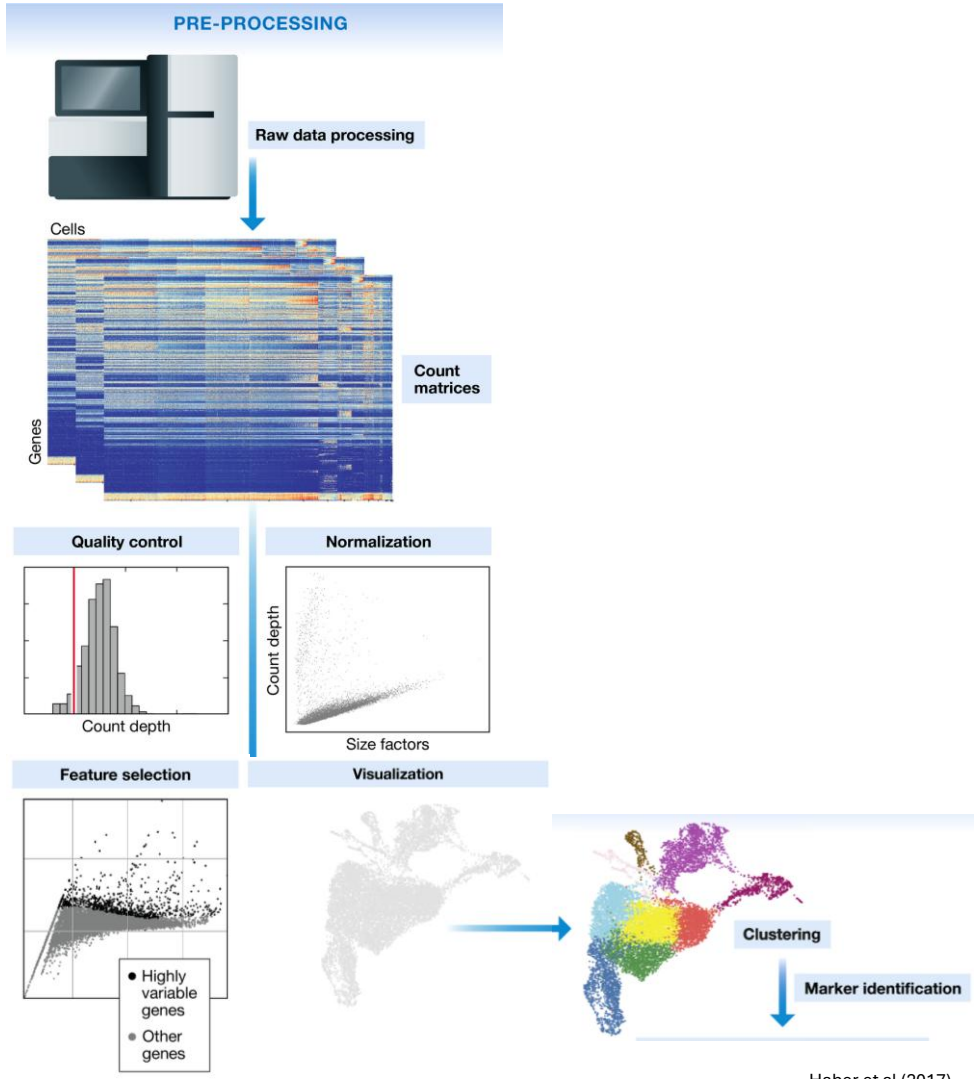
Functional Analysis I

Theoretical Part

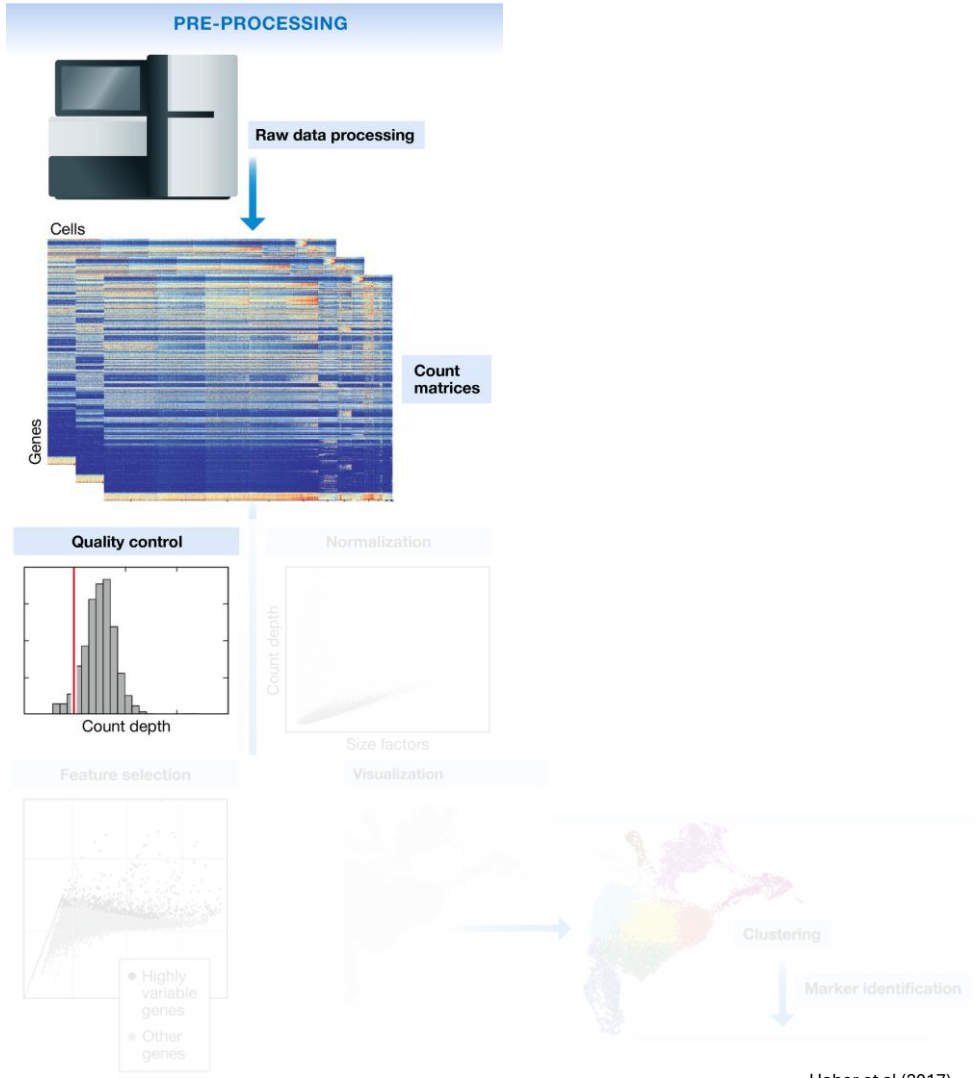


Haber et al (2017)

Count matrices have the dimension: number of feature (number of transcripts, etc) x barcodes.

Containers are organized by keys.

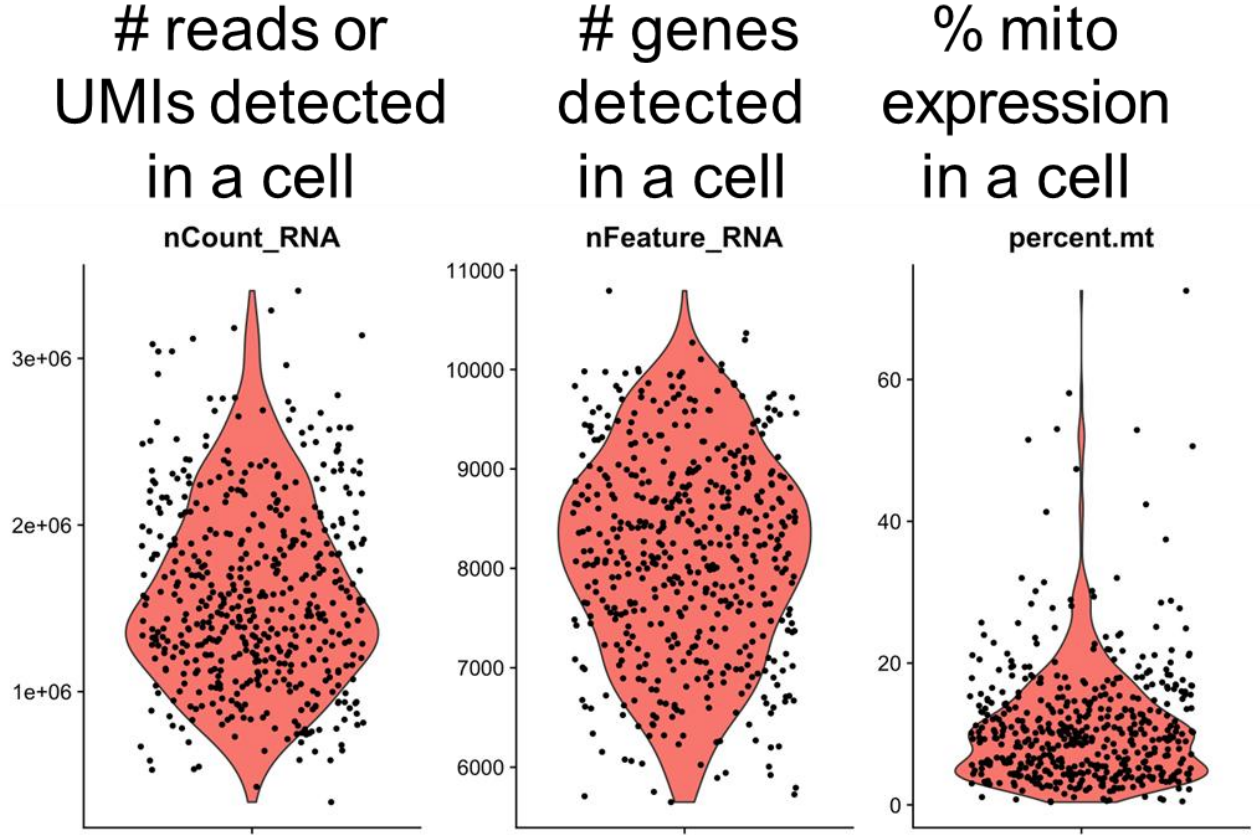
Quality control



Haber et al (2017)

We filter cells based on technical or biological parameters. So, get viable cells before analyzing:

- The number of counts per barcode (count depth)
- The number of genes per barcode
- The fraction of counts from mitochondrial genes per barcode

