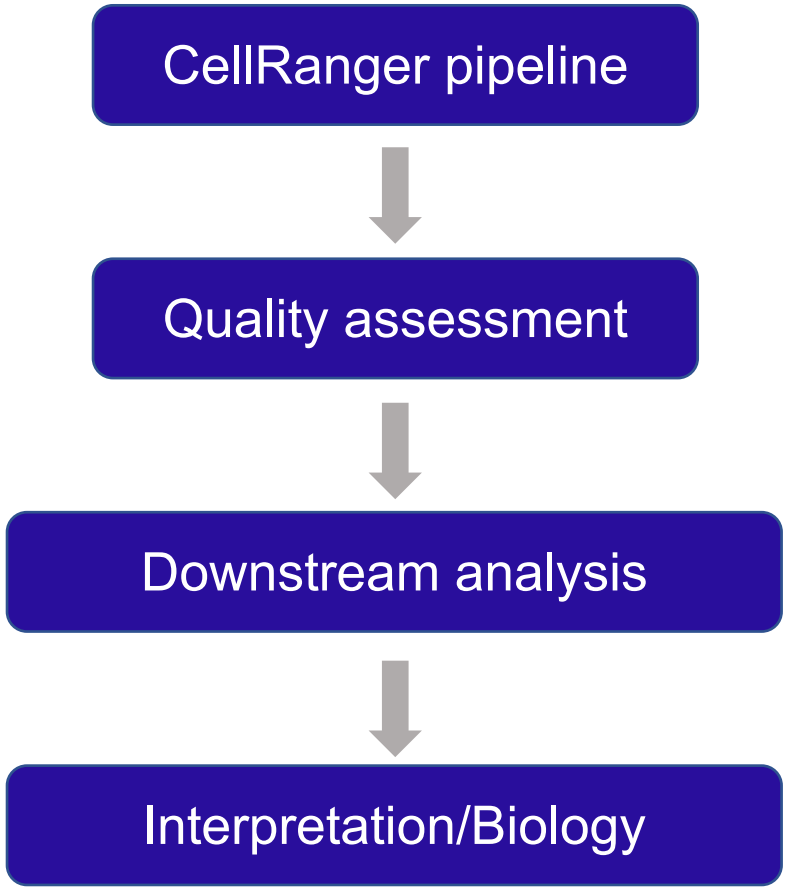
General Rule:
Achieve 90% saturation

Official Recommendations (reads/cell):

3’ - V3: 20K
3’ - V2: 50K
5’ - 20K
5’ with variant discovery - 200K
5’ V(D)J - 5K



Post-sequencing workflow



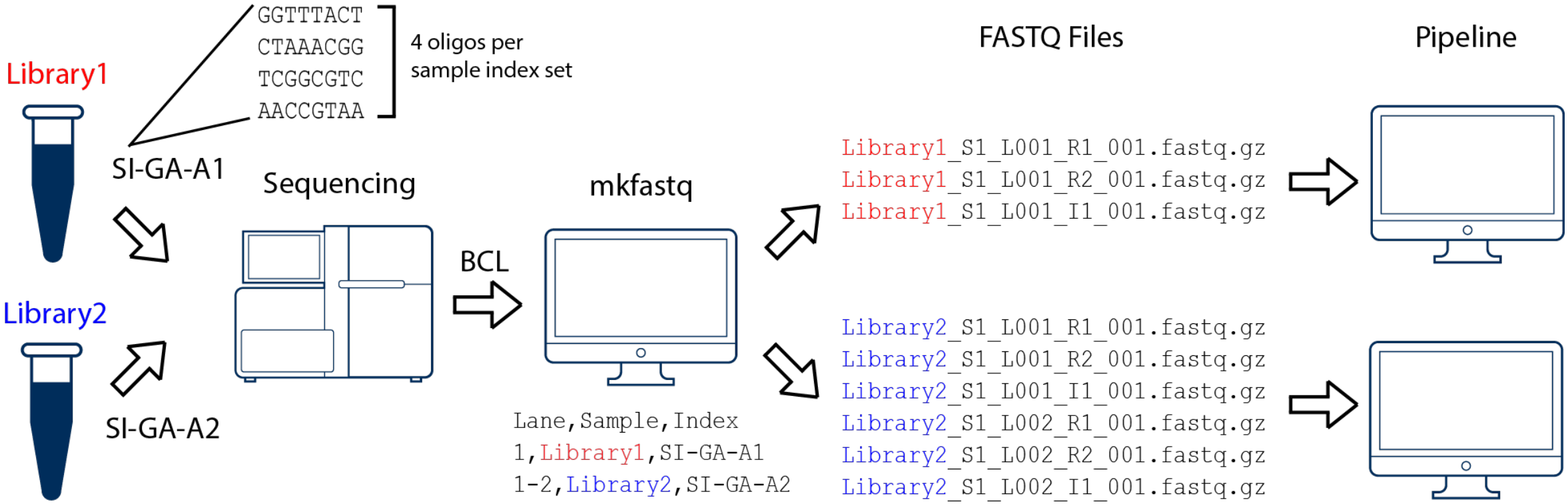
Cellranger

- cellranger and all dependencies (e.g. reference transcriptomes) can be downloaded from the 10x Genomics website:

- https://support.10xgenomics.com/single-cell-gene-expression/software/downloads/latest

- Extensive instructions are provided here: https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/using/tutorial_ov

Cellranger Step 0: Sample Demultiplexing



https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/using/mkfastq

Running the ‘cellranger mkfastq’ command

Usually done by sequencing provider

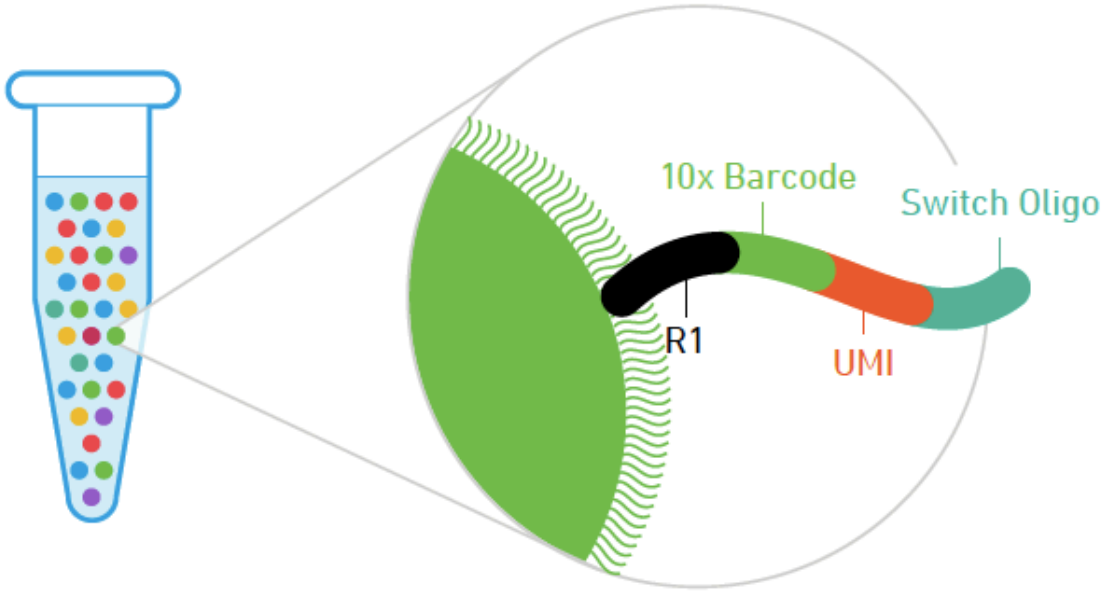
```
cellranger mkfastq –id=SampleID –run=/path/to/machine/data/directory –
samplesheet=SampleSheet.csv –csv -qc
```

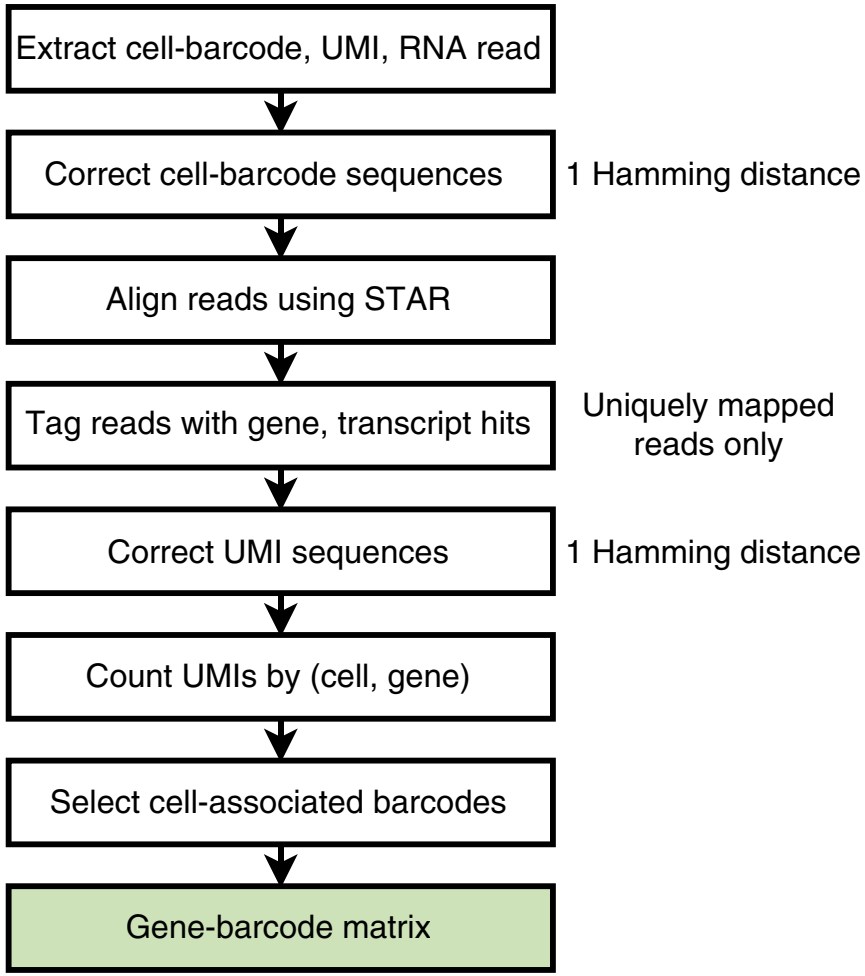
- cellranger mkfastq is a wrapper for Illumina’s bcl2fastq script, which converts basecall (bcl) files to fastq files
- -qc option is not available for NovaSeq sequencers
- If sequencing provider does this step, you should request the SampleSheet file
- Format of the SampleSheet.csv file (example only!):

Lane,Sample,Index
5,Sample1,SI-GA-E8
5,Sample2,SI-GA-E9
6,Sample1,SI-GA-E8
6,Sample2,SI-GA-E9
7,Sample1,SI-GA-E8
7,Sample2,SI-GA-E9
8,Sample1,SI-GA-E8
8,Sample2,SI-GA-E9

https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/using/mkfastq#fastq_output

Cell Ranger Overview



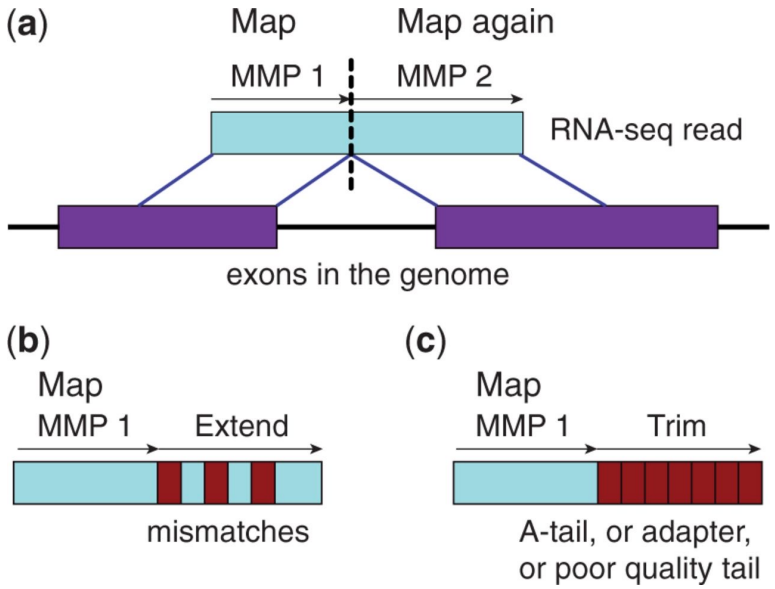


Zheng, et al. 2016. Nature Communications 8:14049

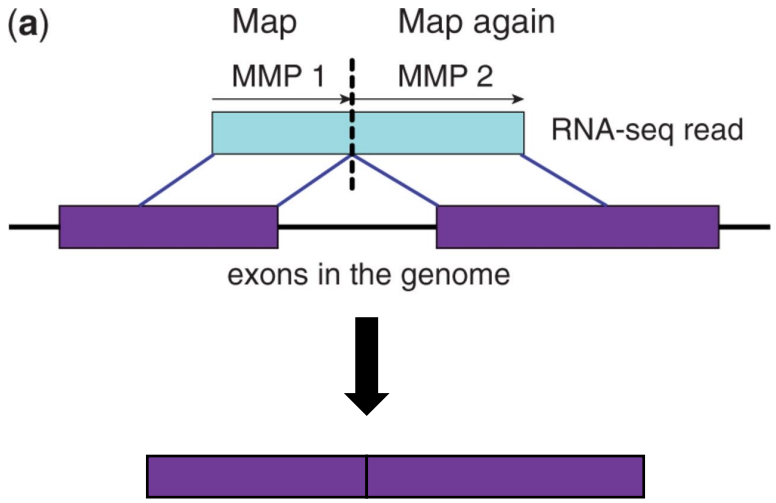
Cell Ranger: STAR alignment

STAR = Spliced Transcripts Alignment to a Reference
Aligns non-contiguous reads directly to the reference genome
Dobin A, et al. (2013) Bioinformatics 29(1):15-21.

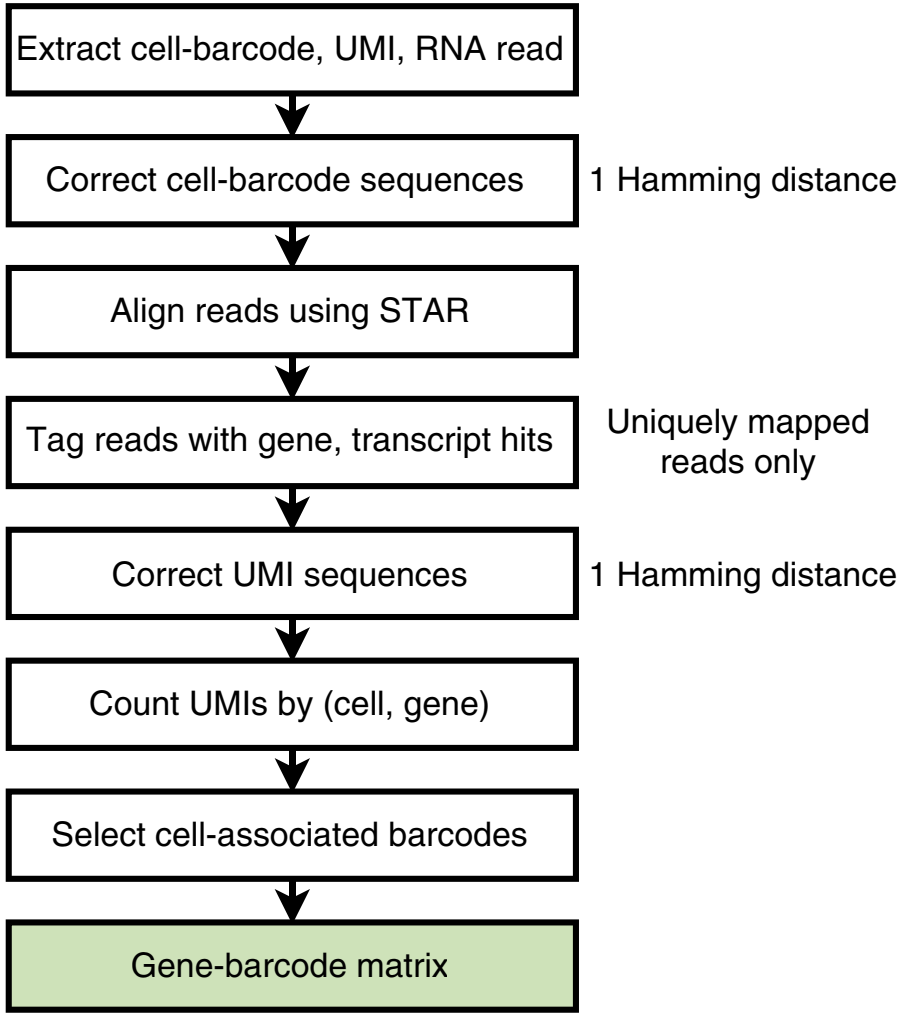
Step 1: Seed search for Maximum Mappable Prefix (MMP)



Step 2: Cluster, stitch, and score the seeds from Step 1

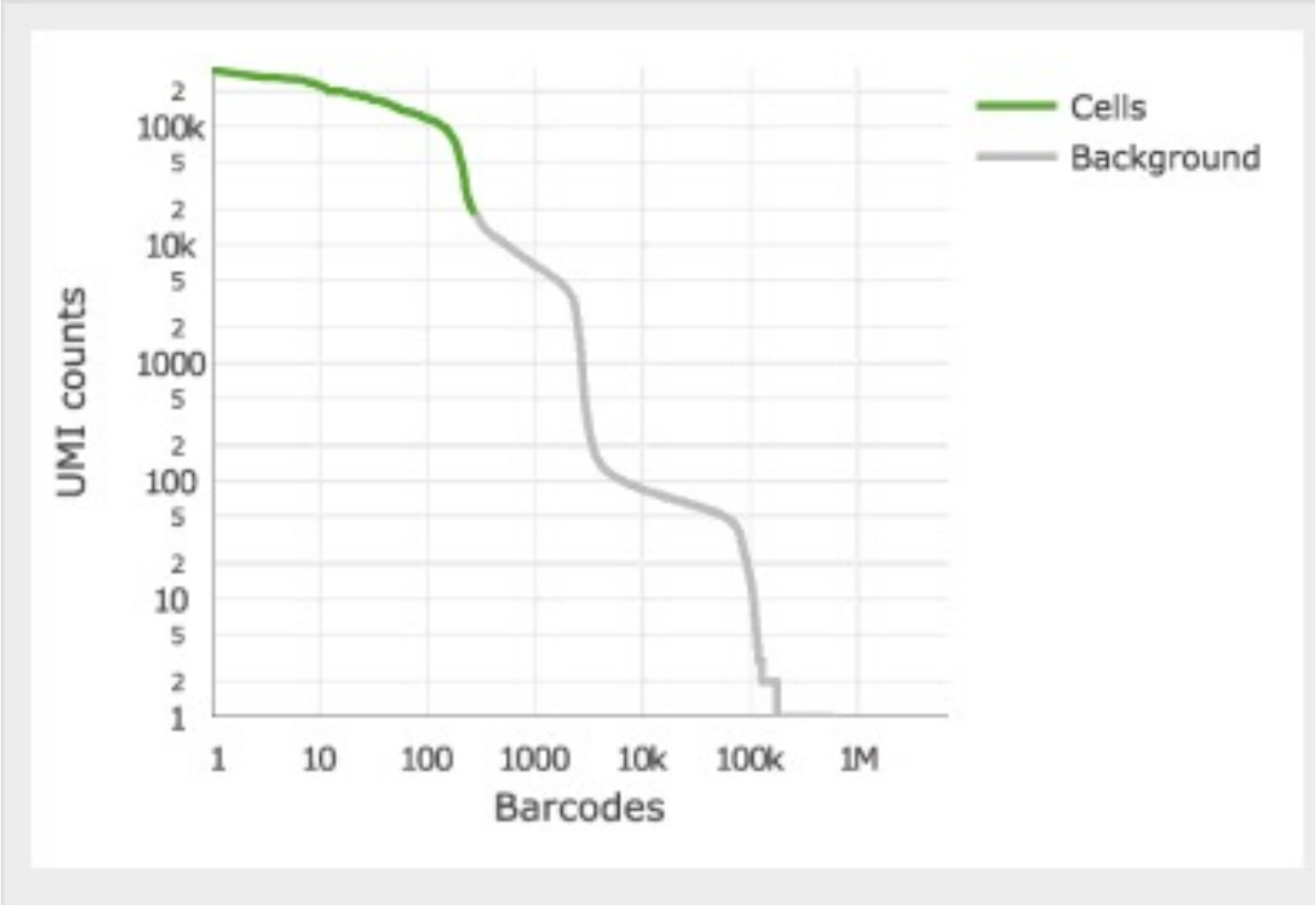


Cell Ranger: Processing Aligned Reads



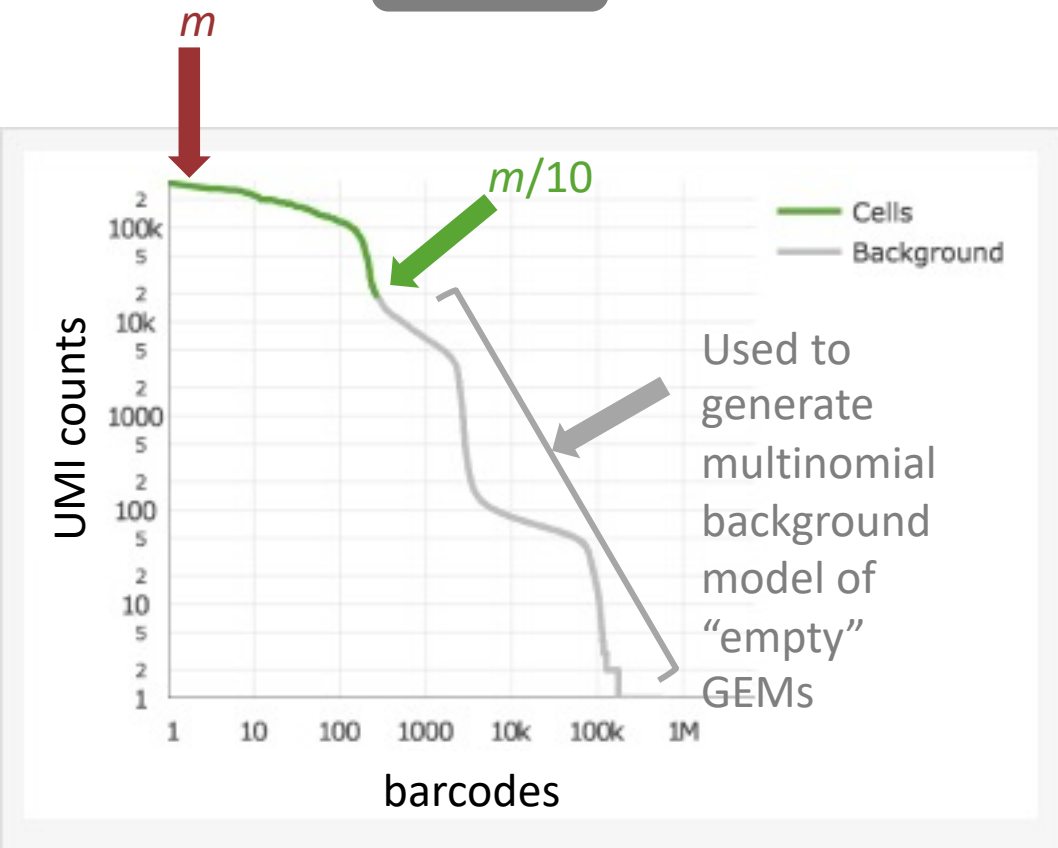
- Bins reads using transcript annotation (GTF)
- If at least 50% of the read overlaps an exon: **exonic**
- Otherwise, if it intersects an intron: **intronic**
- Otherwise, **intergenic**
- Exonic loci are prioritized in the event of multi-mapping
- If an exonic read corresponds to an annotated transcript, aligned to the same strand, and compatible with single-gene annotation, it is used for UMI counting

Cell Ranger: Selecting barcodes (cells)



Cell Ranger: Selecting barcodes (cells)

Step 1

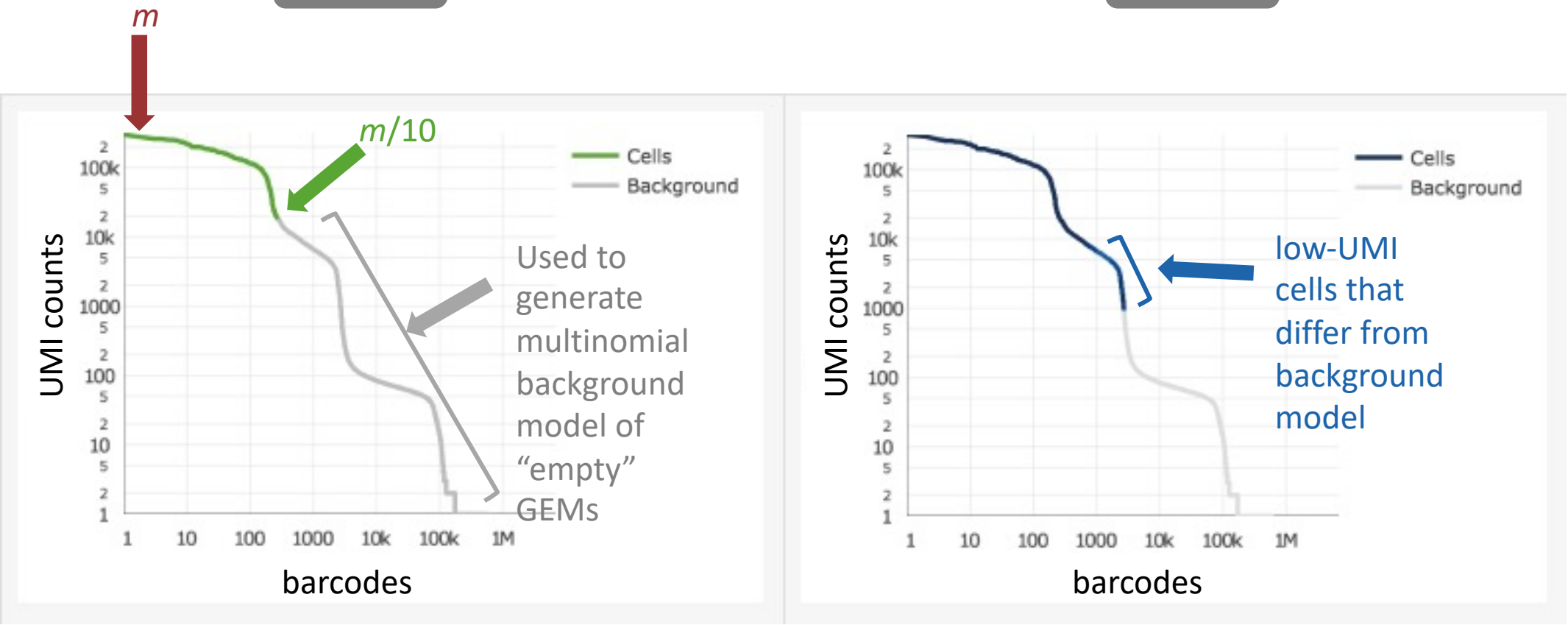


m = 99th %tile of expected cells

Cell Ranger: Selecting barcodes (cells)

Step 1

Step 2



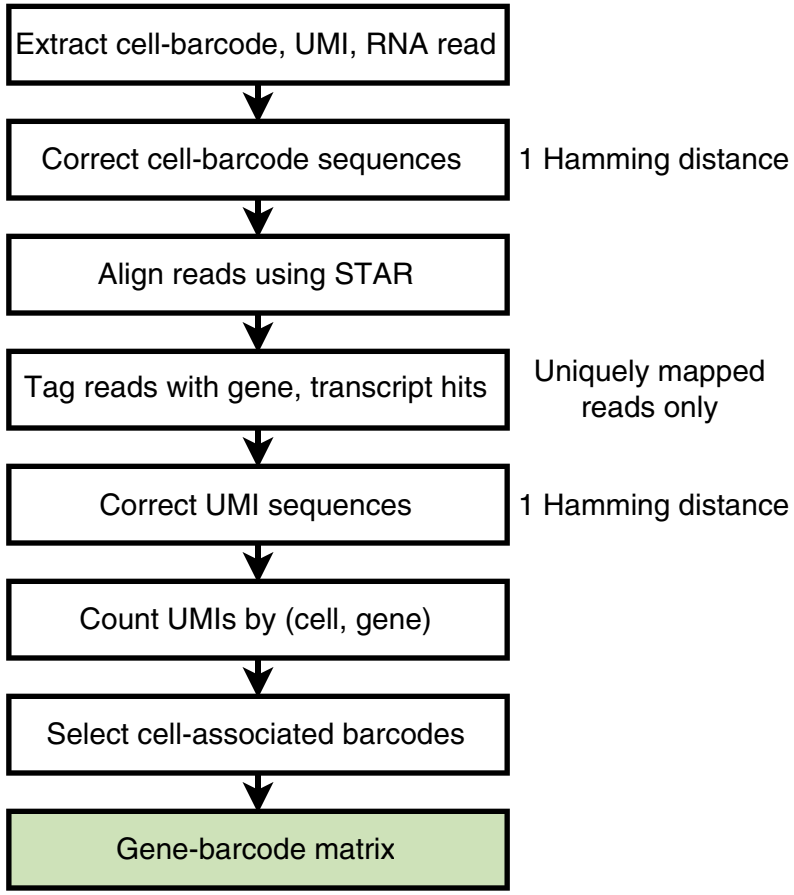
m = 99th %tile of expected cells

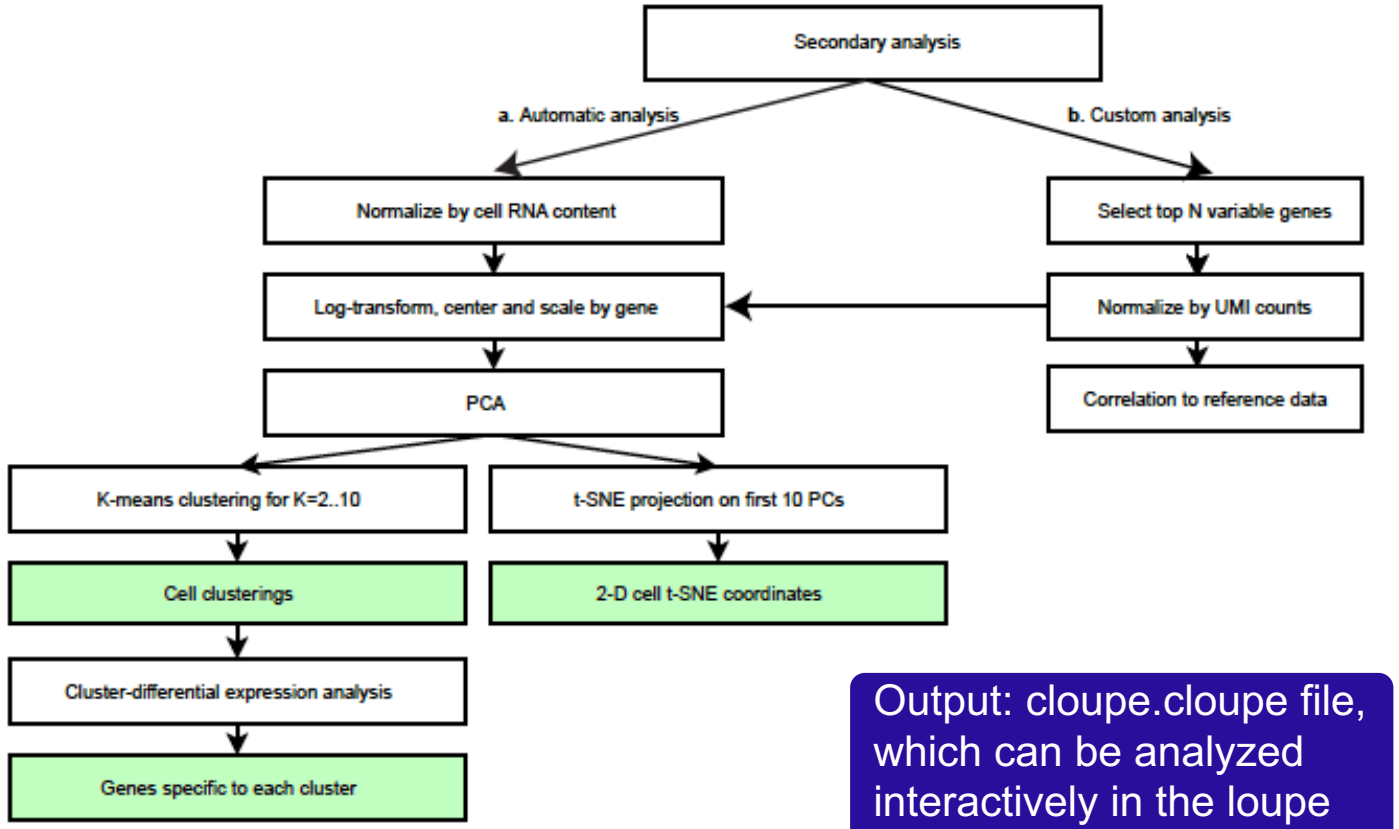
Alternative pipelines

- kallisto bustools: https://www.kallistobus.tools/
- scumi: https://bitbucket.org/jerry00/scumi-dev/src/master/
- STARsolo: https://github.com/alexdobin/STAR/blob/master/docs/STARsolo.md

Analysis of Gene-barcode matrix (better done yourself)

Normalization, dimensionality reduction, data representation





Output: cloupe.cloupe file, which can be analyzed interactively in the loupe browser

Zheng, et al. 2016. Massively parallel digital transcriptional profiling of single cells. Nature Communications 8:14049

Running cellranger using the command line

https://support.10xgenomics.com/single-cell-gene-expression/software/overview/welcome

Transcript alignment, counting, barcode selection, etc, to generate feature-barcode matrix:

```
cellranger count --id=$OutName --sample=$SampleName --fastqs=/path/to/fastqs –
indices=$SampleIndices --transcriptome=/path/to/refdata-cellranger-GRCh38-
3.0.0 --localmem=64 --localcores=12
```

Definitions:
$OutName = what you want the output directory to be called (using the sample name works well)
$SampleName = sample name provided to the sequencer; in fastq file name, e.g. SampleName_S1_L003_R1_001.fastq.gz
$SampleIndices = Set of four oligos, such as CAGTACTG,AGTAGTCT,GCAGTAGA,TTCCCGAC, *OR* a code like SI-GA-A2
Note that 10x Genomics provides oligo/code conversion files. 3' files are here:
https://support.10xgenomics.com/single-cell-gene-expression/index/doc/specifications-sample-index-sets-for-single-cell-3

Cellranger output files

analysis
- clustering - flat file clustering results
- diffexp – DEGs for each cluster
- pca – details about each principal component, projections, etc
- tsne – coordinates of each cell in t-SNE plot

cloupe.cloupe – input to loupe browser for interactive analysis

filtered_feature_bc_matrix
- barcodes.tsv.gz
- features.tsv.gz
- matrix.mtx.gz
- filtered_feature_bc_matrix.h5

metrics_summary.csv – flat file QC information

molecule_info.h5

possorted_genome_bam.bam

possorted_genome_bam.bam.bai

raw_feature_bc_matrix – not filtered for cell-associated barcodes

raw_feature_bc_matrix.h5 – not filtered for cell-associated barcodes

web_summary.html – QC information and minimal interactive analysis

Did your experiment work?

