midterm_646

Sai Sushma Mutyala

2025-03-19

Problem (1)

```r
library(Seurat)
```

```
## Warning: package 'Seurat' was built under R version 4.4.2
## Loading required package: SeuratObject
## Warning: package 'SeuratObject' was built under R version 4.4.2
## Loading required package: sp
## Warning: package 'sp' was built under R version 4.4.2
##
## Attaching package: 'SeuratObject'
## The following objects are masked from 'package:base':
##
##     intersect, t
```

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```r
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.4.3
##
## 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
```

```r
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.4.2
##
## Attaching package: 'data.table'
## The following objects are masked from 'package:dplyr':
##
```

```
##     between, first, last
```

```r
# problem 1 for seger data
file_path_counts <- "C:/Users/sushm/Downloads/E-MTAB-5061/pancreas_refseq_rpkms_counts_3514sc.txt"
file_path_meta <- "C:/Users/sushm/Downloads/E-MTAB-5061/E-MTAB-5061.sdrf.txt"
first_line <- readLines(file_path_counts, n = 1)
cell.barcodes <- unlist(strsplit(first_line, "\t"))[-1]
count <- fread(file_path_counts, sep = "\t", header = TRUE, data.table = FALSE)
gene.meta <- count[, 1:2]
count <- count[, -c(1,2)]
if (ncol(count) >= 7028) count <- as.matrix(count[, 3515:7028])
rownames(count) <- gene.meta[, 1]
if (ncol(count) == length(cell.barcodes)) colnames(count) <- cell.barcodes
rm(gene.meta, cell.barcodes)
cell.meta <- read.csv(file_path_meta, sep = "\t", header = TRUE, check.names = FALSE)
if (ncol(cell.meta) >= 6) cell.meta <- cell.meta[, 1:6]
rownames(cell.meta) <- cell.meta$`Source Name`
cell.meta <- cell.meta[match(colnames(count), rownames(cell.meta)), ]
print(all(colnames(count) == rownames(cell.meta)))
```

```
## [1] TRUE
```

```r
col_name_individual <- "Characteristics [individual]"
cell.meta[[col_name_individual]] <- trimws(cell.meta[[col_name_individual]])
print(unique(cell.meta[[col_name_individual]]))
```

```
##  [1] "H2"   "T2D1" "H1"   "T2D2" "H3"   "H4"   "H5"   "H6"   "T2D3" "T2D4"
```

```r
healthy_ids <- c("H1", "H2", "H3", "H4", "H5", "H6")
healthy_cells <- cell.meta[[col_name_individual]] %in% healthy_ids
cell.meta <- cell.meta[healthy_cells, ]
count <- count[, healthy_cells]
print(ncol(count))
```

```
## [1] 1980
```

```r
print(sum(duplicated(rownames(count))))
```

```
## [1] 745
```

```r
count <- count[!duplicated(rownames(count)), ]
dim(count)
```

```
## [1] 25525  1980
```

```r
dim(cell.meta)
```

```
## [1] 1980    6
```

```r
seger <- CreateSeuratObject(counts = count, project = "seger", min.cells = 2, min.features = 150, meta.
```

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

```
## Warning: Data is of class matrix. Coercing to dgCMatrix.
```

```r
print(seger)
```

```
## An object of class Seurat
## 20814 features across 1871 samples within 1 assay
## Active assay: RNA (20814 features, 0 variable features)
```

```
##  1 layer present: counts
```

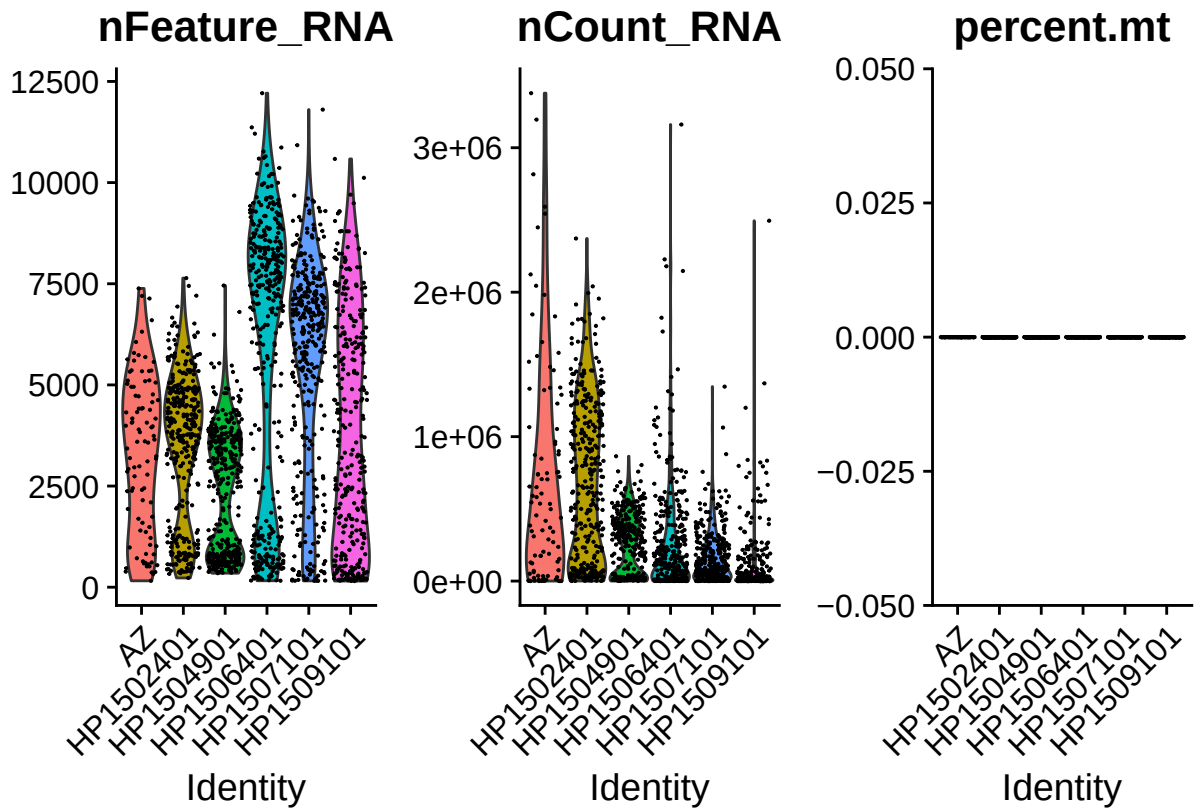
```r
dim(seger)
```

```
## [1] 20814  1871
```

```r
seger[["percent.mt"]] <- PercentageFeatureSet(seger, pattern = "^mt-")
VlnPlot(seger, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)
```

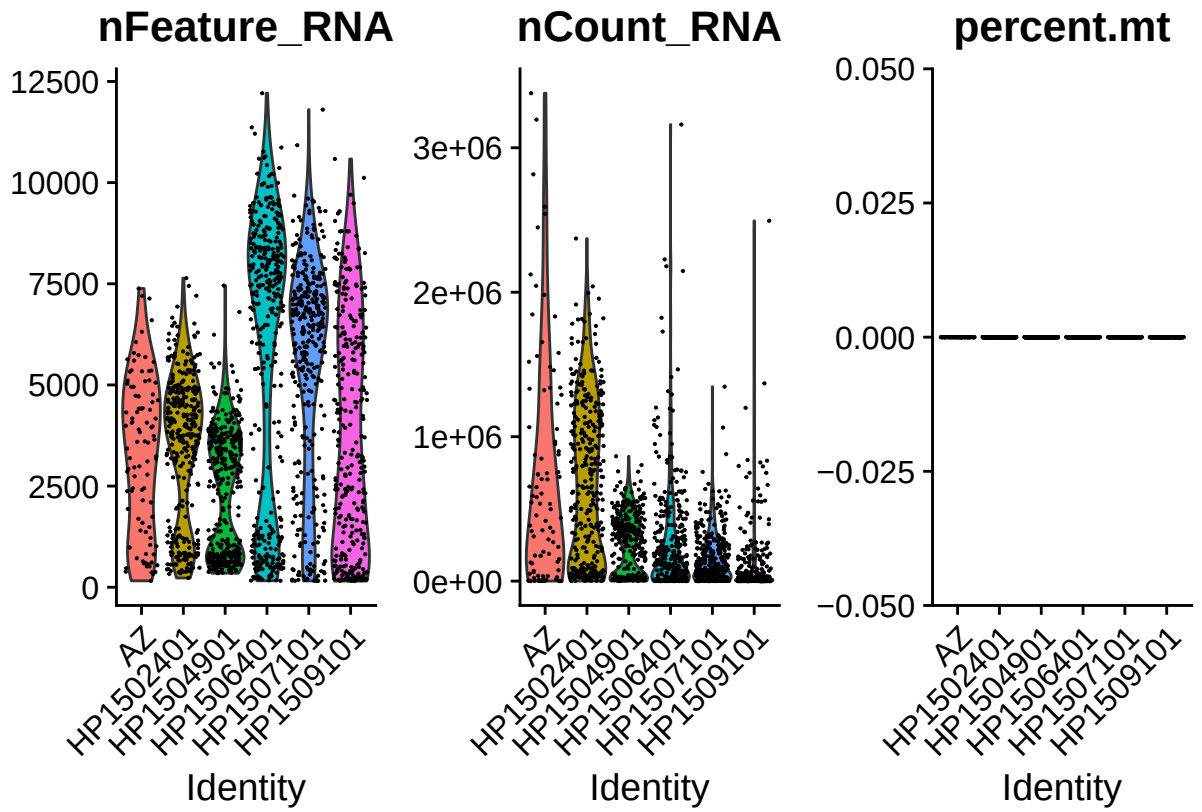
```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
## Warning in SingleExIPlot(type = type, data = data[, x, drop = FALSE], idents =
## idents, : All cells have the same value of percent.mt.
```




