Normalization, PCA, UMAP, K-Means Clustering, K Nearest Neighbors

2025-04-24

```
library(Seurat)
```

```
## Attaching SeuratObject
```

```
## Attaching sp
```

```
library(ggplot2)
library(sctransform)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tibble)
```

```
metadata <- read.csv(
  file = "./metadata.csv"
)
```

```
counts <- read.csv(
  file = "./20200513_Mouse_PatchSeq_Release_count.v2.csv",
  row.names = 1,           # if first column has gene names
  check.names = FALSE      # so column names stay as barcodes
)

# 2. Convert to matrix (optional but recommended)
counts_mat <- as.matrix(counts)

# 3. Create a Seurat object
seurat_obj <- CreateSeuratObject(
  counts = counts_mat,
  project = "MousePatchSeq"
)
```

```
## Warning: Feature names cannot have underscores ('_'), replacing with dashes
## ('-')
```

```
metadata <- metadata %>%
  mutate(Transcriptomics_id = as.character(Transcriptomics_id)) %>%   # safety
  column_to_rownames("Transcriptomics_id")

common <- intersect(colnames(seurat_obj), rownames(metadata))
```

```
seurat_obj <- subset(seurat_obj, cells = common)

metadata <- metadata[colnames(seurat_obj), , drop = FALSE]

seurat_obj <- AddMetaData(seurat_obj, metadata = metadata)
```

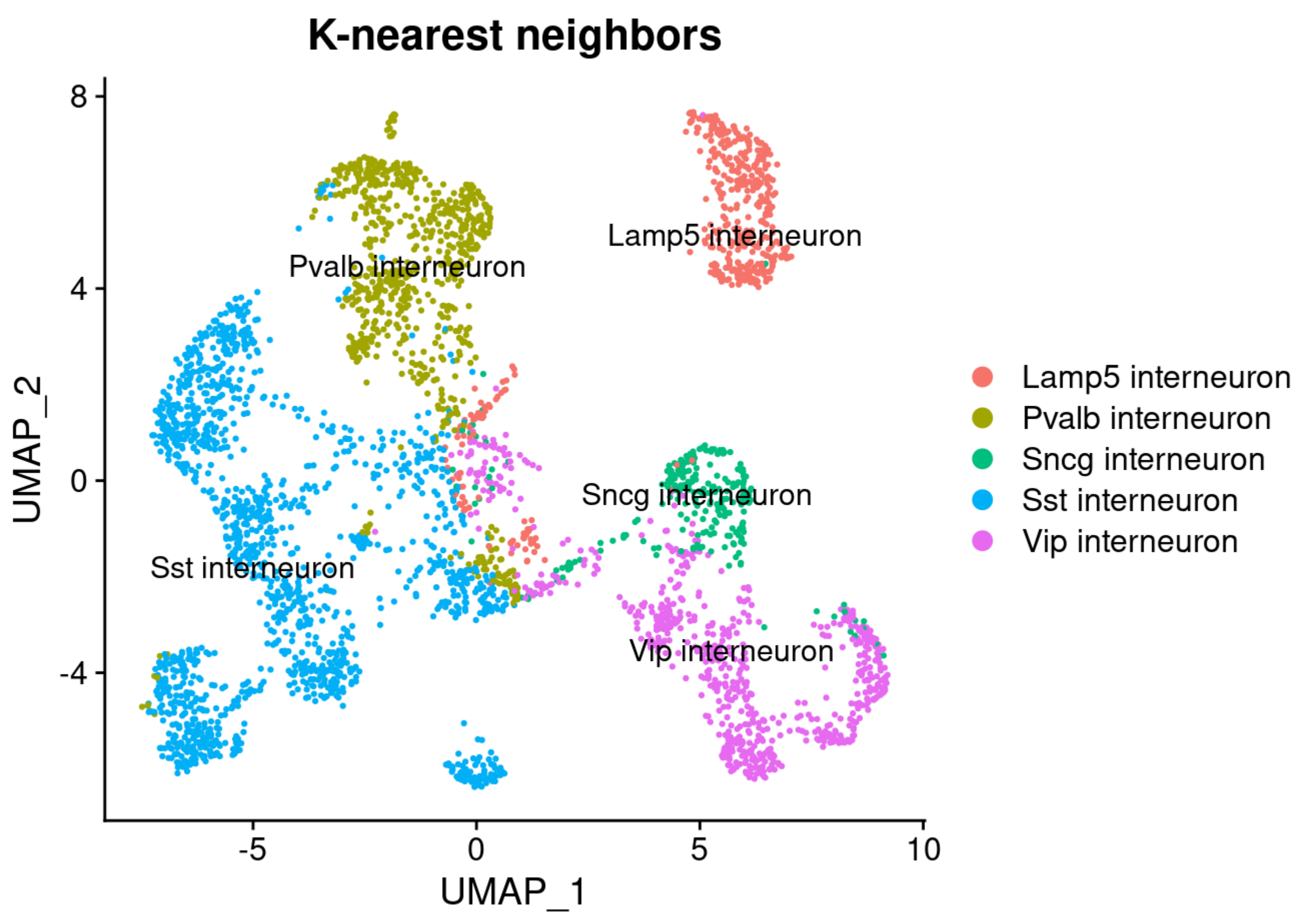
```
# run sctransform, how does it work?
mps <- SCTransform(seurat_obj, verbose = FALSE)
```

```
# These are now standard steps in the Seurat workflow for visualization and clusterin
g
umap_mps <- RunPCA(mps, verbose = FALSE)
umap_mps <- RunUMAP(umap_mps, dims = 1:30, verbose = FALSE)
```

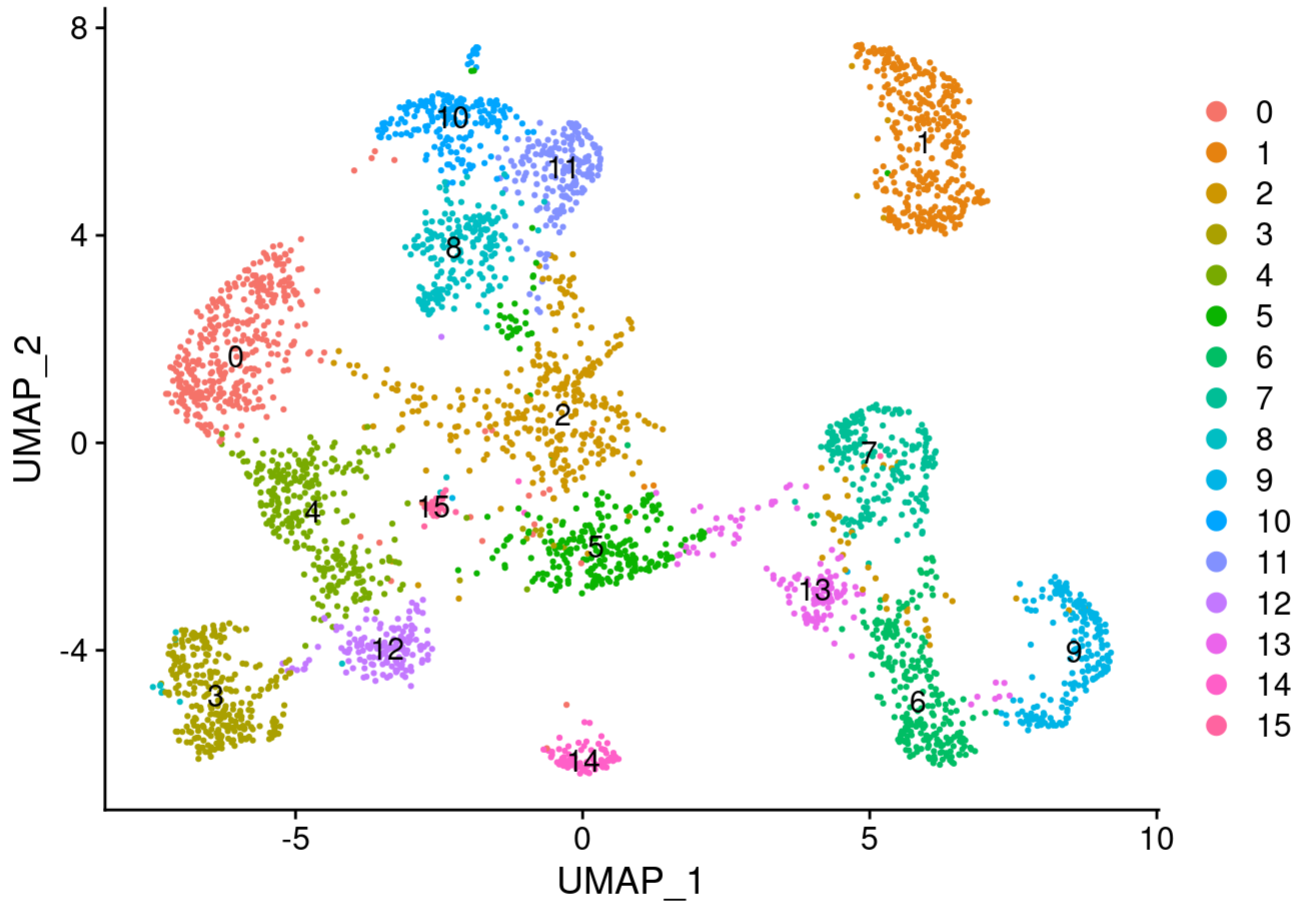
```