Two key QC files:

- metrics_summary.csv
- web_summary.html: http://genomedata.org/rnaseq-tutorial/scrna/508084_5prime_GRCh38_web_summary.html

web_summary.html, metrics_summary.csv

The analysis detected some issues. Details »

Alert	Value	Detail
⚠ **Low Fraction Valid Barcodes**	63.7%	Ideal > 75%. This usually indicates a quality issue with the Illumina I7 read for Single Cell 3' v1 or the R1 read for Single Cell 3' v2. Application performance may be affected.
⚠ **Low Fraction Reads Confidently Mapped To Transcriptome**	26.1%	Ideal > 30%. This can indicate use of the wrong reference transcriptome, a reference transcriptome with overlapping genes, poor library quality, poor sequencing quality, or reads shorter than the recommended minimum. Application performance may be affected.
⚠ **High Fraction of Reads Mapped Antisense to Genes**	16.9%	Ideal < 10%. This can indicate use of an unsupported chemistry type (e.g. using Single Cell V(D)J for gene counting). Application performance may be affected.

Estimated Number of Cells

9,151

Mean Reads per Cell	Median Genes per Cell
147,263	2,206

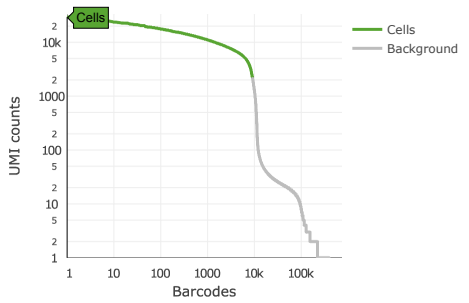
Sequencing

Number of Reads	1,347,604,104
Valid Barcodes	63.7%
Sequencing Saturation	89.0%
Q30 Bases in Barcode	94.9%
Q30 Bases in RNA Read	91.7%
Q30 Bases in RNA Read 2	87.5%
Q30 Bases in Sample Index	92.6%
Q30 Bases in UMI	94.7%

Mapping

Reads Mapped to Genome	92.8%
Reads Mapped Confidently to Genome	70.3%
Reads Mapped Confidently to Intergenic Regions	17.8%
Reads Mapped Confidently to Intronic Regions	10.1%
Reads Mapped Confidently to Exonic Regions	44.7%
Reads Mapped Confidently to Transcriptome	26.1%
Reads Mapped Antisense to Gene	16.9%

Cells



Estimated Number of Cells	9,151
Fraction Reads in Cells	91.9%
Mean Reads per Cell	147,263
Median Genes per Cell	2,206
Total Genes Detected	19,919
Median UMI Counts per Cell	6,407

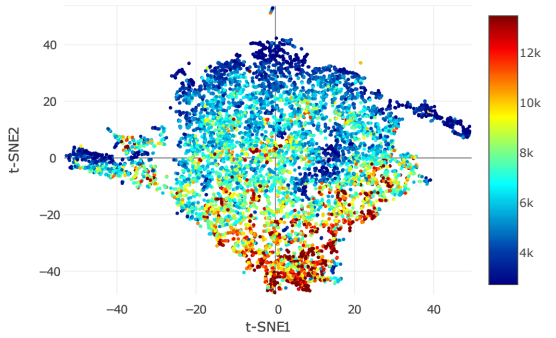
Sample

Name	115225_P_1_Nova
Description	
Transcriptome	GRCh38
Chemistry	Single Cell 5' PE
Cell Ranger Version	2.1.1

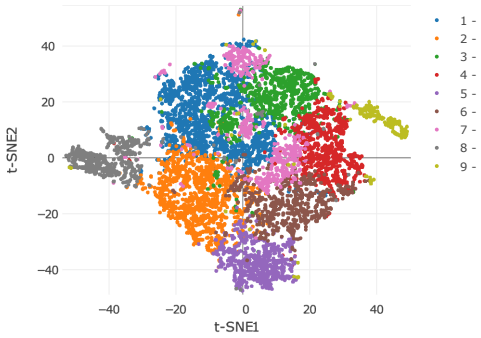
The analysis detected some issues. Details »

Clustering Type: Gra

t-SNE Projection of Cells Colored by UMI Counts



t-SNE projection of Cells Colored by Automated Clustering



Top Genes By Cluster (Log2 fold-change, p-value)

		Cluster 1		Cluster 2		Cluster 3		Cluster 4		Cluster 5		Cluster 6		Cluster 7		Clus
Gene ID	Gene name	L2FC	p-value	L2FC	p-value	L2FC	p-value	L2FC	p-value	L2FC	p-value	L2FC	p-value	L2FC	p-value	L2FC
ENSG00000180573	HIST1H2AC	0.56	1e+00	-1.02	5e-01	0.48	1e+00	0.82	6e-01	-0.89	4e-01	-0.66	1e+00	0.29	1e+00	0.25
ENSG00000125652	ALKBH7	0.54	1e+00	-0.01	1e+00	0.19	1e+00	-0.07	1e+00	-0.80	6e-01	-0.32	1e+00	0.09	1e+00	0.33
ENSG00000257698	RP11-620J15.3	0.53	1e+00	-0.19	1e+00	0.01	1e+00	-0.00	1e+00	-0.23	1e+00	-0.36	1e+00	-0.01	1e+00	0.17
ENSG00000104894	CD37	0.48	1e+00	-0.27	1e+00	0.47	1e+00	0.26	1e+00	-0.74	6e-01	-0.43	1e+00	0.14	1e+00	0.22
ENSG00000150782	IL18	0.46	1e+00	-0.51	1e+00	0.86	9e-01	0.44	8e-01	-0.64	8e-01	-0.43	1e+00	0.02	1e+00	-0.49
ENSG00000267453	AC004791.2	0.46	1e+00	0.29	1e+00	-0.61	1e+00	-0.19	1e+00	-0.36	1e+00	-0.04	1e+00	0.14	1e+00	-0.37
ENSG00000196531	NACA	0.45	1e+00	-0.02	1e+00	0.18	1e+00	0.05	1e+00	-0.67	7e-01	-0.16	1e+00	0.16	1e+00	0.00
ENSG00000095932	SMIM24	0.44	1e+00	0.12	1e+00	0.15	1e+00	-0.06	1e+00	-0.13	1e+00	-0.14	1e+00	-0.00	1e+00	-1.74
ENSG00000204628	GNB2L1	0.44	1e+00	0.10	1e+00	0.12	1e+00	-0.08	1e+00	-0.51	9e-01	-0.21	1e+00	0.01	1e+00	-0.05
ENSG00000145708	CRHBP	0.43	1e+00	-0.26	1e+00	0.36	1e+00	0.52	8e-01	-0.55	9e-01	-0.04	1e+00	0.47	1e+00	-2.77
ENSG00000095917	TPSD1	0.42	1e+00	-0.53	1e+00	-1.03	1e+00	0.41	9e-01	-0.55	9e-01	0.06	1e+00	0.91	1e+00	0.41
ENSG00000105373	GLTSCR2	0.41	1e+00	0.06	1e+00	0.12	1e+00	-0.00	1e+00	-0.41	1e+00	-0.26	1e+00	-0.20	1e+00	-0.04
ENSG00000104408	EIF3E	0.40	1e+00	0.05	1e+00	0.23	1e+00	0.09	1e+00	-0.61	8e-01	-0.13	1e+00	0.02	1e+00	-0.15
ENSG00000263961	C1orf186	0.40	1e+00	-0.08	1e+00	0.40	1e+00	0.35	9e-01	-0.67	7e-01	-0.08	1e+00	0.25	1e+00	-1.46
ENSG00000170891	CYTL1	0.40	1e+00	-0.18	1e+00	0.86	9e-01	0.22	1e+00	-0.47	1e+00	-0.49	1e+00	0.13	1e+00	-1.12
ENSG00000269893	SNHG8	0.39	1e+00	0.18	1e+00	0.09	1e+00	-0.09	1e+00	-0.59	9e-01	-0.09	1e+00	0.17	1e+00	-0.33

Sequencing Saturation

Median Genes per Cell

Values from some real experiments

Metric	Human – Cryo. Bone Marrow	Human – Fresh GBM Cell lines	Mouse – Cryo. Bone Marrow
Estimated Cells	7000	8500	5000
Target Reads/Cell	20K-50K (expression), 200K (variants)		
Median Genes/Cell	2000	5600	2100
% Transcriptome mapping	>50%*	70%	>70%
% Antisense Reads	~3%	~5%	~3%
Fraction reads in cells	80-90%	80-90%	80-90%
Total Genes Detected	20,000	25,000	16,500
Median UMIs/Cell	5000-6000	25000	7000-8000

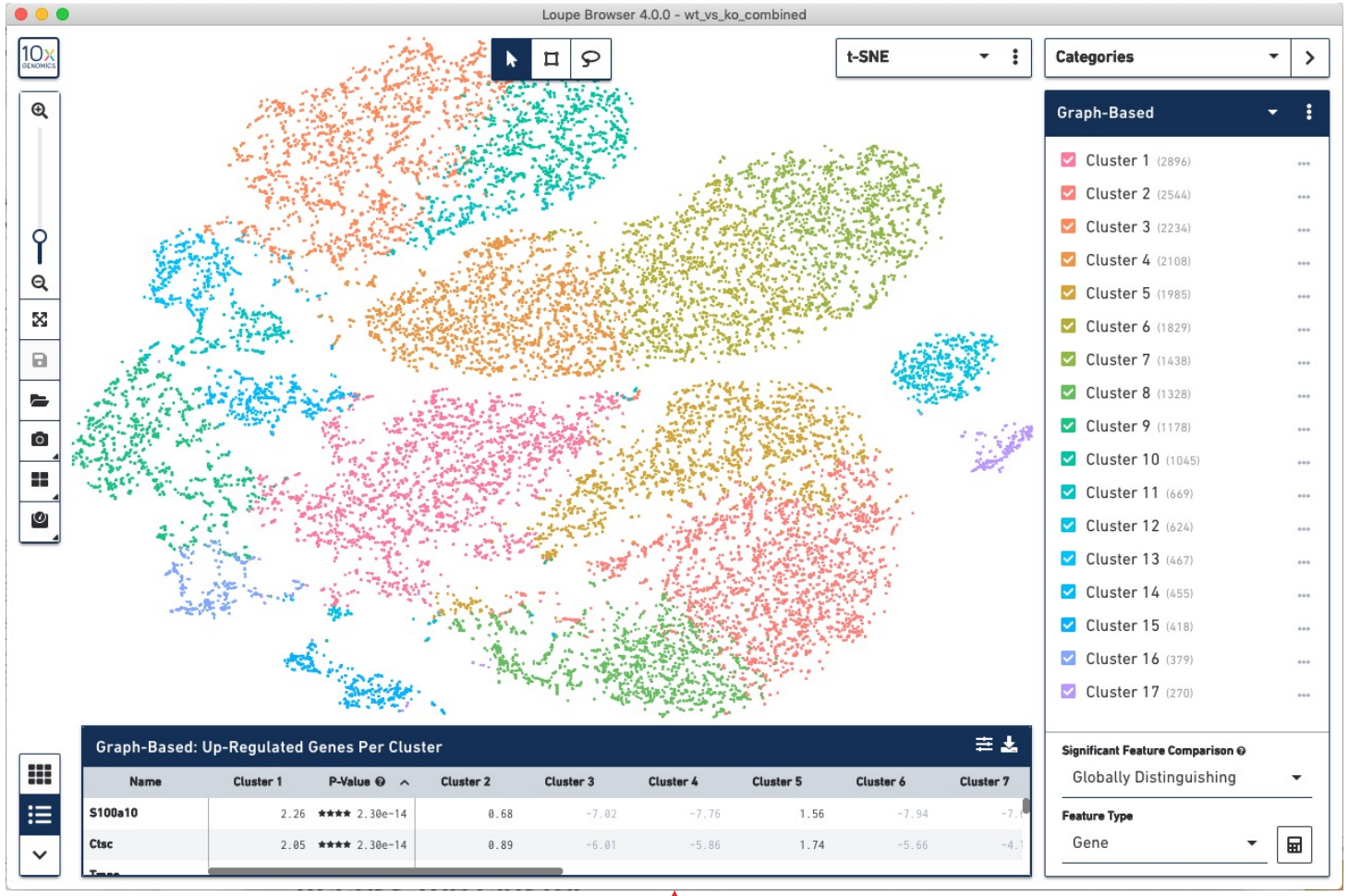
Possible reasons for low quality

Metric	Human
Estimated Cells	Low viability, lysed cells
Target Reads/Cell	Rarely problematic
% Transcriptome mapping	Wrong transcriptome, low sequence quality
% Antisense Reads	Wrong chemistry, low sequence quality
Fraction reads in cells	Lysed cells, extracellular RNA

Exploring the data using the loupe browser

Wild-type mouse (WT) compared to Knock-out (KO) mouse (2 replicates each)

File: wt_vs_ko_combined.cloupe



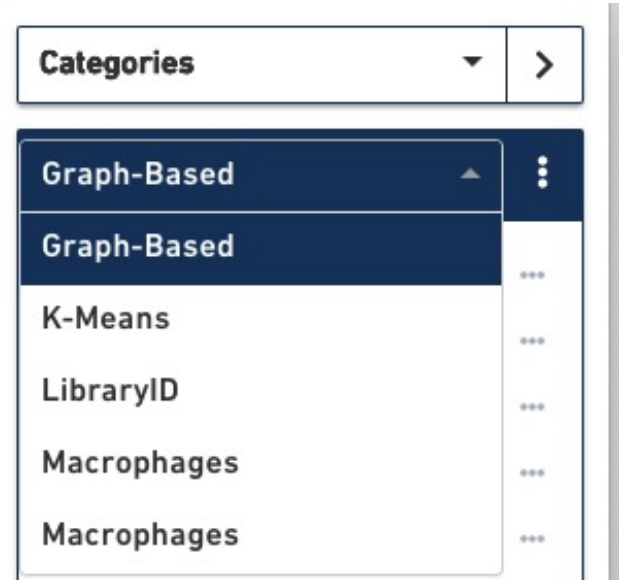
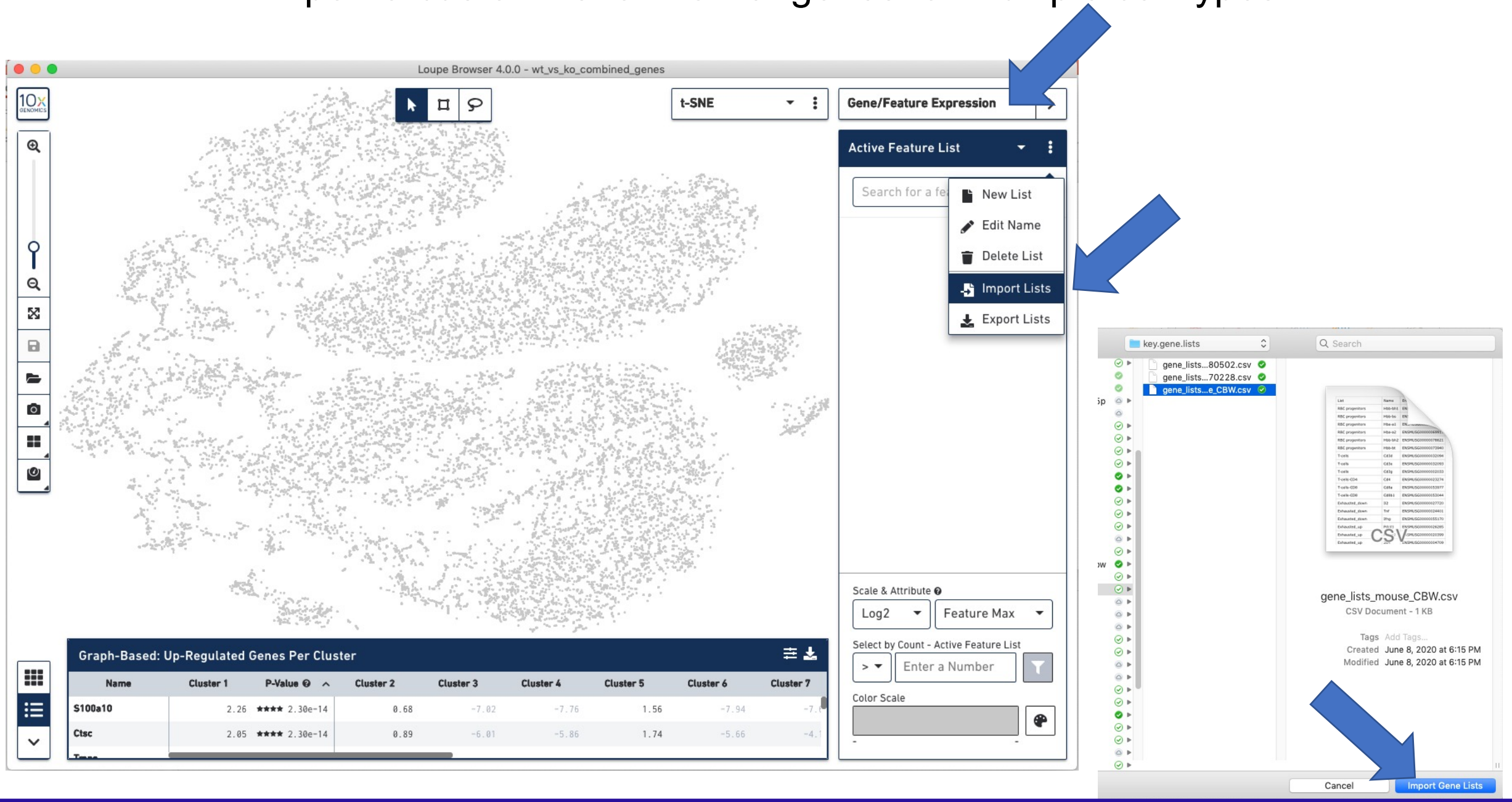


Table of differentially expressed genes in each cluster

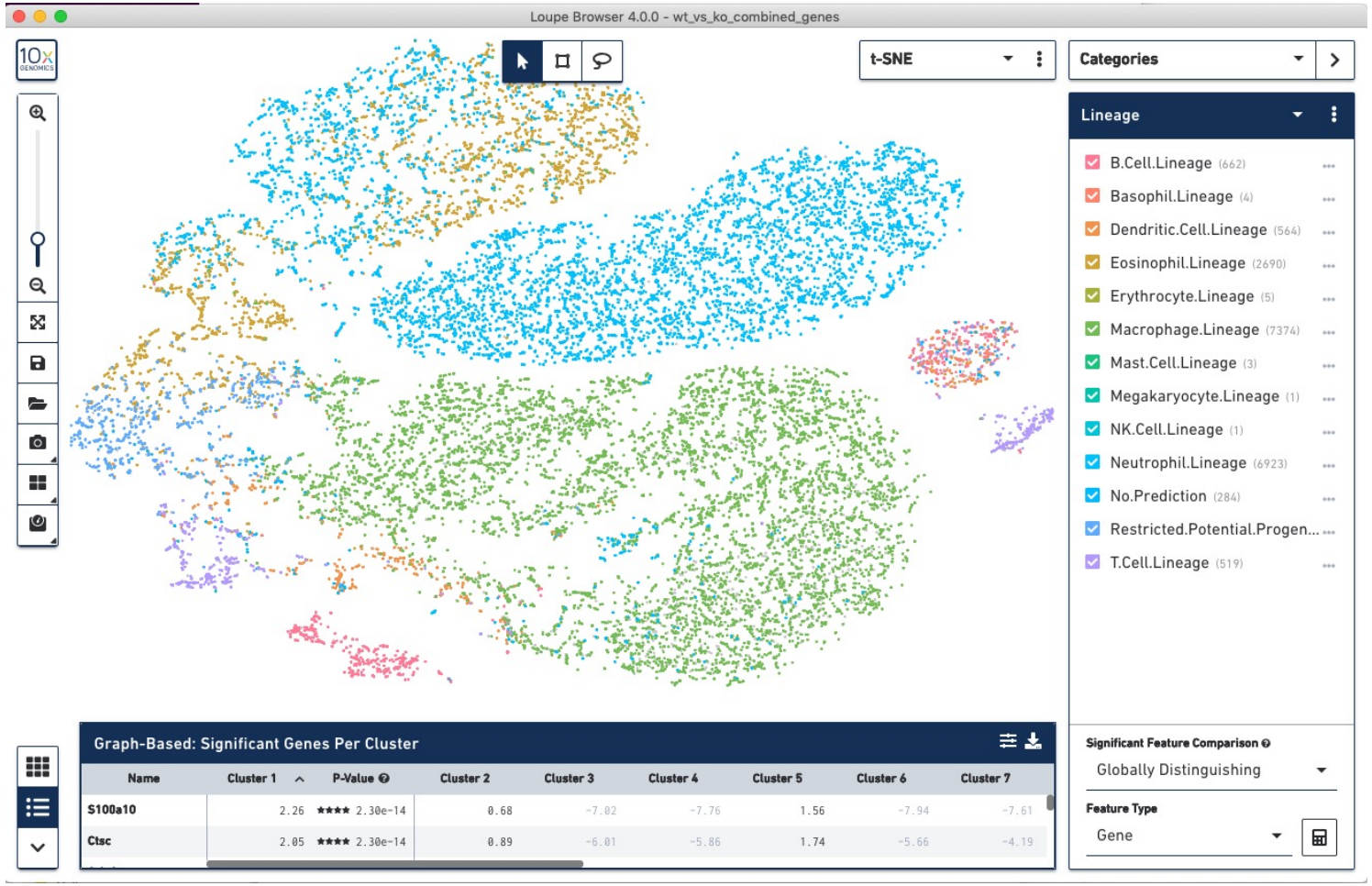

Loupe Browser 4.0.0 - wt_vs_ko_combined_genes
t-SNE
Categories
LibraryID
mouse31_Dnmt3a_KO (7122)
mouse31_Dnmt3a_KO_sort...
mouse32_Dnmt3a_WT (5378)
mouse32_Dnmt3a_WT_sort...
mouse31_Dnmt3a_KO
mouse31_Dnmt3a_KO_sorted
mouse32_Dnmt3a_WT
mouse32_Dnmt3a_WT_sorted
Graph-Based: Up-Regulated Genes Per Cluster
Name
Cluster 1
P-Value
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
S100a10
2.26
**** 2.30e-14
0.68
-7.02
-7.76
1.56
-7.94
Ctsc
2.05
**** 2.30e-14
0.89
-6.01
-5.86
1.74
-5.66
Significant Feature Comparison
Globally Distinguishing
Feature Type
Gene

Import a custom list of marker genes for multiple cell types



Click on a few genes and gene sets

Import a custom list of inferred cell types*



*These cell types were inferred using an in-house nearest-neighbor algorithm and the Haemopedia database. You can use the SingleR package to do something similar.

Compare two clusters using a differential expression analysis

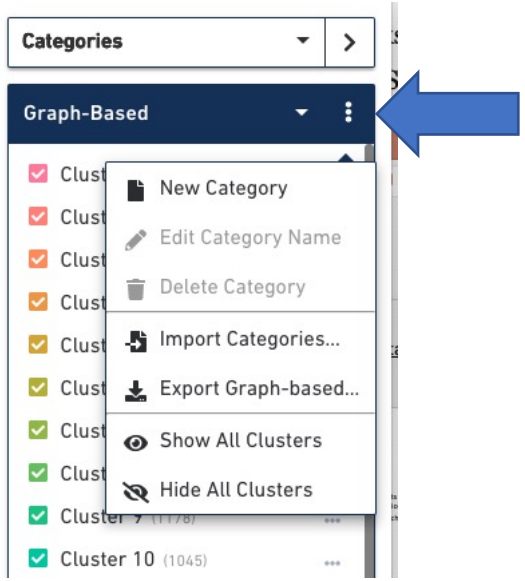
Select Categories…

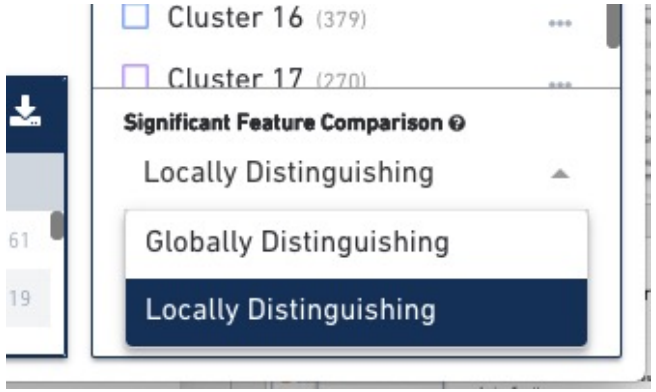
Select Graph-Based…

Click the three vertical dots to access the “Hide all clusters” option

Then click two clusters of your choice

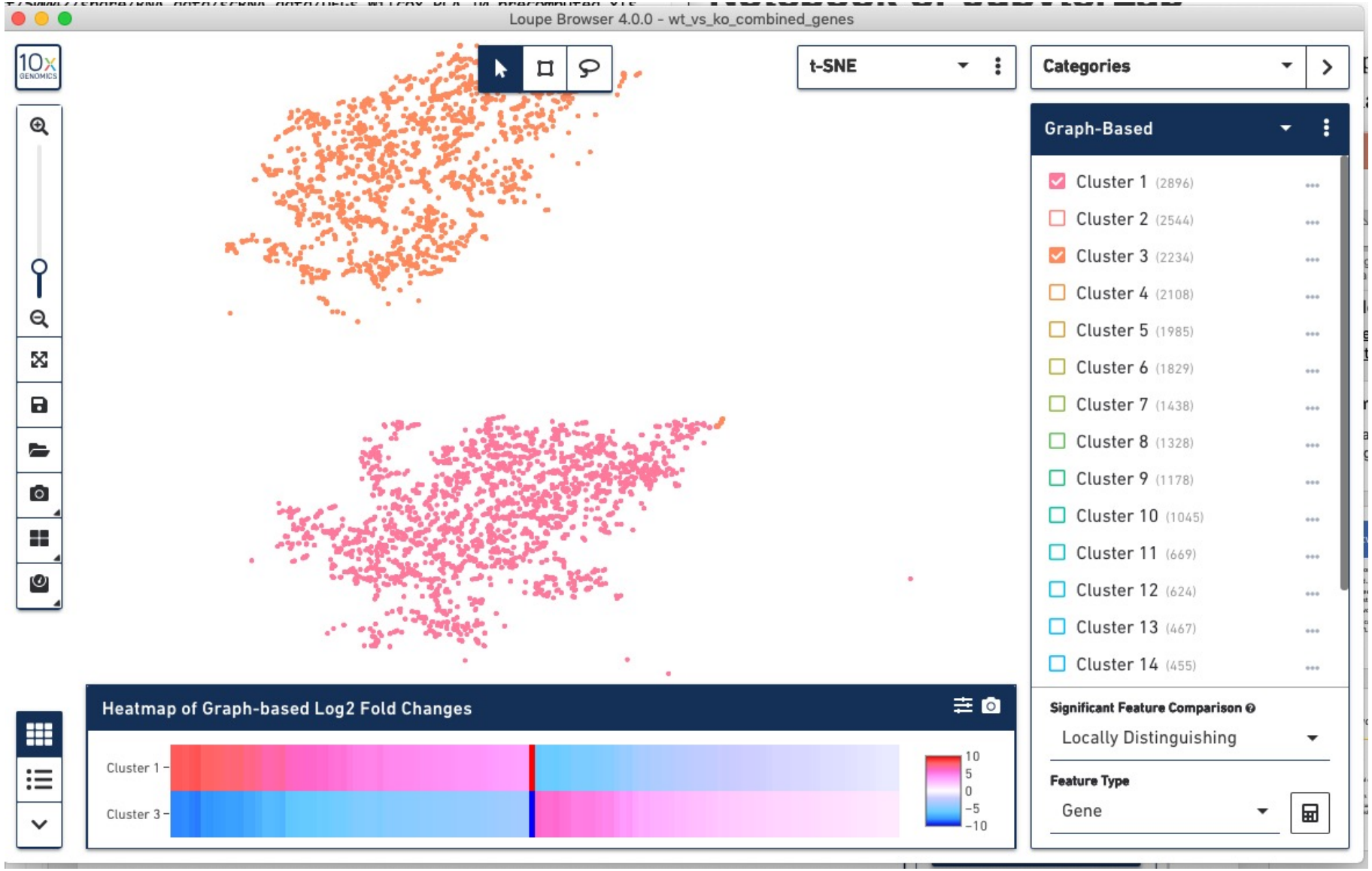
In the Significant Feature Comparison panel, click “Locally distinguishing,” then press calculator icon