```r
seger <- subset(seger,subset = nFeature_RNA > 10 &
           percent.mt < 60)
VlnPlot(seger, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
## Warning: All cells have the same value of percent.mt.
```




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

```
## Normalizing layer: counts
```

```
seger <- FindVariableFeatures(seger, selection.method = "vst", nfeatures = 2000)
```

```
## Finding variable features for layer counts
```

```
seger <- ScaleData(seger, features = rownames(seger))
```

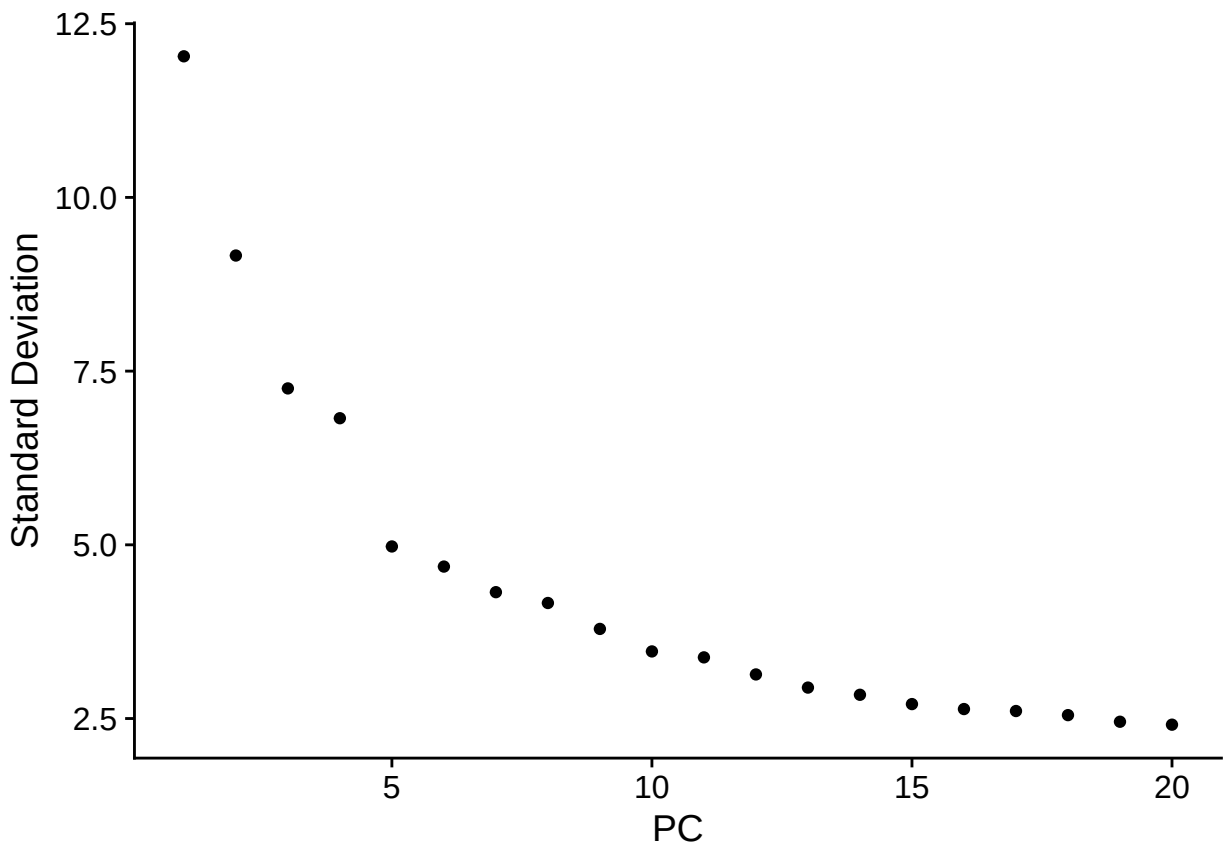
```
## Centering and scaling data matrix
```

```
seger <- RunPCA(seger, features = VariableFeatures(object = seger))
```

```
## PC_ 1
## Positive:  IFITM3, SERPING1, CTSH, CDC42EP1, PRSS8, RHOC, DHRS3, KRT7, TACSTD2, ZFP36L1
##     TMSB4X, ANXA4, CD24, CLDN1, S100A11, LGALS3, LCN2, SDC4, IFITM2, SERPINA5
##     LITAF, CLDN10, KRT18, CFB, KRT19, SERPINA3, TM4SF1, PPAP2C, TNFRSF12A, SAT1
## Negative:  G6PC2, CRYBA2, PEMT, HEPACAM2, SCGB2A1, RFX6, PCP4, RGS4, SLC38A4, FAM159B
##     RGS9, PARM1, LOXL4, CRH, FAM105A, PLCE1, EDIL3, RBP4, SST, PCSK1
##     FAP, SERPINI1, NROB1, RASD1, MLLT11, DHRS2, C1QL1, MLIP, SPTSSB, IAPP
## PC_ 2
## Positive:  SPARC, COL4A1, NID1, LRRC32, IFITM1, COL1A2, BGN, PXDN, TIMP3, PDGFRB
##     COL5A1, COL6A2, COL3A1, LAMA4, F2R, COL1A1, SFRP2, COL6A3, TMEM204, THY1
##     PTRF, IGFBP4, ENG, AEBP1, CDH11, MMP2, ANGPTL2, COL15A1, PLXDC1, LHFP
## Negative:  CD24, KRT8, LCN2, CLDN4, GATM, ELF3, PRSS8, SERPINA3, TACSTD2, SDC4
##     CFB, REG1A, KRT7, MUC1, SPINK1, ANPEP, GSTA1, AMBP, RAB11FIP1, CD74
##     KRT18, ANXA4, SERPINA5, LYZ, GGT6, PRSS1, ADH1C, CLDN1, CD44, CYP3A5
## PC_ 3
```

```
## Positive:  CTRB1, CTRC, CTRB2, PNLIPRP2, CPA2, PRSS3P2, PNLIPRP1, CPA1, PNLIP, REG1B
##     CELA2A, PRSS1, CELA3A, PRSS3, BCAT1, PLA2G1B, RARRES2, CELA3B, REG3G, REG3A
##     CPB1, PDIA2, SPINK1, MT1H, DPEP1, CEL, GP2, REG1A, KLK1, MT1G
## Negative:  CFTR, TINAGL1, SPP1, CMTM7, CCND1, TSPAN8, AQP1, VTCN1, KRT19, ALDH1A3
##     PPAP2C, CA2, IGFBP7, CEACAM7, MMP7, PMEPA1, SLC3A1, HSD17B2, APCDD1, SERPINA1
##     ONECUT2, SERPINA6, FUT3, ANXA2P2, ANXA2, ANXA9, C6, CA4, SEMA6A, HHEX
## PC_ 4
## Positive:  COL1A2, COL5A1, PDGFRB, COL3A1, COL1A1, SFRP2, COL6A3, PCOLCE, BGN, LUM
##     CDH11, LAMC3, MXRA8, NOTCH3, CYGB, COL6A1, THBS2, COL6A2, FN1, EDNRA
##     AEBP1, DCN, COL5A2, MMP11, MSC, CD248, PLXDC1, ISLR, GPNMB, LAMA2
## Negative:  RGCC, FLT1, PCAT19, PECAM1, PODXL, ELTD1, KDR, PLVAP, ERG, ESAM
##     CLEC14A, S1PR1, MYCT1, VWF, MMP1, FAM101B, CDH5, IFI27, ECSCR, CXCR4
##     GPR4, PRDM1, PASK, ACVRL1, F2RL3, CD36, TM4SF18, DYSF, CALCRL, MMRN2
## PC_ 5
## Positive:  HADH, NPTX2, PCSK1, PFKFB2, C1orf127, FAM159B, RASD1, ADCYAP1, PARM1, G6PC2
##     FAM105A, ELMO1, SLC39A14, CASR, NDRG4, MEG3, DHRS2, PDX1, GSN, C1QL1
##     ENTPD3, IGSF1, HEPACAM2, IAPP, ATP1B1, PEMT, SLC6A6, BMP5, TIMP1, TMEM37
## Negative:  ERCC-468.75:mix1-937.5:mix2, AXL, ERCC-468.75:mix1-468.75:mix2, ERCC-29.296875:mix1-43.945
##     ERCC-117.1875:mix1-175.78125:mix2, ERCC-468.75:mix1-117.1875:mix2, PPY, ERCC-117.1875:mix1-234.37
##     ERCC-58.59375:mix1-58.59375:mix2, OLFM4, CRP, LCN2, SERPINA3, PIGR, DEFB1, SPARC, PLVAP, CEACAM7
```

