Introduction to single cell analysis with Seurat V5

Sara Brin Rosenthal, Ph.D.

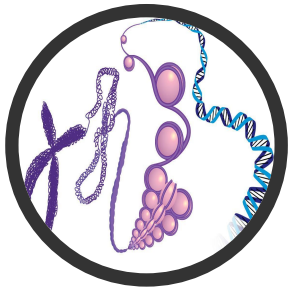
Associate Director for Research

Center for Computational Biology and Bioinformatics (CCBB)

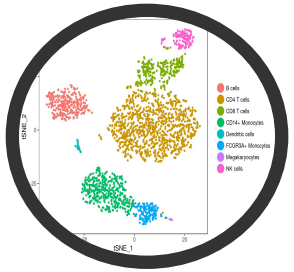
UC San Diego
Altman Clinical and Translational Research Institute

CCBB | CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

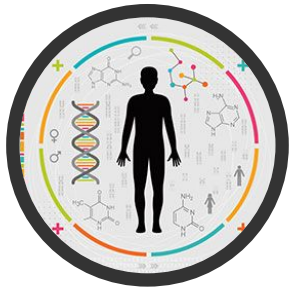
ACTRI
Center for Computational Biology & Bioinformatics (CCBB)



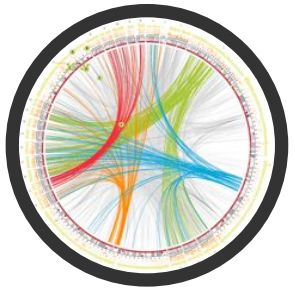
Epigenomics & Transcriptomics



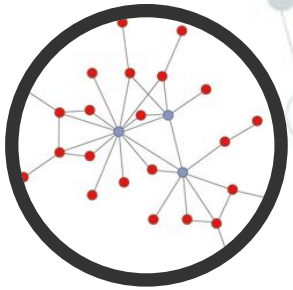
Single cell RNAseq & ATACseq



Microbiome



Whole Genome & Exome



Networks & Integrative Multi-Omics

Contact: Brin Rosenthal sbrosenthal@health.ucsd.edu
Website: Compbio.ucsd.edu

- SDDRC members eligible for 50% subsidy on CCBB analysis
- Vouchers for CCBB analysis (up to $10k/year) available for ACTRI affiliated researchers

UC San Diego
Altman Clinical and Translational Research Institute

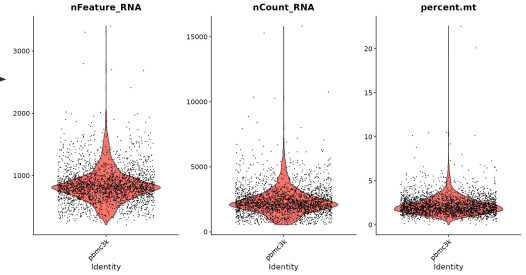
CCBB | CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Outline*

Alignment with Cell Ranger (10x genomics)

Data import to Seurat

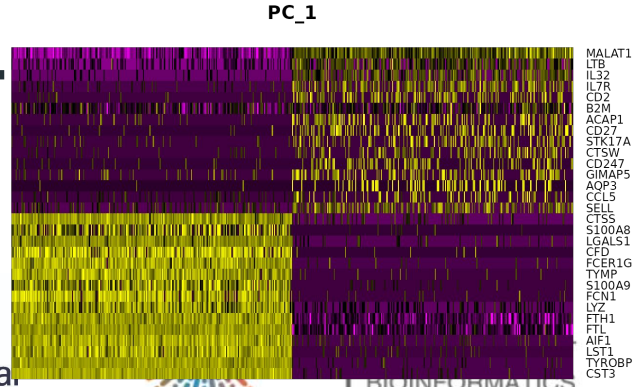
Quality control/ Preprocessing



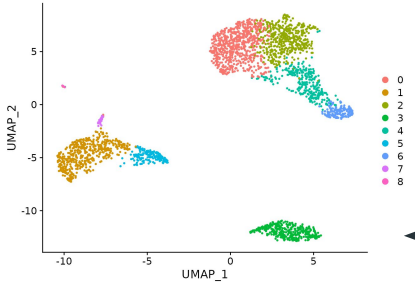
Normalization

Sample integration

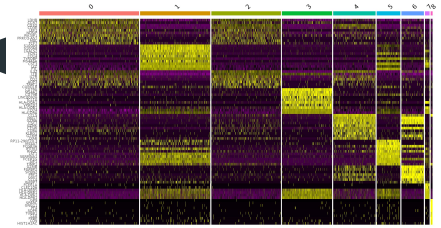
Dimensionality reduction



Cluster identification



Marker gene identification



Cell type classification

Differential expression analysis

* For more detailed information on any of these steps please visit:
https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/what-is-cell-ranger
https://satijalab.org/seurat/

Images from https://satijalab.org/seurat/articles/pbmc3k_tutorial.html

UC San Diego
Altman Clinical and Translational Research Institute
BIOINFORMATICS

Improvements in Seurat V5 (vs V4)

- Efficiency
- More integration methods

- **Integration workflow:**
  Seurat v5 introduces a streamlined integration and data transfer workflows that performs integration in low-dimensional space, and improves speed and memory efficiency. The results of integration are not identical between the two workflows, but users can still run the v4 integration workflow in Seurat v5 if they wish.

  In previous ver
  the data can be
  integration vig

- **Differential expression:**
  Seurat v5 now uses the presto package (from the Korunsky and Raychaudhari labs), when available, to perform differential expression analysis. Using presto can dramatically speed up DE testing, and we encourage users to install it.
  In addition, in Seurat v5 we implement a pseudocount (when calculating log-FC) at the group level instead of the cell level. As a result, users wil
  particularly for ge
  McCarthy and Pa

- **Pseudobulk analysis:**
  Once a single-cell dataset has been analyzed to annotate cell subpopulations, pseudobulk analyses (i.e. aggregating together cells within a given subpopulation and sample) can reduce noise, improve quantification of lowly expressed genes, and reduce the size of the data matrix. In Seurat v5, we encourage the use of the `AggregateExpression` function to perform pseudobulk analysis.
  Check out our differential expression vignette as well as our pancreatic/healthy PBMC comparison, for examples of how to use `AggregateExpression` to perform robust differential expression of scRNA-seq data from multiple different conditions.

Alignment with Cell Ranger

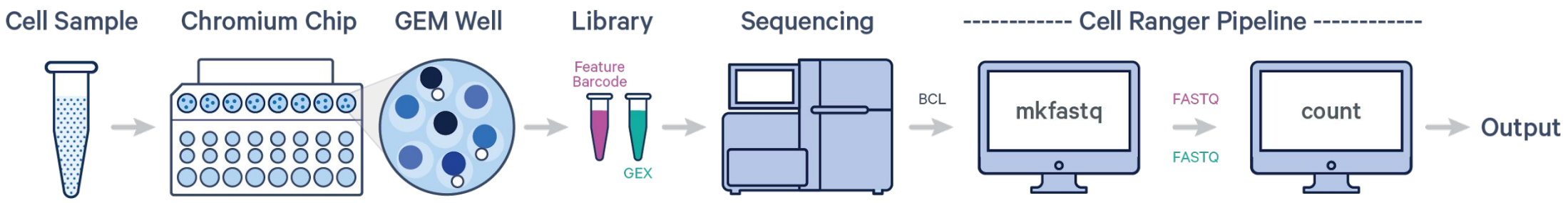


Image from https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/what-is-cell-ranger

Data import to Seurat

Load in data from 10X

Source: `R/preprocessing.R`

Enables easy loading of sparse data matrices provided by 10X genomics.

```
Read10X(
  data.dir,
  gene.column = 2,
  cell.column = 1,
  unique.features = TRUE,
  strip.suffix = FALSE
)
```

Arguments

data.dir
Directory containing the matrix.mtx, genes.tsv (or features.tsv), and barcodes.tsv file
vector can be given in order to load several data directories. If a named vector is give
prefixed with the name.

gene.column
Specify which column of genes.tsv or features.tsv to use for gene names; default is 2

cell.column
Specify which column of barcodes.tsv to use for cell names; default is 1

unique.features
Make feature names unique (default TRUE)

strip.suffix
Remove trailing "-1" if present in all cell barcodes.

```
library(dplyr)
library(Seurat)
library(patchwork)

# Load the PBMC dataset
pbmc.data <- Read10X(data.dir = "/brahms/mollag/practice/filtered_gene_bc_matrices/hg19/")
# Initialize the Seurat object with the raw (non-normalized data).
pbmc <- CreateSeuratObject(counts = pbmc.data, project = "pbmc3k", min.cells = 3, min.features =
200)
pbmc
```