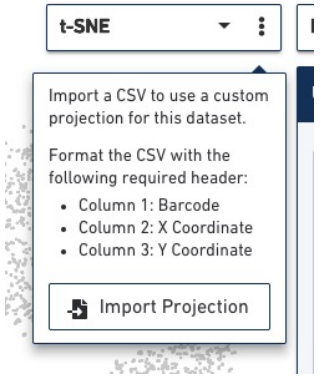
heatmap of results

table of results



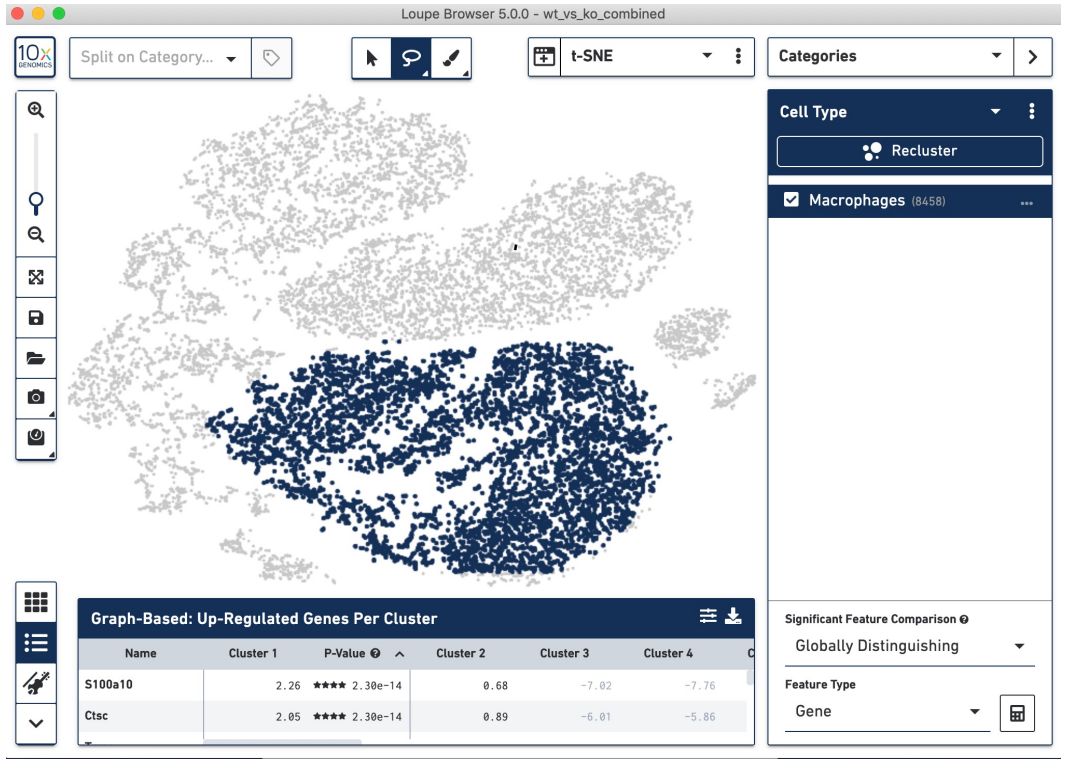
lasso to select cells

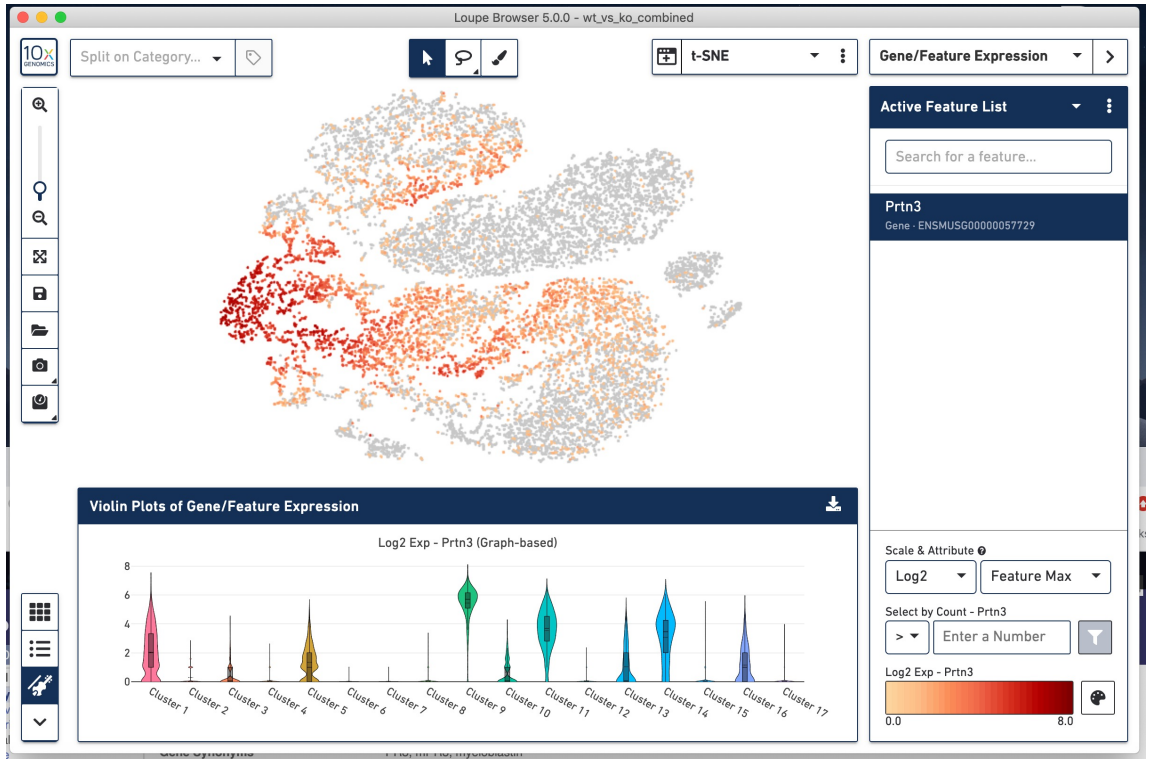
Import custom t-SNE or UMAP coordinates, clusters, etc



Define complex filters to select cells based on expression, cluster membership

Loupe 6.0 enables filtering, complete re-analysis of subpopulations, etc.





Exercise using Loupe browser and the file wt_vs_ko_combined.cloupe

This is bone marrow data from *DNMT3A* wild-type and *DNMT3A* knockout mice. Bulk RNA-seq revealed no expression differences.

1. scRNAseq reveals some significant differences in cell type composition between the WT and KO strains. What are they?
2. There are four clusters of macrophages in this data set. One cluster is missing from the KO mouse. Find this cluster, and generate a heatmap of genes that are differentially expressed across those four clusters. How is the “missing” cluster different from the others in terms of gene expression?

Part II: Analysis of the gene x cell (aka feature-barcode) matrix

By the end of this lecture, you will:

- Understand the rationale underlying downstream scRNAseq analysis steps
- Understand how to assess (and improve) the quality of scRNAseq data
- Understand to interpret scRNA-seq data from a biological standpoint
- Learn how to perform custom analyses of scRNAseq data using R
- Implement a complete gene-barcode matrix analysis pipeline in R

Analysis governed by two main principles

1. Single-cell RNA-seq data is very high-dimensional
2. And very sparse:
   Fraction of transcripts captured per cell:
   10x 3' V2: 14-15%
   10x 3' V3: 30-32%

Additional Quirks:

- Transcripts encoding ribosomal proteins can comprise 30-50% of reads
- Top 100 transcripts often comprise ~50% of reads
- Low sensitivity
- Works best for cell type classification, more subtle signatures may get lost
- Results may favor highly expressed genes (e.g. *VIM*)

Methods Galore

Number of single cell tools ~ 1104 (https://www.scrna-tools.org/)

Point and Click:
Loupe Browser
Partek Flow ($$$)
Flow-Jo ($$$)

Large data sets:
scSVA
SAUCIE

Pseudotemporal Ordering:
Monocle 3 (R)
Slingshot (R)
PAGA (Python)
pCreode (Python)

General-purpose:
Seurat V3
Monocle V3
scran
LIGER (NMF)
CellHarmony
scAlign
Scanorama

Predicting the future:
RNAvelocity
scVelo

Cell type assignment:
Cellassign
CellHarmony
scPred
Moana
Garnet

Mutation Detection:
CONICSmat (CNV)
HoneyBadger (CNV, LOH)
cb_sniffer (SNVs, Indels)
Vartrix (SNVs, Indels)

→ Need thoughtful, creative application of existing tools to extract new biology

Analysis of the gene-by-cell matrix: Overview

- For multiple samples, optionally subsample to achieve comparable sequencing depth using ‘cellranger aggr’ function

- **Read in data, do gene and cell filtering**
  - **Retain genes present in >= x cells**
  - **Retain cells containing >= y genes**

- Merge samples if not done in cellranger

- **Batch correction, if necessary (cellranger does not do this properly)**

- **Data quality overview for subsequent filtering**
  - **Plot distribution of Genes/cell, UMIs/cell, mitochondrial percentage per cell, ribosomal percentage per cell**

- **Secondary cell filtering (depends on data set, questions):**
  - **Genes and/or UMIs**
  - **Mitochondrial transcript %**
  - **Ribosomal transcript %**

Analysis of the gene-by-cell matrix: Overview

- **Calculate G1/S and G2/M scores for each cell**
- **Identify variable genes, scale and normalize expression (or use the one-step alternative, SCTransform)**
- **Remove unwanted sources of variation**
  - **Cell cycle (total cell cycle, not “Cell cycle difference”)**
  - **Mitochondrial percentage**
  - **Ribosomal percentage**
  - **Cell cycle**
  - **Combinations of these variables**

Analysis of the gene-by-cell matrix: Overview

- Principal Component Analysis (PCA) on variable genes

- **Retain and plot key information about each principal component (PC):**
  - Percentage of standard deviation explained
  - P-value (obtained from bootstrapping “Jackstraw”)
  - Plot gene expression heatmaps for each of the top ~12 principal components

- **Choose Principal Components**:
  - Purpose: choose relative importance of minor expression signatures
  - Discontinuity in elbow plot (of standard deviation explained by each PC)
  - All PCs that explain >= 2% of SD
  - P-value from JackStraw analysis < $1x10^{-100}$
  - Clarity of PC heatmaps

- Compute t-SNE and **UMAP** layouts on **n** Principal Components (NB: not raw data)
  - Old data, single samples: 5-10
  - New data, multiple samples: 20-30
  - Cellranger default: 10
  - Partek default: 50

Analysis of the gene-by-cell matrix: Overview

- Clustering
  - A tool for finding patterns in the data
  - Graph-based (unsupervised, must specify resolution (0.7))
  - Alternative: k-means (supervised, must specify k)

- Characterizing Clusters in terms of individual genes
  - Differentially expressed genes (numerous methods)
  - PC-perspective
    - Choose genes that contribute heavily to top principal components
    - Plot heatmaps of these genes in each cluster
    - Independent of clustering
    - Shows relationships of clusters to each other

- Cell lineage inference

- R package (R3.5+) containing functions and data structures for single-cell data, including scRNA-seq, scATAC-seq, CITE-seq, etc.
- Useful references:
  - https://satijalab.org/seurat/
  - Basic workflow and command list: https://satijalab.org/seurat/essential_commands.html
  - Tutorial on newest pipeline (using SCTransform): https://satijalab.org/seurat/v3.1/sctransform_vignette.html
  - Older pipeline: https://satijalab.org/seurat/v3.1/multimodal_vignette.html
  - Details on getting information and data into and out of the Seurat object: https://github.com/satijalab/seurat/wiki
- Key terms:
  - Features = Genes (and/or proteins if using CITE-seq)
  - Counts = UMIs
  - Barcodes = Cells

Getting Help on R (and Seurat) functions

1. In R terminal, type:
   ?FunctionName
   example: ?FindAllMarkers

2. Code available on github, e.g.:
https://github.com/satijalab/seurat/blob/master/man/FindClusters.Rd

Read data, create a Seurat object, and perform initial filtering

Purpose: eliminate genes with essentially no expression, and cells with very few genes

Caution!

- Rare genes may be expressed in only a few cells
- Different cell types have different numbers of genes. Example: Red blood cells express only ~200 genes in some 10x data sets.
  - Don't filter out potentially important cells.
  - Consider cell-type-specific filtering thresholds

picky

```
scrna.counts <- Read10X(data.dir = "/yourpath/outs/filtered_feature_bc_matrix")
scrna <- CreateSeuratObject(counts = scrna.counts, min.cells = 10, min.features = 100, project = Project1)
```

Multiplexed version: Combining multiple samples in Seurat

Purpose: Read, filter, and merge an arbitrary number of samples into a single Seurat object

```
# matrix.dirs is a list of directories containing 10x data

data.10x = list(); # declare a list of 10x data objects

for (i in 1:nsamples) { # nsamples = number of samples
    data.10x[[i]] <- Read10X(data.dir = matrix.dirs[i]);

}

scrna.list = list(); # a list of Seurat objects

for (i in 1:length(data.10x)) {
     scrna.list[[i]] = CreateSeuratObject(counts = data.10x[[i]], min.cells=10, min.feat
ures=100, project = “Project1”);
     scrna.list[[i]][["Sample"]] = samples[i]; # optional: assign a sample name from a v
ector ‘samples’, e.g. samples = c(“A”,”B”,”C”)

}
scrna <- merge(x=scrna.list[[1]], y=c(scrna.list[[2]],scrna.list[[3]]), add.cell.ids = c
(”name1","name2","name3”…)) # create merged Seurat object
```

Alternative: Combining multiple samples in cellranger

cellranger downsamples the data sets to the same [lowest] sequencing depth

```
cellranger aggr --id=$AggOutName --csv=$af --normalize=mapped --mempercore=64"
```

Definitions:
$AggOutName = Name of the output directory for the aggregated samples
$af = Full path to aggregation file, e.g. /path/to/aggregationfile.csv

Aggregation file format:
library_id,molecule_h5
Name1,/PathToData/Name1/outs/molecule_info.h5
Name2,/PathToData/Name2/outs/molecule_info.h5

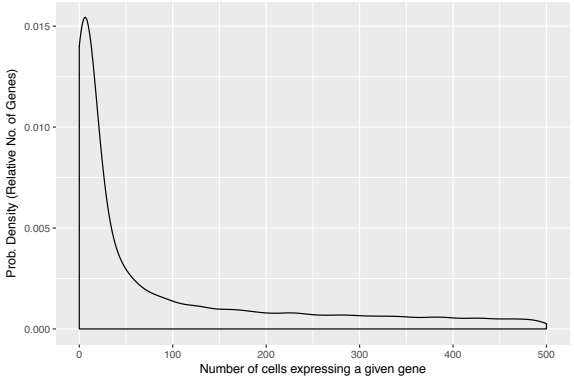
Plot key parameters (per sample) to choose filtering cutoffs

Goals of Plotting and Filtering:

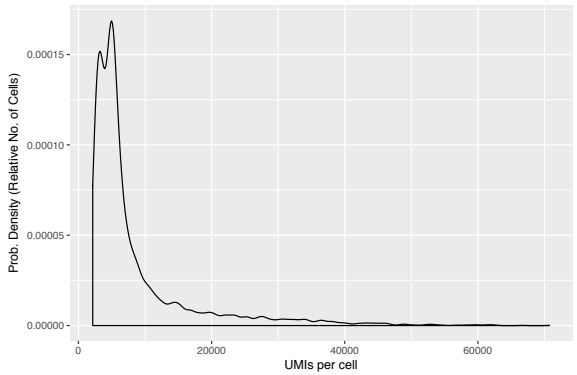
1. Eliminate apoptosing or leaky cells (high percentage of mitochondrial transcripts)
2. Eliminate free-floating transcripts or under-sequenced cells (too few UMIs or genes per cell)
3. Eliminate doublets (too many UMIs or genes per cell)
4. Know what's in your data
    - Is it dominated by a few genes?
    - Is it dominated by ribosomal protein-encoding genes?
    - How heterogeneous are the cells?
    - Does the data suggest that the sample preparation protocol need to be adjusted?

Distributions of key parameters

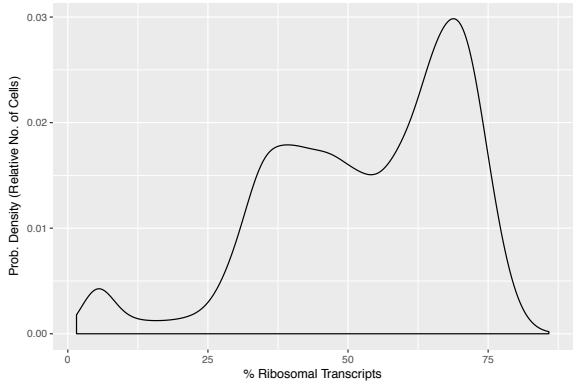
Most genes measured in a few cells!



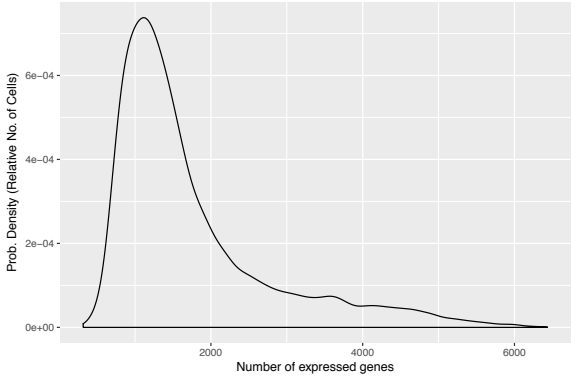
Distribution of UMIs/cell



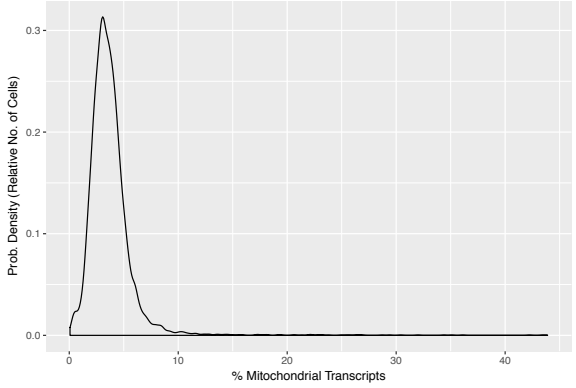
Distribution of Ribosomal transcripts/cell



Most cells: ~1200 genes/cell



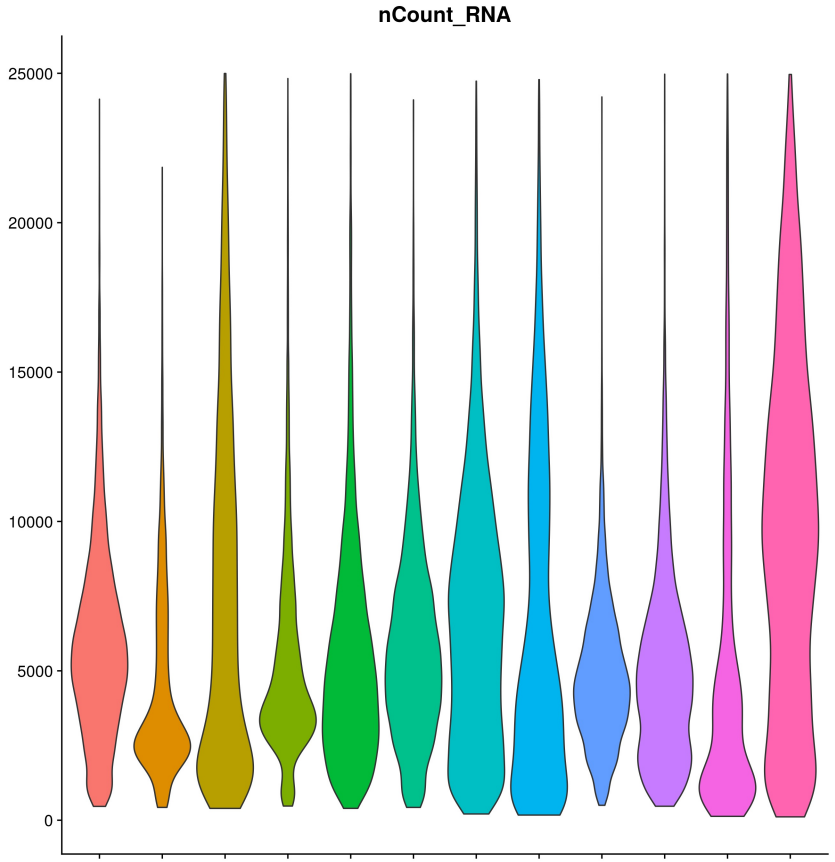
Distribution of MC transcripts/cell

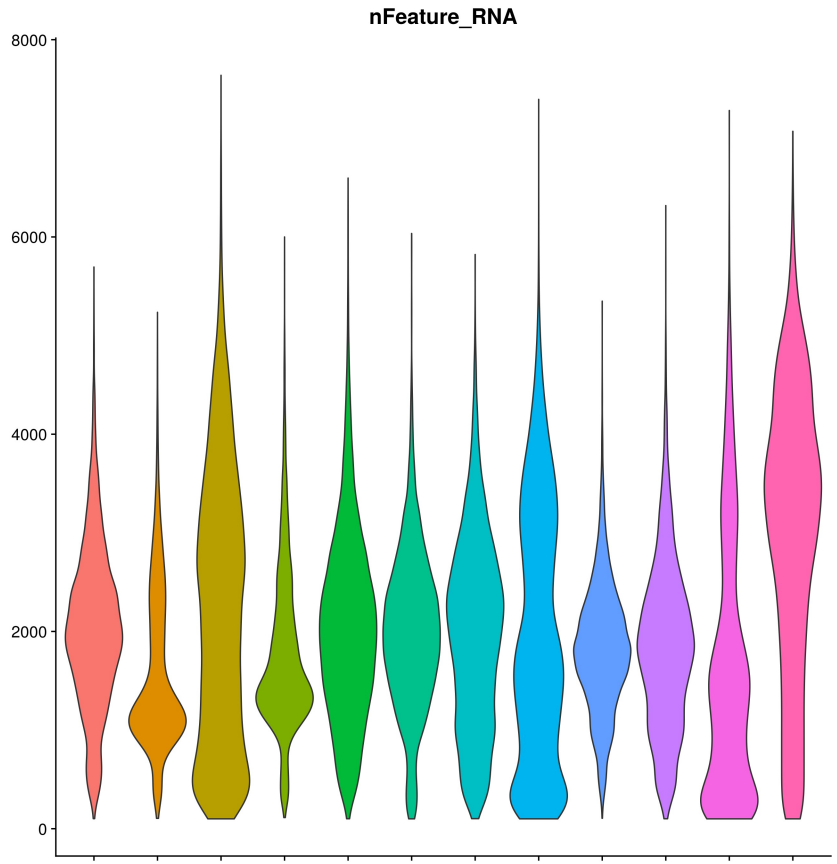


Distribution of HK transcripts/cell

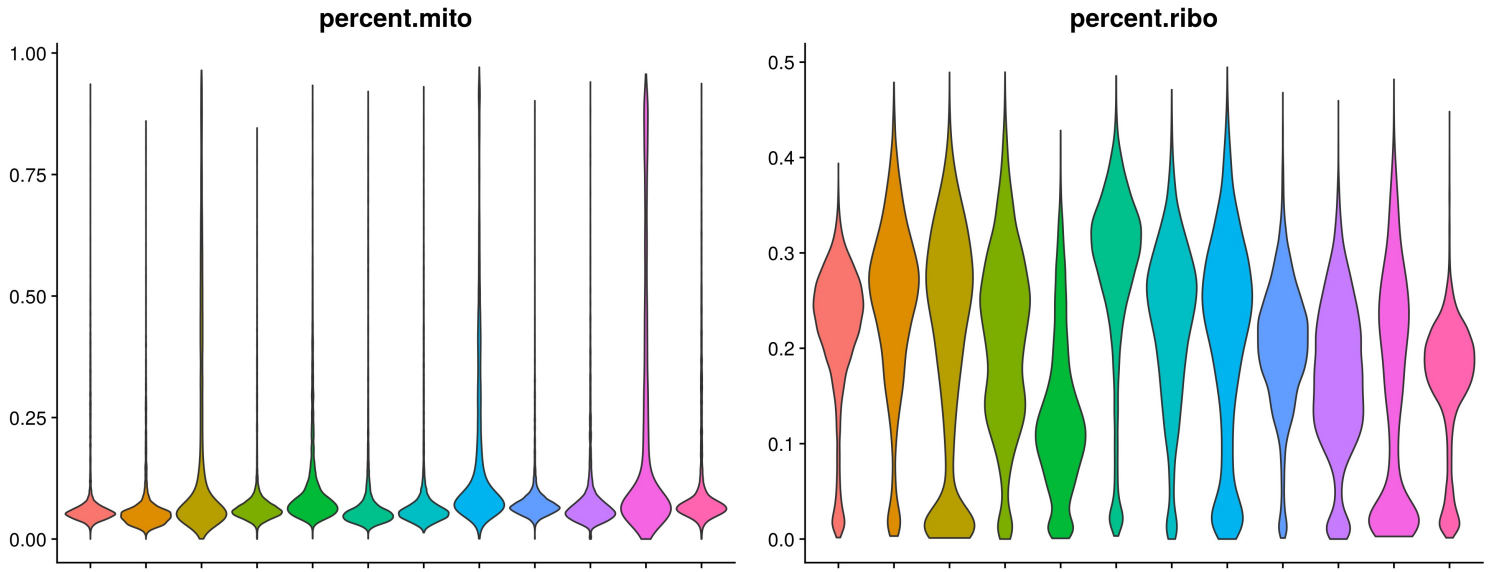
Properties of the data influence interpretation and analysis

Plot counts (UMIs) and features (genes) for every data set





Plot Mitochondrial and Ribosomal Protein content for every data set



- High MC content implies cellular damage (resulting in loss of cytoplasmic transcripts) or impending apoptosis
  - Ilicic, et al. Classification of low quality cells from single-cell RNA-seq data. (2016) Genome Biology 17:29
  - Marquez-Jurado et al. Mitochondrial levels determine variability in cell death by modulating apoptotic gene expression. (2018) Nat. Comm. 9:389
- Ribosomal content: Reflects transcriptional diversity, and probably proliferative potential. (I do not recommend filtering for ribosomal content.)

Sample R and Seurat code for parameter plots

NB: The number of genes and UMIs (nGene and nUMI) are automatically calculated for every object by Seurat and stored in the meta data.

```
# Calculate percentage of mitochondrial genes
mito.genes <- grep(pattern = "^MT-", x = rownames(x = scrna), value = TRUE);
percent.mito <- Matrix::colSums(x = GetAssayData(object = scrna, slot = 'counts')[mito.genes, ]) /
Matrix::colSums(x = GetAssayData(object = scrna, slot = 'counts'));
scrna[['percent.mito']] <- percent.mito; # assign it to the meta data

# ribosomal genes
ribo.genes <- grep(pattern = "^RP[SL][[:digit:]]", x = rownames(x = scrna), value = TRUE);
percent.ribo <- Matrix::colSums(x = GetAssayData(object = scrna, slot = 'counts')[ribo.genes, ]) /
Matrix::colSums(x = GetAssayData(object = scrna, slot = 'counts'));
scrna[['percent.ribo']] <- percent.ribo; # assign it to the meta data

vln <- VlnPlot(object = scrna, features = c("percent.mito", "percent.ribo"), pt.size=0, ncol = 2,
group.by="Batch"); # make a violin plot, and color by batch or sample

vln <- VlnPlot(object = scrna, features = "nCount_RNA", pt.size=0, group.by="Batch", y.max=25000)

vln <- VlnPlot(object = scrna, features = "nFeature_RNA", pt.size=0, group.by="Batch")
```

Some numbers

(50K reads/cell, 3' V2 kit, cellranger V2, circa 2017)

The average gene is detected in ~150 cells

~30-50% of the reads are from transcripts that encode ribosomal proteins

When detected, a gene is represented by one read on average, but the range is huge!

Metric	Sample1	Sample2	Sample3
gene.total	21342	21036	22377
gene.per.cell.mean	1499	1454	1751
gene.per.cell.med	1381	1438	1395
gene.per.cell.min	431	383	317
gene.per.cell.max	4447	4059	6435
gene.per.cell.sd	524	478	1052
cell.per.gene	168	123	123
umi.per.cell.mean	4186	4361	8412
umi.per.cell.med	3570	4144	5296
umi.per.cell.min	1655	1278	2143
umi.per.cell.max	21273	18114	70759
umi.per.cell.sd	2253	2049	8574
umi.per.gene.mean	1	1	1
umi.per.gene.max	1150	2397	17067
umi.per.gene.sd	2	2	5

More numbers (50K reads/cell, 3’ kit)

~50% of the reads come from just 100 genes

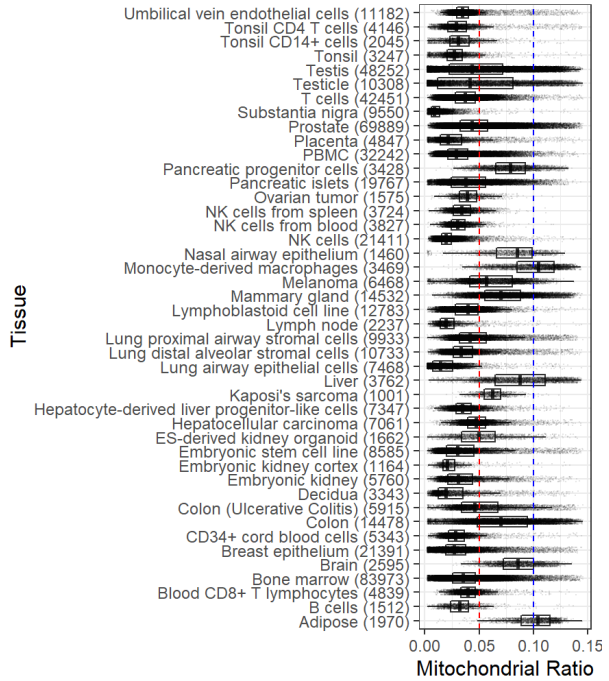
~30-50% of the reads are from transcripts that encode ribosomal proteins

~20-40% of the reads are from housekeeping genes, which are not uniformly expressed

Metric	Sample1	Sample2	Sample3
Topgene %	48	56	57
Ribogene %	32	53	51
ribo.mean %	30	51	53
ribo.med %	29	53	56
ribo.min %	12	14	2
ribo.max %	68	73	86
ribo.sd %	8	8	17
mt.mean %	7	8	4
mt.med %	6	7	3
mt.min %	2	2	0
mt.max %	56	51	44
mt.sd %	2	4	2
hkgene %	22	35	34
hk.mean %	10	16	31
hk.med %	8	15	20
hk.min %	2	2	1
hk.max %	71	72	232
hk.sd %	7	9	31

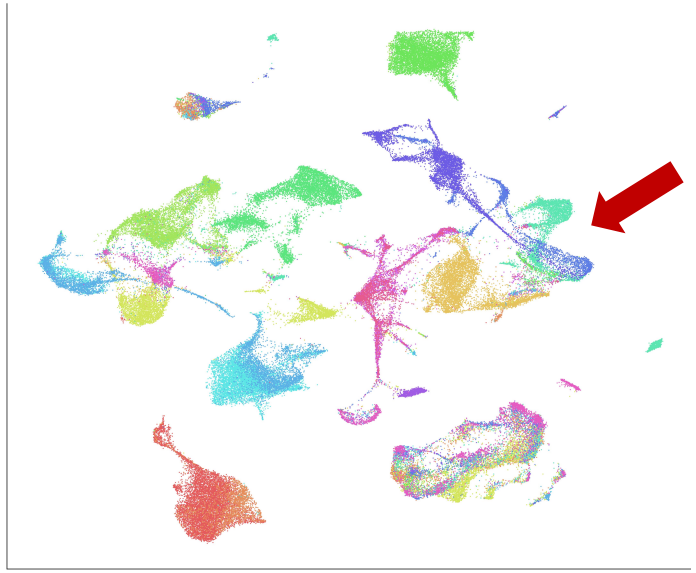
Filter data

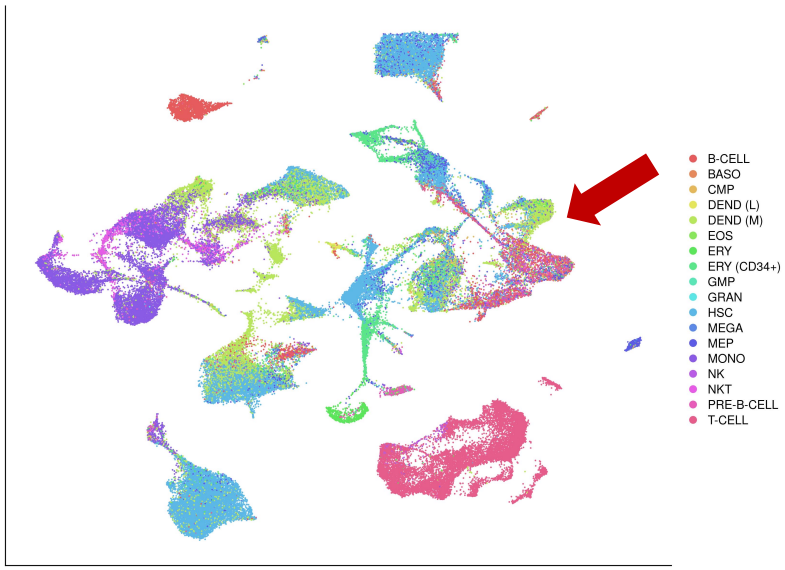
- Use violin plots and/or preliminary clustering results to choose appropriate thresholds for your data.
- Some example thresholds:
  - nFeature_RNA: between 500 and 4000, or between fixed minimum and 95th percentile (for example)
  - nFeature > x, nUMI < y percentile (based on predicted doublet rate of 0.9% per 1000 cells)
  - Mitochondrial content cutoff 5-10%, but is tissue-specific
    - AML 10%
    - 5% for mice, 10% for humans (Osorio and Cai. Systematic determination of the mitochondrial proportion in human and mice tissues for sin- gle-cell RNA sequencing data quality control (2020) BioRxiv)
  - Consider sample specific and/or cell-type specific thresholds