```r
ElbowPlot(seger)
```




```r
ncol(seger)
```

```
## [1] 1871
```

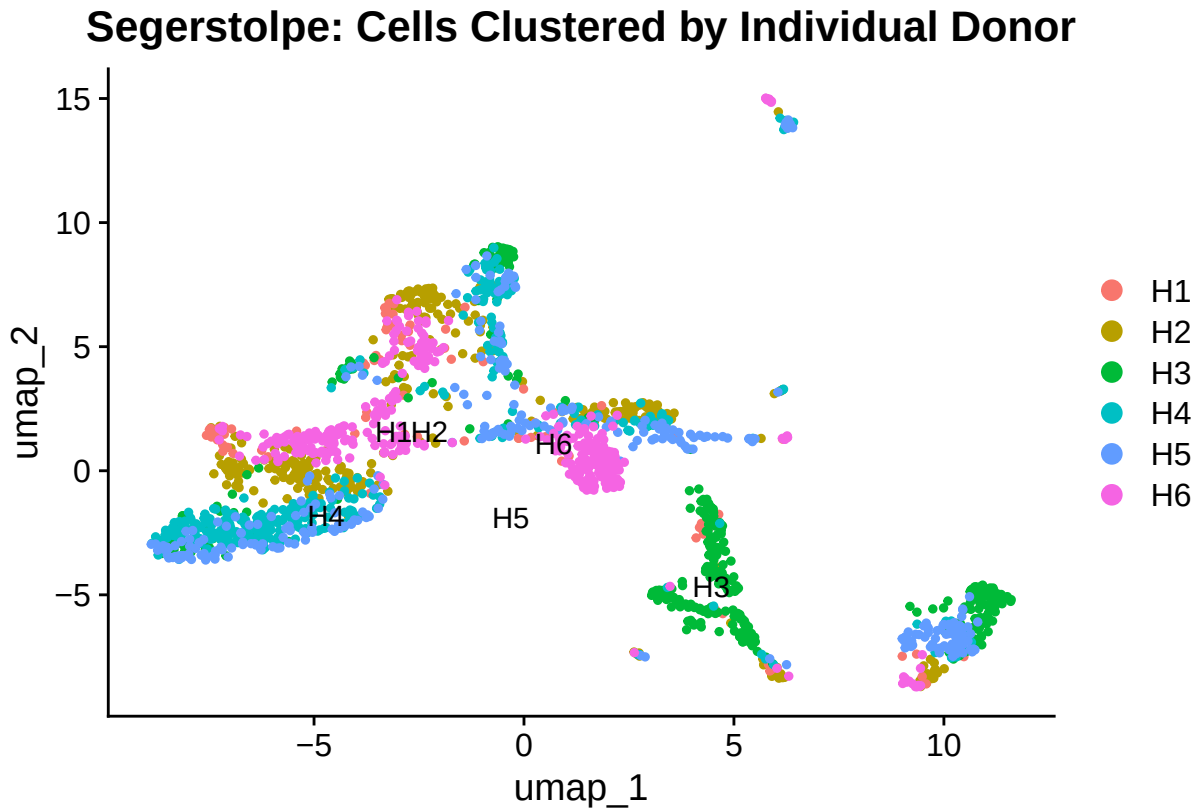
```r
seger <- RunUMAP(seger, dims = 1:10, perplexity = min(3, floor(ncol(seger) / 3)))
```

```
## Warning: The following arguments are not used: perplexity

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

## 12:18:48 UMAP embedding parameters a = 0.9922 b = 1.112

## 12:18:48 Read 1871 rows and found 10 numeric columns

## 12:18:48 Using Annoy for neighbor search, n_neighbors = 30

## 12:18:48 Building Annoy index with metric = cosine, n_trees = 50

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

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

## **************************************************|
## 12:18:48 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc156
## 12:18:48 Searching Annoy index using 1 thread, search_k = 3000
## 12:18:49 Annoy recall = 100%
## 12:18:49 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:18:49 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:18:49 Commencing optimization for 500 epochs, with 73270 positive edges
## 12:18:54 Optimization finished
```

```r
colnames(seger@meta.data)
```

```
##  [1] "orig.ident"                    "nCount_RNA"
##  [3] "nFeature_RNA"                  "Source Name"
##  [5] "Characteristics [organism]"    "Characteristics [individual]"
##  [7] "Characteristics [sex]"         "Characteristics [age]"
##  [9] "Unit [time unit]"              "percent.mt"
```

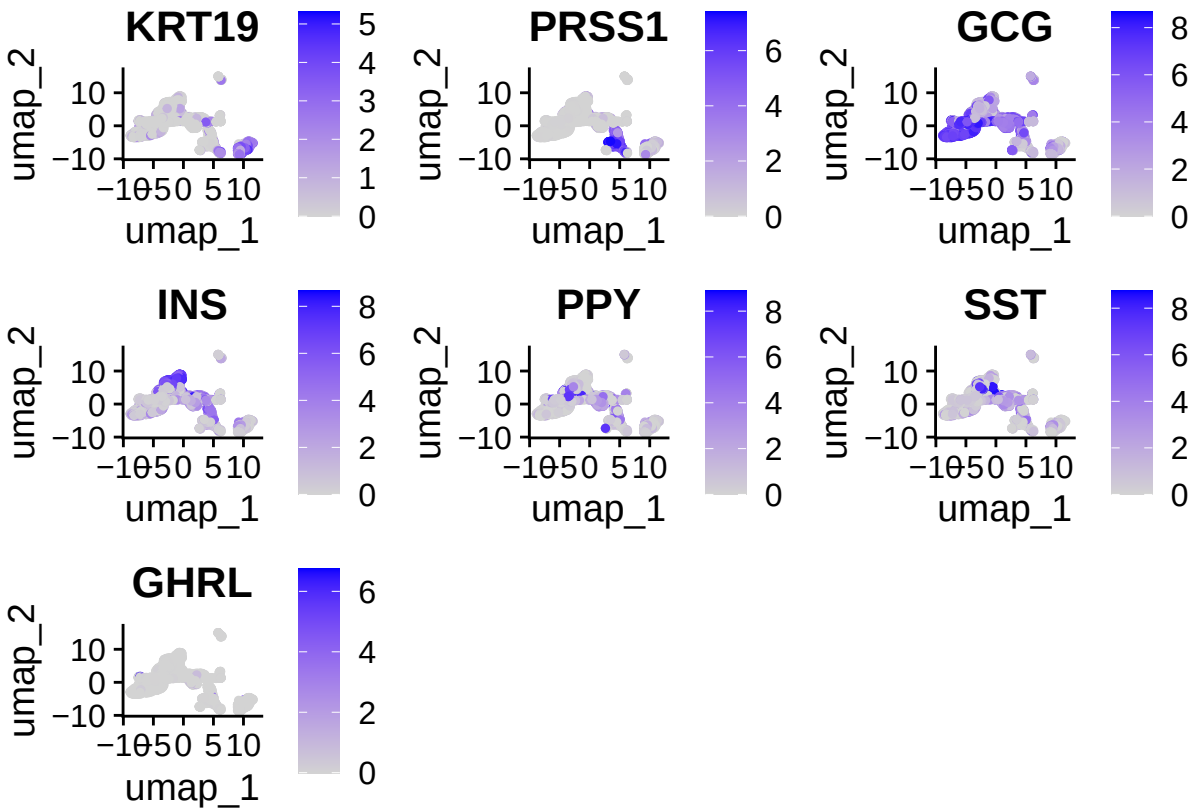
```r
seger$individual <- seger$`Characteristics [individual]`
Idents(seger) <- "individual"
DimPlot(seger, reduction = "umap", group.by = "individual", label = TRUE) +
  ggtitle("Segerstolpe: Cells Clustered by Individual Donor")
```




```
marker_genes <- c("KRT19", "PRSS1", "GCG", "INS", "PPY", "SST", "GHRL")
available_genes <- marker_genes[marker_genes %in% rownames(seger)]
print(available_genes)
```

```
## [1] "KRT19" "PRSS1" "GCG"   "INS"   "PPY"   "SST"   "GHRL"
```

```
FeaturePlot(seger, features = available_genes, reduction = "umap", ncol = 3)
```




```r
# problem 1 for baron data
library(Seurat)
library(ggplot2)
library(dplyr)
library(R.utils)
```