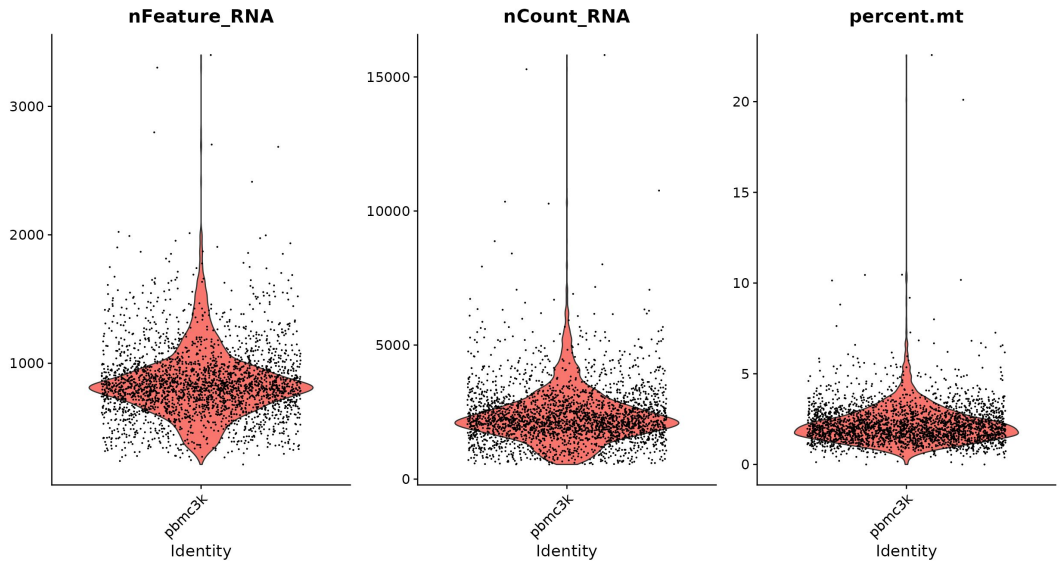
```
## An object of class Seurat
## 13714 features across 2700 samples within 1 assay
## Active assay: RNA (13714 features, 0 variable features)
##  1 layer present: counts
```

Quality control

- Necessary to identify and filter out low quality cells, doublets, etc. Note may also be recommended to use doublet-filtering software (e.g. DoubletFinder https://github.com/chris-mcginnis-ucsf/DoubletFinder)
- Some metrics to approximate these are:
  - Number of unique genes detected per cell (low quality cells have few genes expressed, whereas doublets may have unusually high number of genes
  - Number of molecules detected within cells (similar to gene counts)

ess

```
pbmc <- subset(pbmc, subset = nFeature_RNA > 200 & nFeature_RNA < 2500 & percent.mt < 5)
```



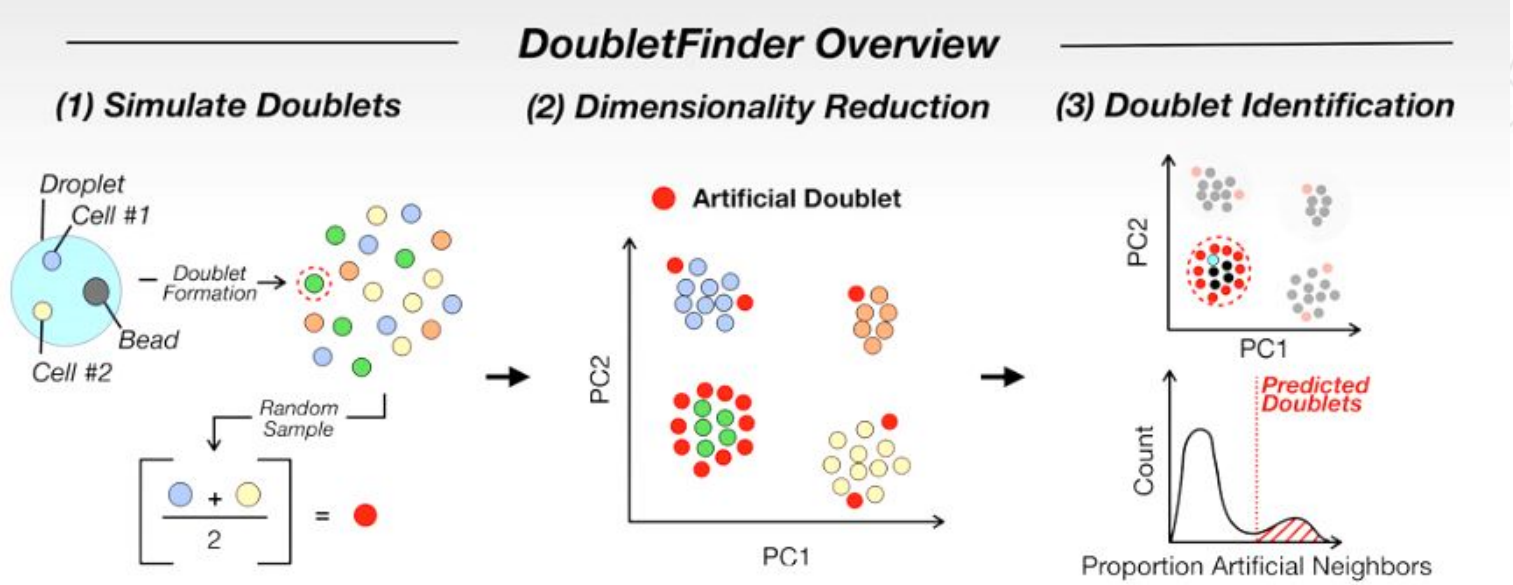
Diego
d Translational
Research Institute

CCBB
CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Doublet detection -- DoubletFinder

In Brief

scRNA-seq data interpretation is confounded by technical artifacts known as doublets—single-cell transcriptome data representing more than one cell. Moreover, scRNA-seq cellular throughput is purposefully limited to minimize doublet formation rates. By identifying cells sharing expression features with simulated doublets, DoubletFinder detects many real doublets and mitigates these two limitations.



UC San Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Preprocessing: Normalization

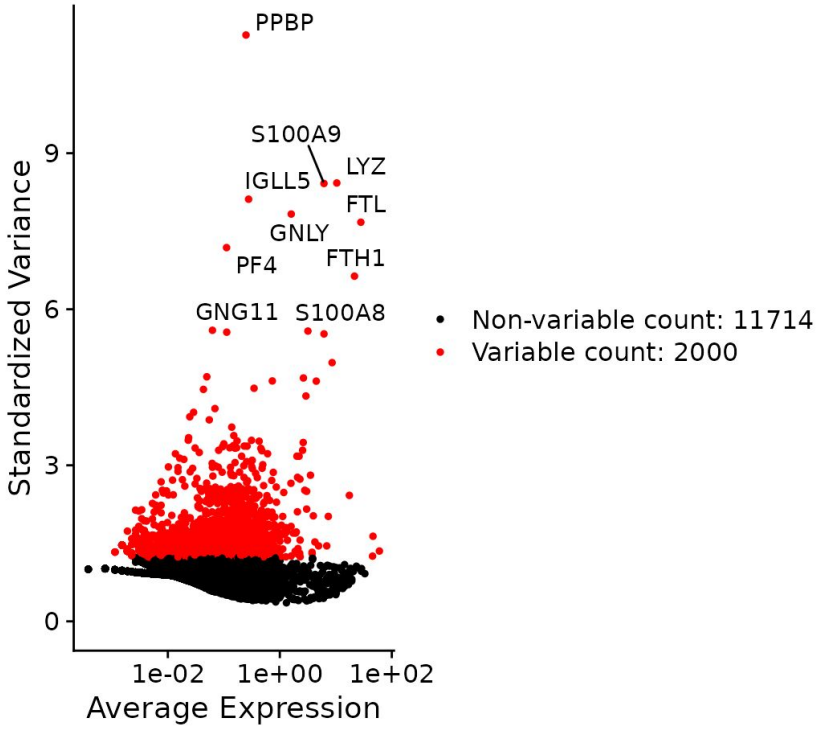
- Data are normalized so gene expression values can be compared across cells
- Normalized by total feature expression, multiplied by a scale factor (10,000 by default).

```
pbmc <- NormalizeData(pbmc, normalization.method = "LogNormalize", scale.factor = 10000)
```

- Note: SCtransform -- alternate normalization method developed by Satija lab: omits the need for heuristic steps including pseudocount addition or log-transformation and improves common downstream analytical tasks such as variable gene selection, dimensional reduction, and differential expression. But unclear if compatible with Harmony, the data integration method we will use

Preprocessing: Identify highly variable features

- Find the genes which change the most cell to cell in the dataset.
- It has been found that these highly variable genes are the most informative for downstream analysis*



*PMID: 24056876

Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

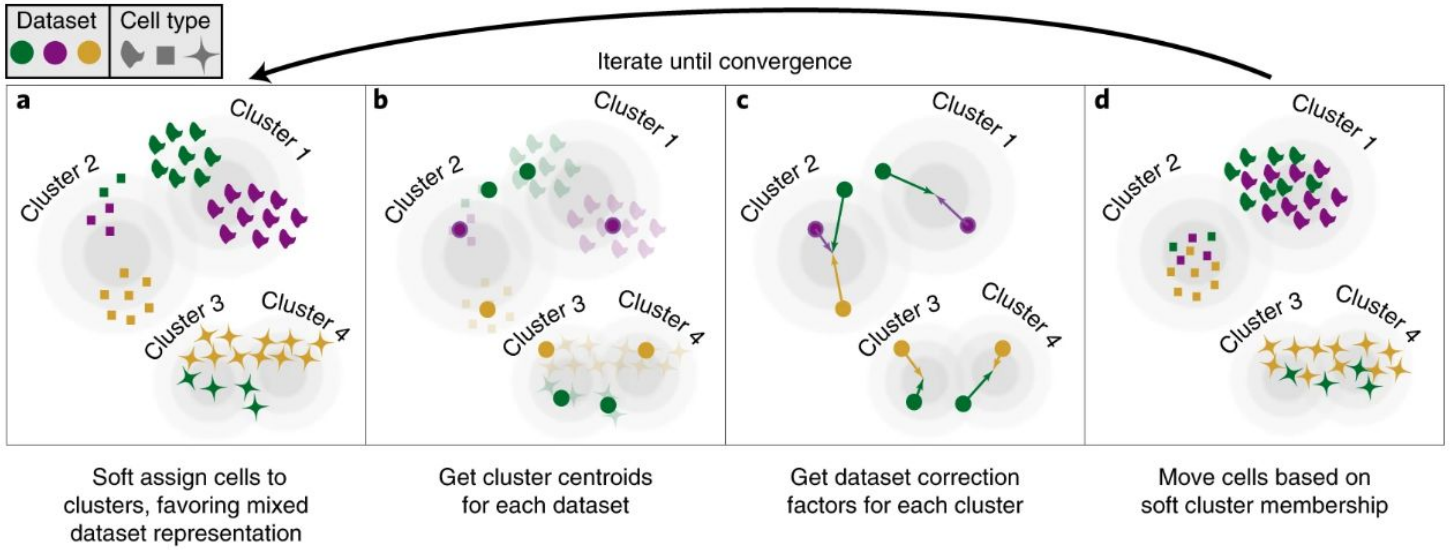
Integration of multiple samples

- Poses a challenge for single cell analysis.