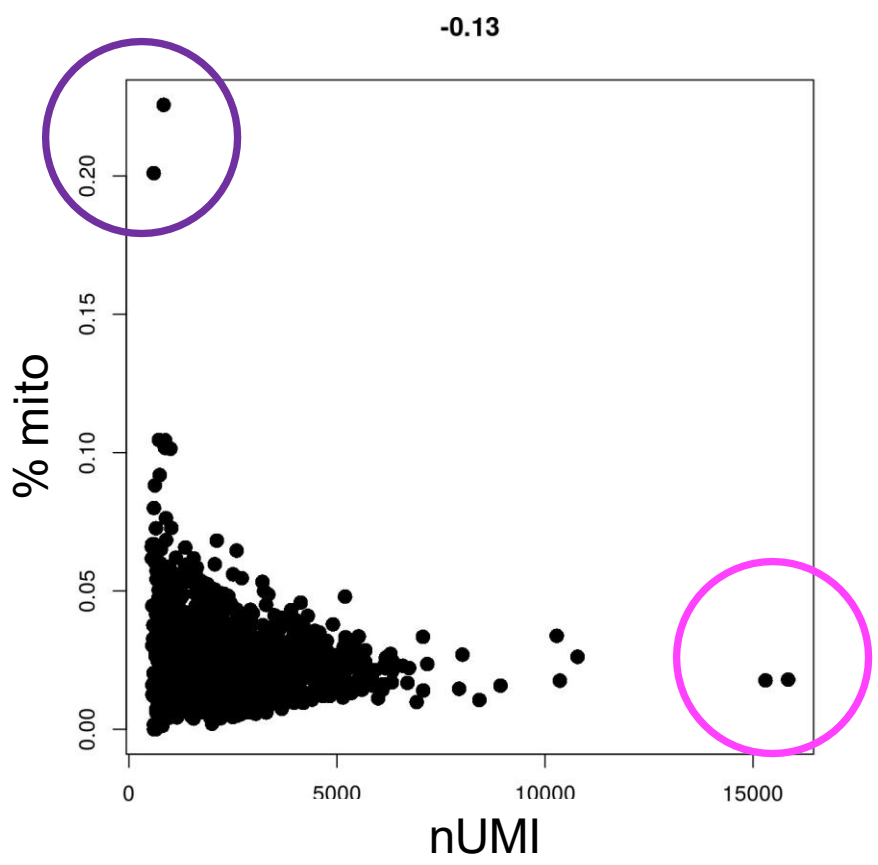


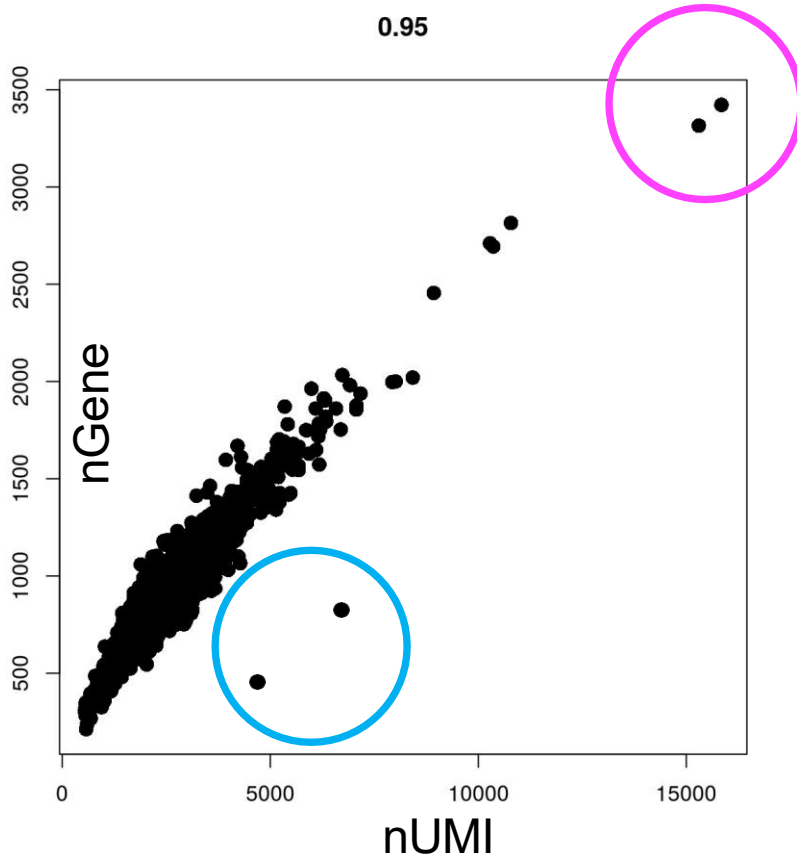
There are many quality control filters for genes and cells

Low nUMI and high %Mt:

High nUMI & low nGene ratio:

High nUMI & high nGene:





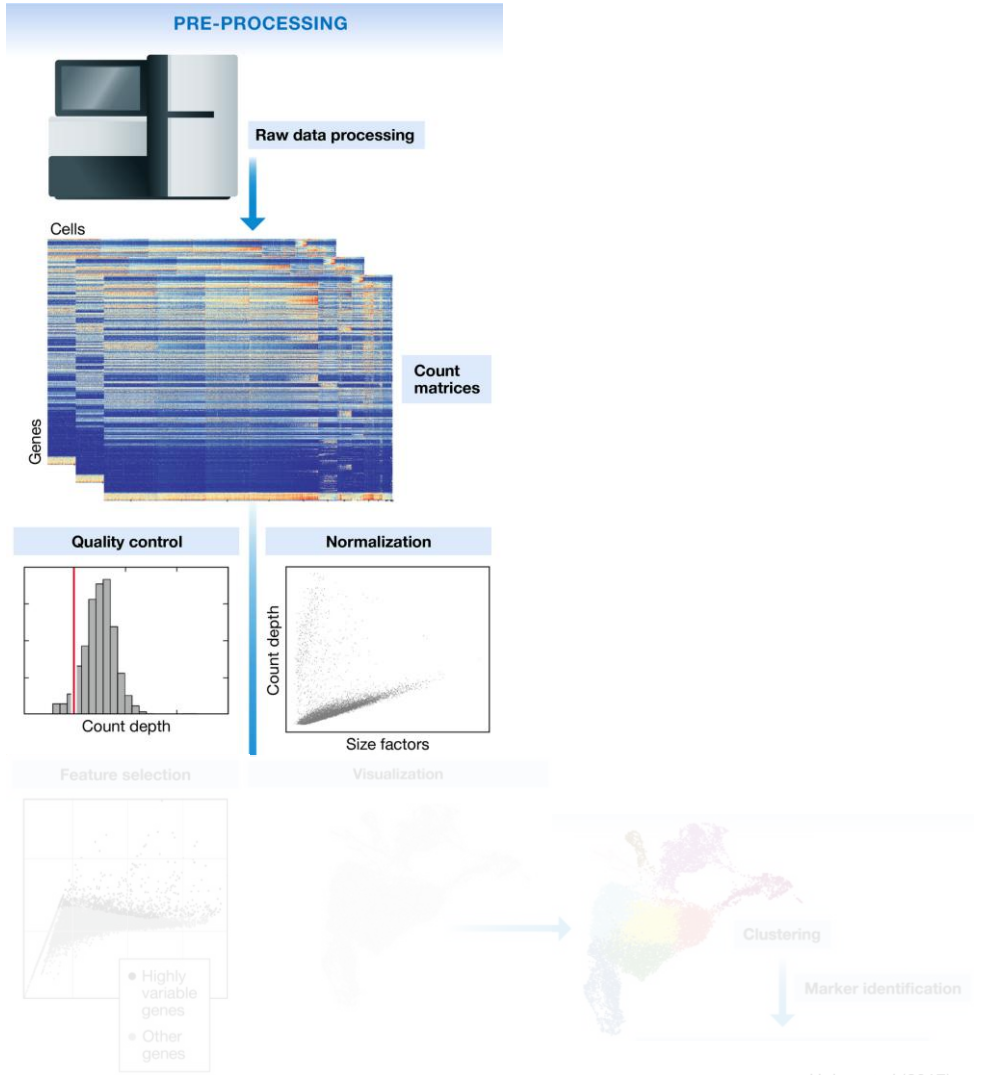
Filtering with combinations of quality control filters