```
## Warning: package 'R.utils' was built under R version 4.4.3

## Loading required package: R.oo
## Loading required package: R.methodsS3
## R.methodsS3 v1.8.2 (2022-06-13 22:00:14 UTC) successfully loaded. See ?R.methodsS3 for help.
## R.oo v1.27.0 (2024-11-01 18:00:02 UTC) successfully loaded. See ?R.oo for help.
##
## Attaching package: 'R.oo'
##
## The following object is masked from 'package:R.methodsS3':
##
##     throw
##
## The following objects are masked from 'package:methods':
##
##     getClasses, getMethods
##
## The following objects are masked from 'package:base':
##
##     attach, detach, load, save
##
```

```
## R.utils v2.13.0 (2025-02-24 21:20:02 UTC) successfully loaded. See ?R.utils for help.
##
## Attaching package: 'R.utils'
##
## The following object is masked from 'package:utils':
##
##     timestamp
##
## The following objects are masked from 'package:base':
##
##     cat, commandArgs, getOption, isOpen, nullfile, parse, use, warnings
```

```r
options(repos = c(CRAN = "https://cloud.r-project.org/"))
if (!requireNamespace("Seurat", quietly = TRUE)) install.packages("Seurat")
if (!requireNamespace("ggplot2", quietly = TRUE)) install.packages("ggplot2")
if (!requireNamespace("R.utils", quietly = TRUE)) install.packages("R.utils")
base_dir <- "C:/Users/sushm/Downloads/GSE84133_RAW"

if (!file.exists(file.path(base_dir, "GSM2230757_human1_umifm_counts.csv"))) {
  untar("C:/Users/sushm/Downloads/GSE84133_RAW.tar", exdir=base_dir)
} else {
  message("Files already extracted. Skipping extraction.")
}
```

```
## Files already extracted. Skipping extraction.
```

```r
gz_files <- list.files(base_dir, pattern = "\\.gz$", full.names = TRUE)
for (f in gz_files) {
  unzipped_file <- sub(".gz$", "", f)
  if (!file.exists(unzipped_file)) {
    gunzip(f, remove = FALSE)
  } else {
    message(paste("File already exists:", unzipped_file, "- Skipping decompression."))
  }
}
```

```
## File already exists: C:/Users/sushm/Downloads/GSE84133_RAW/GSM2230757_human1_umifm_counts.csv - Skipp
## File already exists: C:/Users/sushm/Downloads/GSE84133_RAW/GSM2230758_human2_umifm_counts.csv - Skipp
## File already exists: C:/Users/sushm/Downloads/GSE84133_RAW/GSM2230759_human3_umifm_counts.csv - Skipp
## File already exists: C:/Users/sushm/Downloads/GSE84133_RAW/GSM2230760_human4_umifm_counts.csv - Skipp
```

```r
base_dir <- "C:/Users/sushm/Downloads/GSE84133_RAW"
baron1 <- read.csv(file.path(base_dir, "GSM2230757_human1_umifm_counts.csv"), row.names = 1)
rownames(baron1) <- paste0("donor1_", rownames(baron1))

baron2 <- read.csv(file.path(base_dir, "GSM2230758_human2_umifm_counts.csv"), row.names = 1)
rownames(baron2) <- paste0("donor2_", rownames(baron2))

baron3 <- read.csv(file.path(base_dir, "GSM2230759_human3_umifm_counts.csv"), row.names = 1)
rownames(baron3) <- paste0("donor3_", rownames(baron3))

baron4 <- read.csv(file.path(base_dir, "GSM2230760_human4_umifm_counts.csv"), row.names = 1)
rownames(baron4) <- paste0("donor4_", rownames(baron4))
stopifnot(all(colnames(baron1) == colnames(baron2)))
stopifnot(all(colnames(baron1) == colnames(baron3)))
stopifnot(all(colnames(baron1) == colnames(baron4)))
```

```r
baron <- rbind(baron1, baron2, baron3, baron4)
rm(baron1, baron2, baron3, baron4)
cell.meta_b <- baron[, 1:3]
colnames(cell.meta_b)[1] <- "cell"
colnames(cell.meta_b)[3] <- "celltype"
cell.meta_b$cell <- rownames(baron)
cell.meta_b$individual <- sub("_.*", "", cell.meta_b$cell)
print(table(cell.meta_b$individual))
```

```
##
## donor1 donor2 donor3 donor4
##   1937   1724   3605   1303
```

```r
cell.meta_b <- cell.meta_b[!duplicated(cell.meta_b$cell), ]
rownames(cell.meta_b) <- cell.meta_b$cell
count_mat <- as.matrix(baron[, -(1:3)])
rm(baron)
rownames(count_mat) <- cell.meta_b$cell
count_b <- t(count_mat)
count_b <- count_b[!duplicated(rownames(count_b)), ]
common_cells <- intersect(colnames(count_b), cell.meta_b$cell)
count_b <- count_b[, common_cells]
cell.meta_b <- cell.meta_b[cell.meta_b$cell %in% common_cells, ]
rownames(cell.meta_b) <- cell.meta_b$cell
stopifnot(all(colnames(count_b) == rownames(cell.meta_b)))
library(Seurat)
seurat_baron <- CreateSeuratObject(
  counts = count_b,
  project = "baron",
  min.cells = 1,
  min.features = 50,
  meta.data = cell.meta_b
)
```

```
## Warning: Data is of class matrix. Coercing to dgCMatrix.
```

```r
dim(seurat_baron)
```

```
## [1] 17498  8569
```

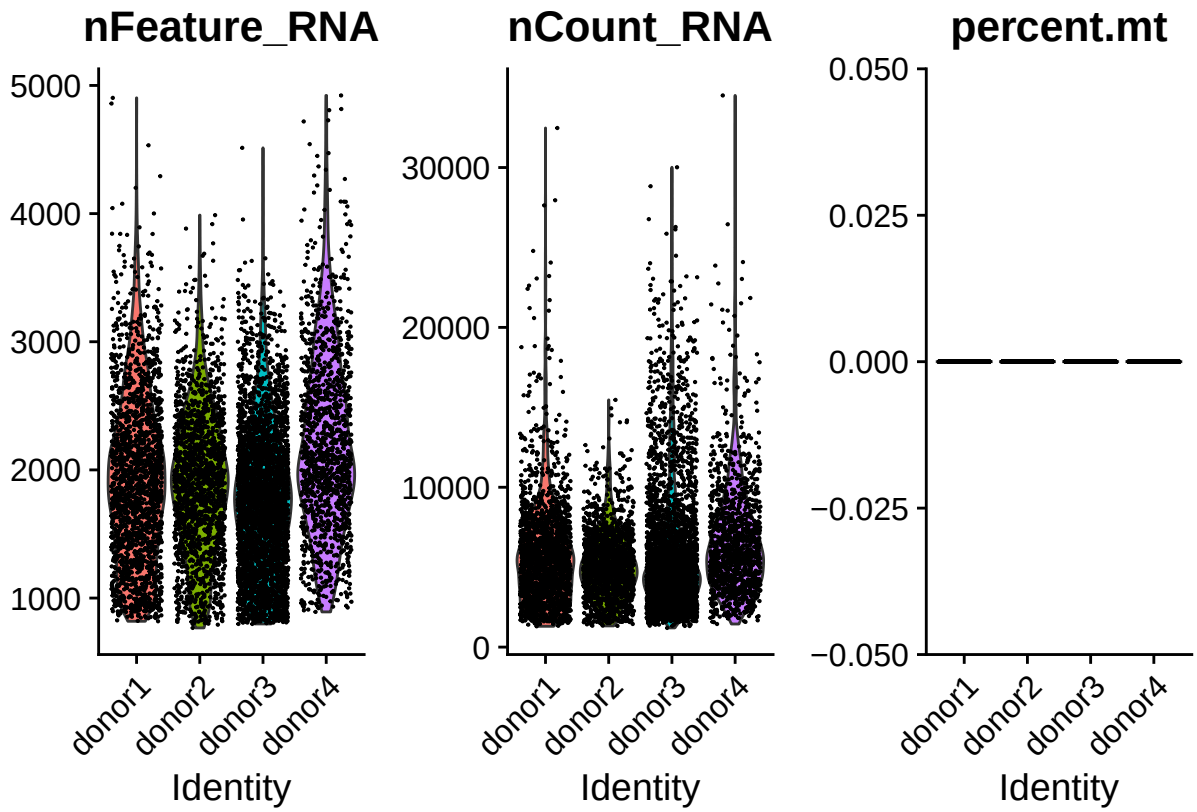
```r
print(table(seurat_baron$individual))
```

```
##
## donor1 donor2 donor3 donor4
##   1937   1724   3605   1303
```

```r
seurat_baron[["percent.mt"]] <- PercentageFeatureSet(seurat_baron, pattern = "^mt-")
VlnPlot(seurat_baron, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
## Warning in SingleExIPlot(type = type, data = data[, x, drop = FALSE], idents =
## idents, : All cells have the same value of percent.mt.
```