- Want to keep true biological variation (between conditions/treatments etc), but remove sample-specific effects
- Harmony (https://github.com/immunogenomics/harmony):
    - projects cells into a shared embedding in which cells group by cell type rather than dataset-specific conditions.
    - simultaneously accounts for multiple experimental and biological factors.
    - Authors demonstrate the superior performance of Harmony to previously published algorithms while requiring fewer computational resources

Fig. 1: Overview of Harmony algorithm.

From: Fast, sensitive and accurate integration of single-cell data with Harmony

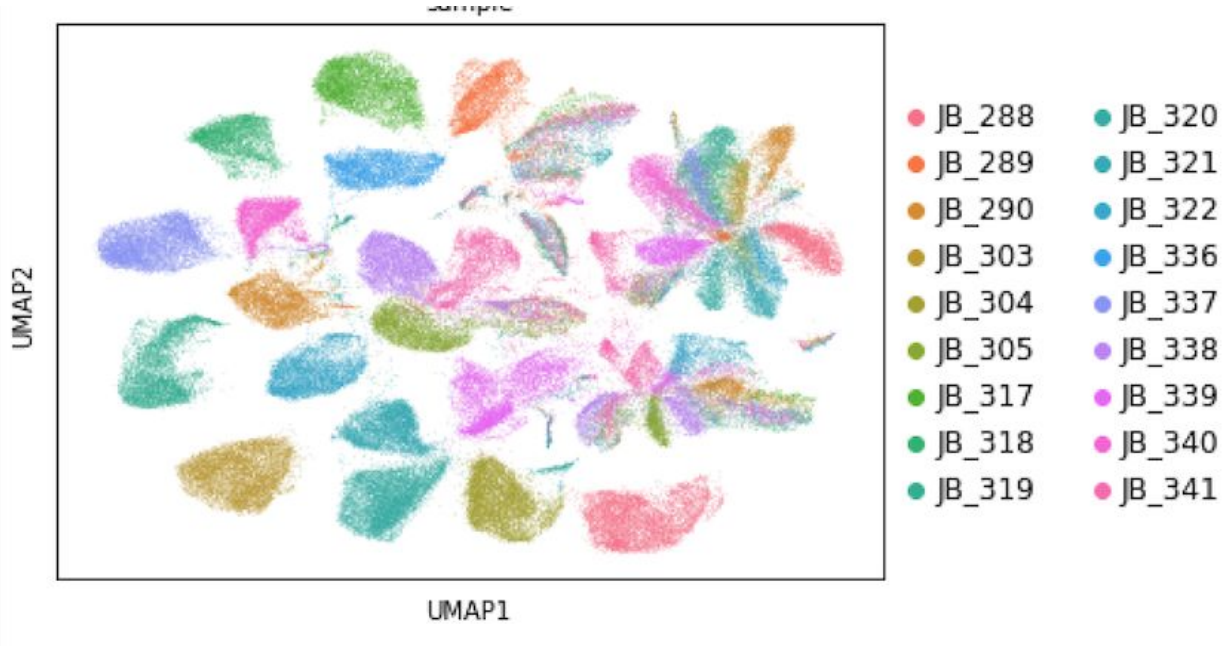


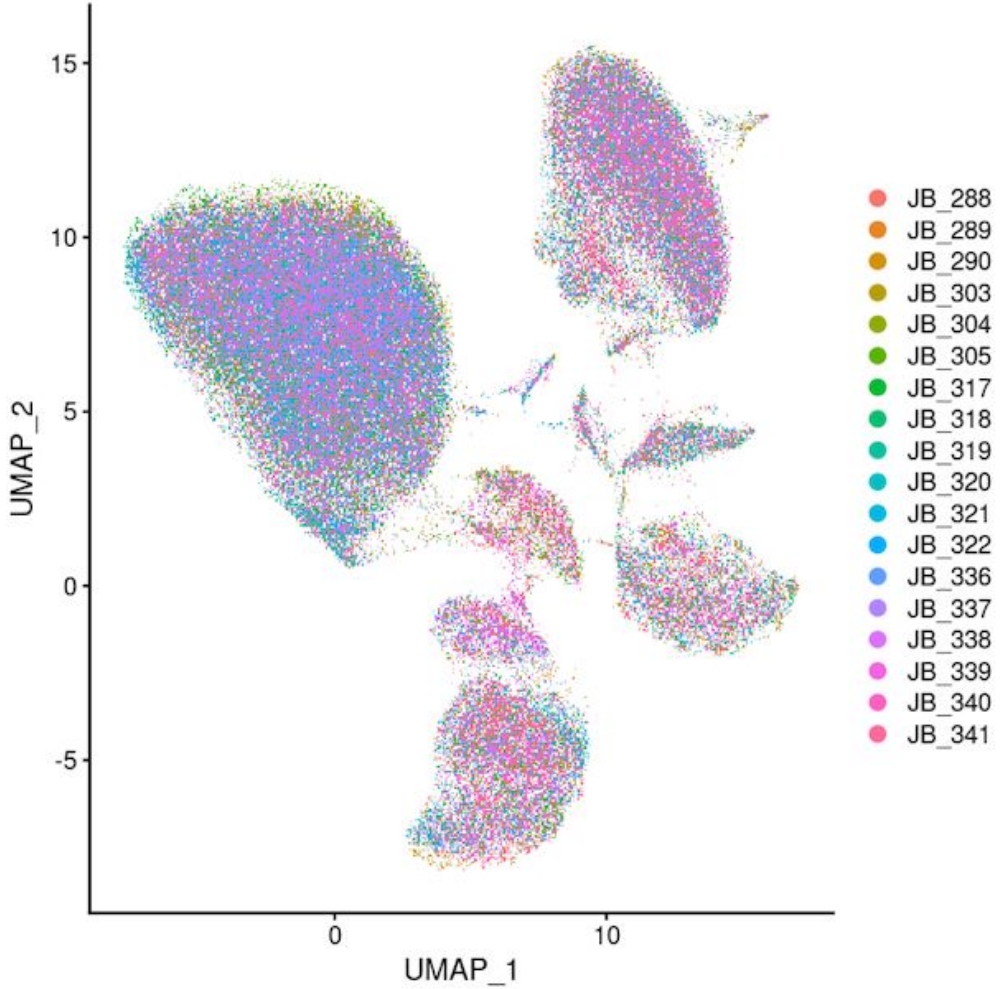
https://www.nature.com/articles/s41592-019-0619-0

UC San Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Importance of proper sample integration





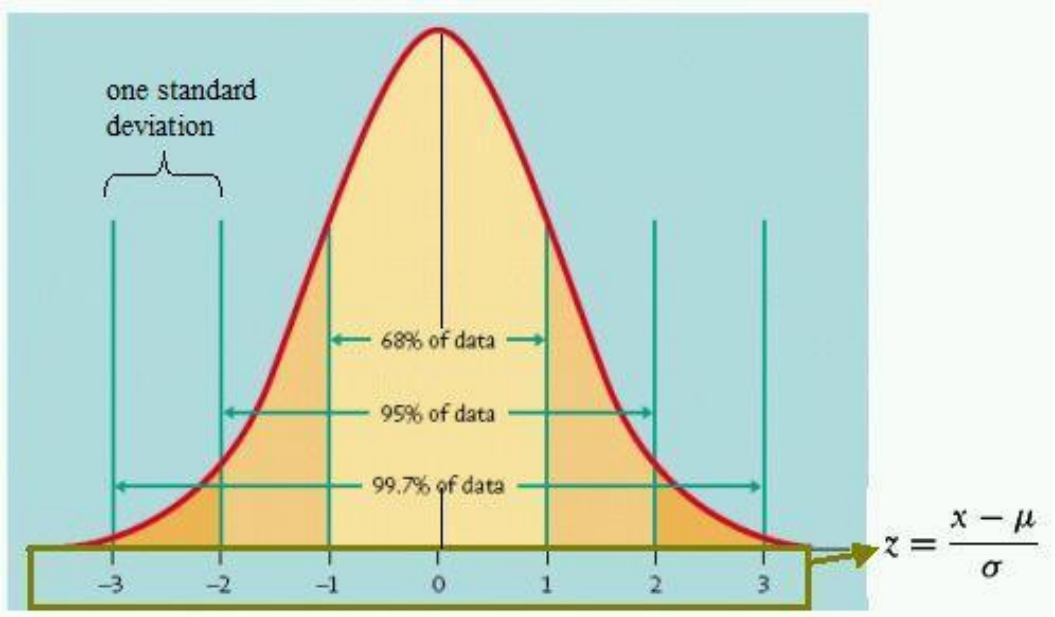
UC San Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Preprocessing: Data scaling