1. Different platforms set different expectations
   - Smart-Seq2 often yields more genes detected per cell than 10x Chromium.

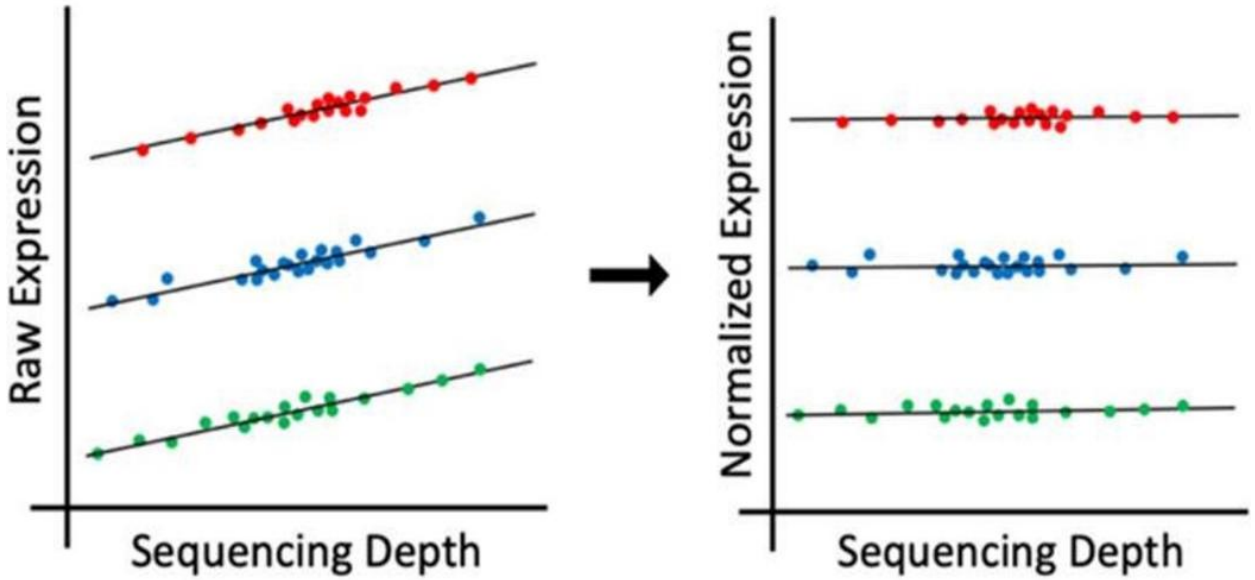
2. Different cell types set different expectations
   - Immune cells normally have fewer genes detected per cell than non-immune cells
   - Malignant cells normally have more genes detected per cell than non-malignant cells