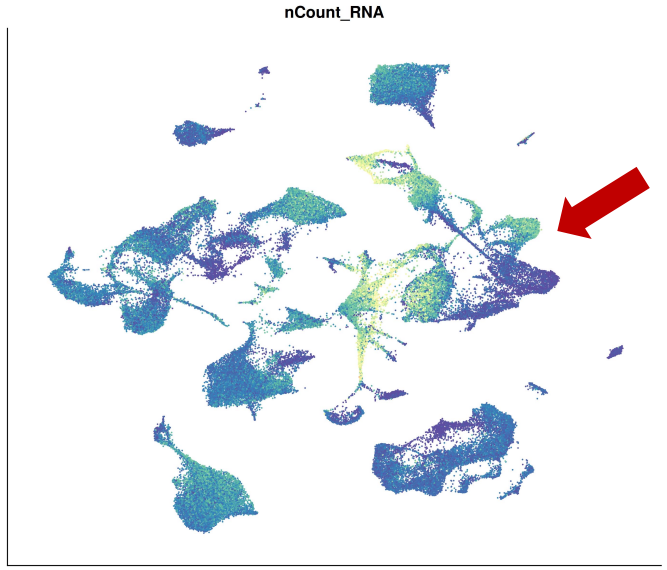



```
scrna <- subset(x = scrna, subset = nFeature_RNA > x & nUMI_RNA < y & percent.mito < z)
```

Example of possibly inadequate filtering



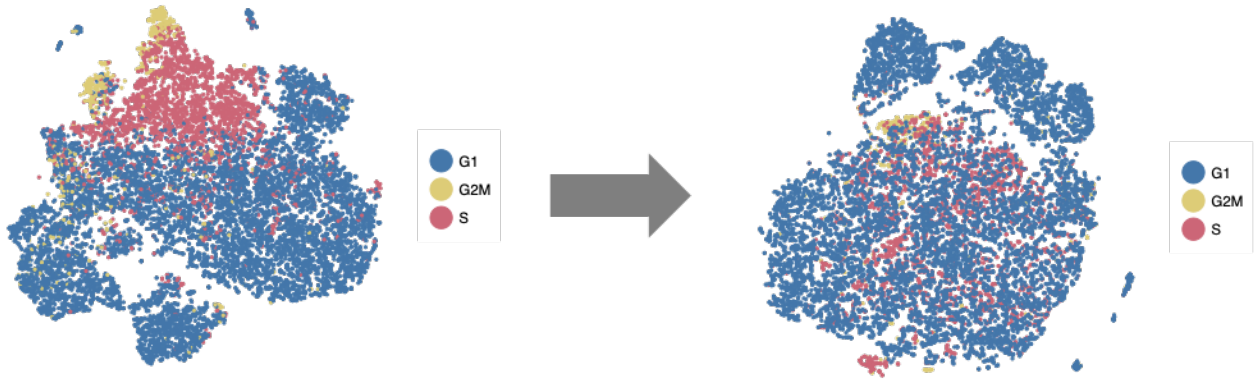




Metric	Cutoff/threshold/range
Genes	>700
UMI	< 93 percentile
Mitochondrial %	<10%

Calculate cell cycle phase of each cell

- Purpose: Cell cycle signature can dominate tSNE/UMAP plots and may need to be removed
- Seurat has a built-in function that calculates relative expression level of G1/S and G2/M genes defined in Tirosh et al 2016



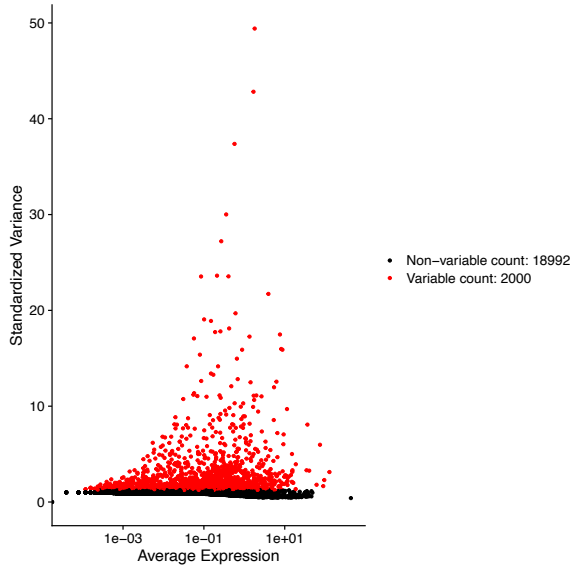

```
cell.cycle.tirosh <- read.table("CellCycleTirosh.txt", sep='\t', header=FALSE);
s.genes = cell.cycle.tirosh$V2[which(cell.cycle.tirosh$V1 == "G1/S")];
g2m.genes = cell.cycle.tirosh$V2[which(cell.cycle.tirosh$V1 == "G2/M")];
scrna <- CellCycleScoring(object=scrna, s.features=s.genes, g2m.features=g2m.genes, set.ident=FALSE)
```

Step 5: Normalize, scale, control for unwanted variation

- **Goal: remove technical effects while preserving biological variation**
- Sequencing depth of a cell: total UMI (nCounts)
- Even in the same experiment, different cells have very different sequencing depths
- Expression level of a gene in a cell is proportional to the sequencing depth of the cell, unless the data is normalized to sequencing depth
- Older normalization approach(es) scale every gene in the cell by the same factor: the sequencing depth
  - **NormalizeData**:
    - Divide by total counts in each cell
    - Scale to fixed counts (default is $1x10^4$ use $1x10^6$ for CPM)
    - Add 1
    - natural log
  - **FindVariableFeatures:** use a variance stabilizing transformation
  - **ScaleData:** subtract mean, divide by standard deviation, remove unwanted signal using multiple regression

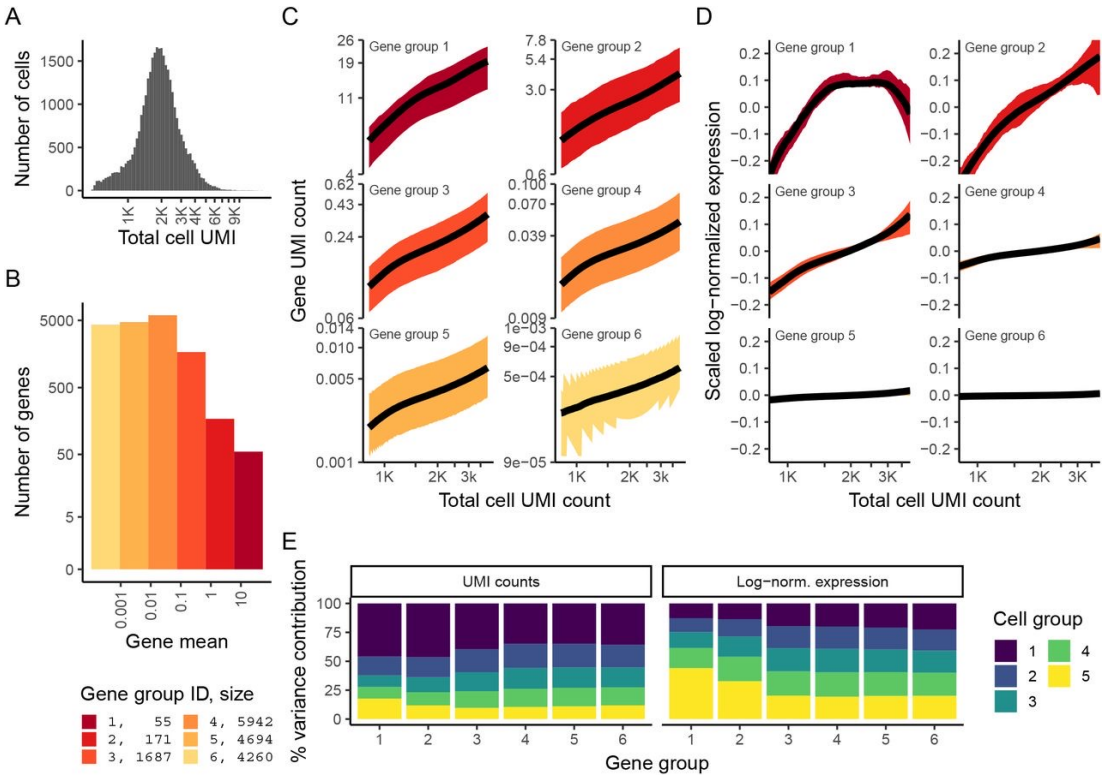
Note on FindVariableFeatures

- vst: Variance-Stabilizing Transformation
  - For each gene, plot log(Var) vs log(mean)
  - Use loess linear regression to calculate expected variance based on observed mean
  - Standardize the expression of each gene so that its variance matches the calculated expected variance
  - Plot standardized variance vs log(mean) and choose outliers
  - Choose 2000-3000 outliers
- mean.var.plot
- dispersion



Regularized negative binomial normalization with Seurat's SCTransform function

- Variance stabilizing transformation used in FindVariableFeatures affects different genes differently
- Highly and lowly expressed genes (B, C) are affected differently by scaling factor (D)
- Variance related to sequencing depth, and traditional normalization unevenly changes contribution of each gene to overall variance
- Solution: Use negative binomial regression (with parameters derived from groups of genes with similar expression) to remove impact of sequencing depth. Seurat function SCTransform.



Hafemeister and Satija. bioRxiv 2019

Two approaches to normalization and scaling in Seurat:

```
# I. Variance Stabilizing Transformation (with removal of cell cycle signal):

scrna <- NormalizeData(object = scrna, normalization.method = "LogNormalize", scale.factor = 1e4) # fea
ture counts divided by total, multiplied by scale factor, add 1, ln-transformed

scrna <- FindVariableFeatures(object = scrna, selection.method = 'vst', mean.cutoff = c(0.1,8), dispers
ion.cutoff = c(1, Inf)) # designed to find ~2000 variable genes

scrna <- ScaleData(object = scrna, features = rownames(x = scrna), vars.to.regress = c("S.Score","G2M.S
core"), display.progress=FALSE) # center and regress out unwanted variation

# II. SCTransform (with removal of cell cycle signal):

scrna <- SCTransform(scrna, vars.to.regress = c("S.Score", "G2M.Score"), verbose=FALSE)
```

The fine print:

• scrna[["SCT"]]@scale.data contains the residuals (normalized values), and is used directly as input to PCA. To save memory, we store these values only for variable genes, by setting the return.only.var.genes = TRUE by default in the SCTransform function call.

• To assist with visualization and interpretation, we also convert Pearson residuals back to 'corrected' UMI counts. You can interpret these as the UMI counts we would expect to observe if all cells were sequenced to the same depth.

• The 'corrected' UMI counts are stored in scrna[["SCT"]]@counts. We store log-normalized versions of these corrected counts in pbmc[["SCT"]]@data, which are very helpful for visualization.

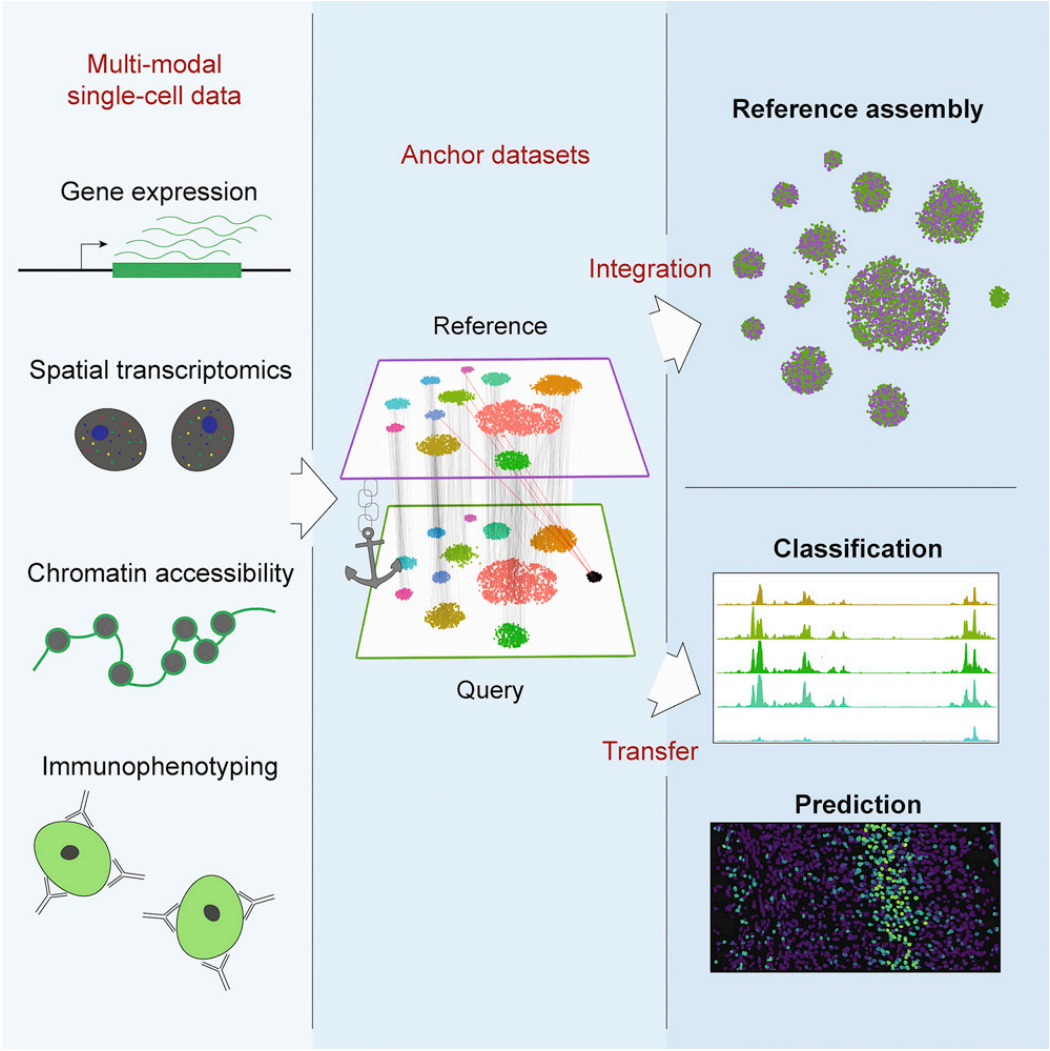
• You can use the corrected log-normalized counts for differential expression and integration. However, in principle, it would be most optimal to perform these calculations directly on the residuals (stored in the scale.data slot) themselves. This is not currently supported in Seurat v3, but will be soon.

Batch correct/integrate instead of merging (optional)

Avoid batch correction unless absolutely necessary

When to use it:

- Correct for different technologies (e.g. 3’ and 5’)
- Correct for different batches
- Discover conserved biology by finding corresponding cells across different data sets
- Combining data of different types (e.g. scRNA-seq, ATAC-seq)



Batch correct/integrate instead of merging (cont'd)

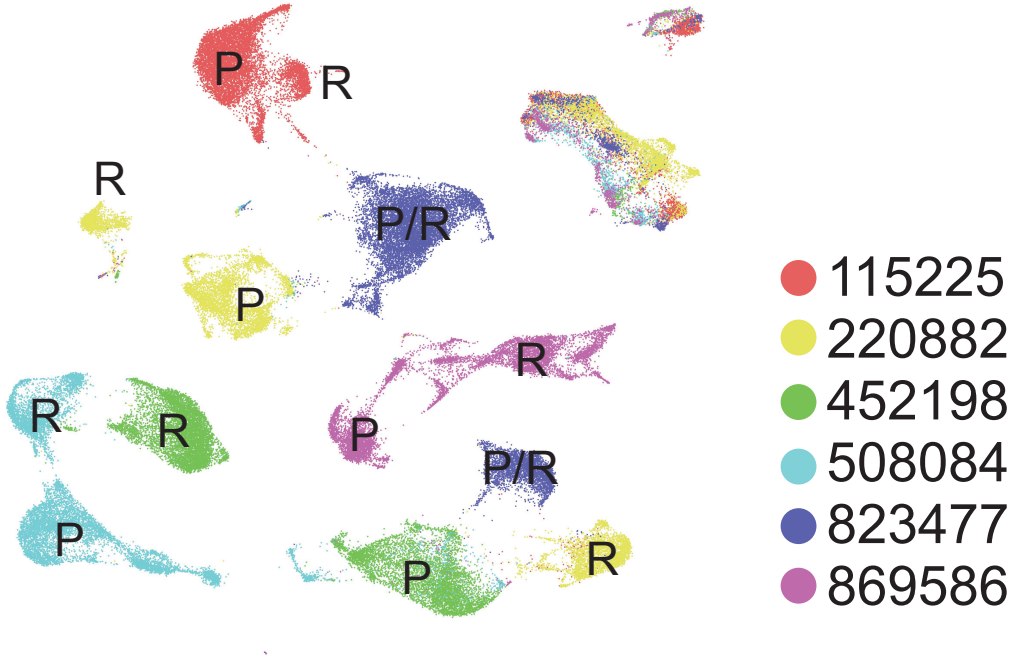
```
# matrix.dirs is a list of directories containing 10x data
data.10x = list(); # declare list of 10x data sets
for (i in 1:length(matrix.dirs.)) { # for each data set
     data.10x[[i]] <- Read10X(data.dir = matrix.dirs[i]); # add it to the list
}
scrna.list = list(); # declare list of Seurat objects
for (i in 1:length(data.10x)) { # for each data set…
    scrna.list[[i]] = CreateSeuratObject(counts = data.10x[[i]], min.cells=10, min.features=100,project
=batch[i]); # …create a Seurat object for each data set
    scrna.list[[i]][["Batch"]] = batch[i]; # assign a batch label from 'batch'
    scrna.list[[i]][["Sample"]] = samples[i]; # assign a sample name from 'samples'
}
# Not shown: Filter each sample separately and normalize (using same method for all samples)
anchors <- FindIntegrationAnchors(object.list = scrna.list, dims = 1:30) # find anchors
scrna.int <- IntegrateData(anchorset = anchors, dims = 1:30) # Integrate data
DefaultAssay(object = scrna.int) <- "integrated" # make integrated data the default for downstream anal
yses
```

Batch correction and integration: The fine print

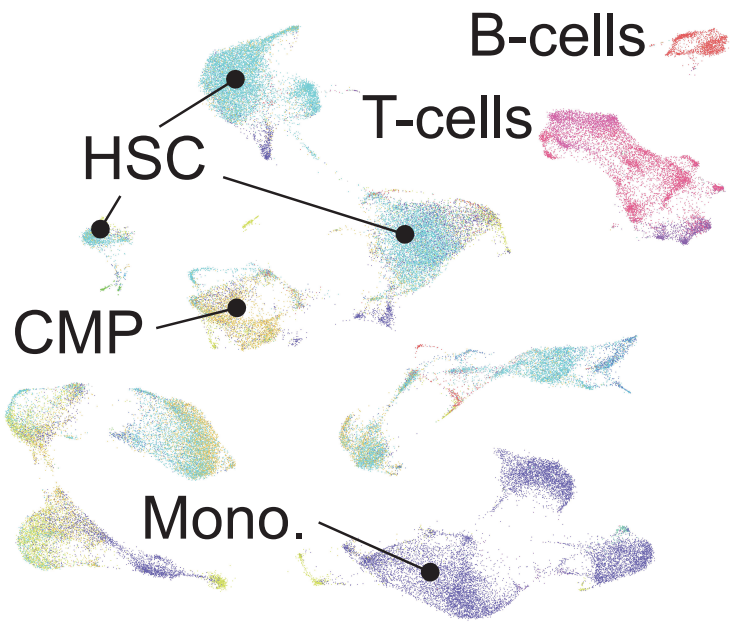
- Integrated values not intended for use with differential expression calculations.
    - We recommend running your differential expression tests on the “unintegrated” data. By default this is stored in the “RNA” Assay. There are several reasons for this.
    - The integration procedure inherently introduces dependencies between data points. This violates the assumptions of the statistical tests used for differential expression.
- SCTransformed data requires a different batch-correction workflow. (DE is not supported yet). See https://satijalab.org/seurat/v3.0/integration.html
- TransferData function uses data integration to classify cells based on a reference data set.
- Cellranger does faux batch-correction (corrected values are discarded), but batch-corrected tSNE can be visualized in the loupe browser.

Paired presentation/relapse samples from 6 AML patients

- Every patient a different batch
- No batch correction
- Malignant cells from all samples are unique
- Normal cells co-cluster

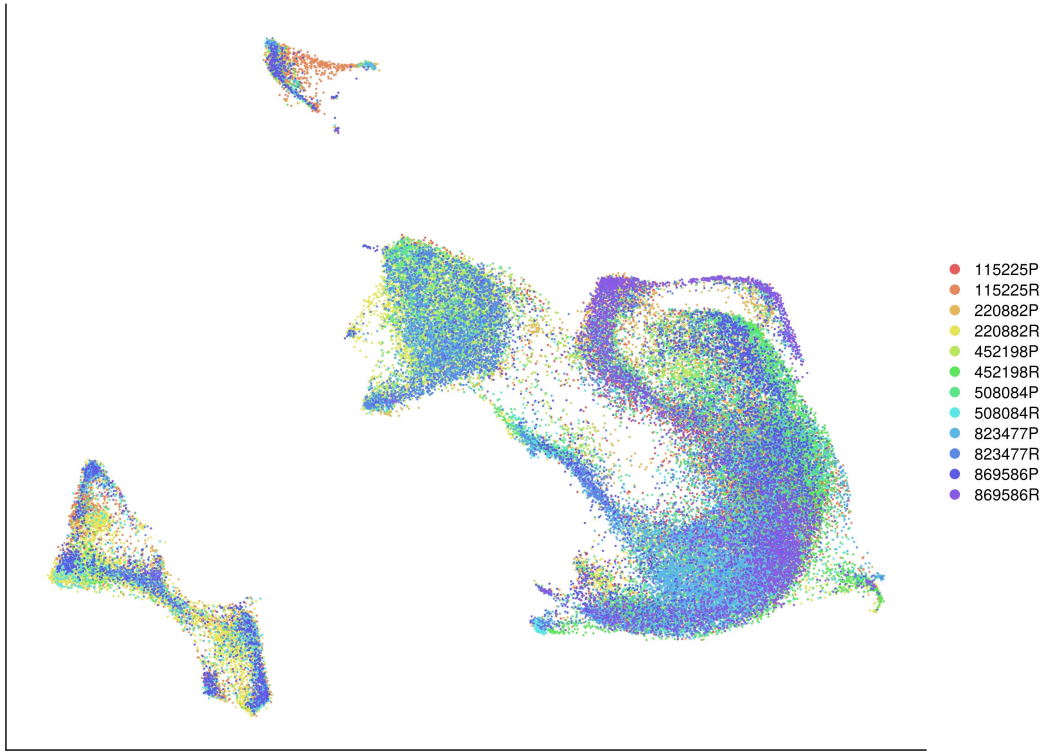


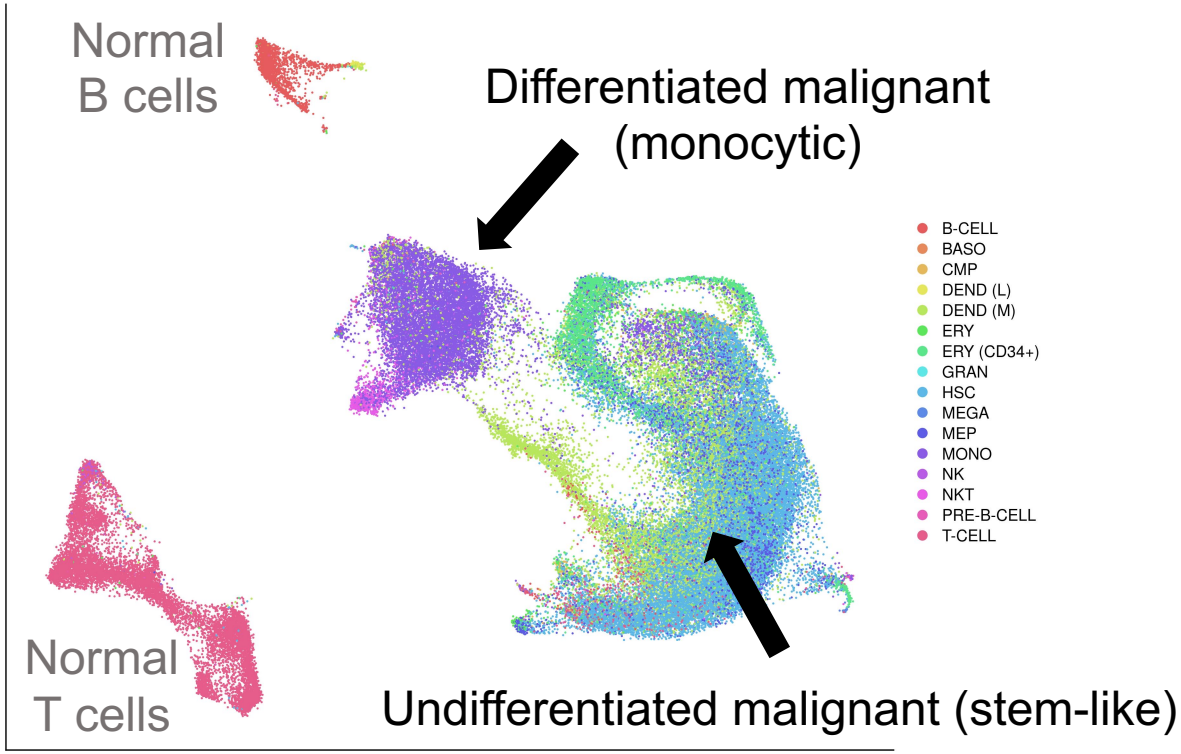
115225
220882
452198
508084
823477
869586



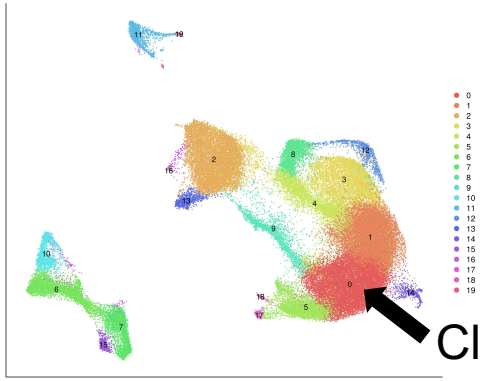
B-cell
BASO
CMP
DEND
EOS
ERY
GMP
GRAN
HSC
MEGA
MEP
MONO
NK
NKT
No.Prediction
T-Cell(CD4+)
T-Cell(CD8+)

Removing inter-individual differences reveals two malignant cell populations

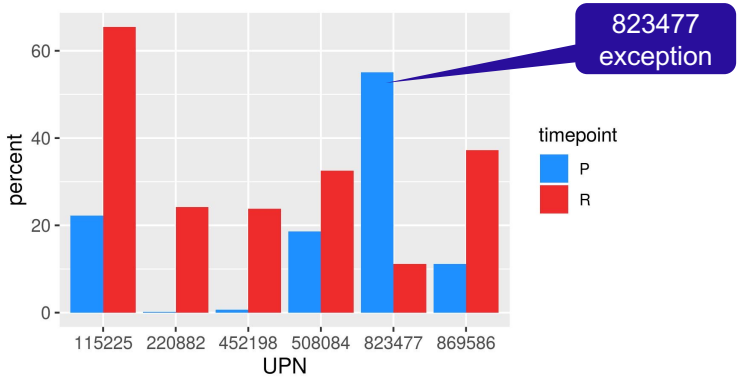




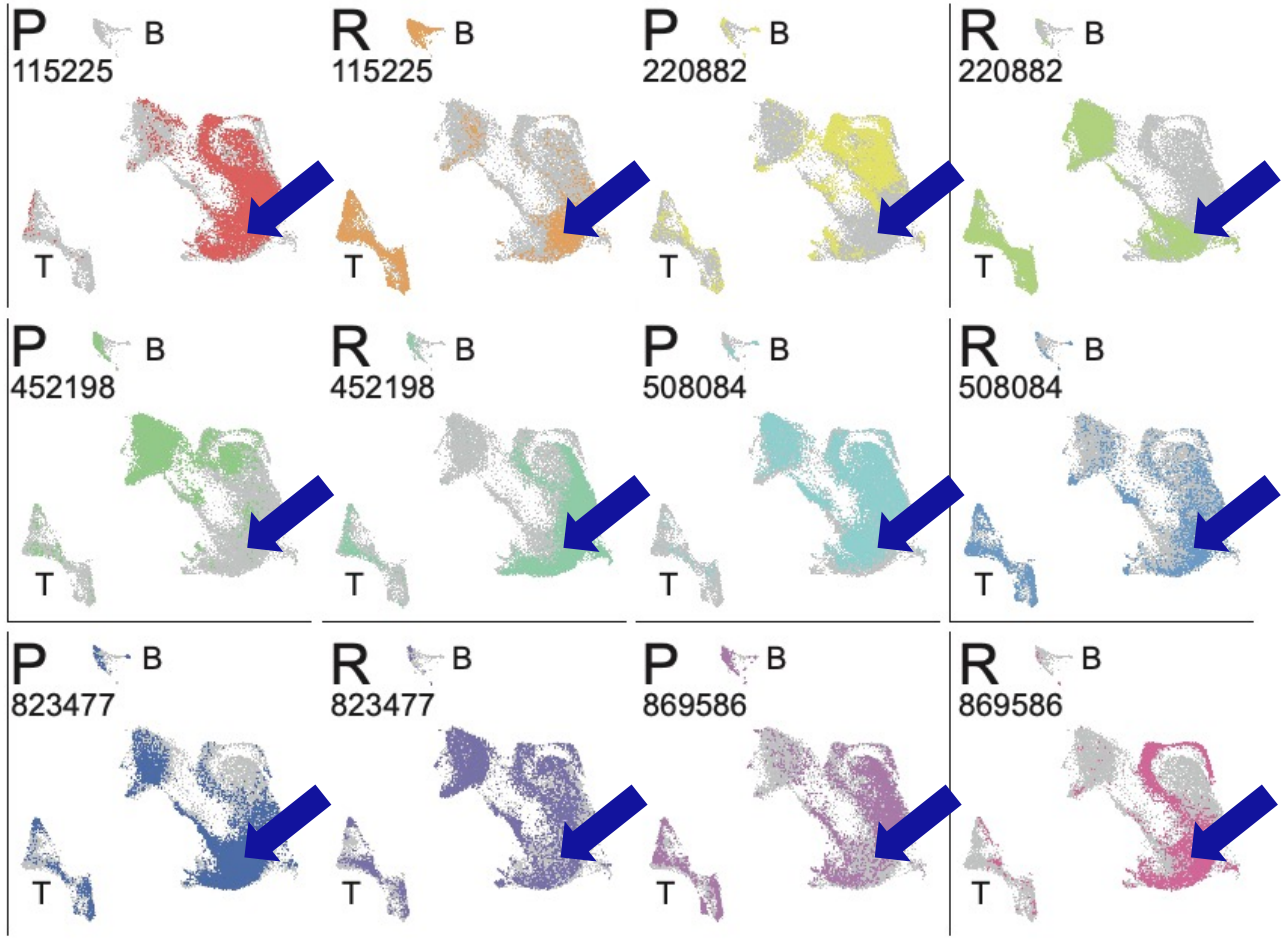
…and a relapse-enriched cluster



Proportion of sample in Cluster 0:



Relapses shift towards
CD34/CYTL1/CD9-high cluster
(Immaturity and adhesion/migration)



Exercise: A Complete Seurat Workflow

In this exercise, we will analyze and interpret a small scRNA-seq data set consisting of three bone marrow samples. Two of the samples are from the same patient, but differ in that one sample was enriched for a particular cell type. The goal of this analysis is to determine what cell types are present in the three samples, and how the samples and patients differ.