```
seurat_baron <- subset(seurat_baron,subset = nFeature_RNA > 10 & percent.mt < 60)
seurat_baron <- NormalizeData(seurat_baron, normalization.method = "LogNormalize", scale.factor = 10000)
```

```
## Normalizing layer: counts
```

```
seurat_baron <- FindVariableFeatures(seurat_baron, selection.method = "vst", nfeatures = 1000)
```

```
## Finding variable features for layer counts
```

```
seurat_baron <- ScaleData(seurat_baron, features = rownames(seurat_baron))
```

```
## Centering and scaling data matrix
```

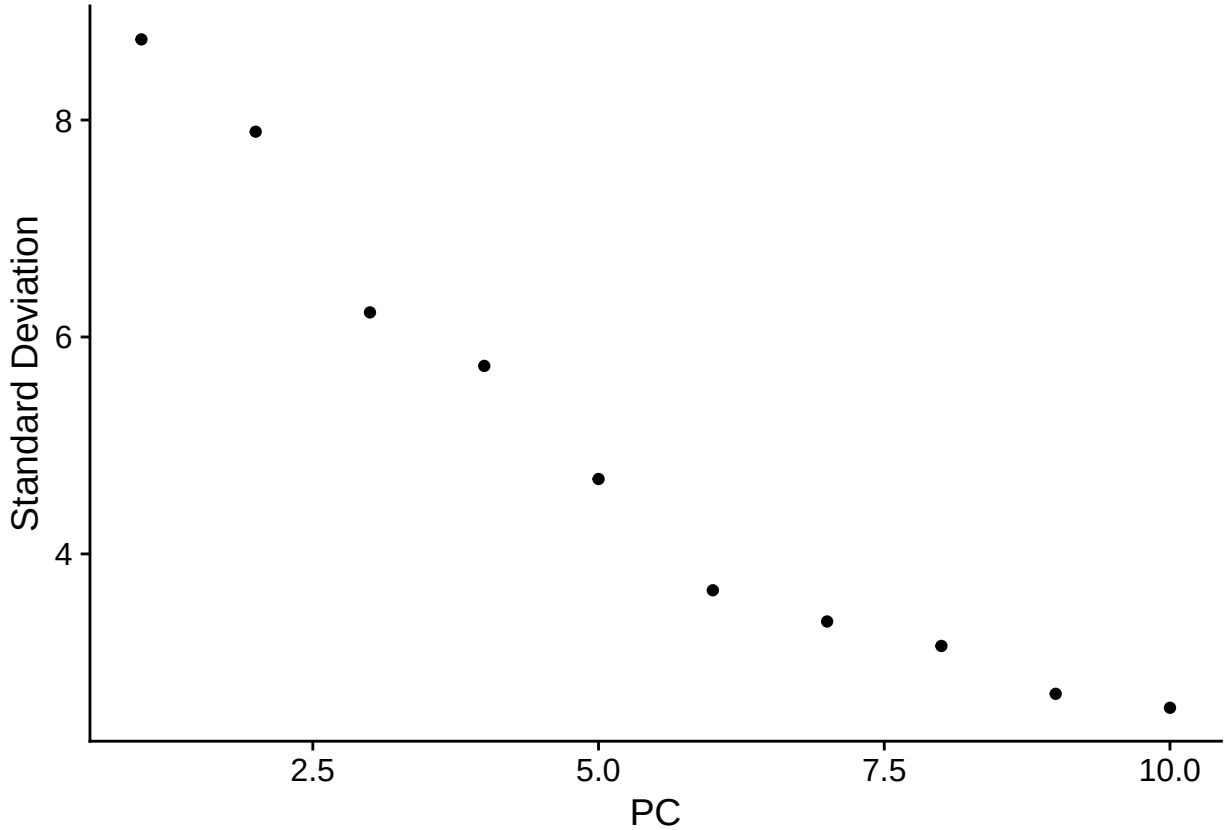
```
seurat_baron <- RunPCA(seurat_baron, features = VariableFeatures(object = seurat_baron), npcs = min(10,
```

```
## PC_ 1
## Positive:  CHGB, VGF, GAD2, SLC30A8, ABCC8, C1QL1, PCSK1, UCN3, ALDH1A1, STX1A
##     IAPP, MAFB, RASD1, PEG10, G6PC2, C10orf10, RGS4, TM4SF4, ADCYAP1, KCNMA1
##     SYNGR4, DHRS2, GC, CRYBA2, EDN3, RBP4, IRX2, DLK1, PAPPA2, FEV
## Negative:  COL18A1, IFITM3, MMP14, IFITM2, SPARC, PMEPA1, LGALS1, SERPINH1, COL6A2, PXDN
##     ITGA5, COL1A1, COL4A1, TMSB10, BGN, PDGFRB, FLNA, COL1A2, COL3A1, ENG
##     NOTCH3, COL15A1, COL6A1, FSTL1, SERPINE1, ZFP36L1, SERPING1, COL6A3, MFGE8, CEBPB
## PC_ 2
## Positive:  SERPINA3, CD24, IL32, CD44, SDC4, KRT18, PDZK1IP1, TACSTD2, KRT8, SPINK1
##     PRSS8, KRT7, CTRB1, PRSS2, CTRB2, REG1A, PRSS1, CPA2, LCN2, CPA1
##     PLA2G1B, CELA3A, CELA3B, KLK1, TM4SF1, CPB1, CTRC, CLDN4, CLPS, MUC1
## Negative:  TIMP1, SPARC, BGN, COL6A2, IGFBP4, COL3A1, COL1A2, PDGFRB, COL15A1, COL4A1
##     COL6A3, CYGB, PXDN, AEBP1, ENG, COL5A1, COL1A1, COL6A1, ITGA1, EMILIN1
##     CRLF1, COL5A3, COL5A2, TIMP3, MMP2, CRISPLD2, SFRP2, HTRA3, FMOD, FKBP10
```

```
## PC_ 3
## Positive:  CTRC, CELA3B, PNLIPRP1, CPA2, CLPS, CELA2A, PLA2G1B, CPA1, PRSS1, KLK1
##     CELA3A, SYCN, CPB1, PNLIP, CTRB1, PNLIPRP2, PRSS2, CTRB2, GP2, REG1B
##     CUZD1, PRSS3, CTRL, CELA2B, REG1A, SPINK1, REG1P, MGST1, FOXD2, PLTP
## Negative:  CFTR, MMP7, KRT19, TINAGL1, SLC4A4, LGALS4, ANXA4, SERPINA5, S100A14, AGRN
##     TSPAN8, CD74, CTSH, MALL, ONECUT2, KRT23, SERPINA1, TNFAIP2, VCAM1, TSPAN15
##     SFRP5, ALDH1A3, EDN1, CCL2, SERPING1, CLDN10, GPRC5B, AQP1, PIGR, CX3CL1
## PC_ 4
## Positive:  COL1A1, COL6A3, COL1A2, COL3A1, CRLF1, COL5A1, SFRP2, PDGFRB, BGN, THBS2
##     FMOD, NOTCH3, COL6A1, VCAN, COL5A2, FN1, EMILIN1, ADAMTS12, CYGB, SPON2
##     MXRA8, LUM, COL5A3, C1S, LAMC3, PRRX1, VSTM4, TPM2, TGFB3, ANO1
## Negative:  PLVAP, PECAM1, CD93, FLT1, KDR, PODXL, SOX18, RGCC, ACVRL1, PCAT19
##     ROBO4, VWF, ESM1, TIE1, ESAM, ADGRL4, SEMA3F, PASK, S1PR1, CXCR4
##     CLDN5, NOTCH4, ANGPT2, CDH5, BCL6B, IFI27, MCAM, IL3RA, ADGRF5, ACKR3
## PC_ 5
## Positive:  TYROBP, LAPTM5, SPI1, FCER1G, CD53, ACP5, HLA.DRA, SDS, C1QC, ITGB2
##     LCP1, ALOX5AP, LSP1, CD300A, CSF1R, C1QB, APOE, HLA.DMB, RGS1, FGL2
##     TNFRSF1B, C1QA, HLA.DPA1, HCK, SRGN, HLA.DRB1, MPEG1, PLA2G7, TBXAS1, AIF1
## Negative:  IGFBP7, PLVAP, HSPG2, PXDN, ESAM, PODXL, FLT1, COL4A1, KDR, STC1
##     VWF, ROBO4, SOX18, PCAT19, ESM1, MCAM, COL18A1, SEMA3F, ADGRL4, LAMA4
##     PASK, ADGRF5, PRSS23, SERPINE1, ANGPT2, SPARC, TIE1, CLDN5, S100A16, PPAP2B
```