- Scaling shifts the expression of each gene so mean=0 and variance=1. Useful so highly-expressed genes don't dominate in downstream analysis.

```
all.genes <- rownames(pbmc)
pbmc <- ScaleData(pbmc, features = all.genes)
```



Preprocessing: Linear dimensionality reduction

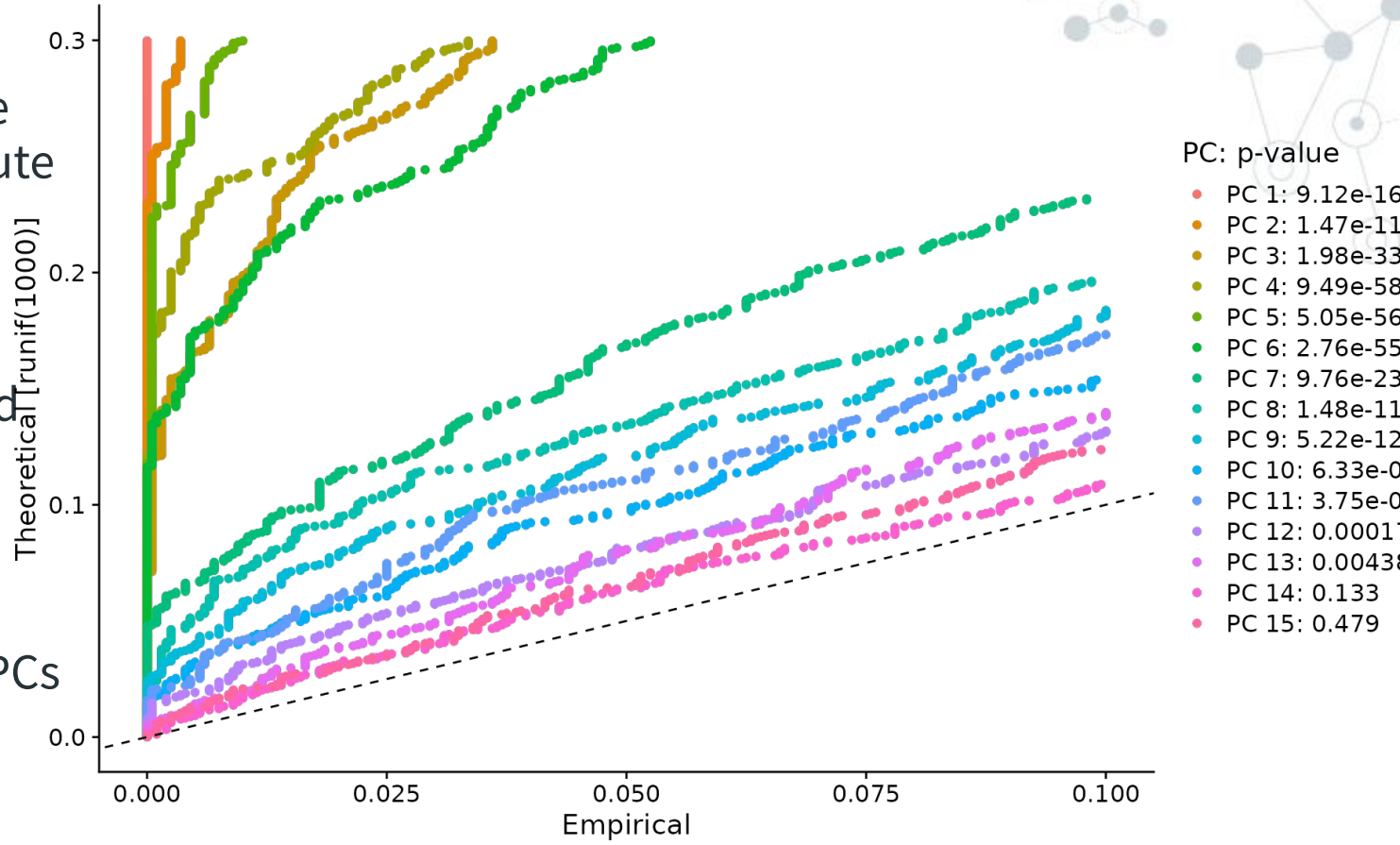
- E.g. principal component analysis (PCA). By default this is computed on variable features identified previously.
- These components will be used for downstream clustering steps.

```r
# Examine and visualize PCA results a few different ways
print(pbmc[["pca"]], dims = 1:5, nfeatures = 5)
```

```
## PC_ 1
## Positive:  CST3, TYROBP, LST1, AIF1, FTL
## Negative:  MALAT1, LTB, IL32, IL7R, CD2
## PC_ 2
## Positive:  CD79A, MS4A1, TCL1A, HLA-DQA1, HLA-DQB1
## Negative:  NKG7, PRF1, CST7, GZMB, GZMA
## PC_ 3
## Positive:  HLA-DQA1, CD79A, CD79B, HLA-DQB1, HLA-DPB1
## Negative:  PPBP, PF4, SDPR, SPARC, GNG11
## PC_ 4
## Positive:  HLA-DQA1, CD79B, CD79A, MS4A1, HLA-DQB1
## Negative:  VIM, IL7R, S100A6, IL32, S100A8
## PC_ 5
## Positive:  GZMB, NKG7, S100A8, FGFBP2, GNLY
## Negative:  LTB, IL7R, CKB, VIM, MS4A7
```

Preprocessing: Identify data dimensionality

- How many PCs should we keep for downstream clustering analysis? More PCs explain more variance, but compute becomes more of a barrier.
- Jackstraw procedure implemented in Seurat, to determine statistically significant PCs. E.g. compare observed to a null distribution created from permuting a subset of the data.
- Here we would choose the top 10-12 PCs



Clustering

- Graph-based clustering, based on nearest N neighbors in PC space. In this example, we use 10 PCs, so the neighbors are computed in 10-dimensional space.
- Common misconception… clusters are NOT computed from the UMAP coordinates. UMAP is mostly used for visualization, and clusters often, but not always, can be seen in the UMAP plots.
- Resolution parameter: tune based on expected biology.

```
pbmc <- FindNeighbors(pbmc, dims = 1:10)
pbmc <- FindClusters(pbmc, resolution = 0.5)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 2638
## Number of edges: 95927
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8728
## Number of communities: 9
## Elapsed time: 0 seconds
```

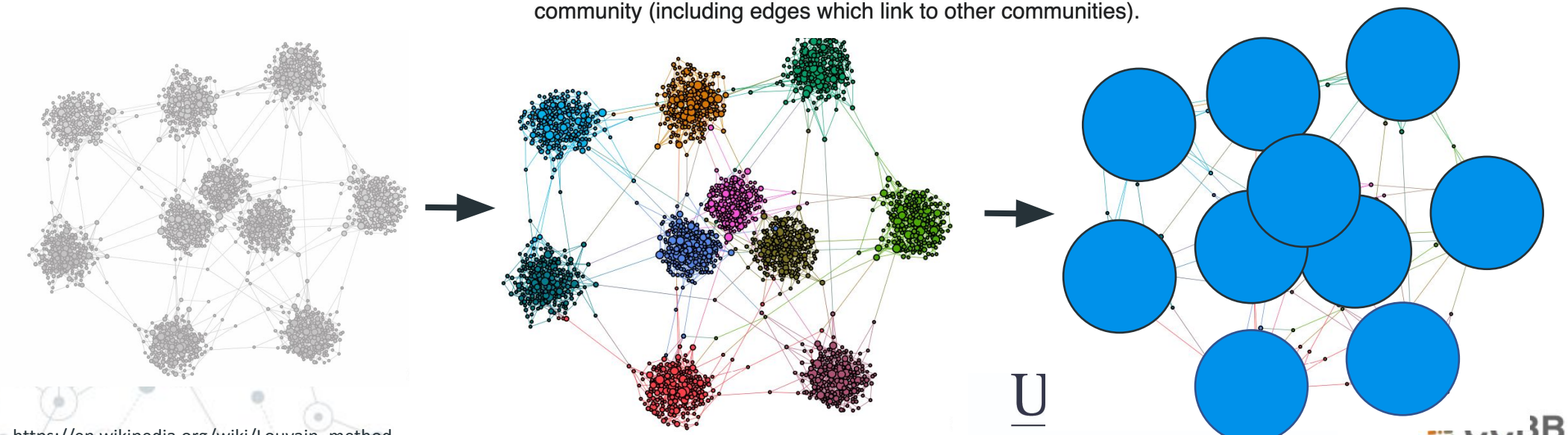
Clustering: modularity optimization with Louvain

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j),$$

- A_{ij} represents the edge weight between nodes i and j;
- k_i and k_j are the sum of the weights of the edges attached to nodes i and j, respectively;
- m is the sum of all of the edge weights in the graph;
- c_i and c_j are the communities of the nodes; and
- δ is Kronecker delta function ($\delta(x, y) = 1$ if $x = y$, 0 otherwise).

$$Q_c = \frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2,$$

- Σ_{in} is the sum of edge weights between nodes within the community c (each edge is considered twice); and
- Σ_{tot} is the sum of all edge weights for nodes within the community (including edges which link to other communities).



https://en.wikipedia.org/wiki/Louvain_method

Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." *Journal of statistical mechanics: theory and experiment* 2008.10 (2008): P10008.