There is enough information in this presentation (functions, code) to analyze the three patients on your own if you want a challenge. (ND050119_CD34_3pV3, ND050119_WBM_3pV3, and ND050819_WBM_3pV3)

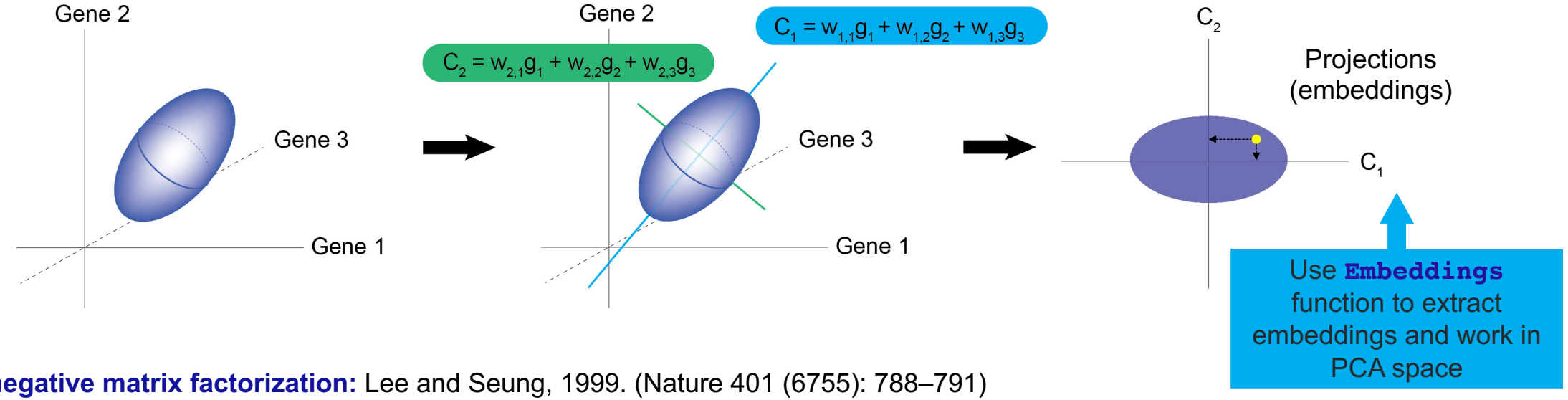
Follow the workflow (with cut-and-paste code) outlined at https://rnabio.org/module-08-scrna/0008/02/01/scRNA/. We will return to the lecture during the saveRDS step at the end of step 9.

Once you finish Step 9, aka the ScaleData step, please click your 'tada' emoji.

Dimensionality reduction using PCA

Purpose: Approximate original data using fewer dimensions. Define new axes that capture as much "information" as possible in as few dimensions as possible.

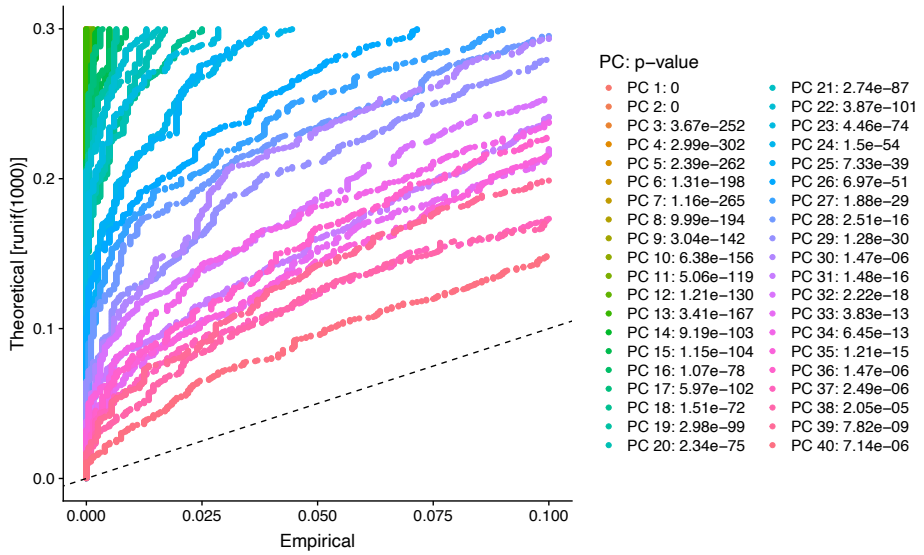
PCA = Principal Component Analysis (similar to SVD - Singular Value Decomposition)
Principal Axes, Eigen Decomposition: Euler (1751)…Cauchy (1829)
SVD: Eugenio Beltrami, 1873 (etc)
PCA: Karl Pearson, 1901
Computation: Gene Golub, Christian Reinsch, 1970
Gene Expression: Orly Alter, Patrick Brown, David Botstein, 2000 (PNAS 97 (18): 10101-10106)
Other applications: image processing, video games, math, statistics, computer science, machine learning, finance, etc.



Non-negative matrix factorization: Lee and Seung, 1999. (Nature 401 (6755): 788–791)
Similar to PCA, but axes are not mutually orthogonal, and clustering and factorization are coupled.

Selecting PCs: Jackstraw analysis

- Calculate statistical significance of each PC
- Randomly permute data
- Recalculate PCs of randomized data
- Compare “real” PCs to “random” PCs to derive significance




```
# NB: jackstraw analysis is slow
scrna <- JackStraw(object = scrna, num.replicate = 100, dims=N) # specify N:
30, 50, etc
scrna <- ScoreJackStraw(object = scrna, dims = 1:50)
js <- JackStrawPlot(object = scrna, dims = 1:20) # make plot shown above
pc.pval <- scrna@reductions$pca@jackstraw@overall.p.values # get overall
pvalues for each PC
```

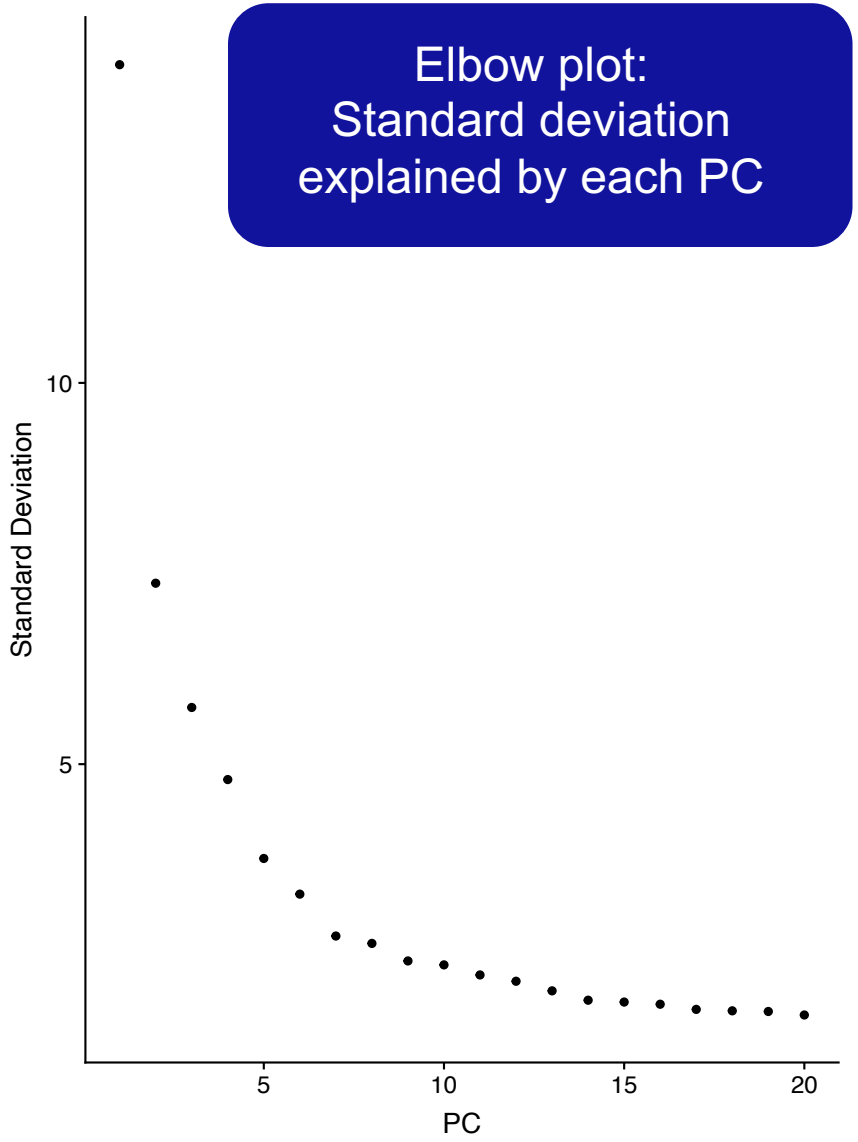
Chung and Storey. Statistical significance of variables driving systematic variation in high-dimensional data. Bioinformatics 2015

Selecting Principal Components

All downstream calculations are done on PCs, not raw data

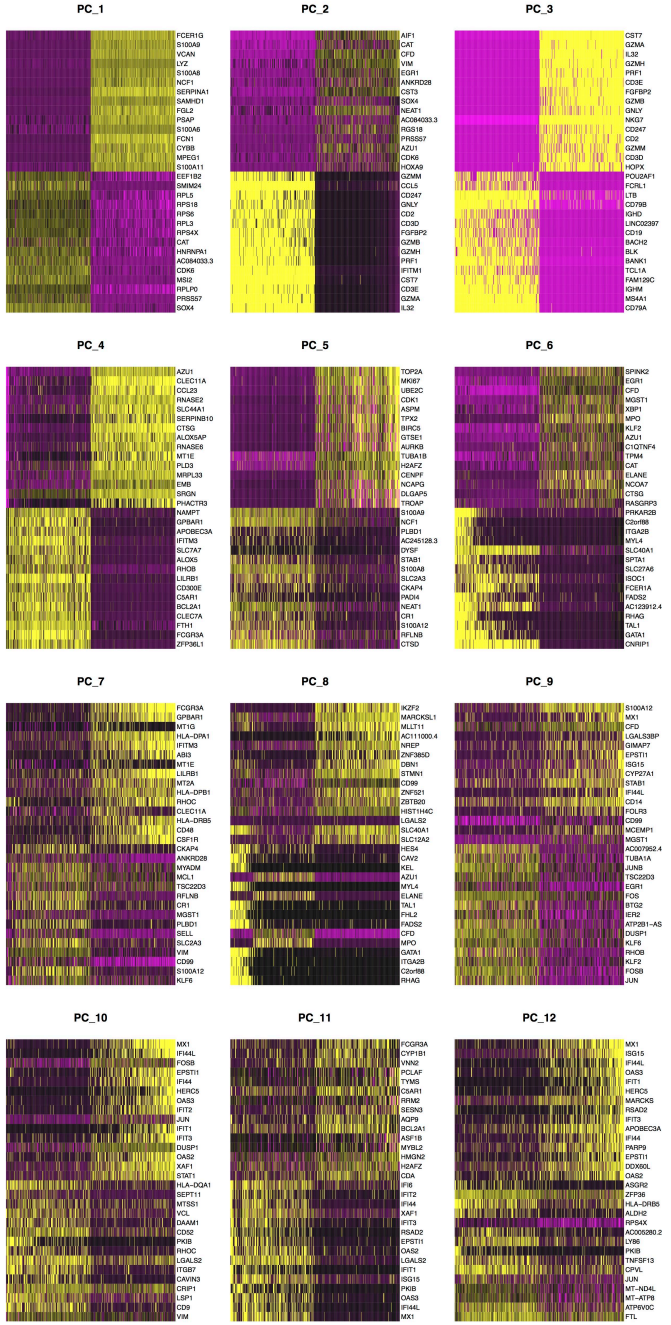
- Retain and plot key information about each principal component (PC):
  - Percentage of standard deviation explained
  - P-value (obtained from bootstrapping “Jackstraw”)
  - Plot gene expression heatmaps for each of the top ~12 principal components
- Choose Principal Components:
  - Purpose: choose relative importance of minor expression signatures
  - Discontinuity in elbow plot
  - All PCs that explain >= x% of SD (e.g. 2%)
  - P-value from JackStraw analysis < $1x10^{-n}$ (e.g. $1x10^{-100}$)
  - Clarity of PC heatmaps
- Number of PCs:
  - Single samples: 5-10
  - Multiple samples: 20-50
  - Cellranger default: 10
  - Partek default: 50

Elbow plot:
Standard deviation
explained by each PC



Minor components contain signal that is not represented in t-SNE/UMAP

PC Tradeoff: More components → more signal, more noise



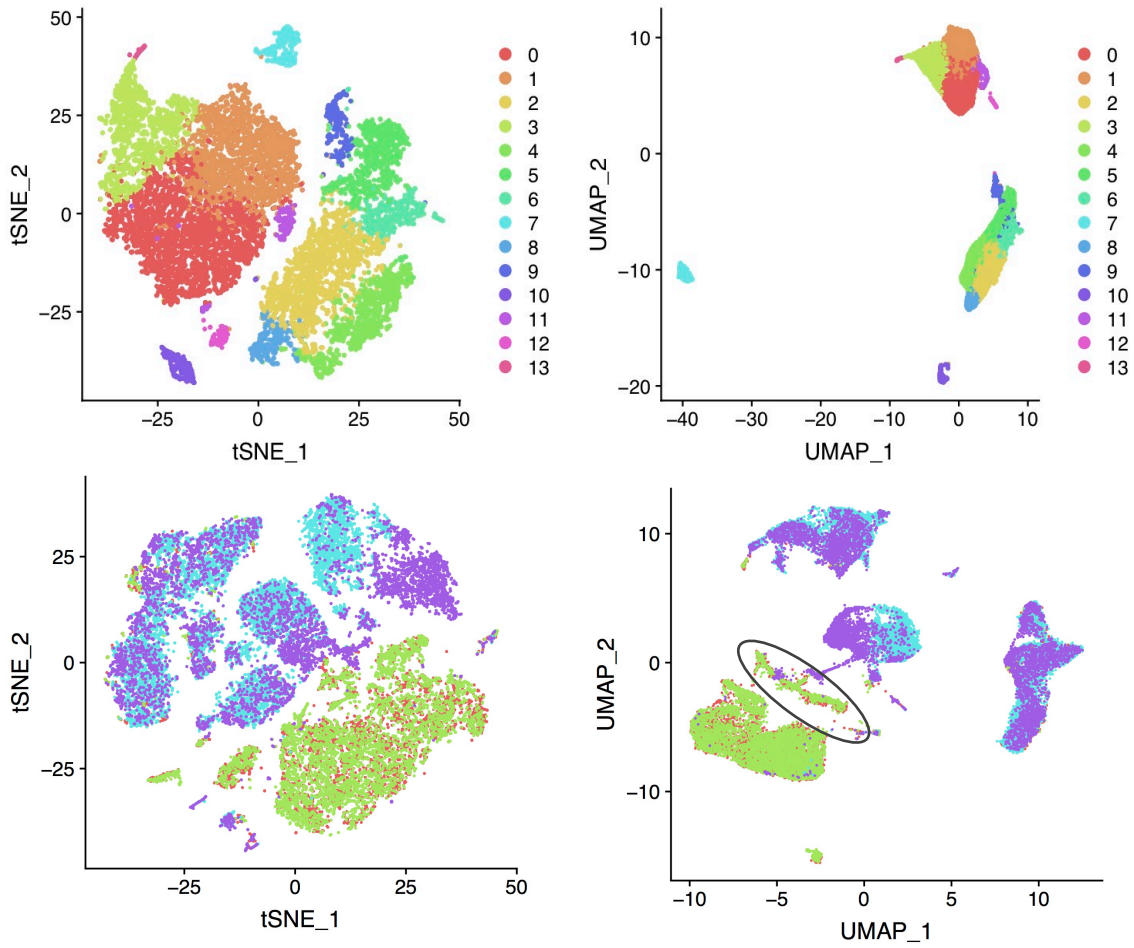
Plotting using t-SNE/UMAP

t-SNE = *t*-distributed Stochastic Neighbor Embedding
UMAP = Uniform Manifold Approximation and Projection

Goal: Embed high-dimensional data in low-dimensional space

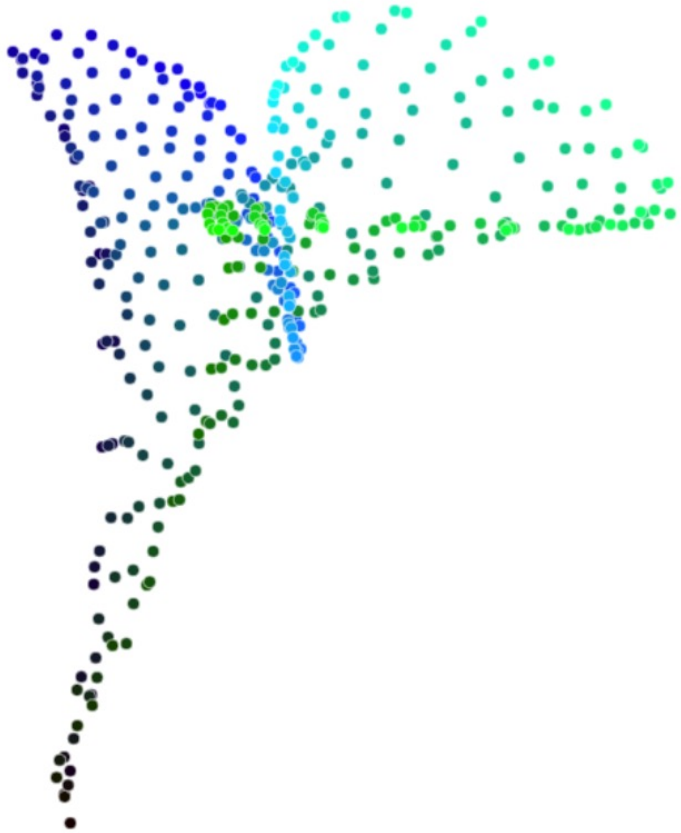
End product: 2D plot where cells are positioned near each other if they have similar gene expression profiles. “Units” are relative and data-dependent.

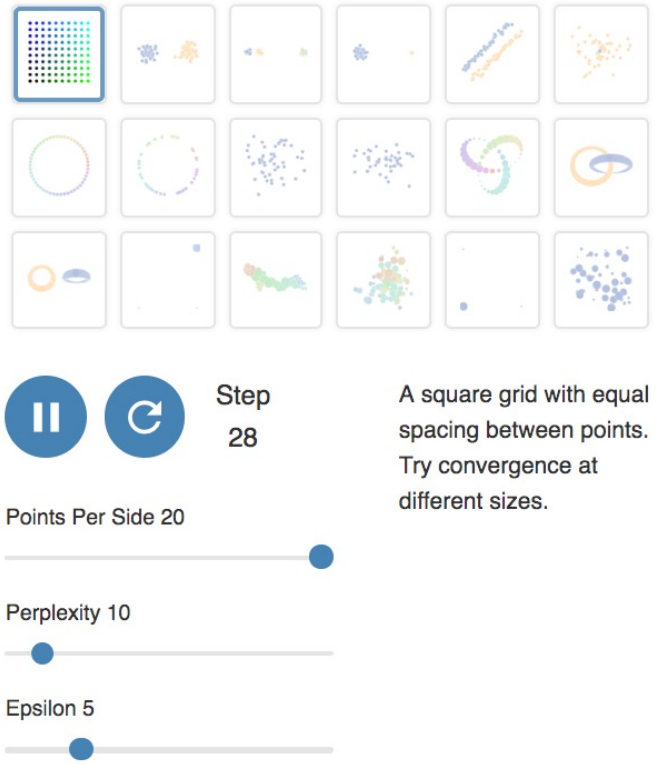
- Expression “distances” between points (ie cells) in high-dimensional space are modeled using a gaussian distribution.
- Distances in low-D space are modeled using a t-distribution.
- Locations in 2D space are chosen s.t. distance (Kulback-Leibler divergence) between the two distributions is minimized.
- Operates in “PCA space”
- “clusters” are not clusters. This is not clustering.
- t-SNE preserves local structure only.
- UMAP preserves local AND global structure.
- Implication: In UMAP, distances between points *and* clusters in UMAP are more interpretable in terms of expression distances/similarity.



How to Use t-SNE Effectively

Although extremely useful for visualizing high-dimensional data, t-SNE plots can sometimes be mysterious or misleading. By exploring how it behaves in simple cases, we can learn to use it more effectively.



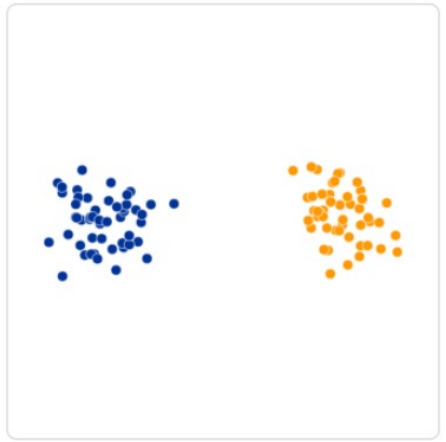


https://distill.pub/2016/misread-tsne/

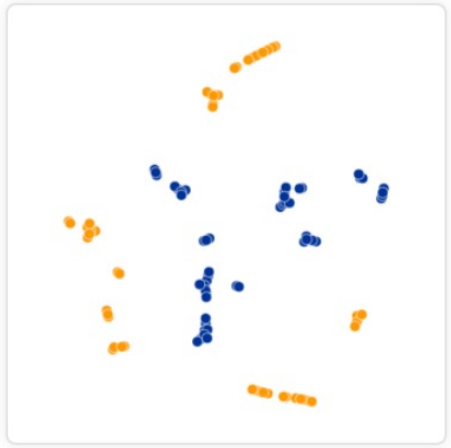
Background

- t-SNE (stochastic neighbor embedding)
- Dimensionality reduction
- Perplexity parameter - tradeoff between local and global aspects

Perplexity matters

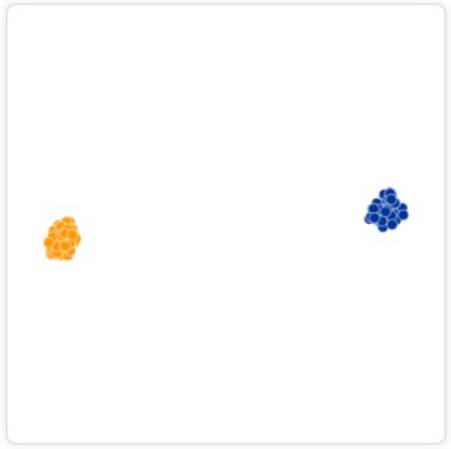


Original



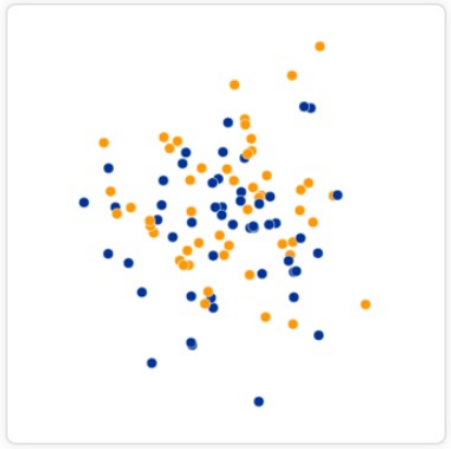
Perplexity: 2
Step: 5,000

Perplexity: 5
Step: 5,000



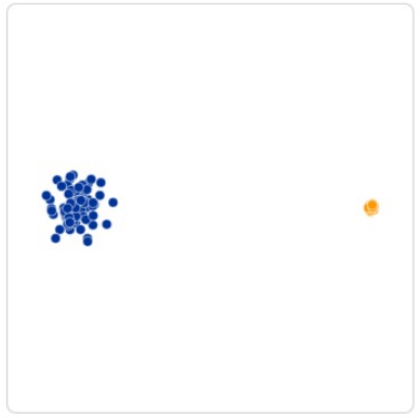
Perplexity: 30
Step: 5,000

Perplexity: 50
Step: 5,000

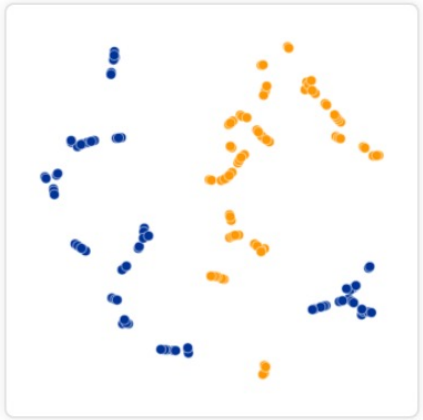


Perplexity: 100
Step: 5,000

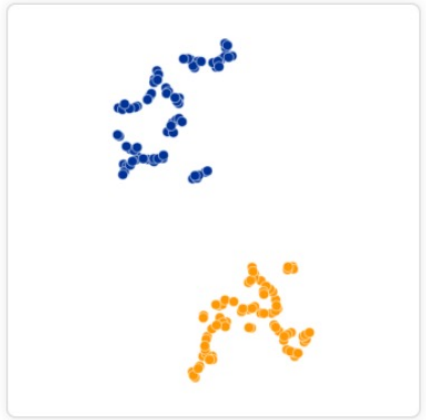
Cluster sizes mean nothing



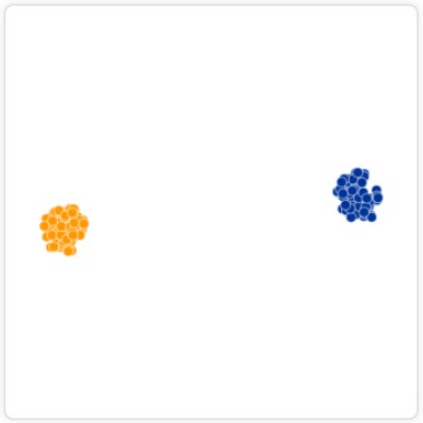
Original



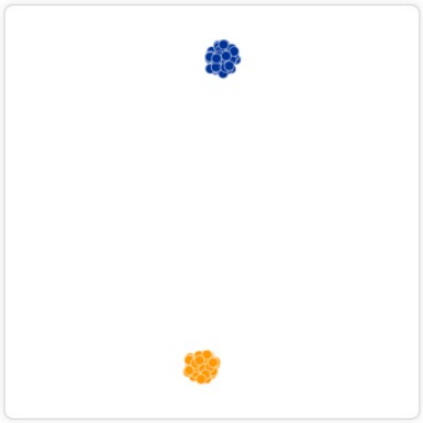
Perplexity: 2
Step: 5,000



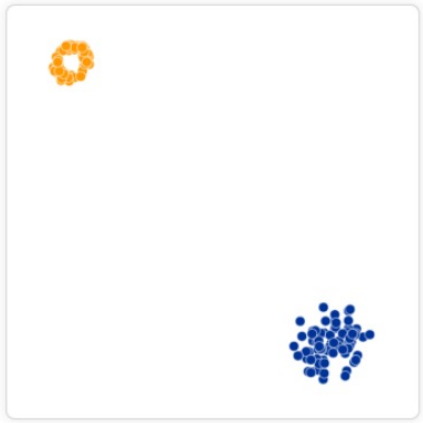
Perplexity: 5
Step: 5,000



Perplexity: 30
Step: 5,000

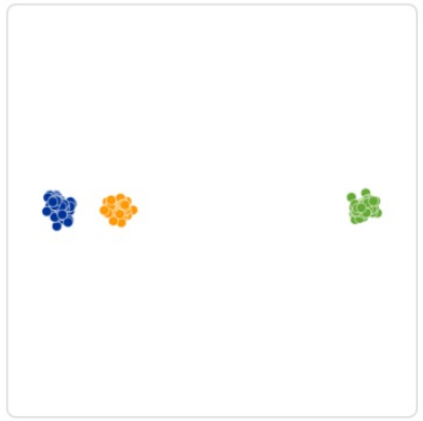


Perplexity: 50
Step: 5,000

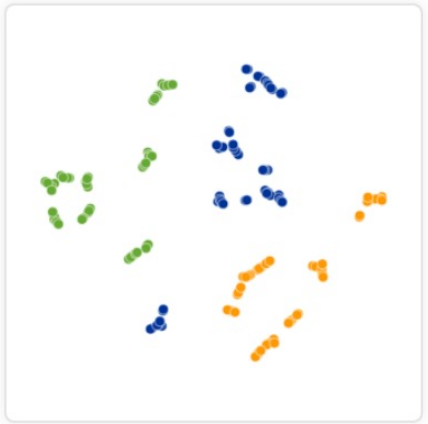


Perplexity: 100
Step: 5,000

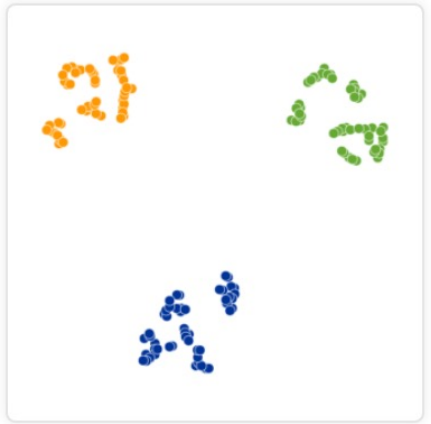
Distances between clusters might not mean anything



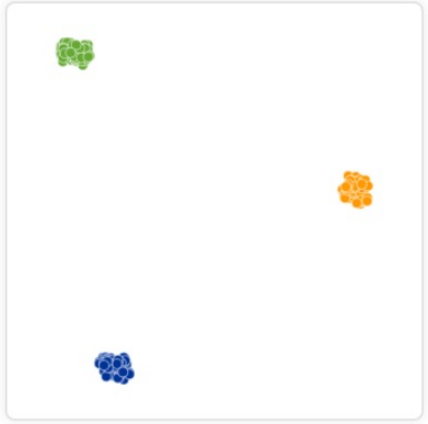
Original



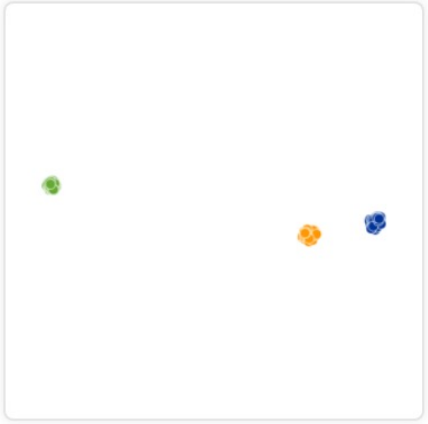
Perplexity: 2
Step: 5,000



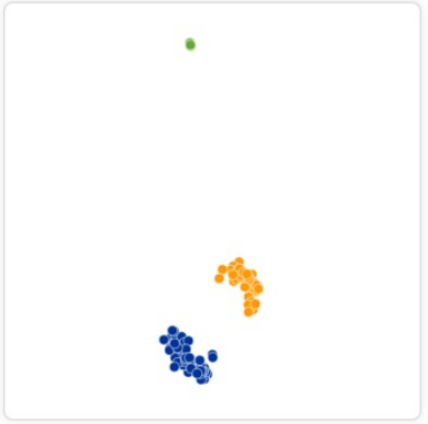
Perplexity: 5
Step: 5,000



Perplexity: 30
Step: 5,000

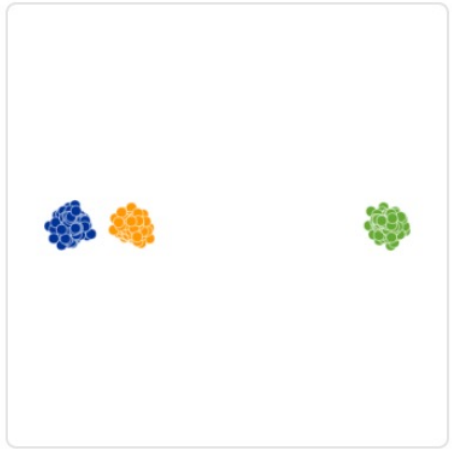


Perplexity: 50
Step: 5,000

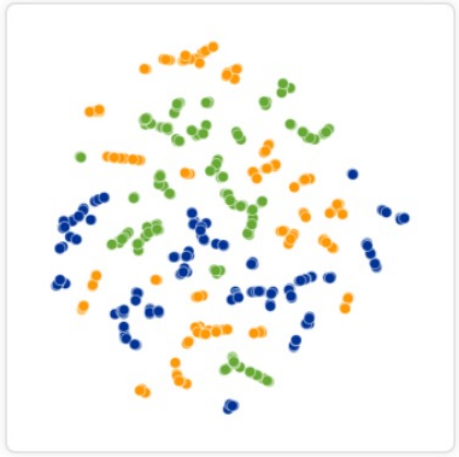


Perplexity: 100
Step: 5,000

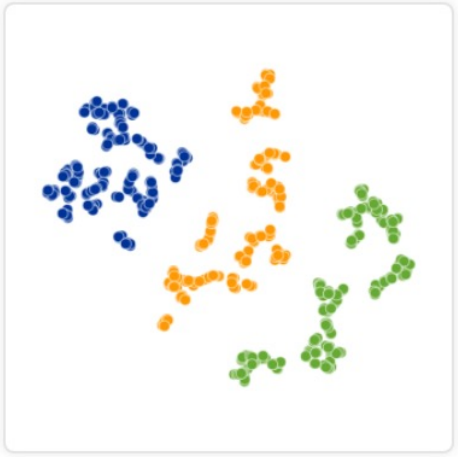
Distances between clusters might not mean anything