3. Different tools help with filtering
   - Detecting doublets (Scrublet) or Detecting ambient RNA (CellBender)

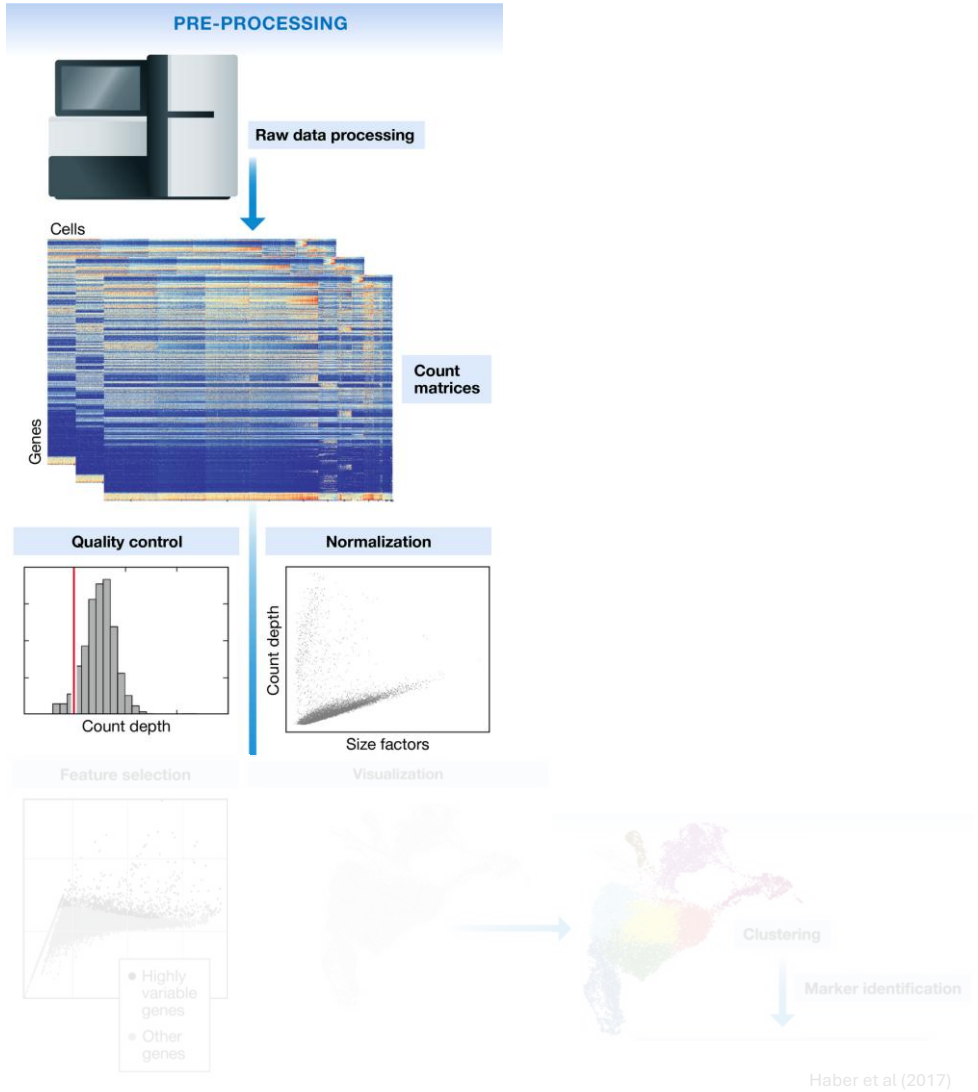
Normalization



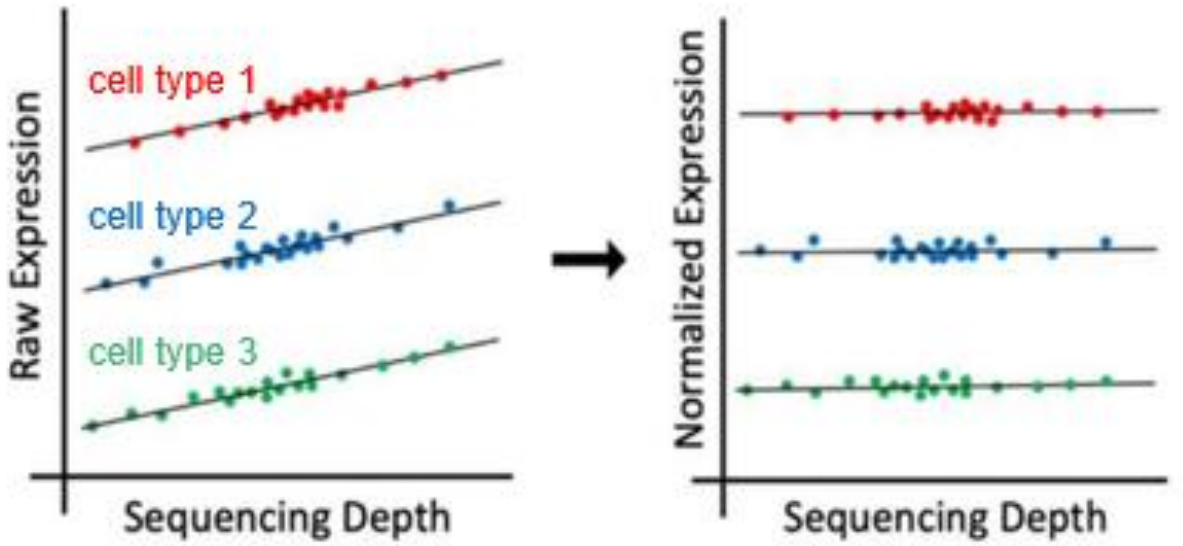
- No normalize gene expression within a cell can lead to false differences in gene expression.