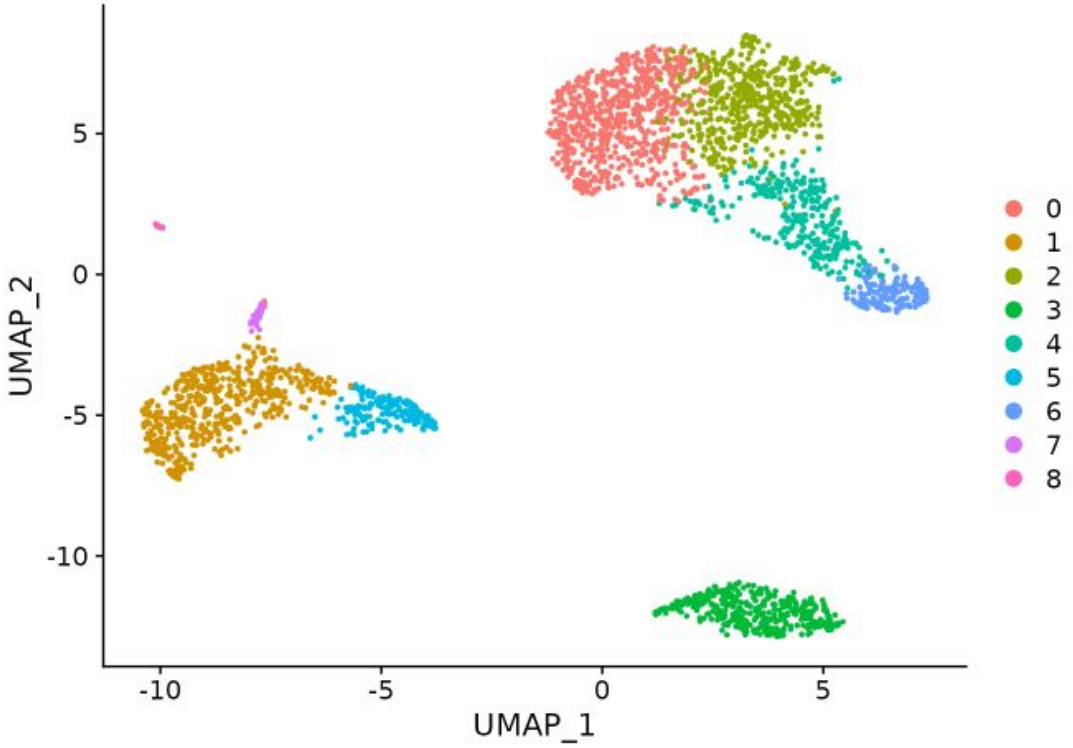
Network image from https://www.nature.com/articles/s41598-018-27506-x.pdf

Non-linear dimensional reduction (UMAP)

- As mentioned previously, UMAP computed independently from clusters. Useful for visualization.

```
# If you haven't installed UMAP, you can do so via reticulate::py_install(packages =
# 'umap-learn')
pbmc <- RunUMAP(pbmc, dims = 1:10)
```

```
# note that you can set `label = TRUE` or use the LabelClusters function to help label
# individual clusters
DimPlot(pbmc, reduction = "umap")
```



Find cluster-specific markers

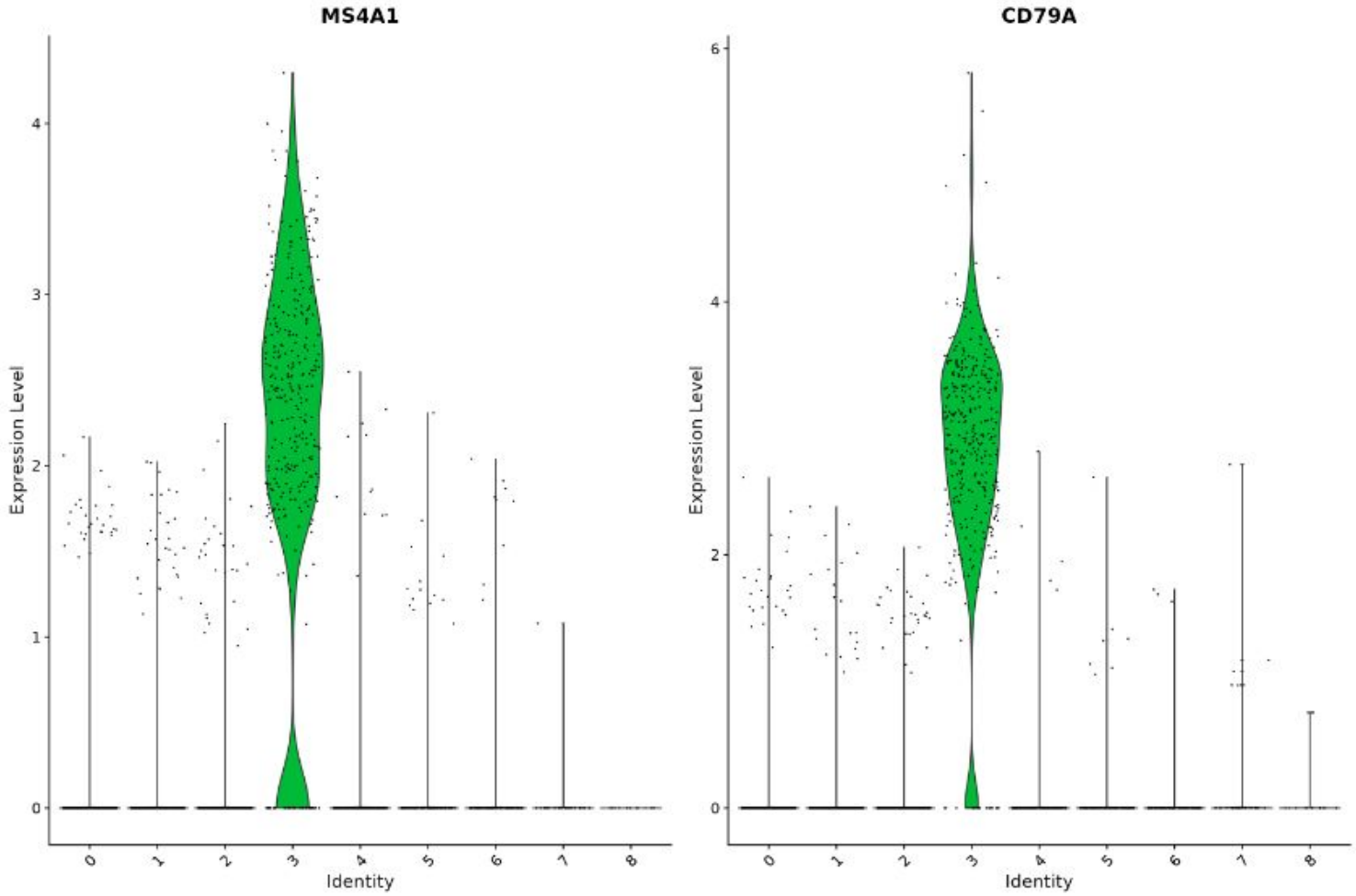
◎ Identify genes which significantly change between cells in each cluster, and all other cells.

```r
# find markers for every cluster compared to all remaining cells, report only the positive
# ones
pbmc.markers <- FindAllMarkers(pbmc, only.pos = TRUE, min.pct = 0.25, logfc.threshold = 0.25)
pbmc.markers %>%
    group_by(cluster) %>%
    slice_max(n = 2, order_by = avg_log2FC)
```

```
## # A tibble: 18 × 7
## # Groups:   cluster [9]
##        p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##        <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 9.57e- 88       1.36 0.447 0.108 1.31e- 83 0       CCR7
##  2 3.75e-112       1.09 0.912 0.592 5.14e-108 0       LDHB
##  3 0               5.57 0.996 0.215 0         1       S100A9
##  4 0               5.48 0.975 0.121 0         1       S100A8
##  5 1.06e- 86       1.27 0.981 0.643 1.45e- 82 2       LTB
##  6 2.97e- 58       1.23 0.42  0.111 4.07e- 54 2       AQP3
##  7 0               4.31 0.936 0.041 0         3       CD79A
##  8 9.48e-271       3.59 0.622 0.022 1.30e-266 3       TCL1A
##  9 5.61e-202       3.10 0.983 0.234 7.70e-198 4       CCL5
## 10 7.25e-165       3.00 0.577 0.055 9.95e-161 4       GZMK
## 11 3.51e-184       3.31 0.975 0.134 4.82e-180 5       FCGR3A
## 12 2.03e-125       3.09 1     0.315 2.78e-121 5       LST1
## 13 3.13e-191       5.32 0.961 0.131 4.30e-187 6       GNLY
## 14 7.95e-269       4.83 0.961 0.068 1.09e-264 6       GZMB
## 15 1.48e-220       3.87 0.812 0.011 2.03e-216 7       FCER1A
## 16 1.67e- 21       2.87 1     0.513 2.28e- 17 7       HLA-DPB1
## 17 1.92e-102       8.59 1     0.024 2.63e- 98 8       PPBP
## 18 9.25e-186       7.29 1     0.011 1.27e-181 8       PF4
```