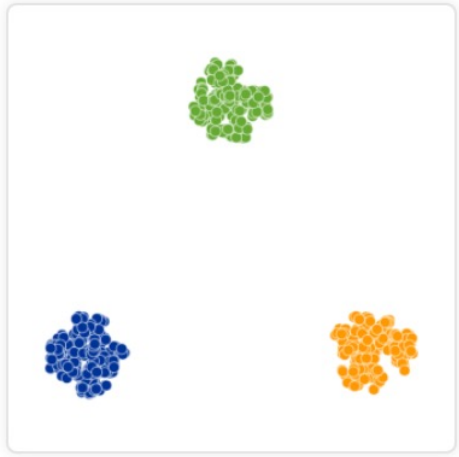
Original



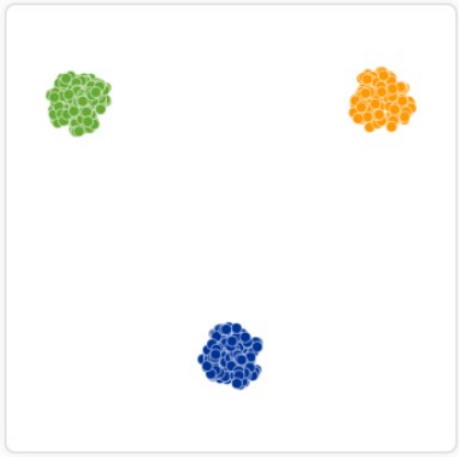
Perplexity: 2
Step: 5,000



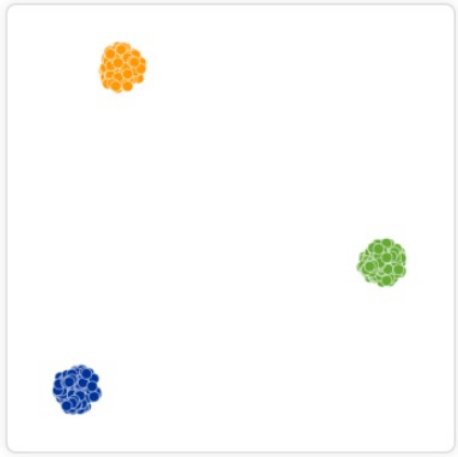
Perplexity: 5
Step: 5,000



Perplexity: 30
Step: 5,000

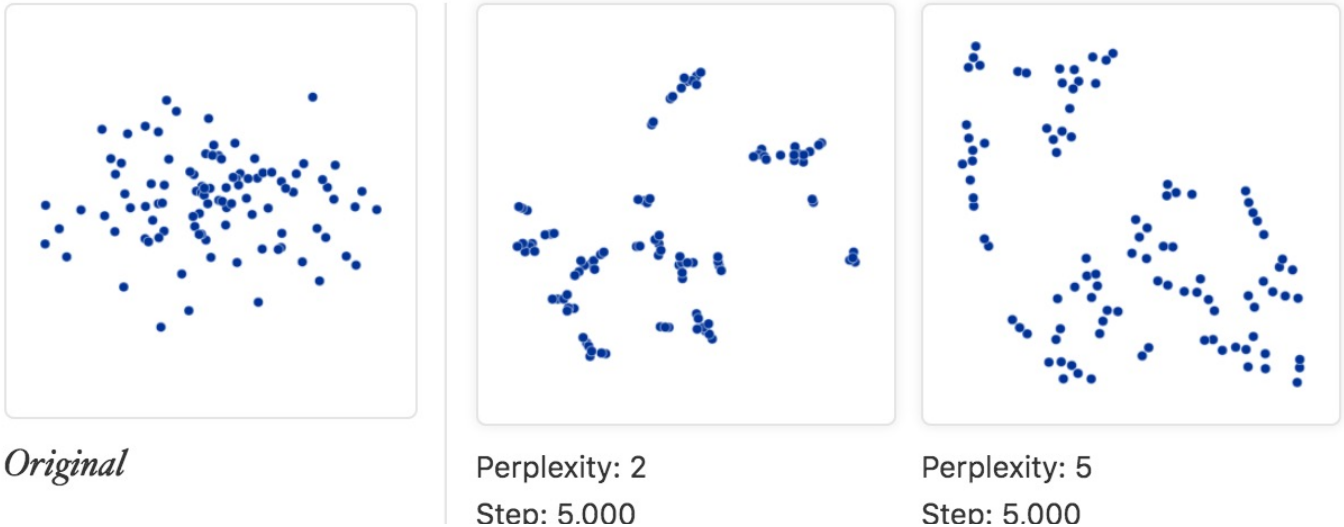


Perplexity: 50
Step: 5,000



Perplexity: 100
Step: 5,000

You can see some shapes, sometimes



Original

Perplexity: 2
Step: 5,000

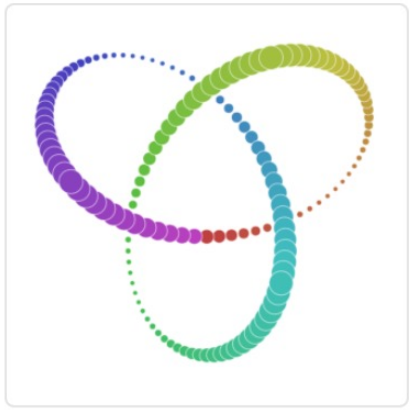
Perplexity: 5
Step: 5,000

Perplexity: 30
Step: 5,000

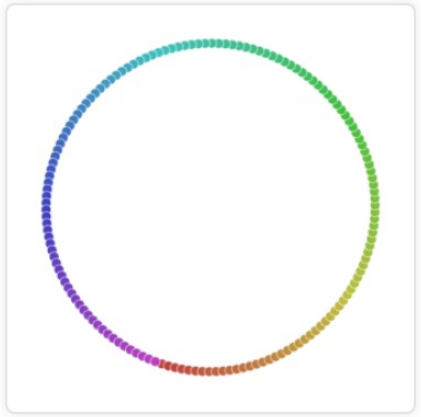
Perplexity: 50
Step: 5,000

Perplexity: 100
Step: 5,000

Topology may or may not be preserved

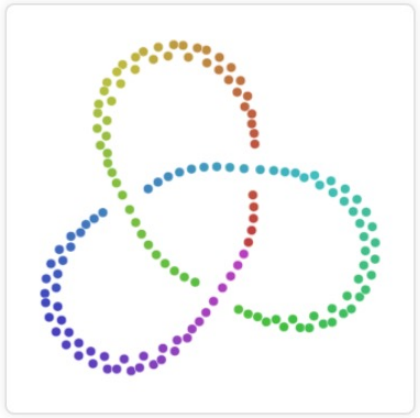


Original

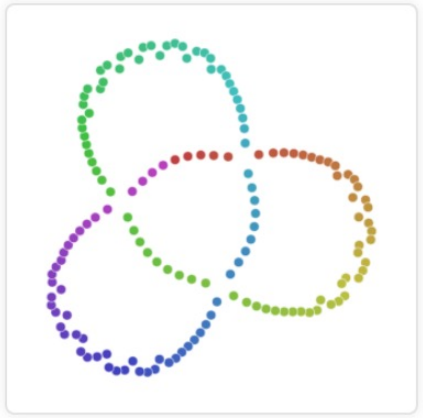


Perplexity: 2
Step: 5,000

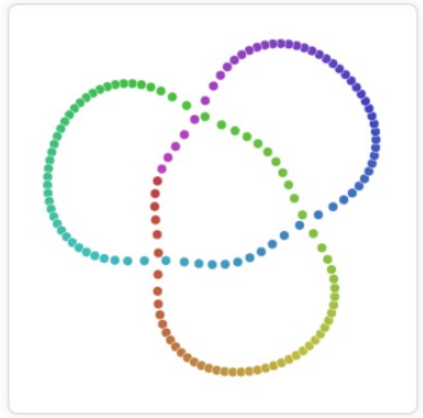
Perplexity: 5
Step: 5,000



Perplexity: 30
Step: 5,000



Perplexity: 50
Step: 5,000

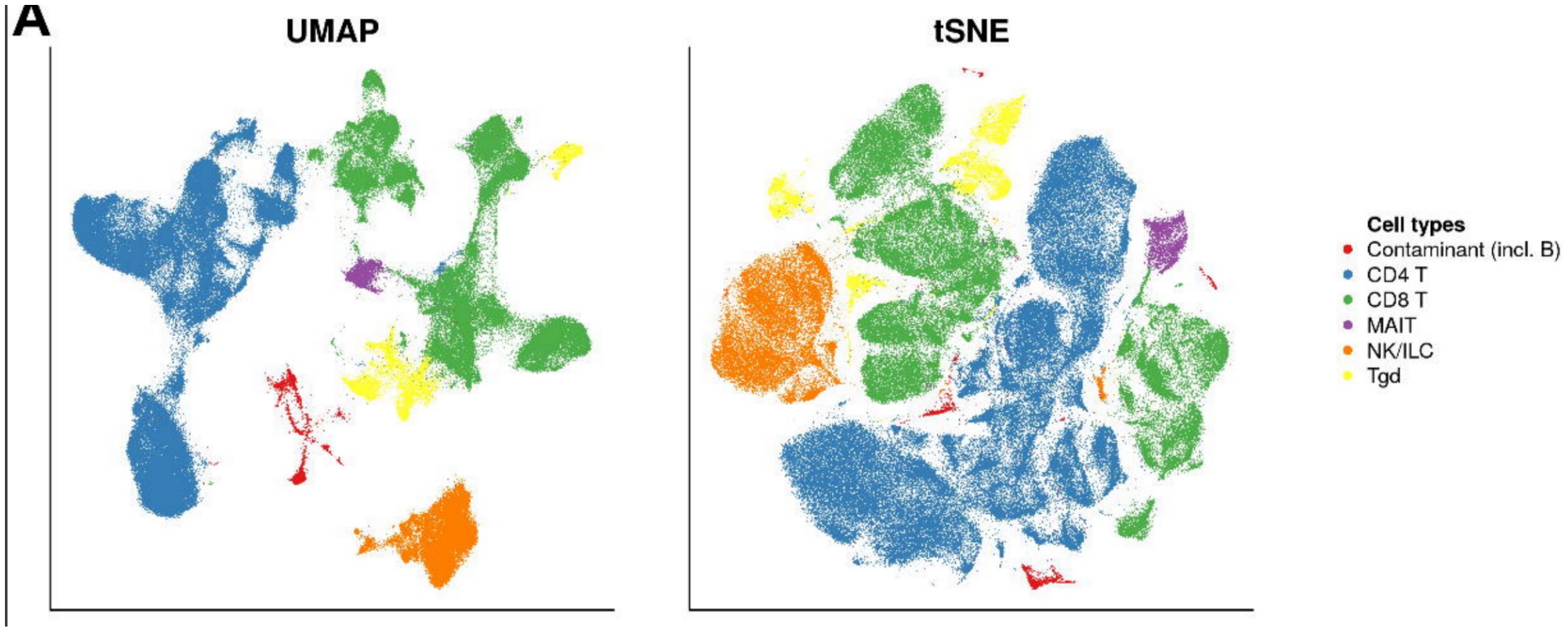


Perplexity: 100
Step: 5,000

Takeaways

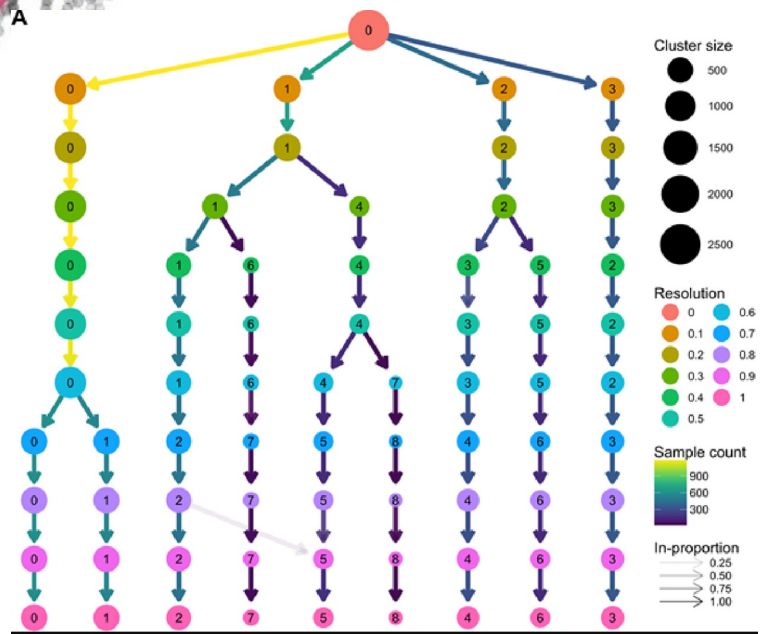
- t-SNE plots are an imperfect representation of the data
- They may not even be the best imperfect representation of the data, depending on your application

UMAP



https://www.biorxiv.org/content/early/2018/04/10/298430

Clustering Trees



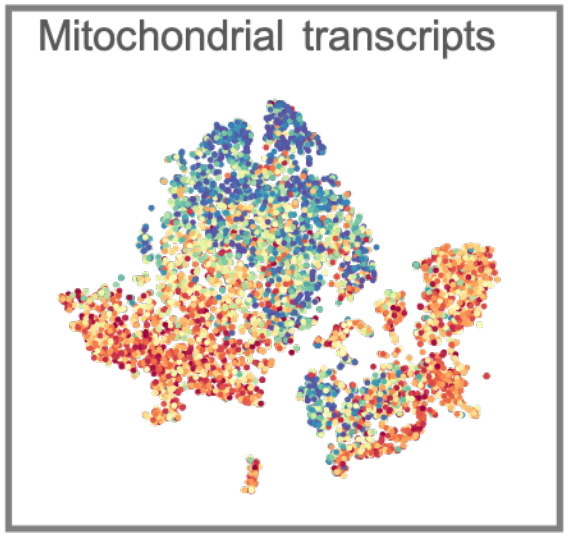
http://dx.doi.org/10.1093/gigascience/giy083

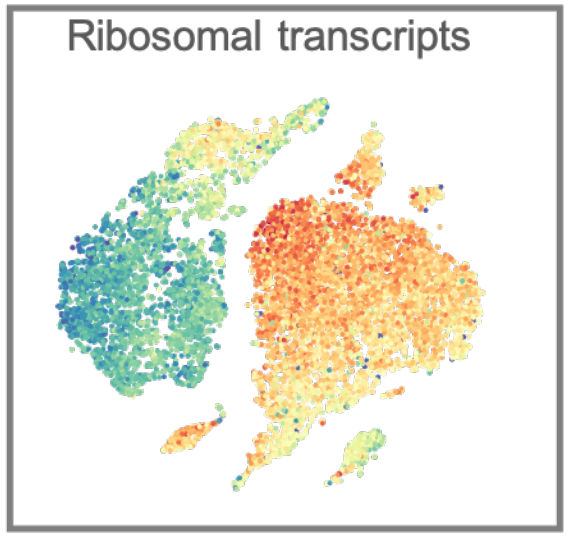
Plotting using t-SNE/UMAP

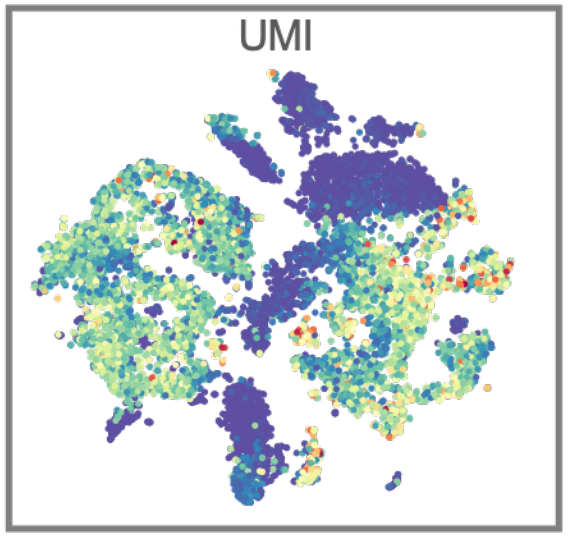
```
n = 10; # choose number of dimensions - experiment-specific!
scrna <- RunUMAP(object = scrna, reduction = "pca", dims = 1:n) # calculate
UMAP
scrna <- RunTSNE(object = scrna, reduction = "pca", dims = 1:n) # calculate
tSNE

# make some plots:
DimPlot(object = scrna, reduction = "tsne", group.by = ”Batch", pt.size=0.1) #
color by Batch
FeaturePlot(object = scrna, features = c("nCount_RNA"), reduction=“tsne”) #
plot one or more genes or variable on t-SNE
```

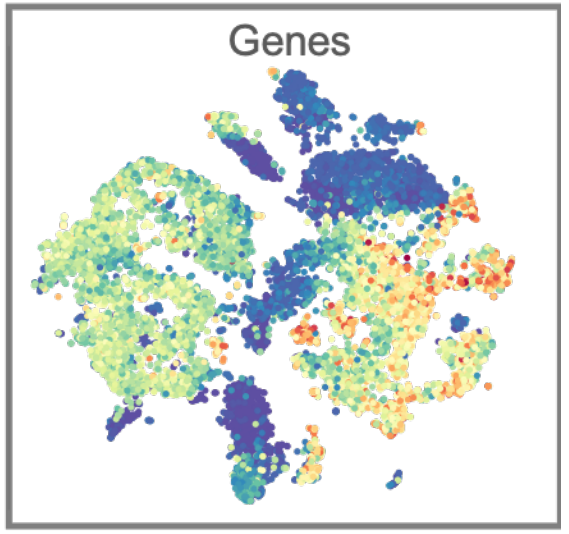
“How to use t-SNE effectively” https://distill.pub/2016/misread-tsne/

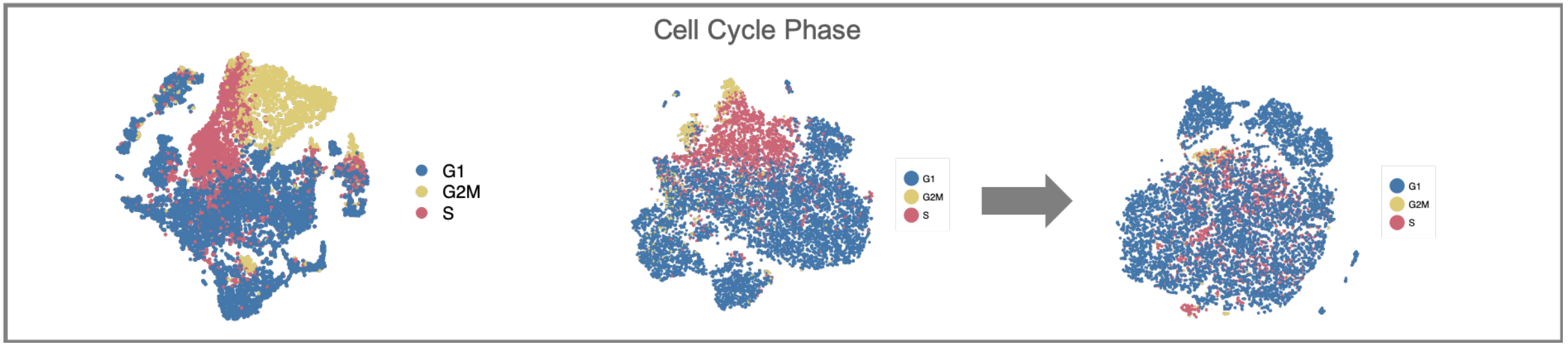
Interpreting the t-SNE/UMAP, Part I: Potentially misleading sources of variation





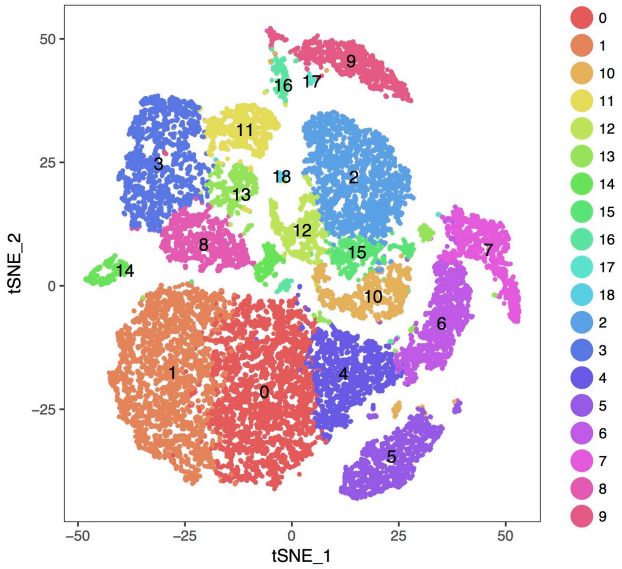


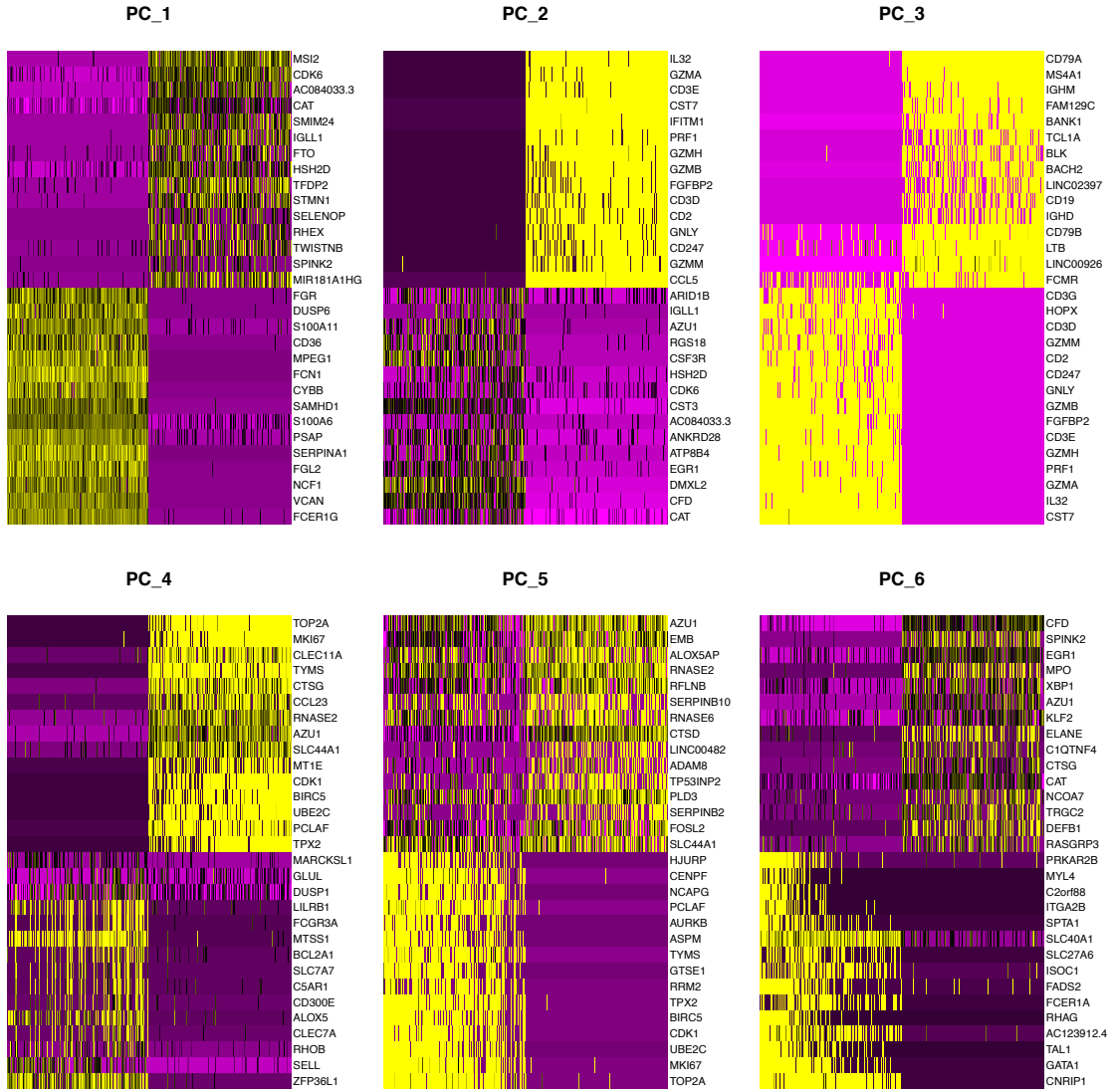




Interpreting the t-SNE/UMAP, Part II: Systematic analysis of variation

- What is driving the t-SNE/UMAP layout?
- Find genes that vary:
  - Principal components
  - Individual cluster-specific genes
- Examine across clusters/t-SNE/UMAP



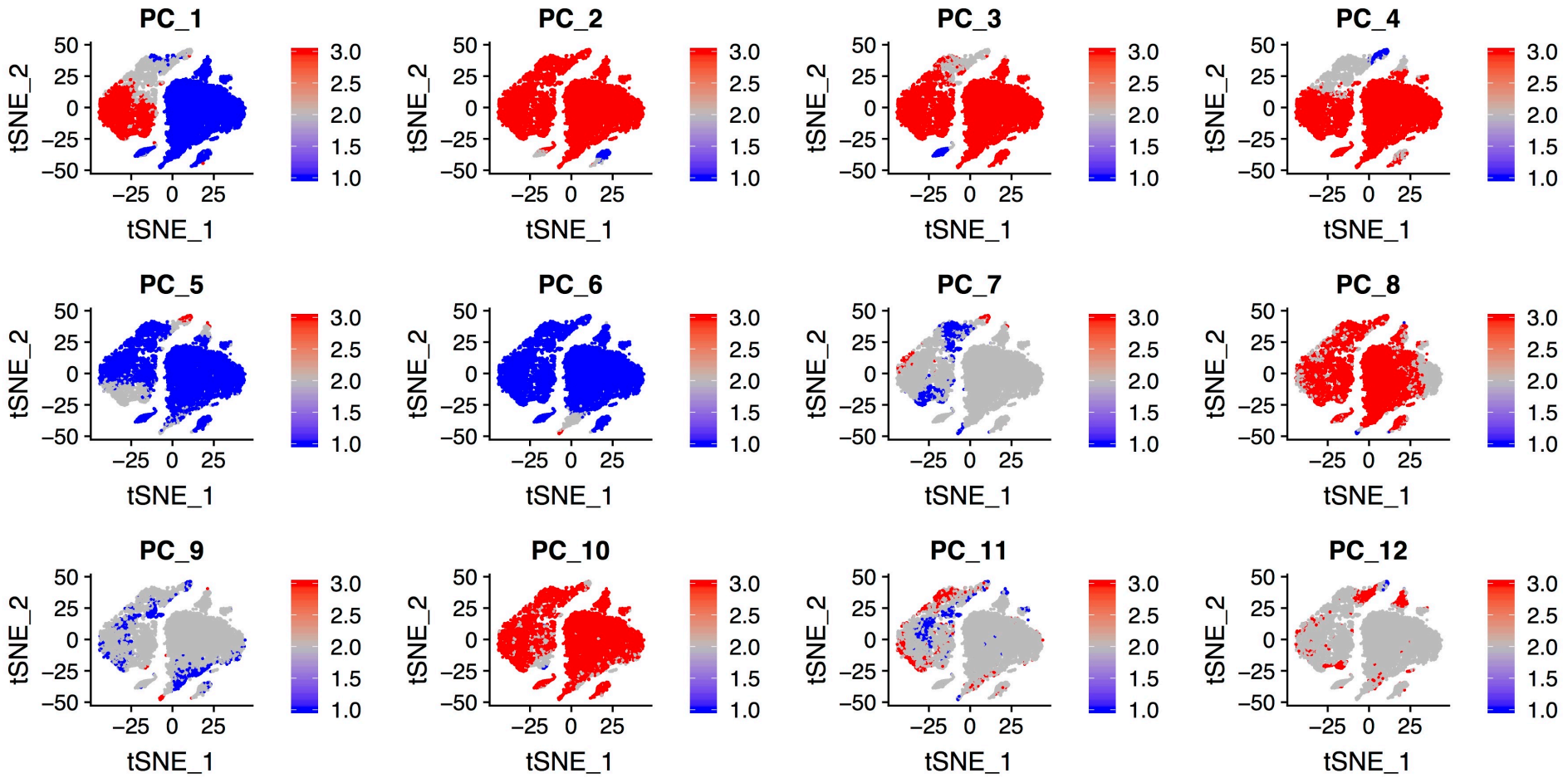



```
DimHeatmap(object = scrna, dims = 1, cells = 500, balanced = TRUE)
```

Part II, cont'd: Visualizing sources of variation

Small but distinct cell clusters may contribute heavily to the layout (may need to be removed)

PC #1 captures the biggest source of variation

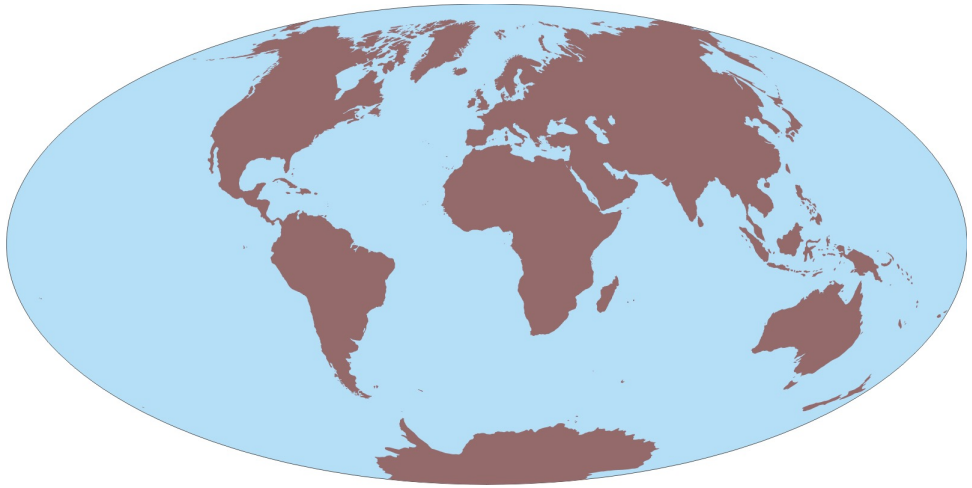


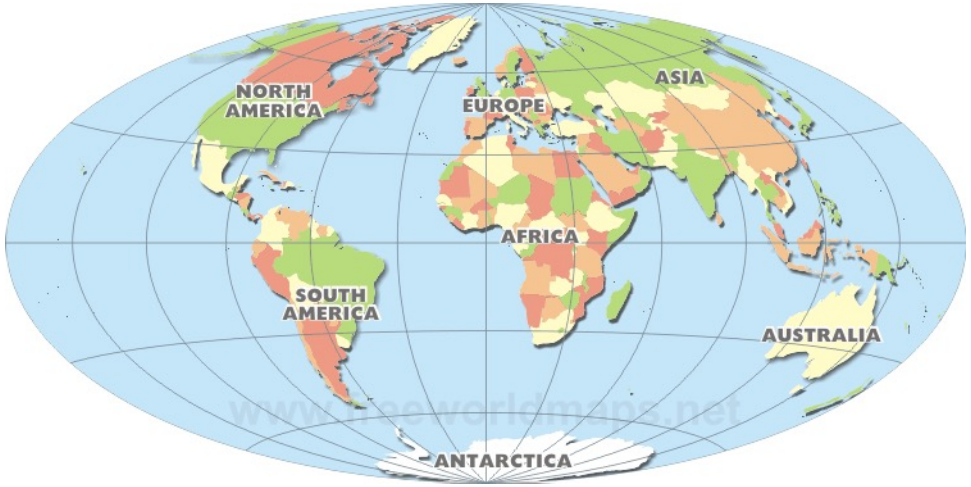
Post-PCA analysis of the gene x cell matrix

- Compute t-SNE and UMAP layouts on *n* Principal Components (NB: not raw data)
- **Clustering**
  - A tool for finding patterns in the data
  - Graph-based (unsupervised, must specify resolution (0.7))
  - Alternative: k-means (supervised, must specify k)
- Characterizing Clusters in terms of individual genes
  - Differentially expressed genes (numerous methods)
  - PC-perspective
    - Choose genes that contribute heavily to top principal components
    - Plot heatmaps of these genes in each cluster
    - Independent of clustering
    - Shows relationships of clusters to each other
- Cell lineage inference

2-D layout vs. Clustering

- tSNE and UMAP reflects natural organization of data by approximating high-dimensional relationships in low-dimensional space
- Clustering imposes structure by assigning cells to non-overlapping groups based on relative expression similarity

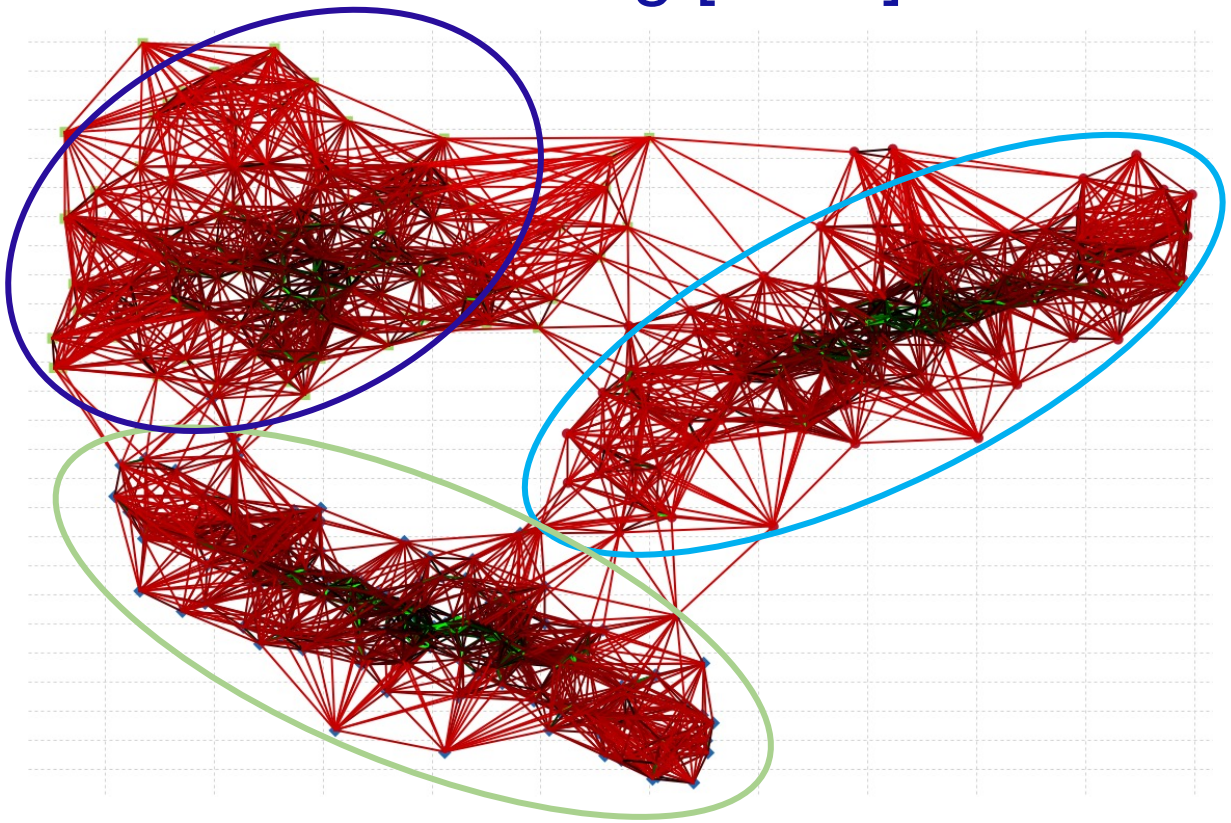




Clustering [Cells]

Step 1: Build KNN graph using Euclidian distance in PCA-space

Step 2: Find clusters, or cliques, or modules using Louvain modularity optimization algorithm (alternatives available)



```
nPC = 20; # specify number of PCs to use
cluster.res = 0.7; # specify resolution of clustering.
scrna <- FindNeighbors(object=scrna, dims=1:nPC);
scrna <- FindClusters(object=scrna, resolution=cluster.res);
scrna <- StashIdent(object = scrna, save.name = sprintf("ClusterNames_%.1f_%dPC",
cluster.res, nPC)) # save cluster names in a new identity (ClusterNames_0.7_20 in this
example) if desired
```

Clustering [Cells]

- Clustering helps organize and identify patterns in data.
- There is no “correct” or “perfect” clustering of any data set.
- Corollary: Even the best clustering may be misleading (aka “wrong”).
- Don’t take clusters too seriously – they don’t prove anything.