```r
ElbowPlot(seurat_baron)
```

```
## Warning in ElbowPlot(seurat_baron): The object only has information for 10
## reductions
```

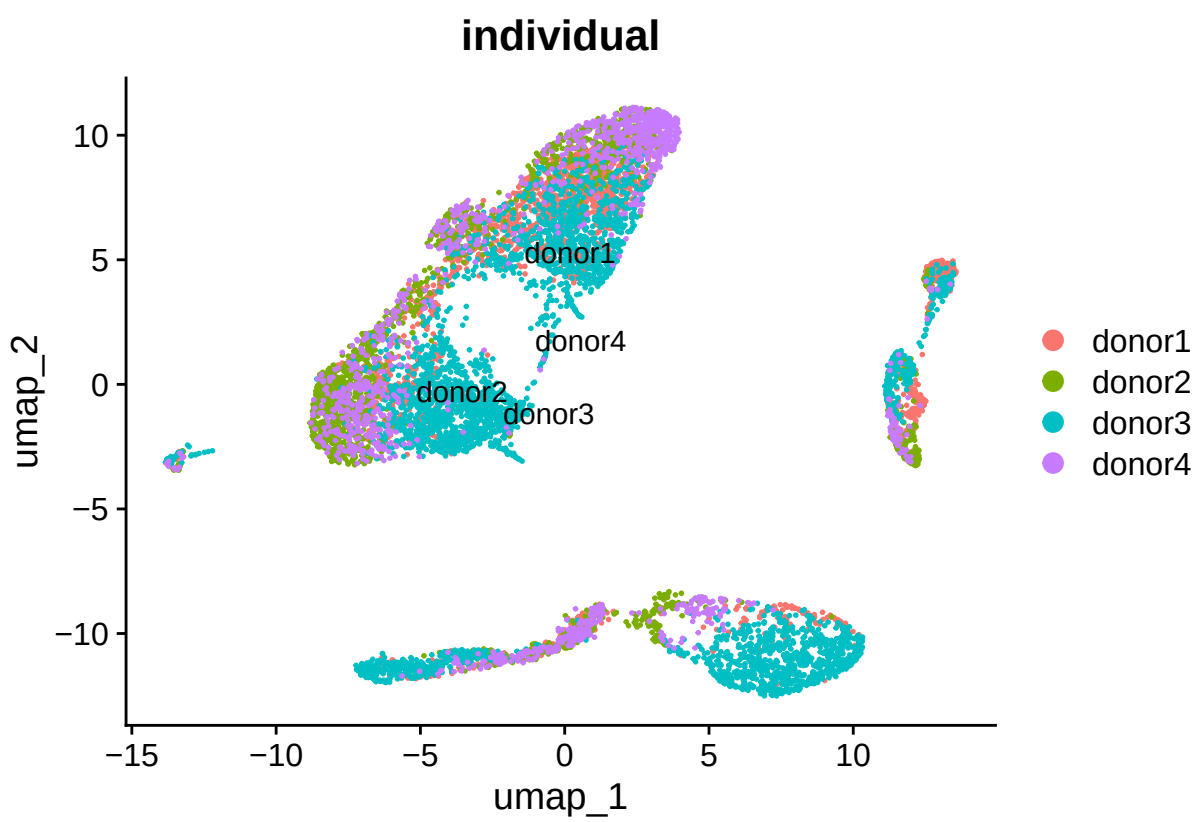


```r
seurat_baron <- RunUMAP(seurat_baron, dims = 1:7, perplexity = min(10, floor(ncol(seurat_baron) / 3)))
```

```
## Warning: The following arguments are not used: perplexity
```

```
## 12:20:12 UMAP embedding parameters a = 0.9922 b = 1.112
## 12:20:12 Read 8569 rows and found 7 numeric columns
## 12:20:12 Using Annoy for neighbor search, n_neighbors = 30
## 12:20:12 Building Annoy index with metric = cosine, n_trees = 50
## 0%   10   20   30   40   50   60   70   80   90   100%
## [----|----|----|----|----|----|----|----|----|----|
## **************************************************|
## 12:20:13 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc4bf
## 12:20:13 Searching Annoy index using 1 thread, search_k = 3000
## 12:20:15 Annoy recall = 100%
## 12:20:15 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:20:16 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:20:17 Commencing optimization for 500 epochs, with 347332 positive edges
## 12:20:36 Optimization finished
```

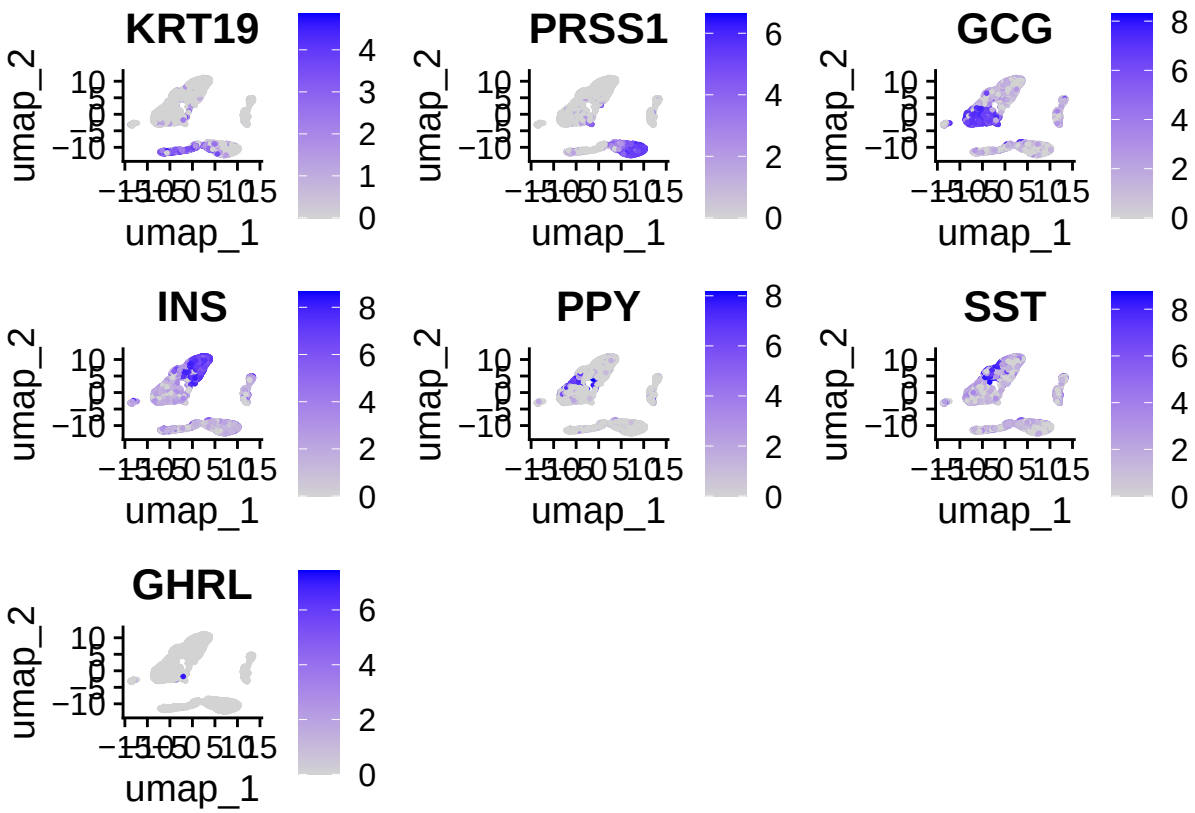
```r
DimPlot(seurat_baron, reduction = "umap", group.by = "individual", label = TRUE)
```




```r
marker_genes <- c("KRT19", "PRSS1", "GCG", "INS", "PPY", "SST", "GHRL")
available_genes <- marker_genes[marker_genes %in% rownames(seurat_baron)]
print(available_genes)
```

```
## [1] "KRT19" "PRSS1" "GCG"   "INS"   "PPY"   "SST"   "GHRL"
```

```r
FeaturePlot(seurat_baron, features = available_genes, reduction = "umap", ncol = 3)
```



Answer 1

Pancreatic Cell Clustering Analysis We processed the six healthy individuals from Segerstolpe et al. and the four donors from Baron et al. separately, including normalization, dimensionality reduction, and clustering. In both datasets, cells clustered by cell type rather than by individual or donor. FeaturePlot and VlnPlot of marker genes confirmed distinct clusters: KRT19 (ductal), PRSS1 (acinar), GCG (-cells), INS (-cells), PPY (-cells), and SST (-cells). GHRL expression was absent, indicating no detectable -cells. Overall, clustering reflects clear segregation of major pancreatic cell types.

Problem (2)

```r
library(Seurat)
library(dplyr)
library(ggplot2)

Idents(seurat_baron) <- "assigned_cluster"
baron.markers <- FindAllMarkers(seurat_baron, only.pos = TRUE, min.pct = 0.25, logfc.threshold = 0.25)
```