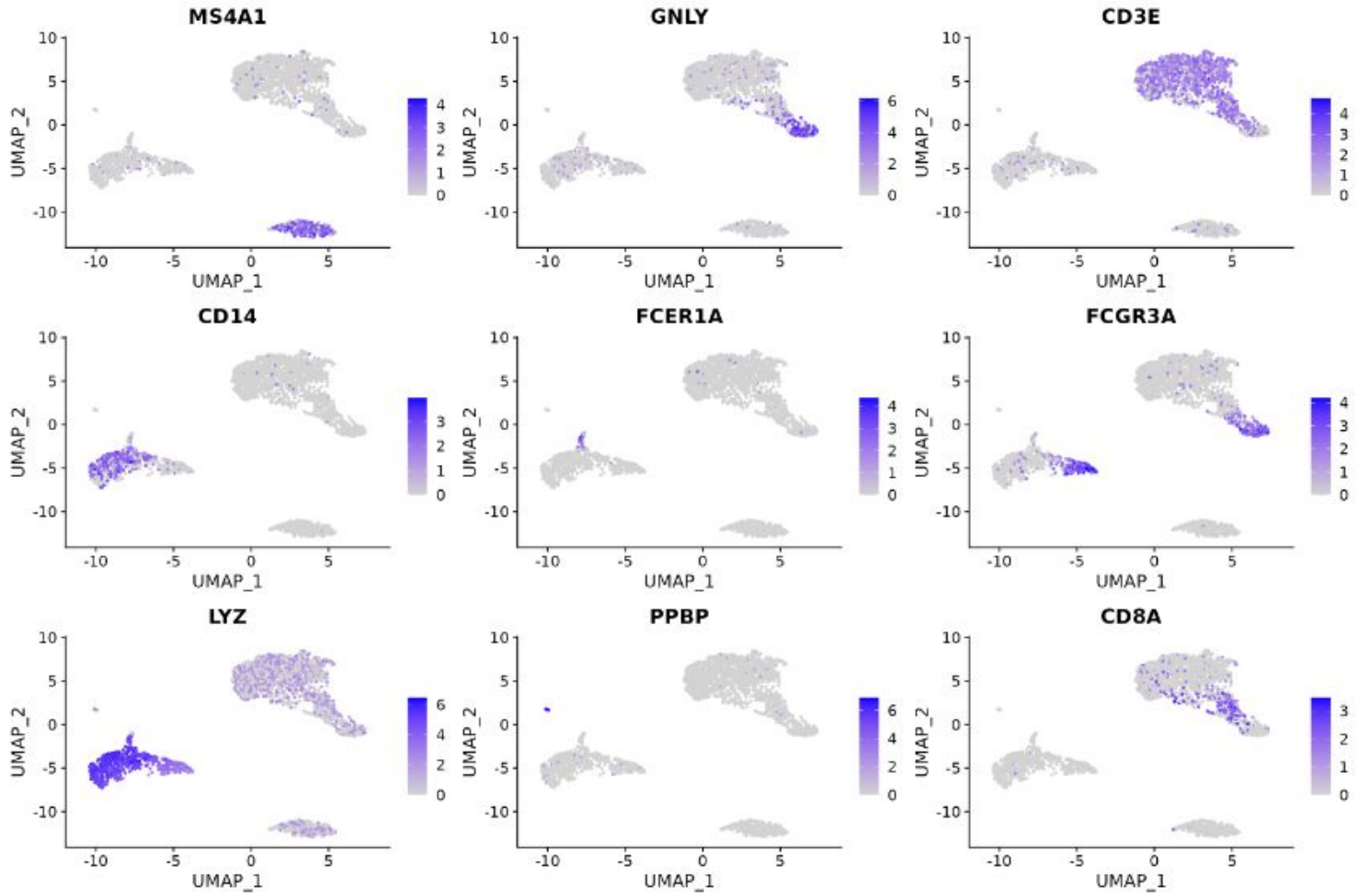
Visualization options: violin plots

```
VlnPlot(pbmc, features = c("MS4A1", "CD79A"))
```



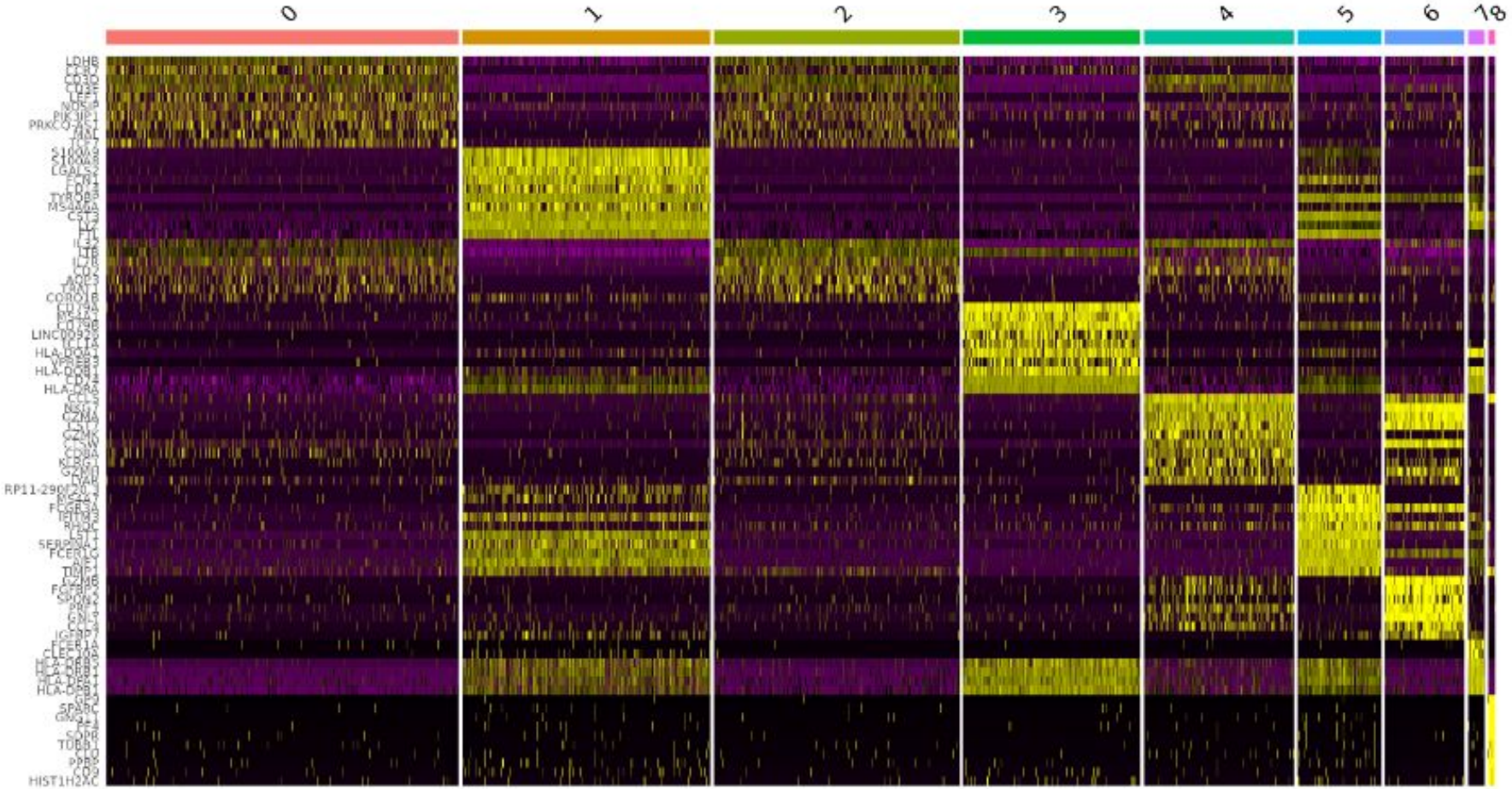
Visualization options: expression UMAP

```
FeaturePlot(pbmc, features = c("MS4A1", "GNLY", "CD3E", "CD14", "FCER1A", "FCGR3A", "LYZ", "PPBP",
    "CD8A"))
```