Common methods:

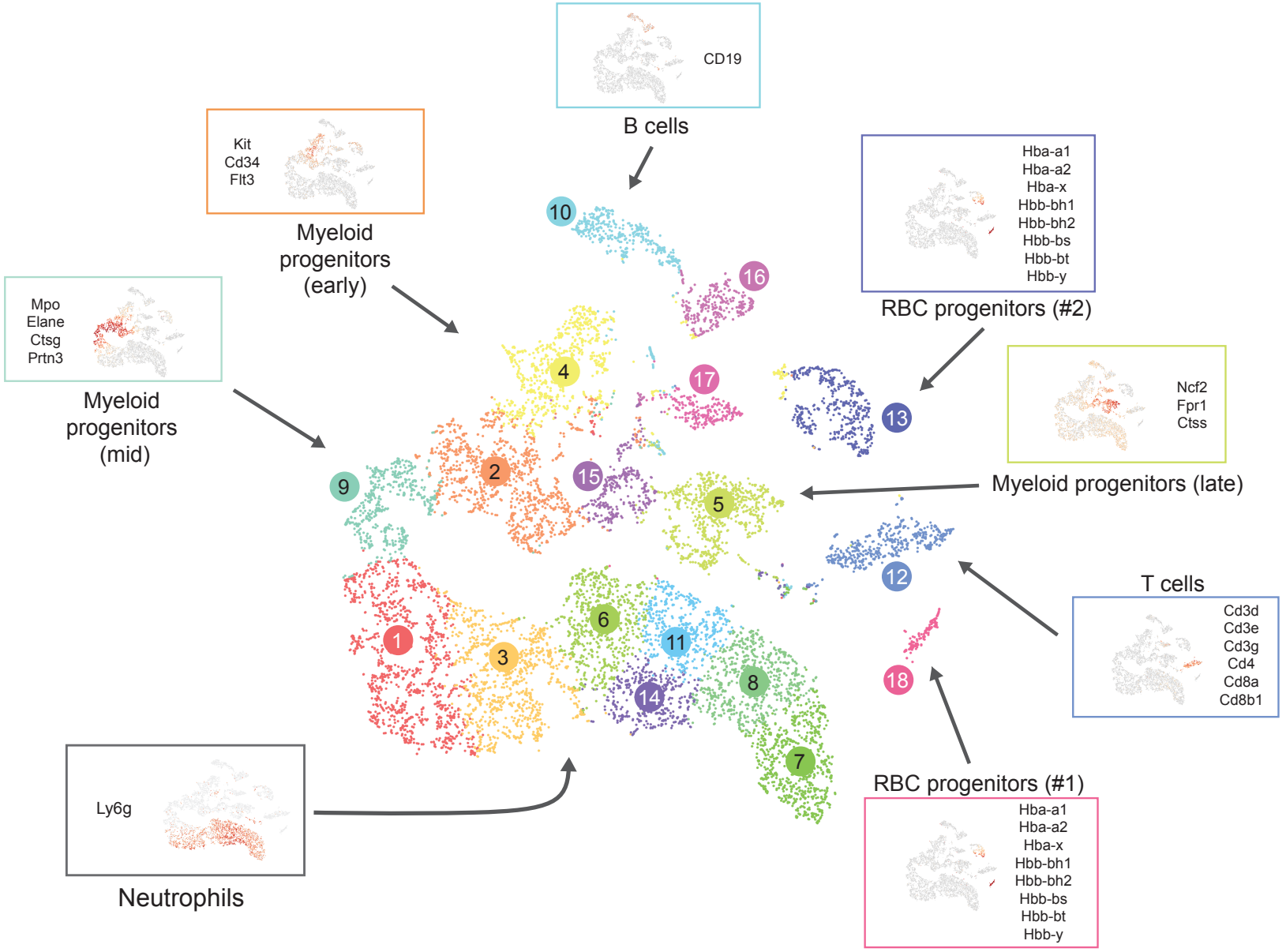
- Graph-based: Unsupervised, but “adjustable” using ”resolution” parameter
  - Calculates k-nearest neighbors for each cell using Euclidian distance in PCA-space; constructs shared nearest-neighbor (SNN) graph; optimize graph modularity (Waltman and van Eck algorithm)
- K-means: Supervised: specify number of clusters (available in cellranger, not Seurat).

Seurat exercise, continued

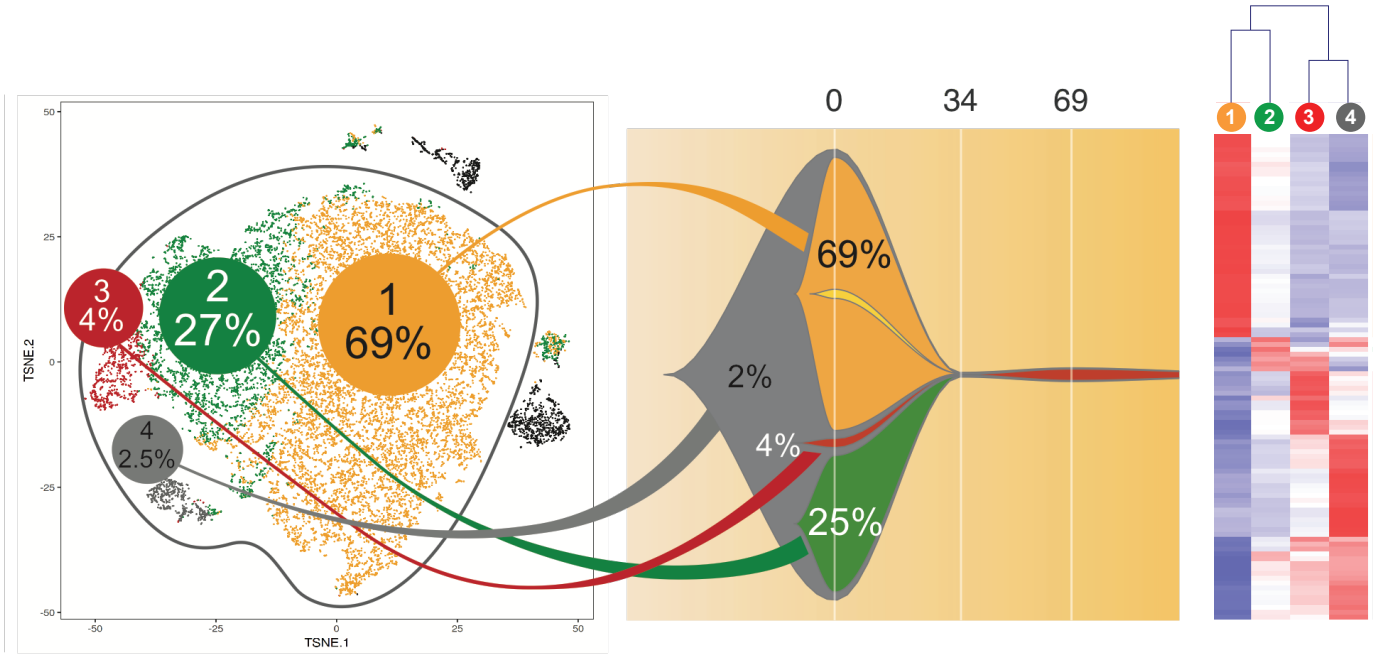
Once you run Step 13, aka the FindClusters step, please click your ‘tada’ emoji.

Characterizing clusters using marker genes

Cells colored by graph-based cluster

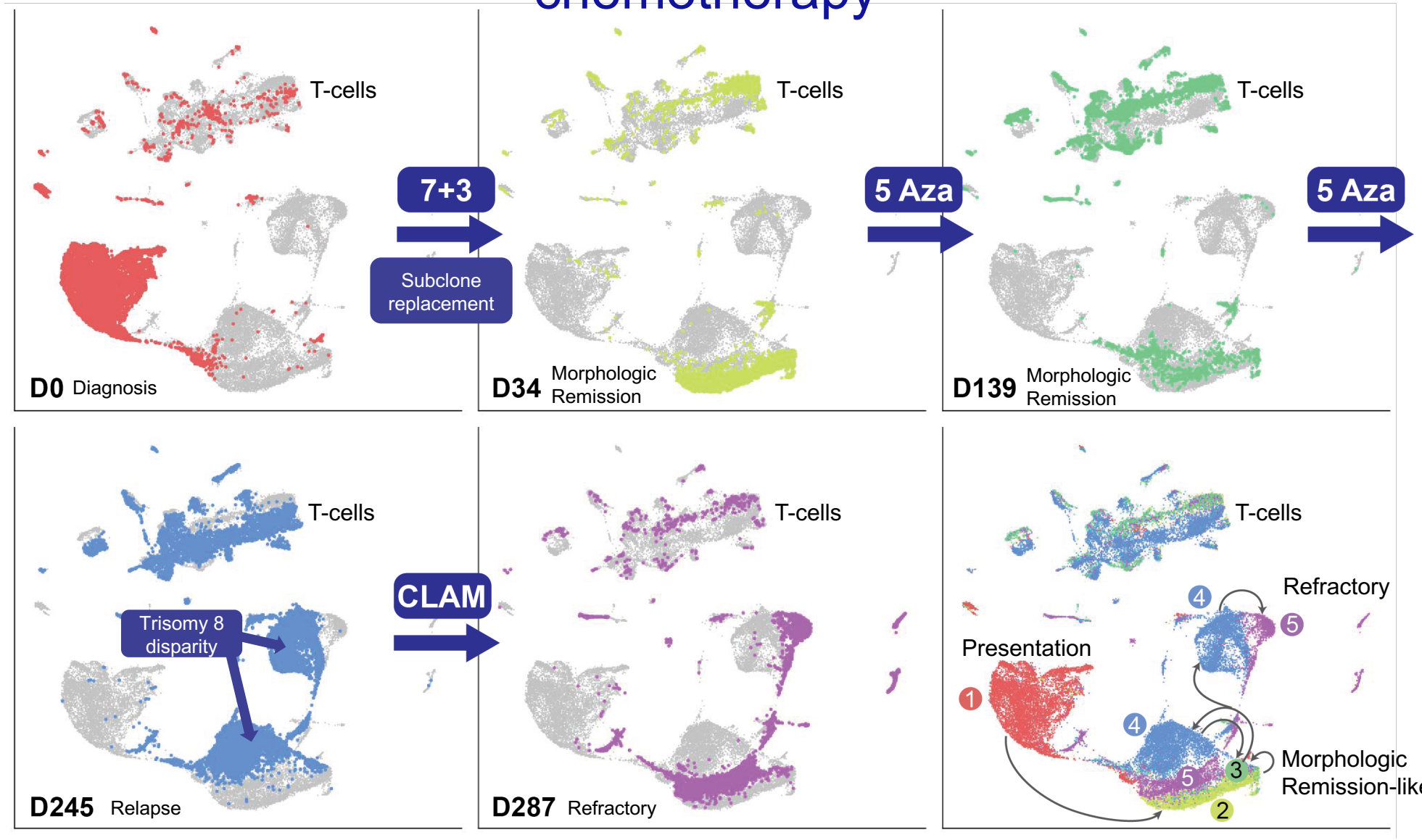


Characterizing clusters using differential gene expression

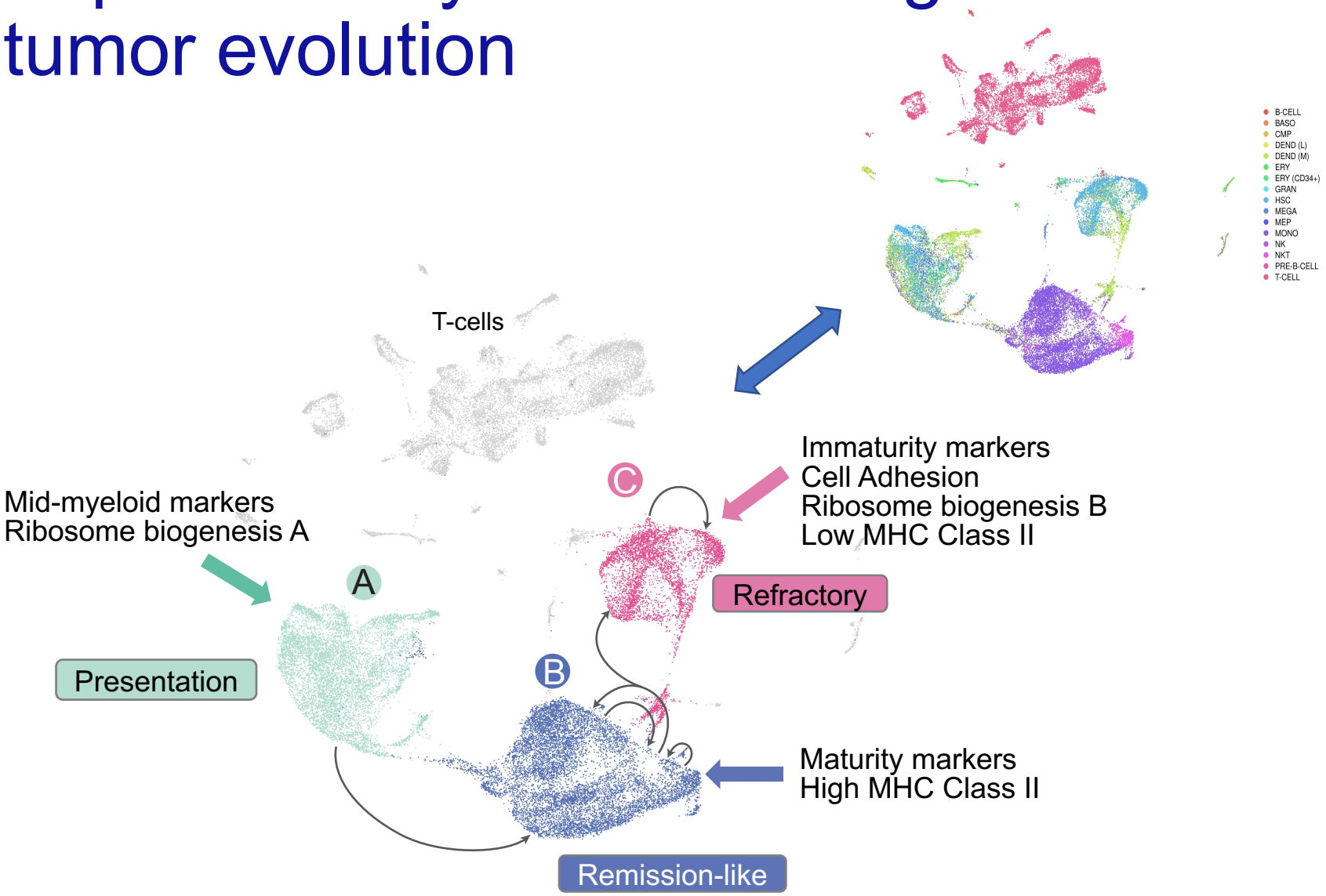


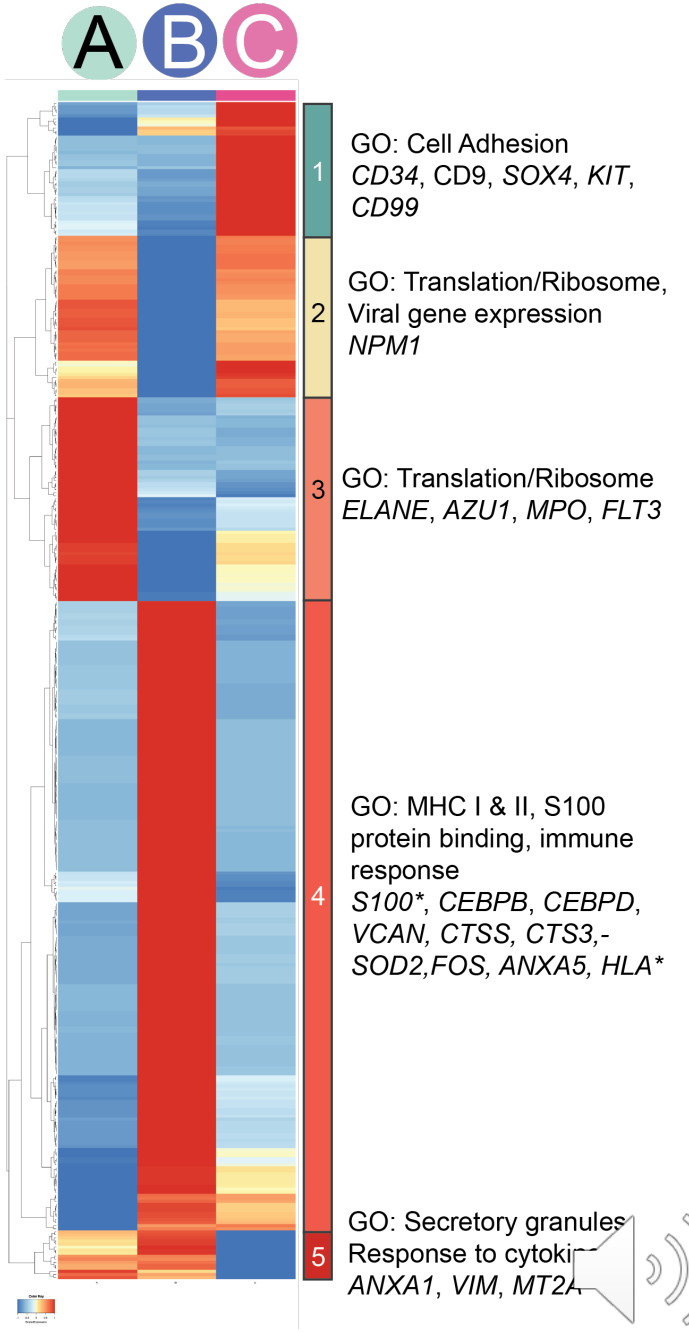
- Data has zero-inflated negative binomial distribution (lots of zeros, overdispersed) so can't use bulk methods
- Default in Seurat: Wilcoxon rank-sum test
- Nonparametric version of t-test
- For two clusters (A and B), and one gene, rank each cell in each cluster according to expression
- Determine whether sum-of-ranks for cluster A is significantly different than sum-of-ranks for cluster B
- Clear explanation of Wilcoxon rank-sum test: http://statweb.stanford.edu/~susan/courses/s141/hononpara.pdf
- Numerous other tests in Seurat and other packages

Coordinated evolution of the tumor and microenvironment during chemotherapy



Expression dynamics during tumor evolution

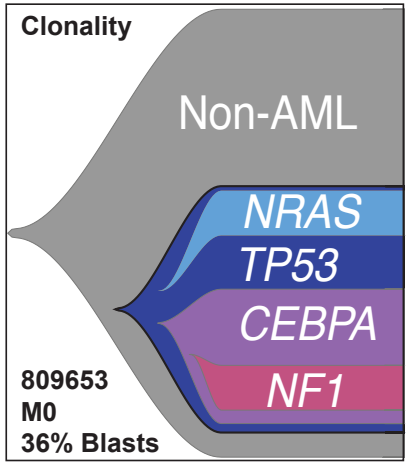


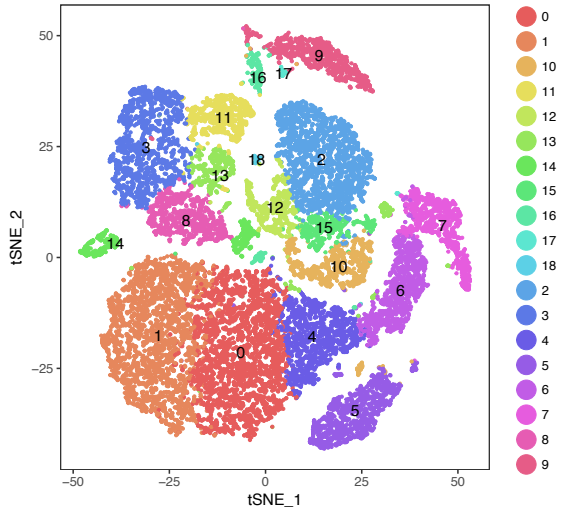


Characterizing clusters using differential gene expression

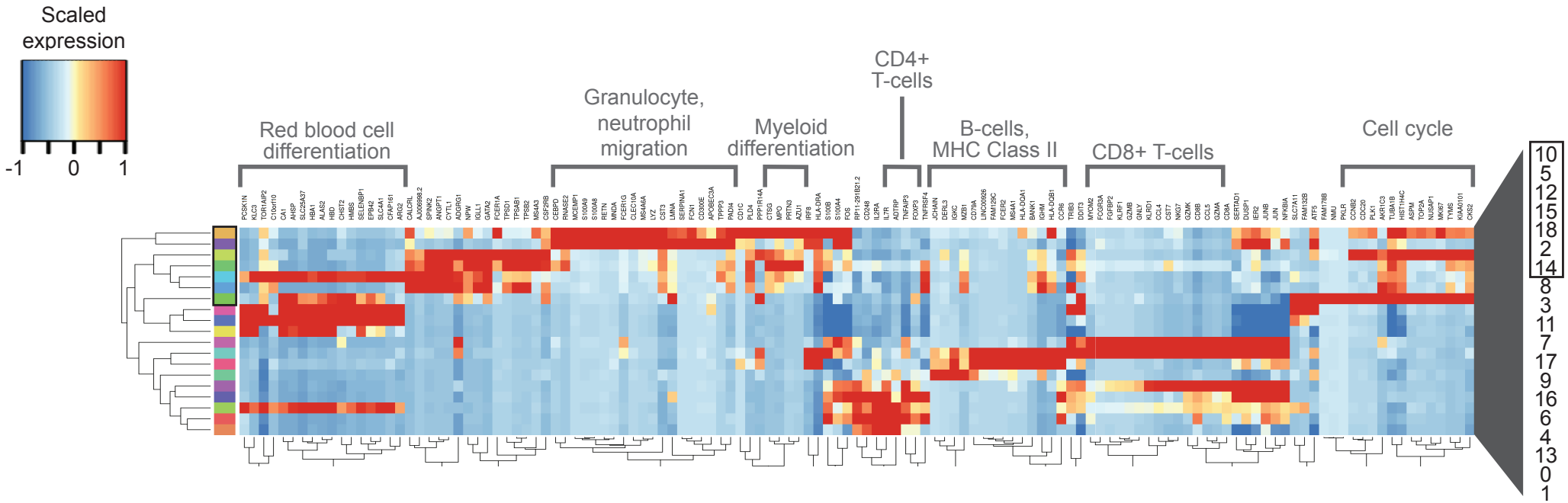
```
DEGs <- FindAllMarkers(object=scrna); # Compare each clusters to all
other cells. output is a matrix.
de.markers <- FindMarkers(scrna, ident.1 = ”1", ident.2 = ”2") #
compare identity 1 to identity 2
# NB: May first need to set default identities, e.g.:
Idents(object = scrna) <- ”seurat_clusters”; # sets the default
identity to Seurat_clusters

# Do the DEGs make sense? Plot them
FeaturePlot(object = scrna, features = ‘CD34’)
# Prettier version:
FeaturePlot(object = scrna, features = genesToPlot, cols =
c("gray","red"), ncol=2, reduction = "umap") +
theme(axis.title.x=element_blank(),axis.title.y=element_blank(),axis.
text.x=element_blank(),axis.text.y=element_blank(),axis.ticks.x=eleme
nt_blank(),axis.ticks.y=element_blank())
```

Characterizing clusters using principal components

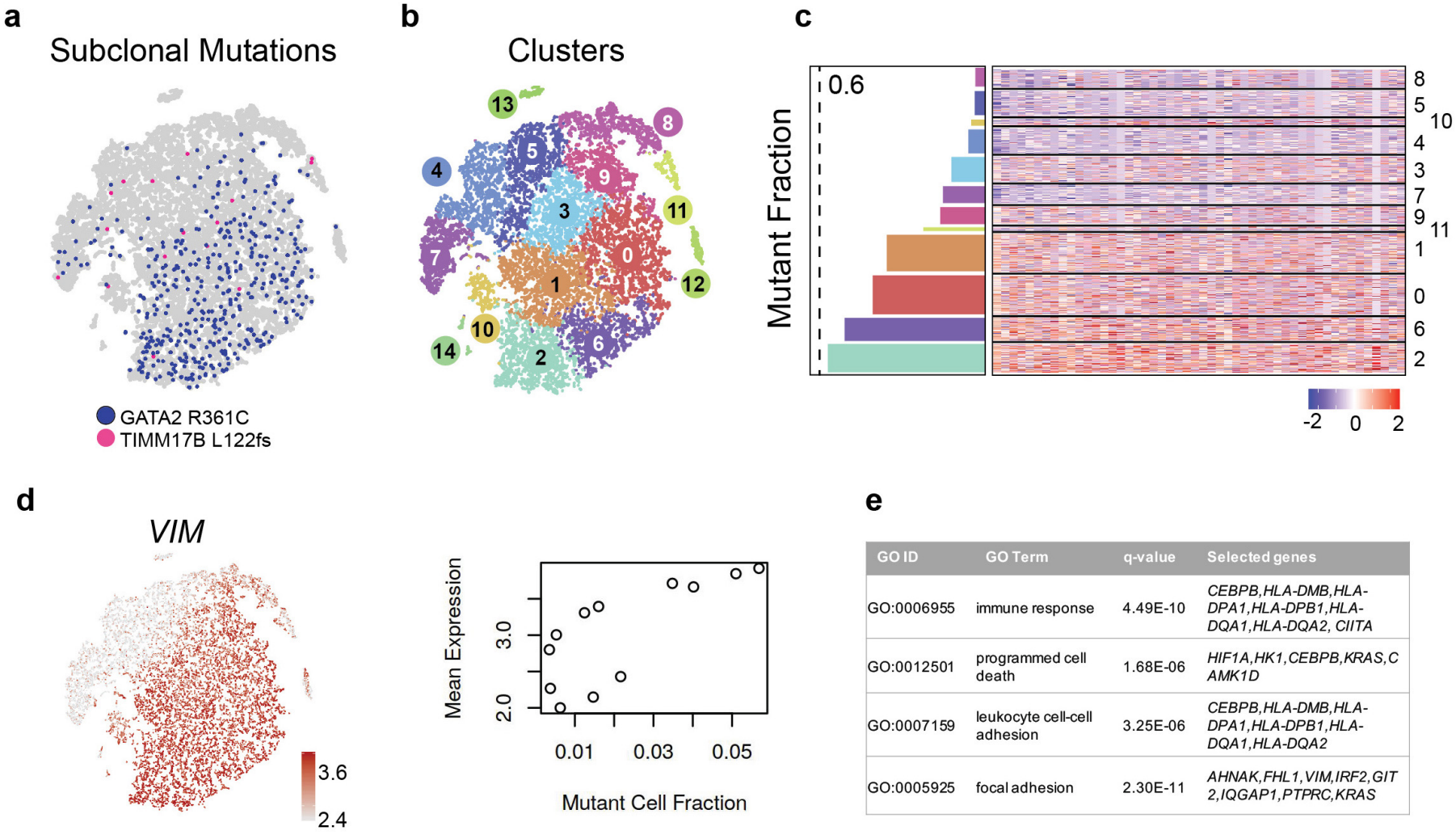




- Cluster-specific DEGs are problematic: require “perfect” clusters, oversimplify, can miss signal, too stringent for the data.
- Instead, visualize principal components across clusters to view relationships among clusters
- Choose most highly-weighted genes in the top principal components
- Calculate average expression of each gene in each cluster
- Use hierarchical clustering to cluster the clusters based on these genes