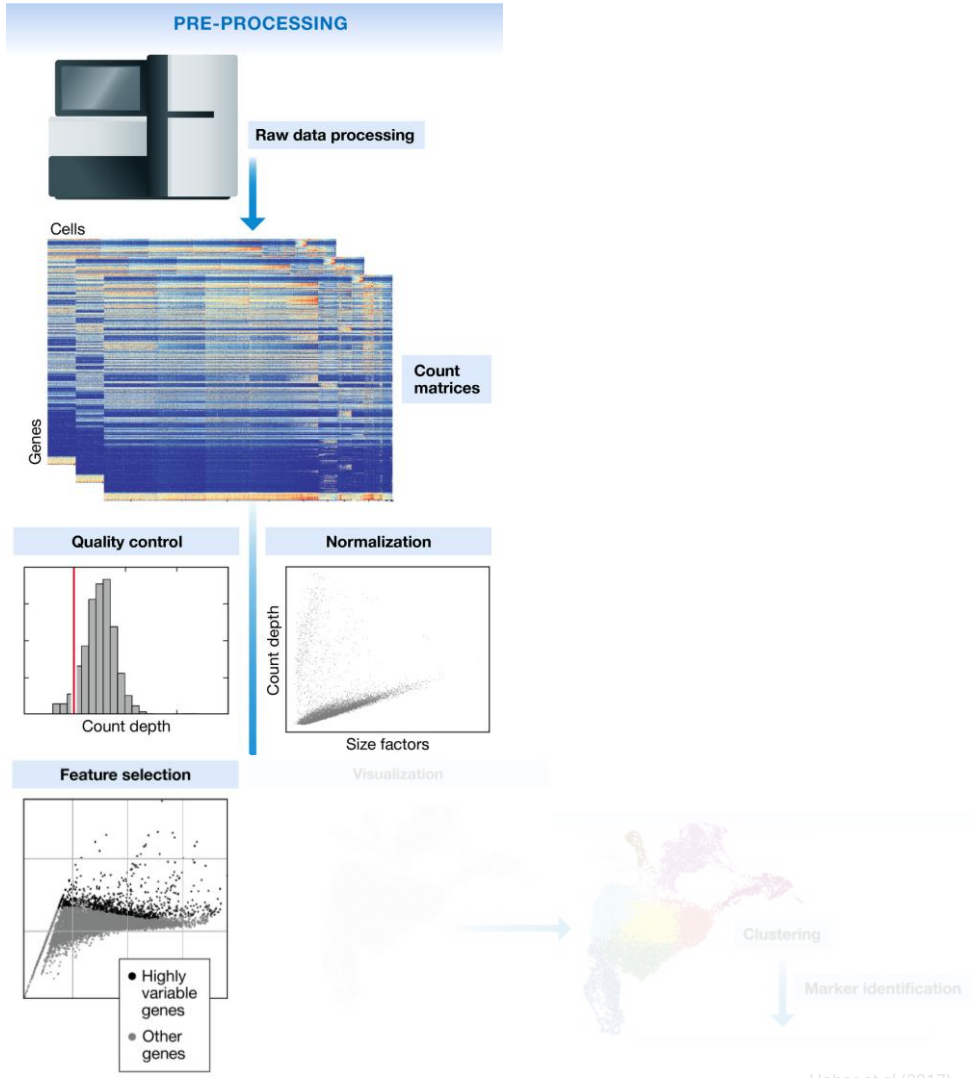
Normalization



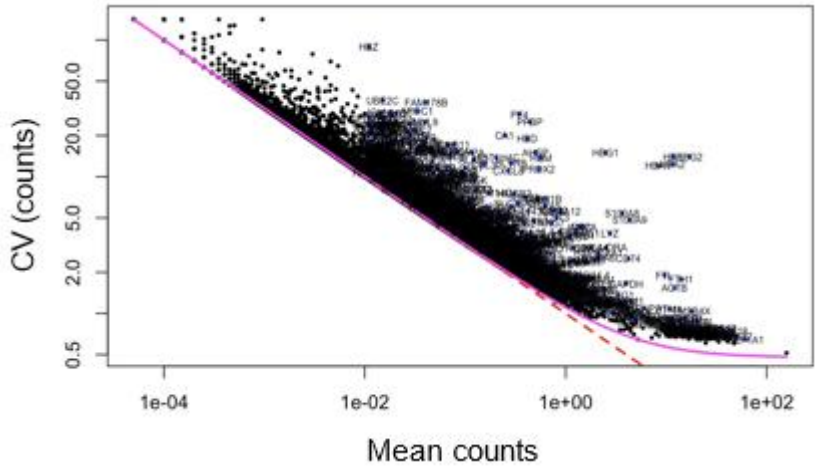
- More highly expressed genes are more variable.
- Variance is a measure of how much the data spread out around the ~mean.
- When it normalizes, this variance is stabilized (genes with high counts don't look more variable just because they have larger numbers)



Dimensionality reduction and clustering

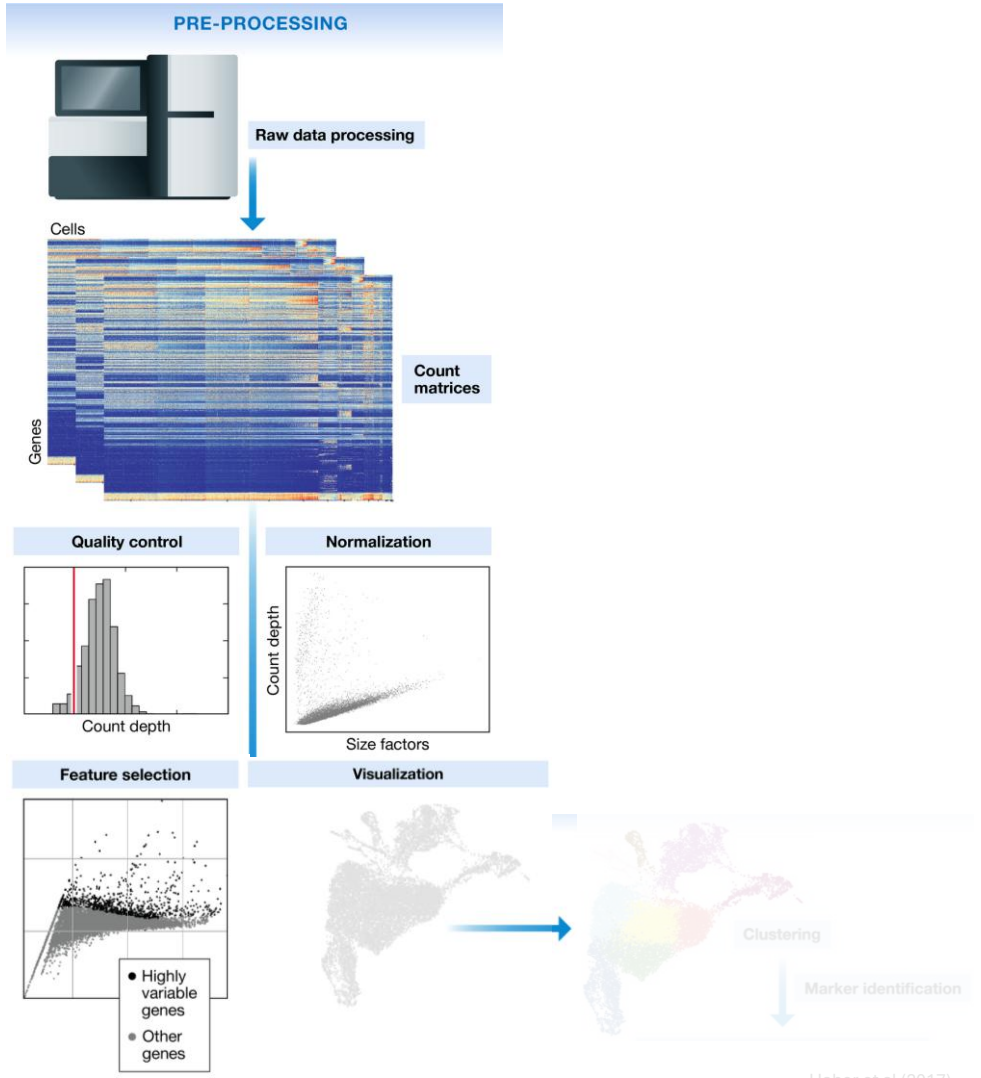


Cells are in ~10K dimensional space (one dimension for each gene)