Visualization options: heatmap of top marker genes

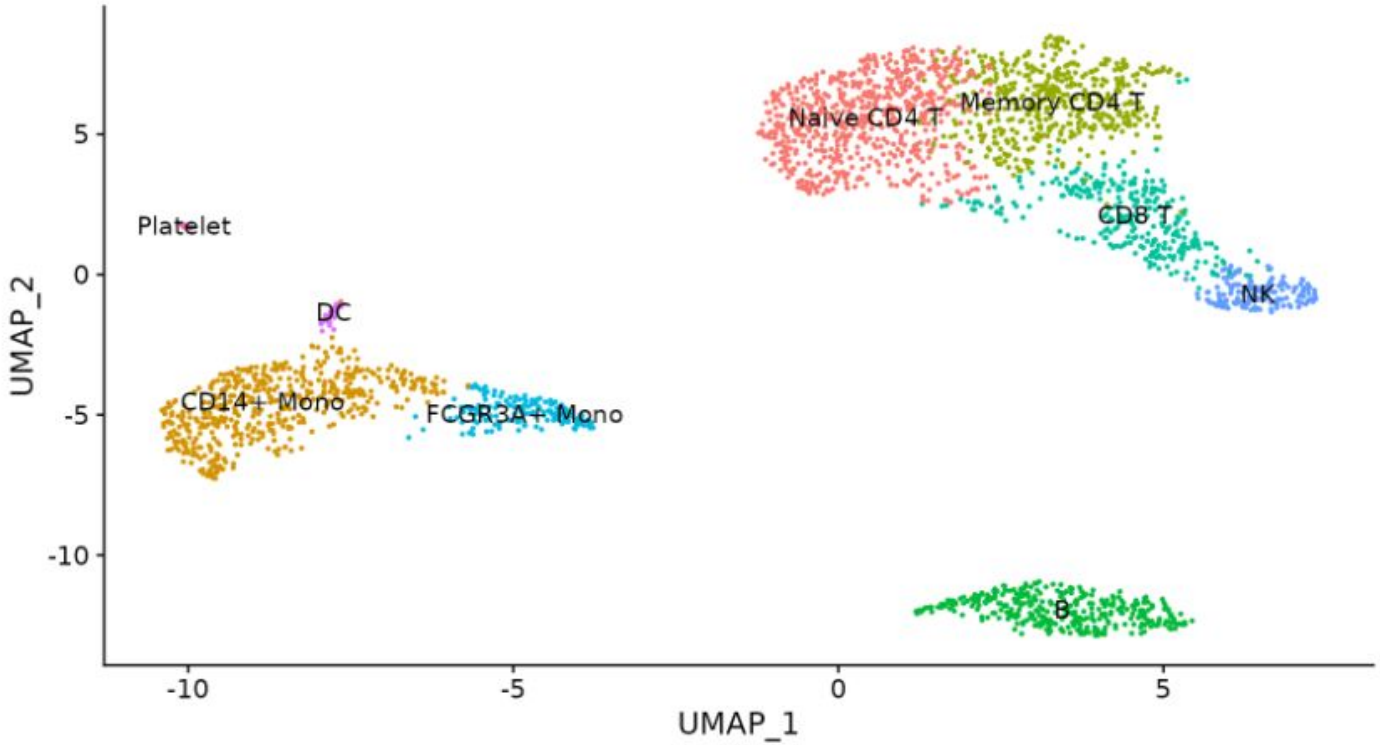
```
pbmc.markers %>%
    group_by(cluster) %>%
    top_n(n = 10, wt = avg_log2FC) -> top10
DoHeatmap(pbmc, features = top10$gene) + NoLegend()
```



Identifying cell types from cluster marker genes: Canonical/known marker genes

Cluster ID	Markers	Cell Type
0	IL7R, CCR7	Naive CD4+ T
1	CD14, LYZ	CD14+ Mono
2	IL7R, S100A4	Memory CD4+
3	MS4A1	B
4	CD8A	CD8+ T
5	FCGR3A, MS4A7	FCGR3A+ Mono
6	GNLY, NKG7	NK
7	FCER1A, CST3	DC
8	PPBP	Platelet

```r
new.cluster.ids <- c("Naive CD4 T", "CD14+ Mono", "Memory CD4 T", "B", "CD8 T", "FCGR3A+ Mono",
    "NK", "DC", "Platelet")
names(new.cluster.ids) <- levels(pbmc)
pbmc <- RenameIdents(pbmc, new.cluster.ids)
DimPlot(pbmc, reduction = "umap", label = TRUE, pt.size = 0.5) + NoLegend()
```



Identifying cell types from cluster marker genes: Cross-referencing with cell type databases

PanglaoDB | Home | Search | Datasets | Tools | Papers | FAQ/Help | About

PanglaoDB is a database for the scientific community interested in exploration of single cell RNA sequencing experiments from mouse and human. We collect and integrate data from multiple studies and present them through a unified framework.

Usage examples

- Run a gene search for SOX2, PECAM1 or ACE2
- Browse the full list of samples
- Explore the list of cell type markers for Schwann cells
- Browse cell types of the mouse retina
- Look at the expression of *CRX* in photoreceptor cells
- Find cell clusters where both *PECAM1* and *VCAM1* are expressed using a boolean search with the 'and' operator
- Find quiescent neural stem cells using AND+NOT

Database statistics

	Mus musculus	*Homo sapiens*
Samples	1063	305
Tissues	184	74
Cells	4,459,768	1,126,580
Clusters	8,651	1,748

Dataset of the day

https://panglaodb.se/

UC San Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Automated tools for cell type identification

- Useful for a starting point, but often need to be refined with input from researcher (using known marker genes for expected cell types, subtypes, etc). Often work best with large, well-defined cell types (fibroblasts, macrophages, t-cells, etc). Less useful for smaller, more novel cell subtypes.

Article | Published: 14 January 2019

Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage

Dvir Aran, Agnieszka P. Looney, Leq
Naikawadi, Paul J. Wolters, Adam R.

Nature Immunology **20**, 163–172 (2

Article | Open access | Published: 10 March 2022

Fully-automated and ultra-fast cell-type identification using specific marker combinations from single-cell transcriptomic data

Aleksandr Ianevski, Anil K. Giri & Tero Aittokallio

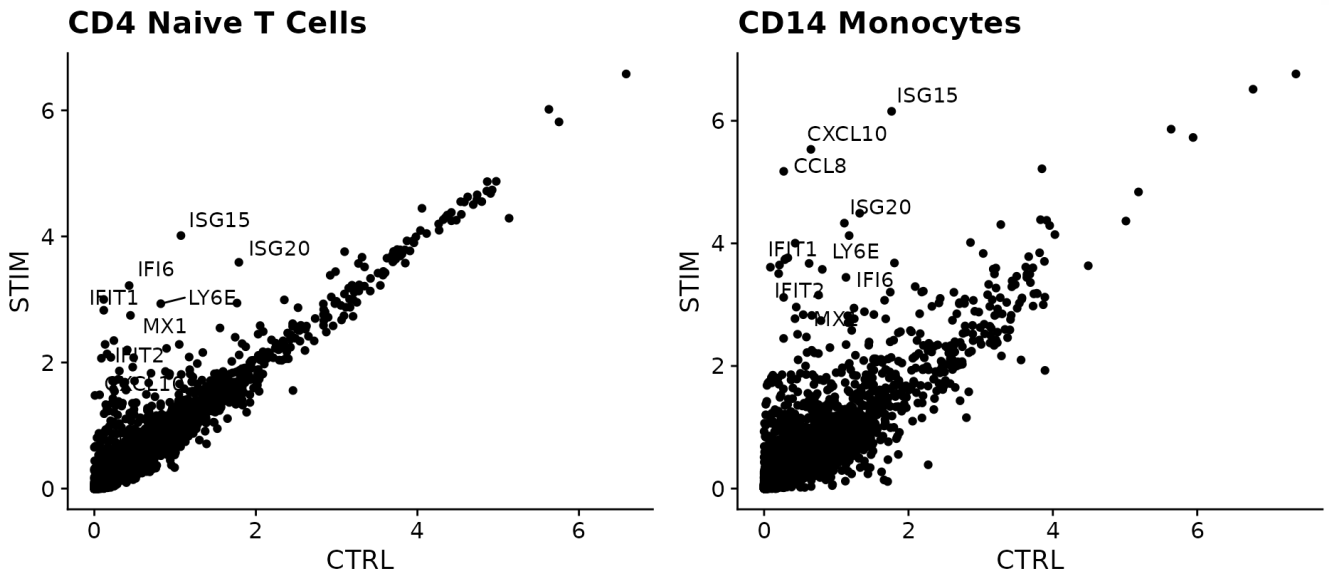
Nature Communications **13**, Article number: 1246 (2022) | Cite this article

Pseudobulk analysis for identifying differentially expressed genes by condition, within clusters

Confronting false discoveries in single-cell differential expression

Jordan W. Squair[1,2,3], Matthieu Gautier[1,2], Claudia Kathe[1,2], Mark A. Anderson[1,2], Nicholas D. James[1,2], Thomas H. Hutson[1,2], Rémi Hudelle[1,2], Taha Qaiser[3], Kaya J. E. Matson[4], Quentin Barraud[1,2], Ariel J. Levine[4], Gioele La Manno[1], Michael A. Skinnider[1,2,5,6]✉ & Grégoire Courtine[1,2,6]✉

Differential expression analysis in single-cell transcriptomics enables the dissection of cell-type-specific responses to perturbations such as disease, trauma, or experimental manipulations. While many statistical methods are available to identify differentially expressed genes, the principles that distinguish these methods and their performance remain unclear. Here, we show that the relative performance of these methods is contingent on their ability to account for variation between biological replicates. Methods that ignore this inevitable variation are biased and prone to false discoveries. Indeed, the most widely used methods can discover hundreds of differentially expressed genes in the absence of biological differences. To exemplify these principles, we exposed true and false discoveries of differentially expressed genes in the injured mouse spinal cord.