Characterizing clusters using regression



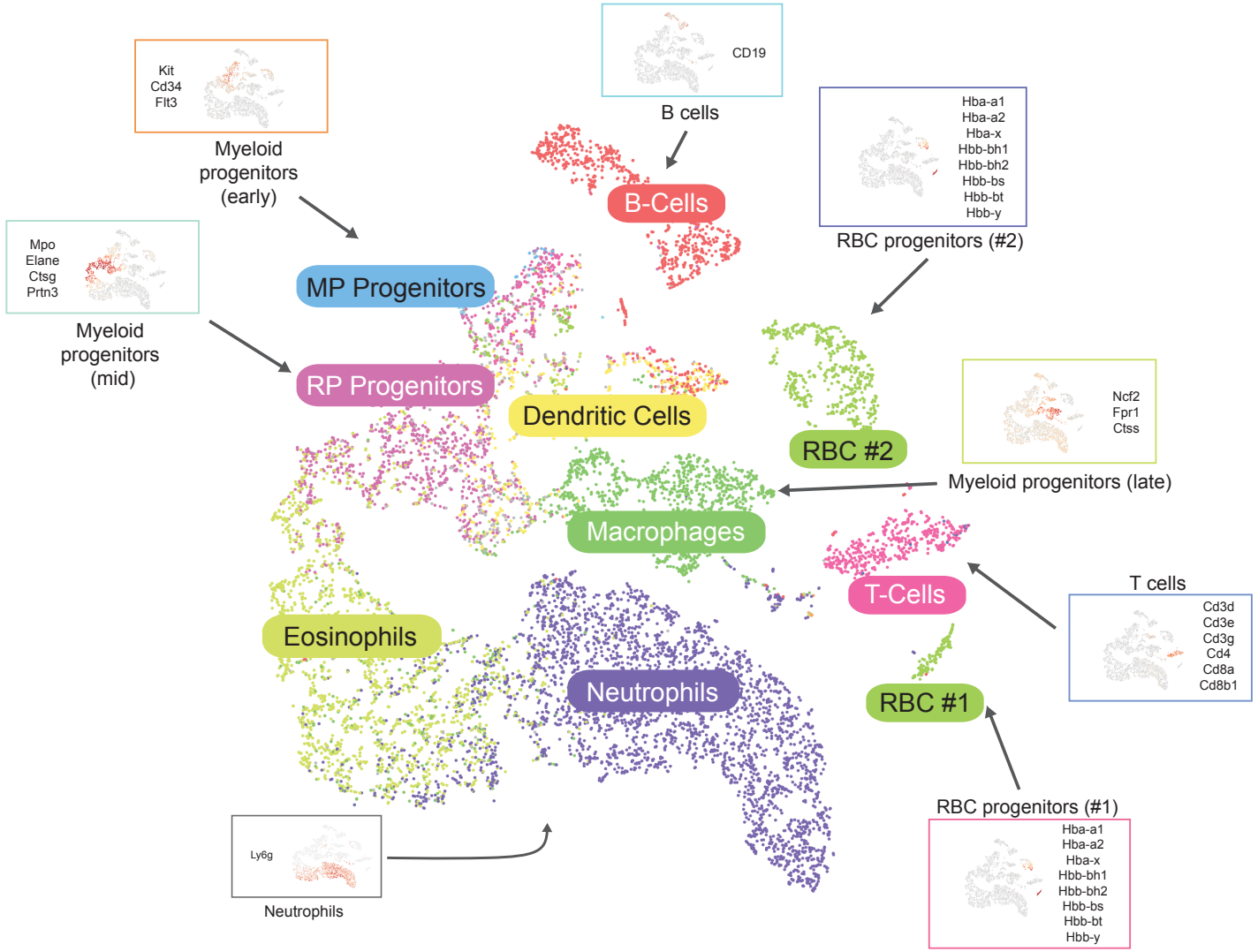
e

GO ID	GO Term	q-value	Selected genes
GO:0006955	immune response	4.49E-10	*CEBPB,HLA-DMB,HLA-DPA1,HLA-DPB1,HLA-DQA1,HLA-DQA2, CIITA*
GO:0012501	programmed cell death	1.68E-06	*HIF1A,HK1,CEBPB,KRAS,CAMK1D*
GO:0007159	leukocyte cell-cell adhesion	3.25E-06	*CEBPB,HLA-DMB,HLA-DPA1,HLA-DPB1,HLA-DQA1,HLA-DQA2*
GO:0005925	focal adhesion	2.30E-11	*AHNAK,FHL1,VIM,IRF2,GIT2,IQGAP1,PTPRC,KRAS*

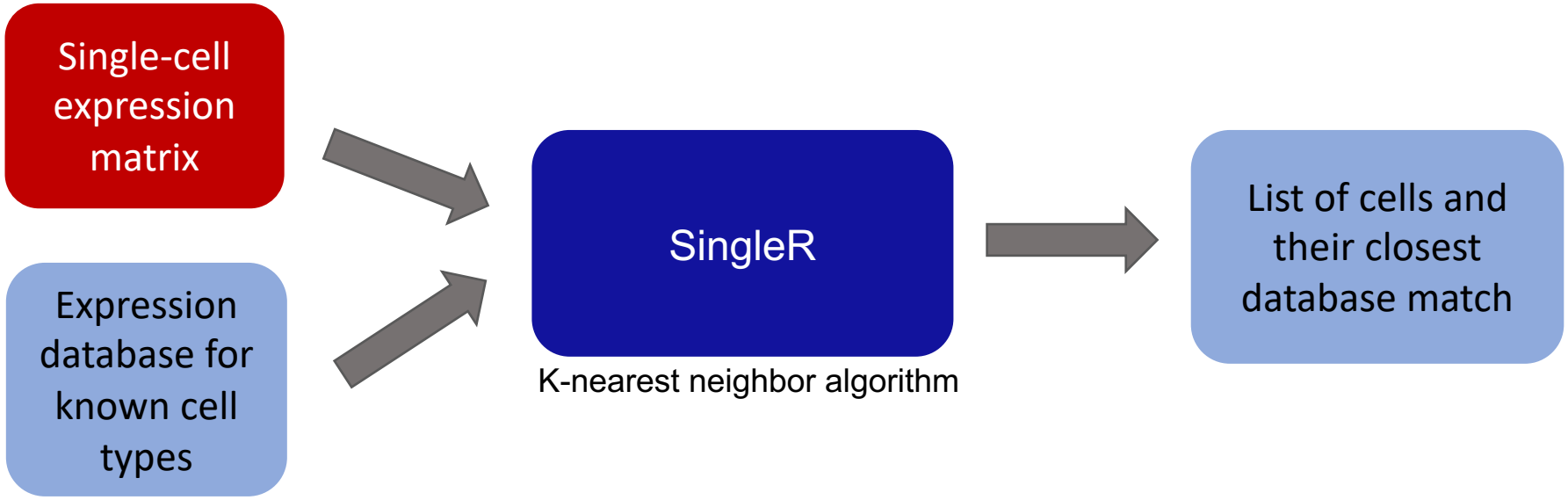
Other questions to consider when comparing two samples/clusters

- Pairwise differential gene expression
- What fraction of cells in each sample express a given gene?
- Of the cells in each sample that express a given gene, does the mean expression in those cells differ?
- Does the distribution of cell types differ between samples?
- Do the samples exhibit cell-type-specific differential gene expression?

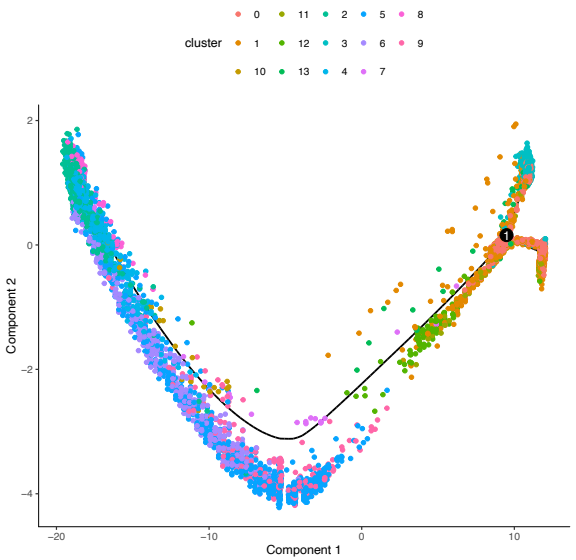
Assigning cell type



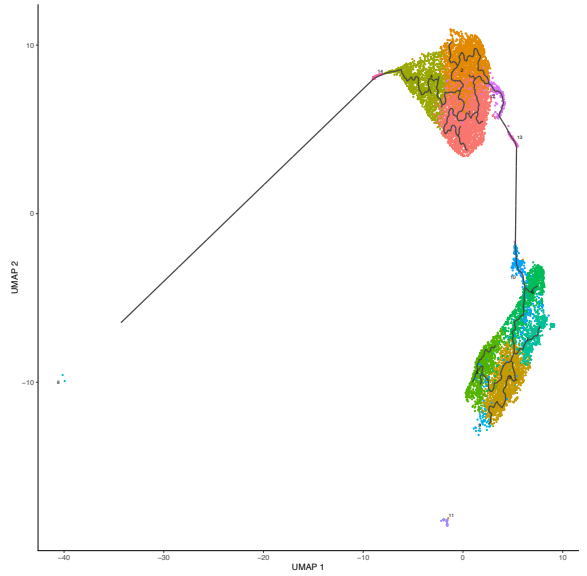
SingleR: Marker-free, Cluster-free Cell Lineage Inference



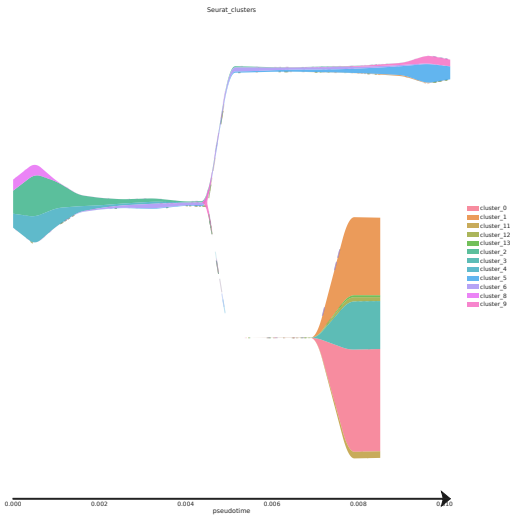
Characterizing gene expression changes with respect to pseudotime



Monocle 2



Monocle 3
(Slingshot is similar)

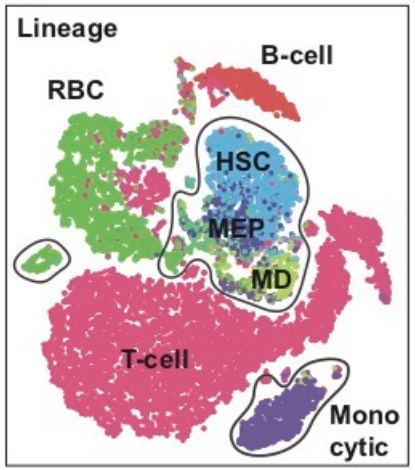


STREAM

Best to benchmark these methods with data from cells with known developmental relationships, e.g. hematopoiesis

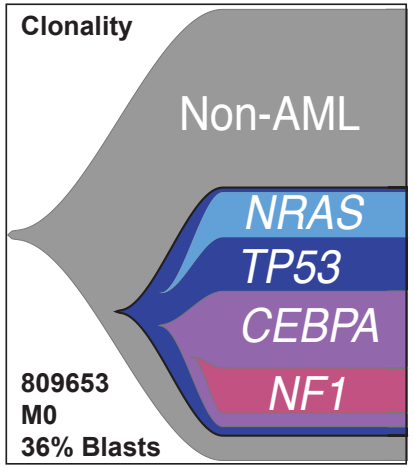
Mutation mapping enables us to answer two key questions:

1. Which cells are tumor cells?
2. Where are the subclones?



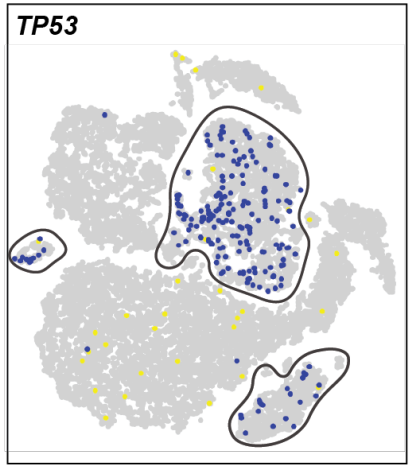
eWGS sequencing

Founding clone
TP53 mutation



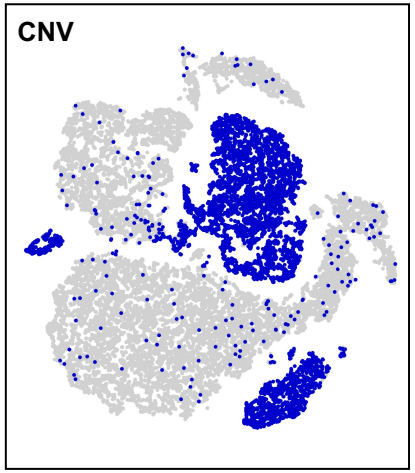
Single-cell mutation mapping: cb_sniffer or Vartrix

Mutant cells mark AML clusters



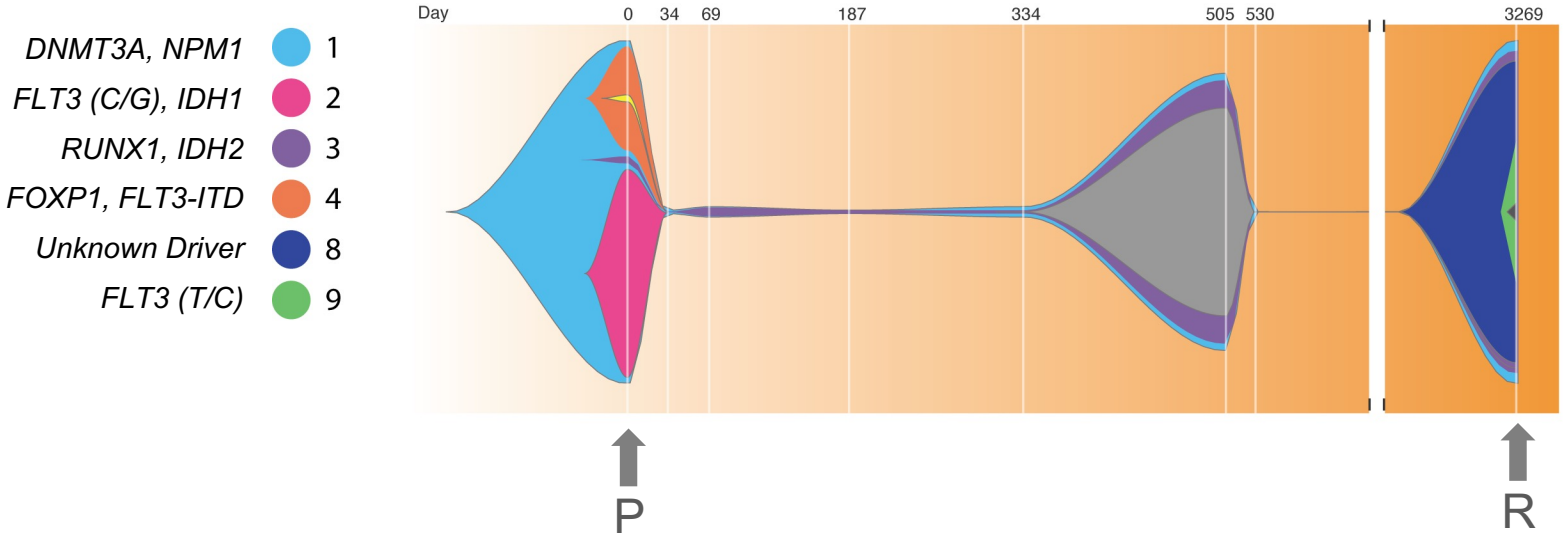
Mutant (E286G)
Unknown genotype
No coverage

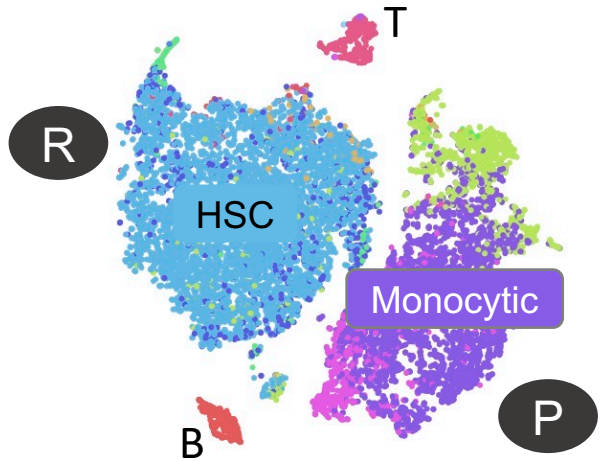
CNV concordance



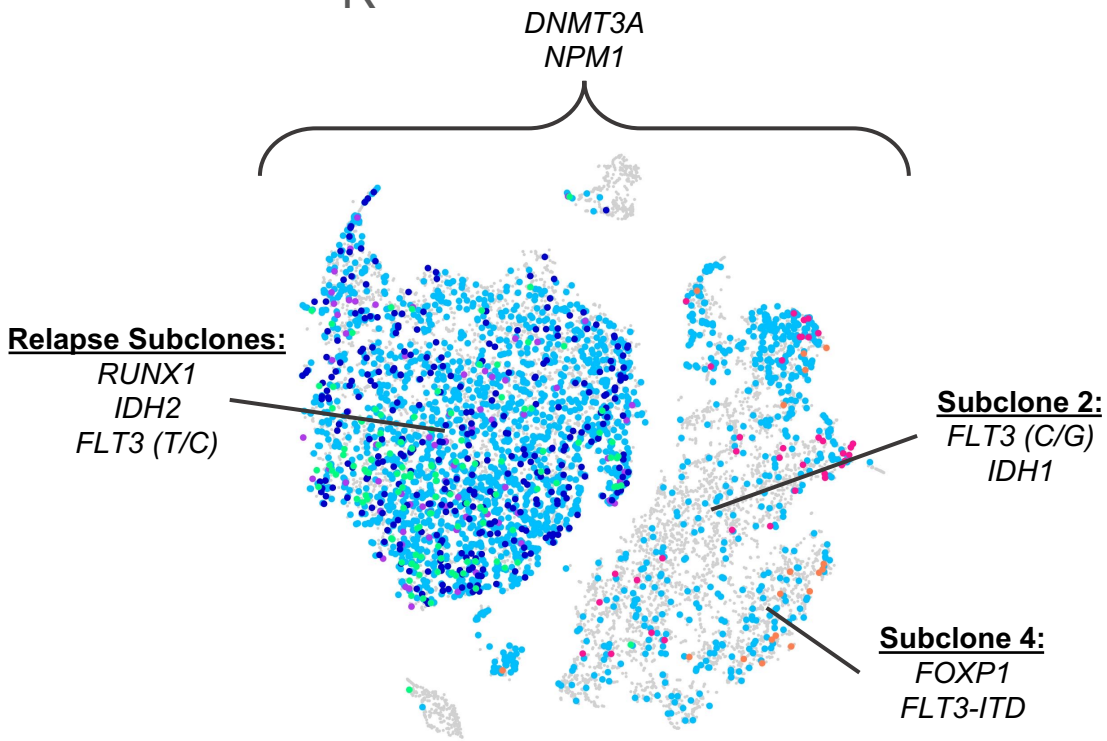
Tools: cb_sniffer, Vartrix, CONICSmat, HoneyBadger

Petti and Williams et al. *Nature Comm.* 2019



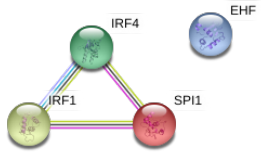


Mutation mapping +
Differential expression +
Pseudotemporal ordering to identify
epigenetically-driven tumor evolution

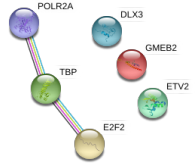


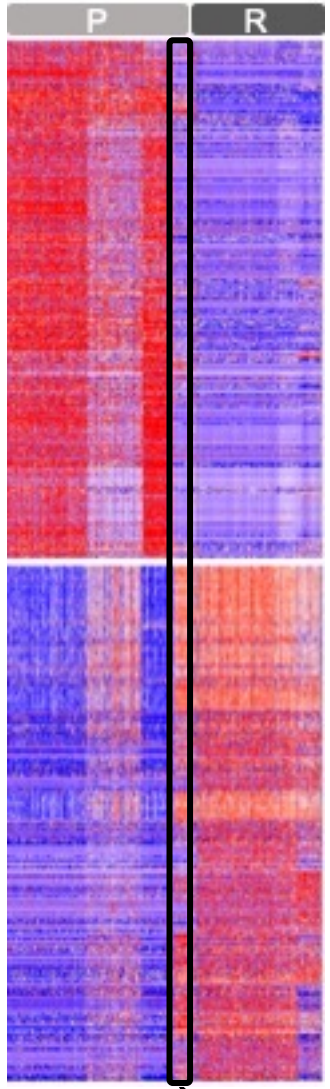
Deciphering the subclonal ecosystem of AML31

Myeloid immune resp.
Antigen presentation
Response to IFNG
Respiratory burst
PU.1
IRF1



Proliferation
Nucleosomes
Translation
FLT3, RUNX1, NPM1, KDM5B, *MYC, MYB, GATA2, XBP1, EGR1, SOX4, HOXA9*



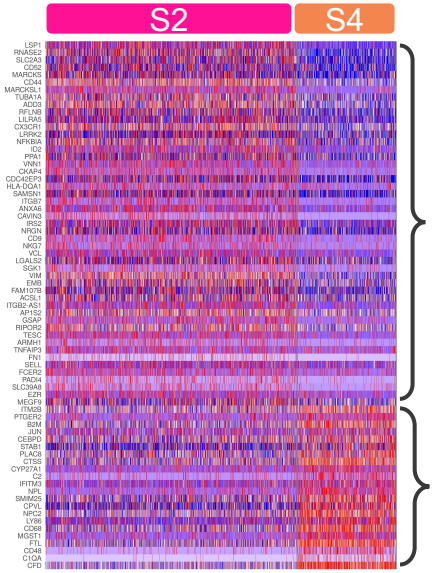


DNMT3A
NPM1

Relapse:
RUNX1
IDH2
FLT3 (T/C)

Subclone 2:
FLT3 (C/G)
IDH1

Subclone 4
FOXP1
FLT3-ITD

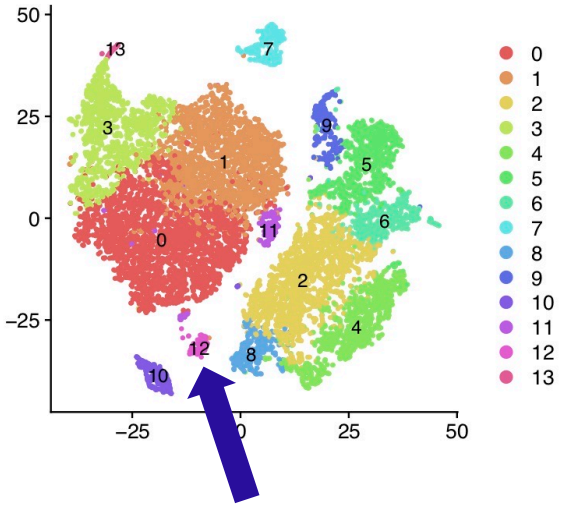


Acquired Immunity, adhesion

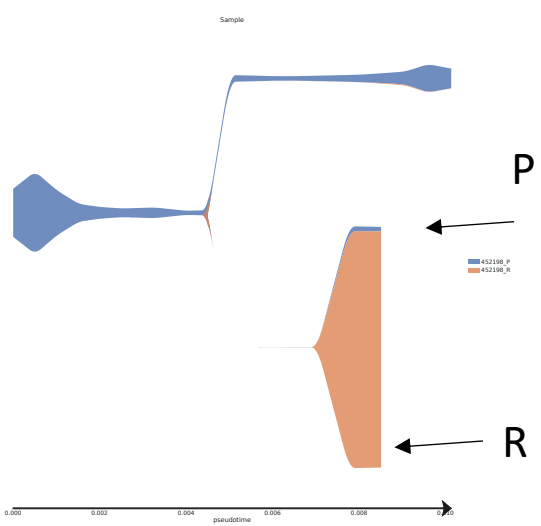
Innate Immunity

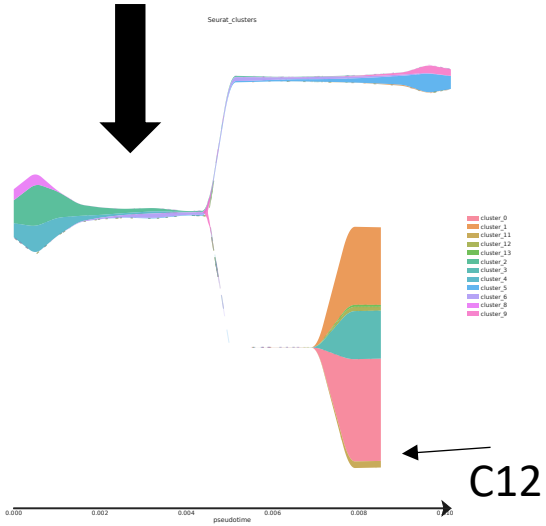
Pseudotemporal ordering (STREAM)

- Reconstructs step-wise changes in gene expression
- Suggests that Presentation Cluster 12 links the presentation and relapse samples

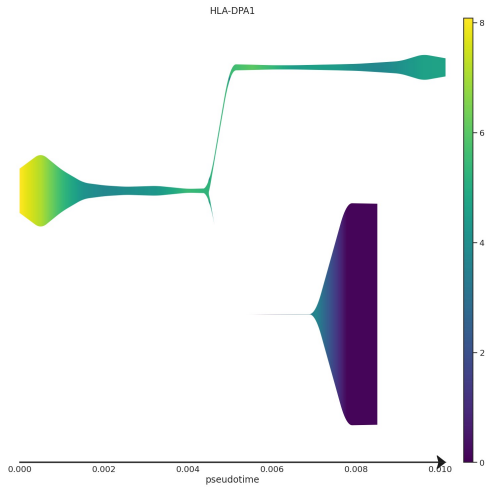


Cluster 12





HLA-DPA1 Expression



Seurat exercise, continued

Option 1: Continue the analysis on your own **up to and including DEG analysis on all clusters and heatmap of top DEGs in each cluster,** by referring to the functions in this powerpoint. Optionally, perform batch correction and repeat the analysis, and/or subset the T cells and perform the analysis on those.

Option 2: Follow the workflow for steps 14 (and optionally 15 & 16) (with cut-and-paste code) outlined at https://rnabio.org/module-08-scrna/0008/02/01/scRNA/.

Once you have finished Step 14, aka the DEG & Heatmap step, please click your 'tada' emoji.

Part III:
Useful Seurat functions for future reference

Seurat object: meta data

Meta data (scrna@meta.data) contains:
- summary statistics
- sample name
- cluster membership for each cell
- cell cycle phase for each cell
- batch or sample for each cell
- other custom labels for each cell

Access using:
- `scrna[[]]`
- `scrna@meta.data`
- `str(scrna@meta.data)`

Combine with R commands such as head and str, e.g. str(scrna[[]])

Example: Access number of genes (“Features”) for each cell:
- `head(scrna@meta.data$nFeature_RNA)`

Example: Access number of UMIs for each cell:
- `head(scrna@meta.data$nCount_RNA)`

What are the items in the current default cell identity class?

`levels(x=scrna)`

How many clusters are there?
- `length(unique(scrna@meta.data$seurat_clusters))`
- `levels(x=scrna)`

What batches are included in this data set?

`unique(scrna@meta.data$Batch)`

Can add meta data

```
> str(scrna@meta.data)
'data.frame':   13049 obs. of  14 variables:
 $ orig.ident           : Factor w/ 1 level "452198": 1 1 1 1 1 1 1 1 1 1 ...
 $ nCount_RNA           : num  11846 3535 2850 9158 5607 ...
 $ nFeature_RNA         : int  3557 1740 1617 2810 2298 3834 1737 2526 2745 2947 ...
 $ percent.mito         : num  0.0386 0.0526 0.0589 0.0717 0.0462 ...
 $ percent.ribo         : num  0.1033 0.097 0.0821 0.1902 0.0885 ...
 $ S.Score              : num  -0.00337 -0.06351 0.30666 -0.09634 -0.0508 ...
 $ G2M.Score            : num  -0.428 -0.112 0.109 -0.28 -0.233 ...
 $ Phase                : Factor w/ 3 levels "G1","G2M","S": 1 1 3 1 1 1 3 1 1 1 ...
 $ CC.Difference        : num  0.4243 0.0486 0.1976 0.1833 0.182 ...
 $ Sample               : Factor w/ 2 levels "452198_P","452198_R": 1 1 1 1 1 1 1 1 1 1 ...
 $ RNA_snn_res.0.7      : Factor w/ 14 levels "0","1","2","3",..: 3 4 7 3 3 7 3 6 3 3 ...
 $ seurat_clusters      : Factor w/ 14 levels "0","1","2","3",..: 3 4 7 3 3 7 3 6 3 3 ...
 $ ClusterNames_0.7_17PC: Factor w/ 14 levels "0","1","2","3",..: 3 4 7 3 3 7 3 6 3 3 ...
 $ CellType             : Factor w/ 15 levels "B-CELL","BASO",..: 11 4 11 13 13 4 11 11 11 11 ...
```

Seurat object: Assay data and Dimensionality reduction

- Assay data (i.e. RNA-seq measurements): e.g. ‘RNA’
  - Assay = data type
  - Each assay has multiple “slots” that hold the raw data (‘counts’), scaled data (‘scale.data’) or normalized data (‘data)
  - Data transformation, such as scaling and normalization, adds new ‘slots’ to the assay data
  - Access values in the slots as follows:
    - raw RNA counts: `scrna[['RNA']]@counts[1:3,1:3]`
    - scaled data after SCTransform: `scrna[['SCT']]@scale.data[1:3,1:3]`
    - corrected UMI count data after SCTransform: `scrna[['SCT']]@counts[1:3,1:3]`
    - log-normalized data after SCTransform: scrna[['SCT']]@data[1:3,1:3]
  - Alternatively, use the function GetAssayData:
    - `GetAssayData(object = scrna, slot = 'scale.data')[1:3, 1:3]`
    - `GetAssayData(object = scrna, slot = 'counts')[1:3, 1:3]`
  - It’s often important to know what ‘slot’ a function is using. Sometimes you can change it.
- Dimensionality reduction data, e.g. PCA, tSNE, UMAP data
  - Access using scrna[[‘pca’]], scrna[[‘tsne’]], scrna[[‘umap’]]

Common functions for the Seurat object

```
GetAssayData(object = scrna, slot = "counts")
GetAssayData(object = scrna, slot = "scale.data")
FetchData(object = scrna) # returns a data frame
colnames(x = scrna)
rownames(x = scrna)
VariableFeatures(object = scrna)
HVFInfo(object = scrna)
scrna[["assay.name"]] eg scrna[[“RNA”]]
scrna[["pca"]]
Embeddings(object = scrna, reduction = "pca")
Loadings(object = scrna, reduction = "pca")
scrna$name <- vector # assign a vector of identities
scrna$name # e.g. scrna$seurat_clusters
Idents(object = scrna) # get default cell identities
Idents(object = scrna) <- "new.idents” # change the identities
Idents(object = scrna, cells = 1:10) <- "new.idents” # change the identities for specific cells
scrna$saved.idents <- Idents(object = scrna) # change current identities for an existing identity class
levels(x = scrna) # get a list of the default identities (e.g. a list of clusters)
RenameIdents(object = scrna, "old.ident" = "new.ident") # change name of identity classes
WhichCells(object = scrna, idents = "ident.keep") # which cells are in cluster x?
WhichCells(object = scrna, idents = "ident.remove", invert = TRUE)
WhichCells(object = scrna, downsample = 500)
WhichCells(object = scrna, expression = name > low & name < high)
subset(x = scrna, cells = cellist, idents=‘keep’); # subset seurat object and return seurat object
subset(x = scrna, subset = name > low & name < high)
merge(x = object1, y = object2)
```

Additional ways to access data

```
Cells(scrna) # get list of cells
# choose cells that have a given characteristic, here, that are in “sample1”:
subcells <- Cells(scrna)[(which(scrna[["Sample"]]$Sample == sample1))]
# alternative approach, choosing cells in cluster 4:
Idents(object=scrna) <- "ClusterNames” # set default identity to ClusterNames
WhichCells(scrna,idents="4") # use WhichCells
# extract raw expression matrix for all cells in cluster 4
as.matrix(GetAssayData(scrna, slot = "counts")[, WhichCells(scrna, ident = '4')])
FetchData # subset Seurat object and return a data frame
subset # subset Seurat object and return a Seurat object
levels(scrna) # reorder the columns in Do.Heatmap
Command(scrna) # lists the functions that were applied to the Seurat object
```