```
## Calculating cluster acinar
```

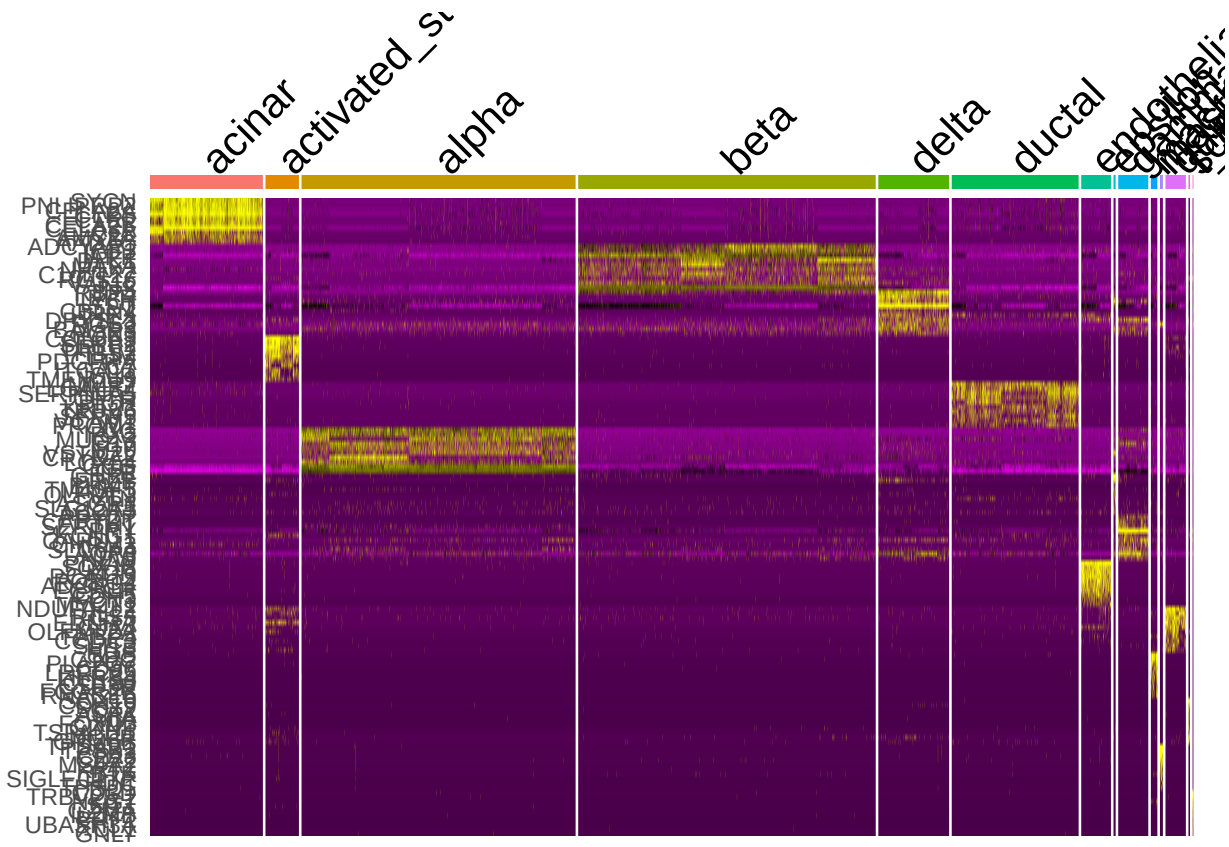
```
## For a (much!) faster implementation of the Wilcoxon Rank Sum Test,
## (default method for FindMarkers) please install the presto package
## --------------------------------------------
```

```
## install.packages('devtools')
## devtools::install_github('immunogenomics/presto')
## ---------------------------------------------
## After installation of presto, Seurat will automatically use the more
## efficient implementation (no further action necessary).
## This message will be shown once per session

## Calculating cluster beta

## Calculating cluster delta

## Calculating cluster activated_stellate

## Calculating cluster ductal

## Calculating cluster alpha

## Calculating cluster epsilon

## Calculating cluster gamma

## Calculating cluster endothelial

## Calculating cluster quiescent_stellate

## Calculating cluster macrophage

## Calculating cluster schwann

## Calculating cluster mast

## Calculating cluster t_cell
```

```
baron.markers %>%
  group_by(cluster) %>%
  slice_max(n = 2, order_by = avg_log2FC)
```

```
## # A tibble: 28 x 7
## # Groups:   cluster [14]
##    p_val avg_log2FC pct.1 pct.2 p_val_adj cluster            gene
##    <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>              <chr>
##  1     0       7.84 0.562 0.005         0 acinar             AMY2A
##  2     0       7.61 0.929 0.016         0 acinar             PNLIPRP2
##  3     0       6.07 0.505 0.016         0 beta               MAFA
##  4     0       6.02 0.966 0.441         0 beta               IAPP
##  5     0       7.21 0.993 0.725         0 delta              SST
##  6     0       5.95 0.606 0.023         0 delta              LEPR
##  7     0       9.76 0.792 0.004         0 activated_stellate SFRP2
##  8     0       9.61 0.303 0.001         0 activated_stellate TMEM119
##  9     0       8.81 0.315 0.002         0 ductal             SFRP5
## 10     0       8.60 0.505 0.005         0 ductal             CFTR
## # i 18 more rows
```

```
baron.markers %>%
  group_by(cluster) %>%
  top_n(n = 10, wt = avg_log2FC) -> top10
DoHeatmap(seurat_baron, features = top10$gene,group.by = "assigned_cluster") + NoLegend()
```




```
seger$predicted.celltype <- "Unknown"
expr_data <- FetchData(seger, vars = c("KRT19", "PRSS1", "GCG", "INS", "PPY", "SST", "GHRL"))
threshold <- 1
seger$predicted.celltype[expr_data$KRT19 > threshold] <- "Ductal"
seger$predicted.celltype[expr_data$PRSS1 > threshold] <- "Acinar"
seger$predicted.celltype[expr_data$GCG > threshold]   <- "Alpha"
seger$predicted.celltype[expr_data$INS > threshold]   <- "Beta"
seger$predicted.celltype[expr_data$PPY > threshold]   <- "Gamma"
seger$predicted.celltype[expr_data$SST > threshold]   <- "Delta"
seger$predicted.celltype[expr_data$GHRL > threshold]  <- "Epsilon"

print(table(seger$predicted.celltype))
```

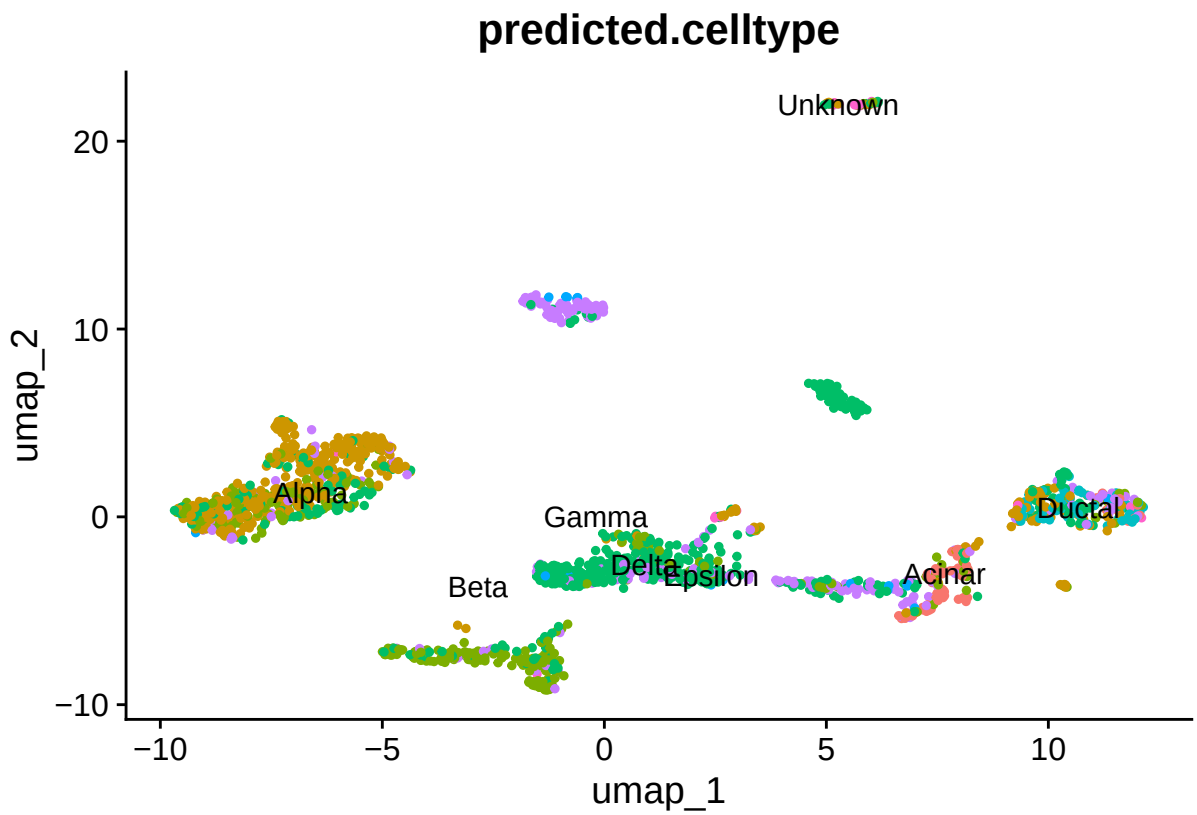
```
##
##  Acinar   Alpha    Beta   Delta  Ductal Epsilon   Gamma Unknown
##     101     465     355     587      62      19     256      26
```