→ aggregate counts to sample level before running comparison to control false discoveries

UC San Diego
Altman Clinical and Translational Research Institute

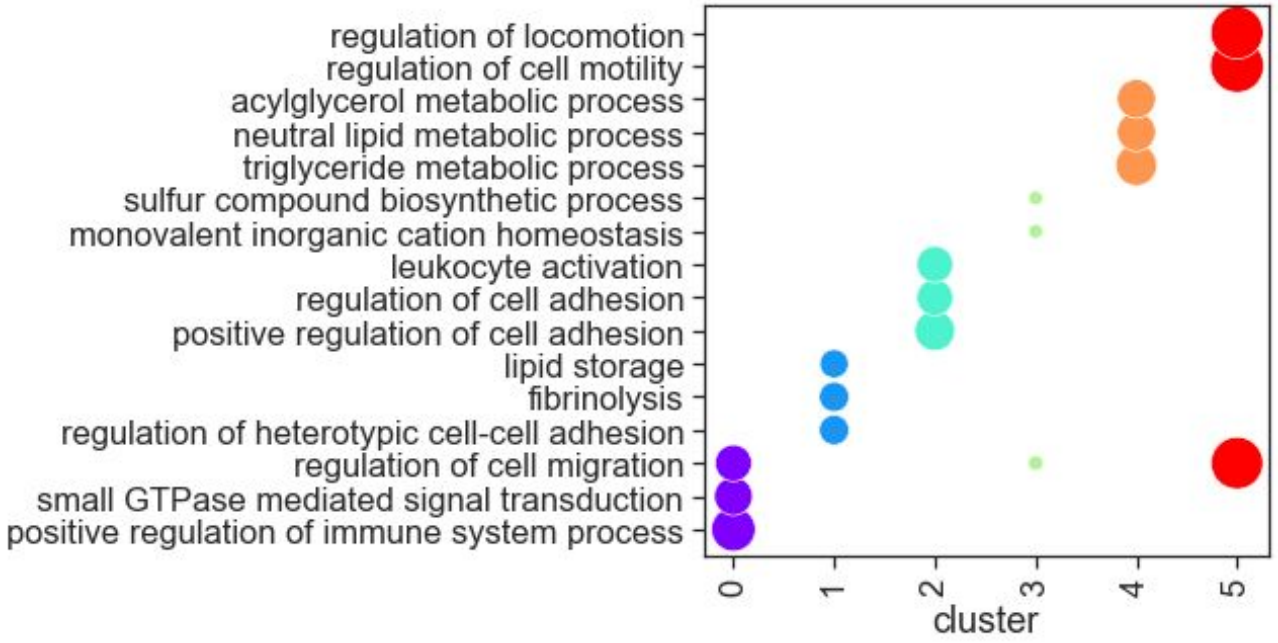
CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Pathway analysis

Investigate enrichment of cluster marker genes with predefined pathways/biological processes/ gene sets

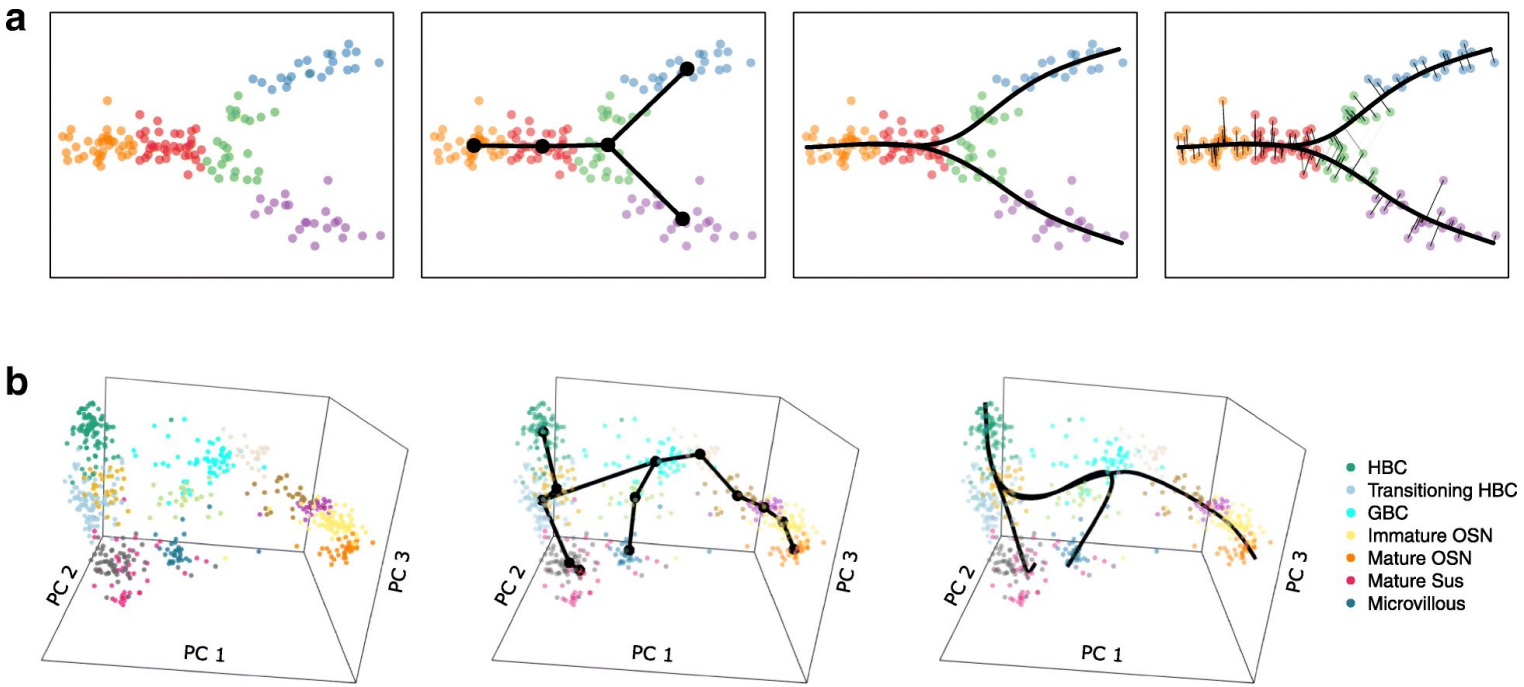
Common tools:

- gprofiler
- toppgene
- GSEA
- enrichR



Pseudotime analysis with slingshot

- Identify lineages and branch points



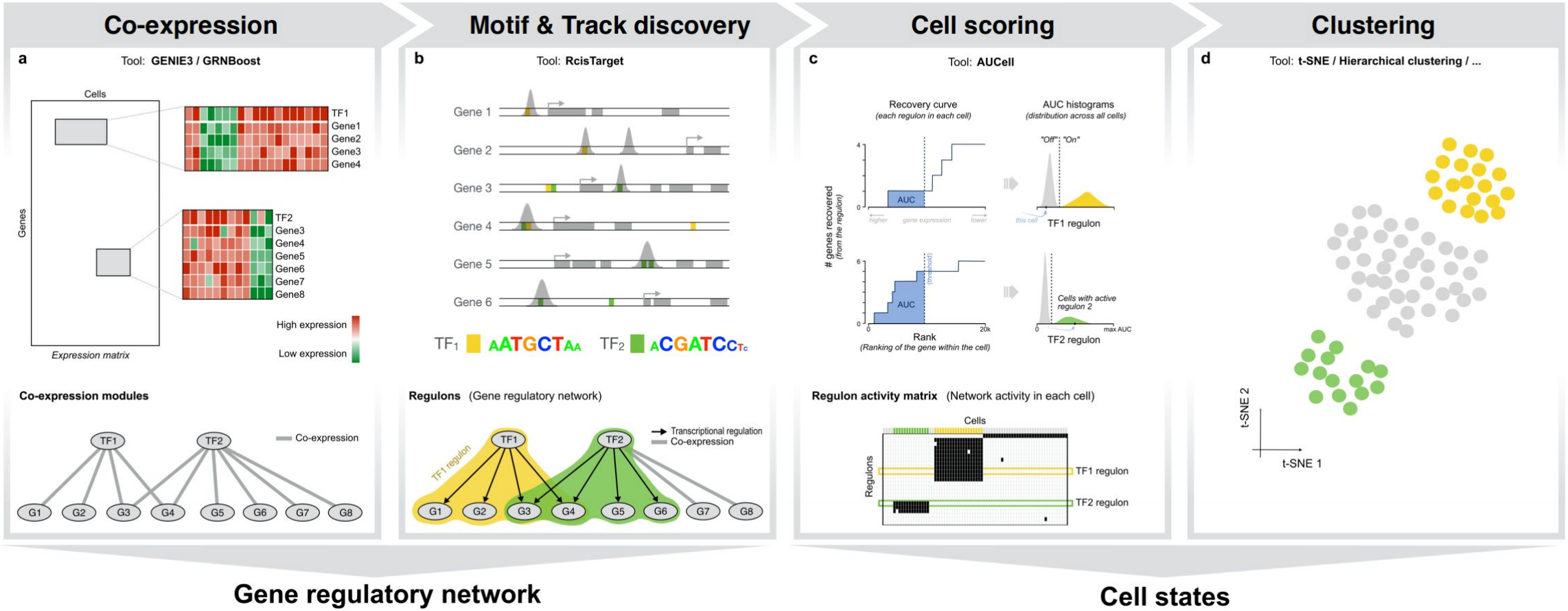
Street, Kelly, et al. "Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics." *BMC genomics* 19.1 (2018): 1-16.

Detailed usage instructions here:
https://bioconductor.org/packages/devel/bioc/vignettes/slingshot/inst/doc/vignette.html

UC San Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Estimating regulon activity with pyscenic



Van de Sande, Bram, et al. "A scalable SCENIC workflow for single-cell gene regulatory network analysis." *Nature Protocols* 15.7 (2020): 2247-2276.

UC San Diego
Altman Clinical and Translational Research Institute

CCBB CENTER FOR COMPUTATIONAL BIOLOGY & BIOINFORMATICS

Thank you!

Now on to the interactive part!