## Warning: The default method for RunUMAP has changed from calling Python UMAP via r
eticulate to the R-native UWOT using the cosine metric
## To use Python UMAP via reticulate, set umap.method to 'umap-learn' and metric to
'correlation'
## This message will be shown once per session
```

```
umap_mps <- FindNeighbors(umap_mps, dims = 1:30, verbose = FALSE)
umap_mps <- FindClusters(umap_mps, verbose = FALSE)
```

```
DimPlot(umap_mps, group.by = "Cell.Type", label = TRUE) + labs(title = "K-nearest nei
ghbors")
```



```
DimPlot(umap_mps, label = TRUE)
```



```
pca_mps <- RunPCA(mps, verbose = FALSE)
pca_embeddings <- Embeddings(pca_mps, "pca")
#    This is an n_cells × n_pcs matrix

# 3. Select how many PCs you want to cluster on (e.g. first 10)
pc_choice <- 1:30
pca_data <- pca_embeddings[, pc_choice]

# 4. Run k-means
set.seed(42)                       # for reproducibility
k <- 5                             # choose number of clusters
kmeans_res <- kmeans(pca_data, centers = k, nstart = 25)
```

```
pca_mps$kmeans_clusters <- as.factor(kmeans_res$cluster)

# 6. (Optional) Visualize clusters on UMAP/t-SNE
pca_mps <- RunUMAP(pca_mps, dims = pc_choice)
```

```
## 19:35:50 UMAP embedding parameters a = 0.9922 b = 1.112
```

```
## 19:35:50 Read 3600 rows and found 30 numeric columns
```

```
## 19:35:50 Using Annoy for neighbor search, n_neighbors = 30
```

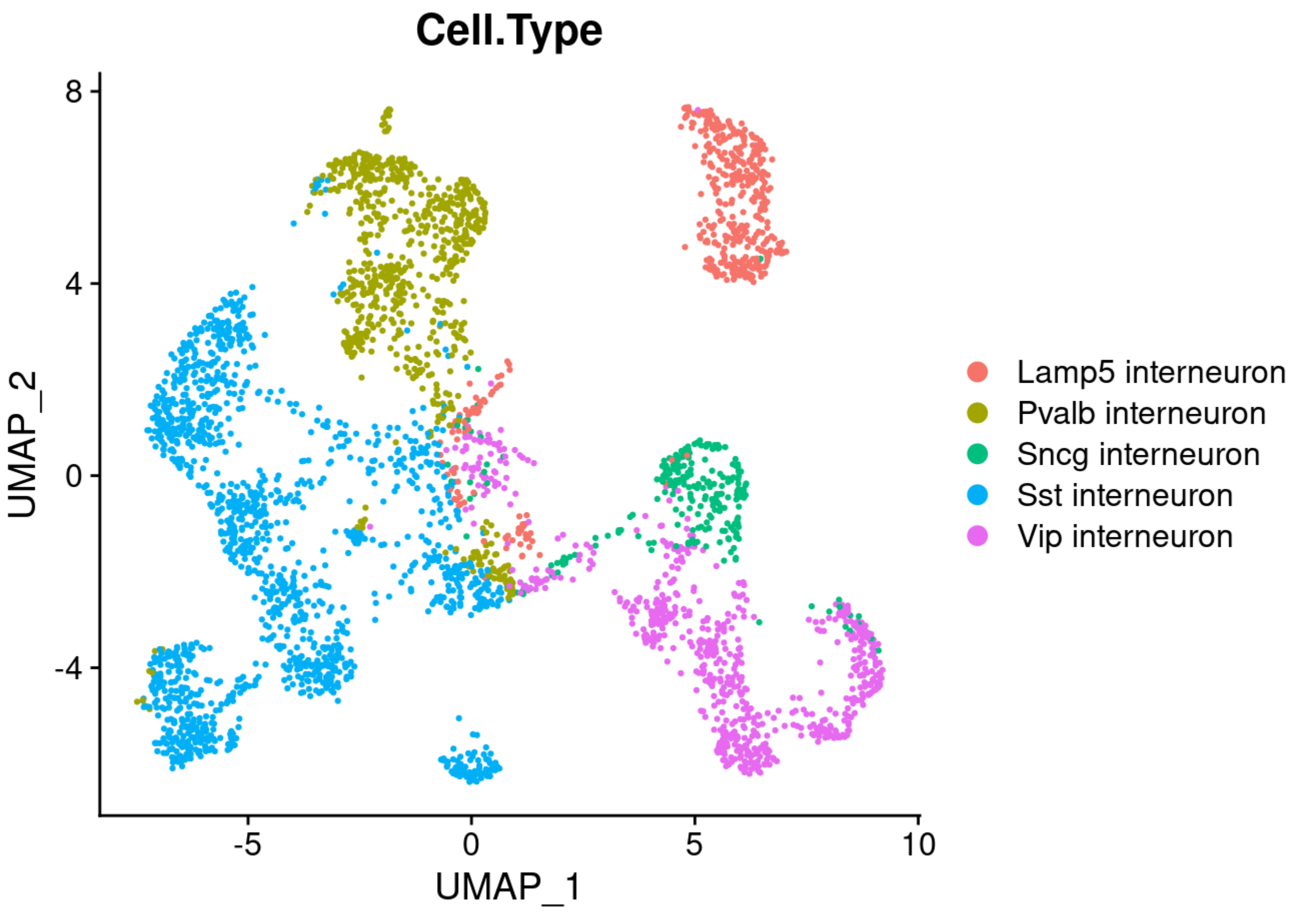
```
## 19:35:50 Building Annoy index with metric = cosine, n_trees = 50
```

```
## 0%   10   20   30   40   50   60   70   80   90   100%
```

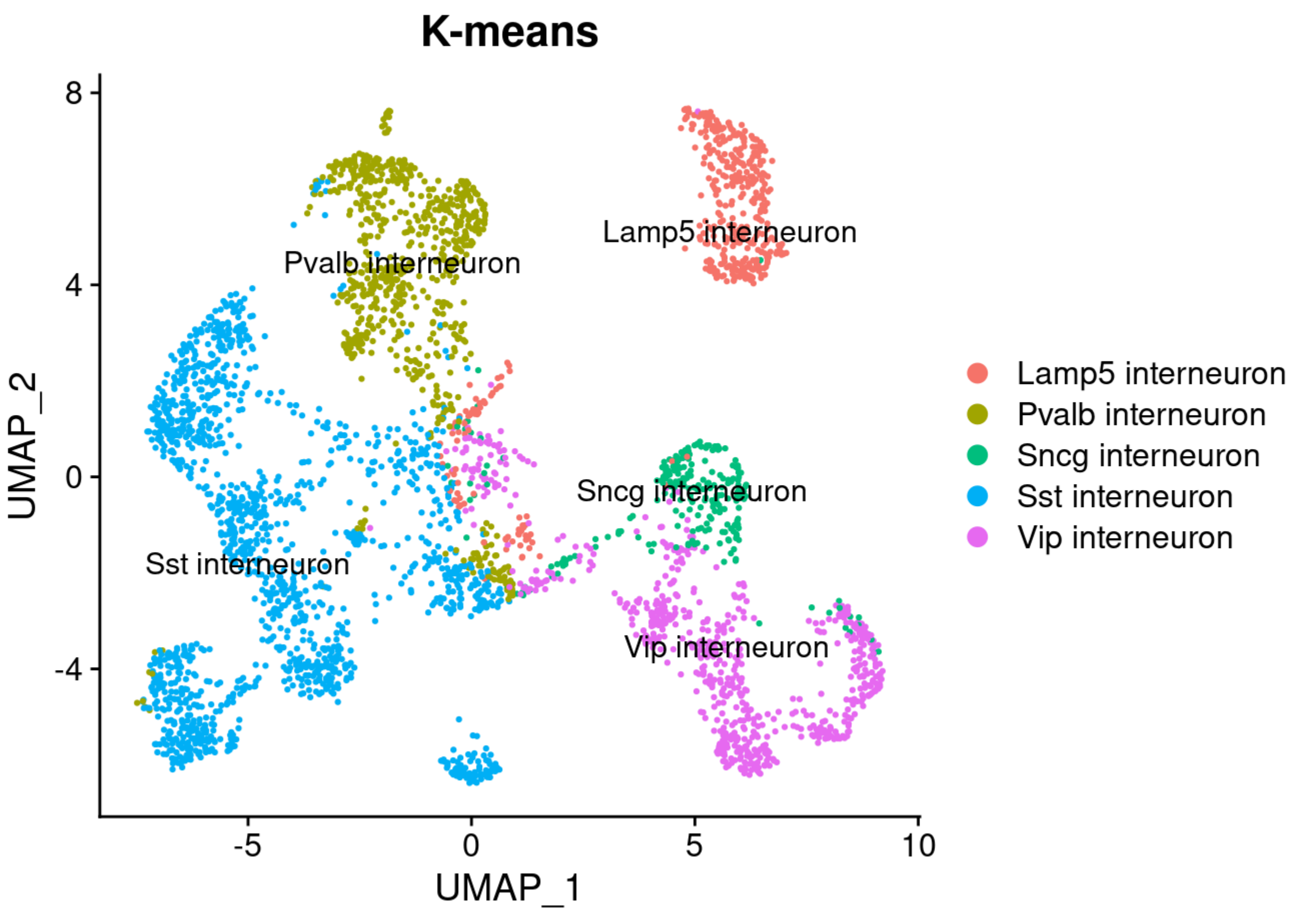
```
## [----|----|----|----|----|----|----|----|----|----|
```

```
## **************************************************|
## 19:35:50 Writing NN index file to temp file /tmp/Rtmpm3vk0q/file17386f34e4f056
## 19:35:50 Searching Annoy index using 1 thread, search_k = 3000
## 19:35:51 Annoy recall = 100%
## 19:35:51 Commencing smooth kNN distance calibration using 1 thread
## 19:35:52 Initializing from normalized Laplacian + noise
## 19:35:52 Commencing optimization for 500 epochs, with 147760 positive edges
## 19:35:56 Optimization finished
```

```
DimPlot(pca_mps, group.by = "Cell.Type", reduction = "umap")
```



```
DimPlot(pca_mps, group.by = "Cell.Type", reduction = "umap", label=TRUE) + labs(title
= "K-means")
```


K-means
UMAP_2
UMAP_1
Lamp5 interneuron
Pvalb interneuron
Sncg interneuron
Sst interneuron
Vip interneuron