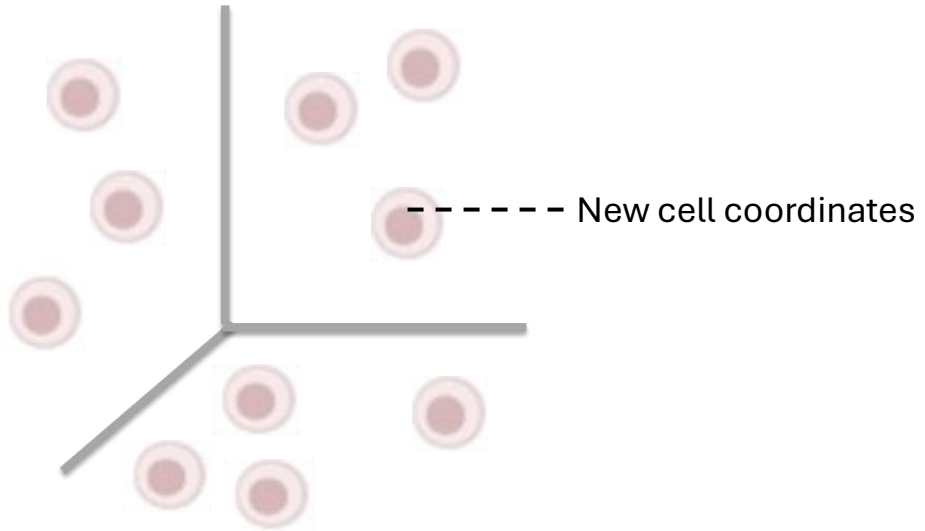


Find genes (features) that are outliers in a plot of mean of gene expression vs variance of gene expression (after Normalizing)

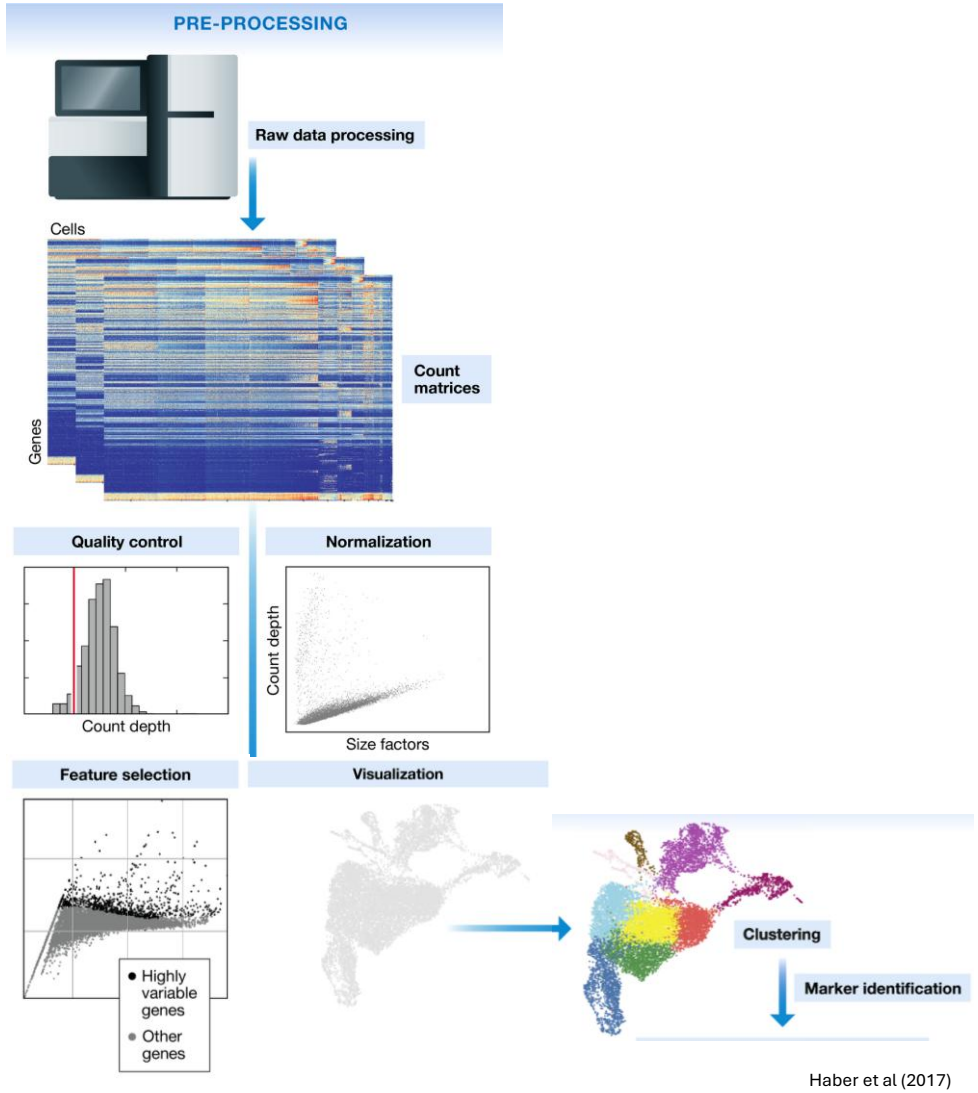
Dimensionality reduction and clustering



- After choosing HVGs, scaling each gene across cells (turns the values into z-scores so all genes are on the same scale).
- We then use this new scaled matrix to compute the PCs (Principal Components).
- The PCA, it analyzes and captures the main global variation and dividing in new dimensions (PCs).



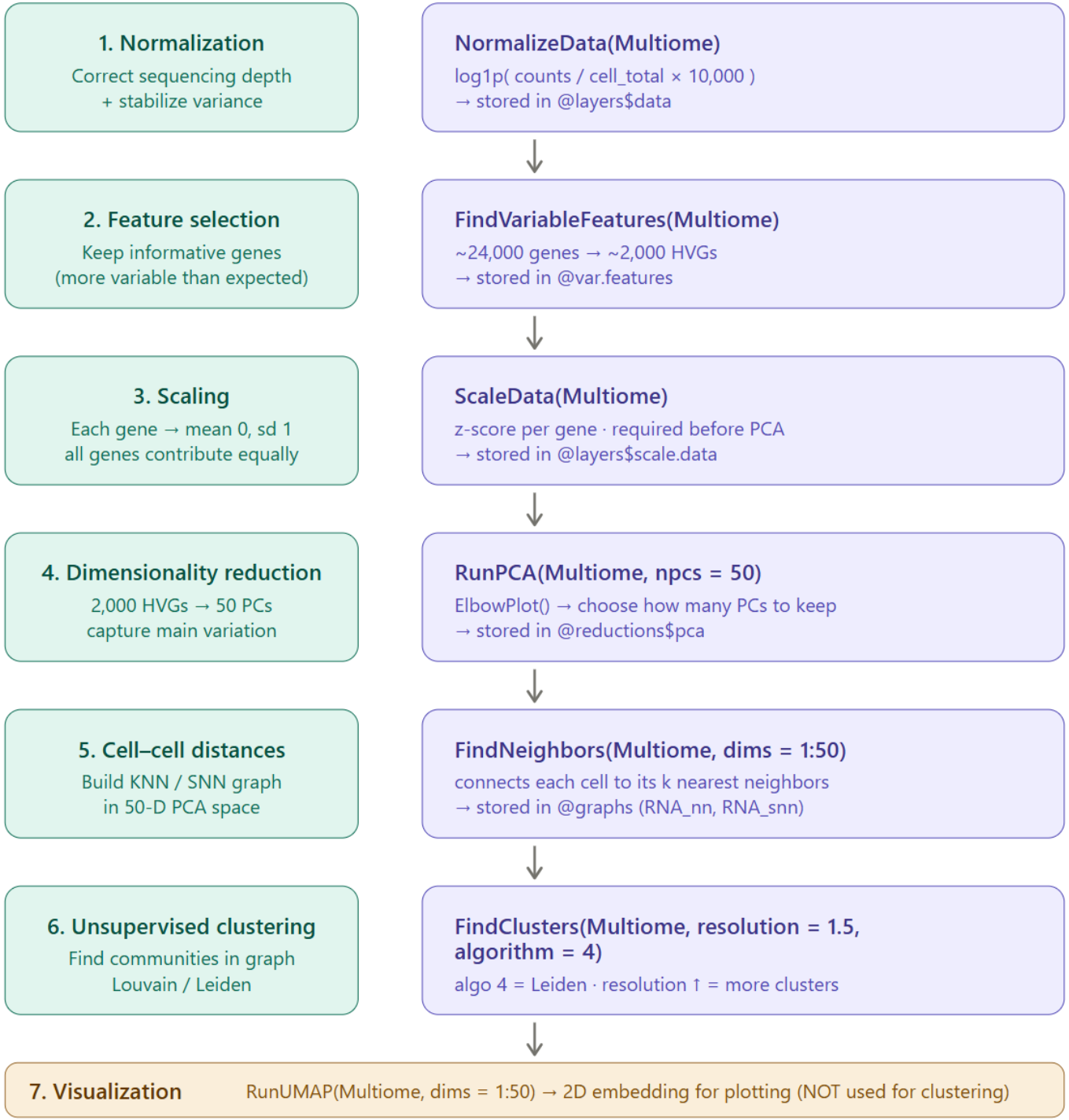
Dimensionality reduction and clustering



Haber et al (2017)

- Shows which cells are close to each other and forms “unsupervised clustering”.
- FindNeighbors (default K = 20) results in KNN and SNN.
  - ✓ KNN (K-Nearest Neighbors) graphs by finding, for each cell, its K closest neighbors in the PC space (its individual coordinate).
  - ✓ While, shared nearest neighbor (SNN) graph giving weighted connections instead of binary ones.
- A community detection algorithm (Louvain/Leiden) uses the KNN/SNN graph.
- UMAP just visualizes, it’s the projection, and it looks at local relationships.

Hands on Section


1. Normalization
Correct sequencing depth
+ stabilize variance
NormalizeData(Multiome)
log1p(counts / cell_total × 10,000)
→ stored in @layers$data
2. Feature selection
Keep informative genes
(more variable than expected)
FindVariableFeatures(Multiome)
~24,000 genes → ~2,000 HVGs
→ stored in @var.features
3. Scaling
Each gene → mean 0, sd 1
all genes contribute equally
ScaleData(Multiome)
z-score per gene · required before PCA
→ stored in @layers$scale.data
4. Dimensionality reduction
2,000 HVGs → 50 PCs
capture main variation
RunPCA(Multiome, npcs = 50)
ElbowPlot() → choose how many PCs to keep
→ stored in @reductions$pca
5. Cell–cell distances
Build KNN / SNN graph
in 50-D PCA space
FindNeighbors(Multiome, dims = 1:50)
connects each cell to its k nearest neighbors
→ stored in @graphs (RNA_nn, RNA_snn)
6. Unsupervised clustering
Find communities in graph
Louvain / Leiden
FindClusters(Multiome, resolution = 1.5,
algorithm = 4)
algo 4 = Leiden · resolution ↑ = more clusters
7. Visualization
RunUMAP(Multiome, dims = 1:50) → 2D embedding for plotting (NOT used for clustering)