```
if (!"pca" %in% names(seger@reductions)) {
  seger <- ScaleData(seger) %>% RunPCA(npcs = 30, verbose = FALSE)
}

seger <- RunUMAP(seger, reduction = "pca", dims = 1:20)
```

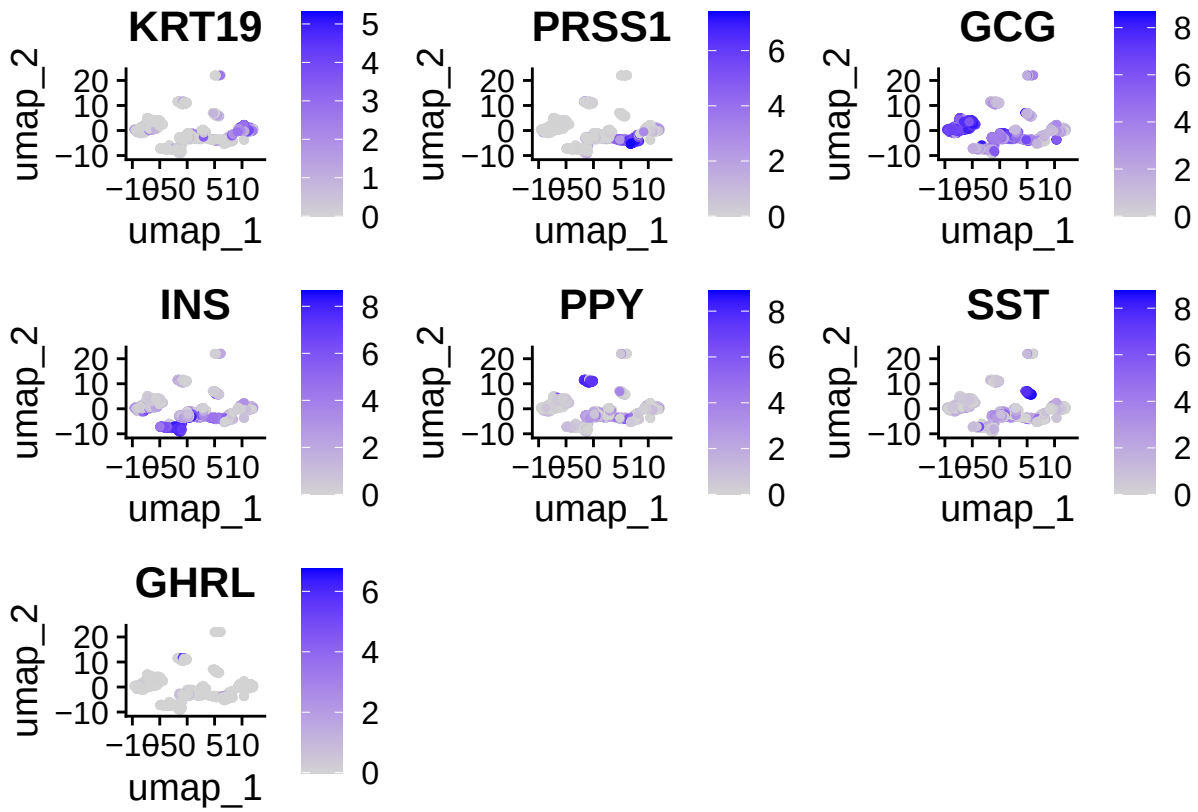
```
## 12:27:28 UMAP embedding parameters a = 0.9922 b = 1.112

## 12:27:28 Read 1871 rows and found 20 numeric columns

## 12:27:28 Using Annoy for neighbor search, n_neighbors = 30
```

```
## 12:27:28 Building Annoy index with metric = cosine, n_trees = 50
## 0%   10   20   30   40   50   60   70   80   90   100%
## [----|----|----|----|----|----|----|----|----|----|
## **************************************************|
## 12:27:30 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc2010
## 12:27:31 Searching Annoy index using 1 thread, search_k = 3000
## 12:27:31 Annoy recall = 100%
## 12:27:31 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:27:32 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:27:32 Commencing optimization for 500 epochs, with 77070 positive edges
## 12:27:37 Optimization finished
```

```
DimPlot(seger, reduction = "umap", group.by = "predicted.celltype", label = TRUE) + NoLegend()
```




```
FeaturePlot(seger, features = c("KRT19", "PRSS1", "GCG", "INS", "PPY", "SST", "GHRL"), reduction = "umap
```



Problem (3)

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

```
## Warning: package 'patchwork' was built under R version 4.4.2
```

```
library(dplyr)

seurat_baron <- NormalizeData(seurat_baron)
```

```
## Normalizing layer: counts
```

```
seurat_baron <- FindVariableFeatures(seurat_baron, selection.method = "vst", nfeatures = 2000)
```

```
## Finding variable features for layer counts
```

```
seurat_baron <- ScaleData(seurat_baron)
```

```
## Centering and scaling data matrix

## Warning: Different features in new layer data than already exists for
## scale.data
```

```
seurat_baron <- RunPCA(seurat_baron, npcs = 10)
```

```
## PC_ 1
## Positive:  CHGA, CHGB, SCG2, PPP1R1A, VGF, PCSK2, GAD2, SLC30A8, ERO1B, NEUROD1
```

```
##     SYT7, ABCC8, C1QL1, PCSK1, C2CD4A, UCN3, PAPSS2, GCH1, IAPP, PCP4
##     TMEM176B, STX1A, CFC1, MAFB, TMEM176A, INSM1, C2CD4B, PAK3, G6PC2, PLCXD3
## Negative:  IFITM3, PMEPA1, TMSB10, COL18A1, IFITM2, ZFP36L1, CDC42EP1, SOX4, MMP14, FLNA
##     SERPING1, ITGA5, MSN, YBX3, RHOC, MYH9, TMSB4X, LGALS1, S100A16, SERPINH1
##     S100A11, TACSTD2, HSPG2, LGALS3, CYR61, LITAF, CEBPB, KRT7, PFKFB3, DDIT4
## PC_ 2
## Positive:  SERPINA3, CD24, KRT8, KRT18, TACSTD2, SDC4, PDZK1IP1, KRT7, PRSS8, CD44
##     IL32, LCN2, SPINK1, CLDN4, CTRB1, PRSS2, MUC1, REG1A, CTRB2, ELF3
##     PRSS1, CPA2, CFB, LAD1, KRT19, CPA1, GATM, PLA2G1B, SERINC2, CELA3A
## Negative:  SPARC, COL6A2, BGN, IGFBP4, COL4A1, PDGFRB, COL3A1, COL1A2, COL15A1, PXDN
##     ENG, COL6A3, COL1A1, CYGB, AEBP1, COL6A1, COL5A1, TIMP1, EMILIN1, ITGA1
##     COL5A3, CRLF1, COL5A2, TIMP3, MMP2, CRISPLD2, HTRA3, LRRC32, MFGE8, FMOD
## PC_ 3
## Positive:  CFTR, MMP7, TINAGL1, SLC4A4, KRT19, LGALS4, ANXA4, SERPINA5, PPAP2C, AGRN
##     SERPINA1, TSPAN8, S100A14, CD74, CTSH, VWA1, ONECUT2, FXYD2, AQP1, KRT23
##     VCAM1, SFRP5, MALL, TSPAN15, TNFAIP2, TRPV6, GPRC5B, CX3CL1, RHPN2, ALDH1A3
## Negative:  CELA3B, CTRC, CPA2, PNLIPRP1, CLPS, PLA2G1B, PRSS1, CELA2A, CELA3A, CPA1
##     KLK1, CTRB1, SYCN, CPB1, PRSS2, CTRB2, PNLIP, PNLIPRP2, REG1B, GP2
##     REG1A, CUZD1, PRSS3, CTRL, CELA2B, SPINK1, REG1P, MGST1, FOXD2, PLTP
## PC_ 4
## Positive:  PLVAP, PECAM1, CD93, FLT1, KDR, PODXL, SOX18, RGCC, ACVRL1, PCAT19
##     ROBO4, VWF, ESM1, TIE1, ESAM, ADGRL4, ECSCR, SEMA3F, S1PR1, PASK
##     CXCR4, CLDN5, DYSF, NOTCH4, ERG, ANGPT2, CDH5, BCL6B, IFI27, PTPRB
## Negative:  COL1A1, COL1A2, COL6A3, COL3A1, CRLF1, COL5A1, PDGFRB, BGN, SFRP2, NOTCH3
##     FMOD, THBS2, COL6A1, COL5A2, VCAN, EMILIN1, FN1, CYGB, ADAMTS12, SPON2
##     MXRA8, LUM, COL5A3, C1S, LAMC3, VSTM4, TPM2, PRRX1, COL6A2, CRISPLD2
## PC_ 5
## Positive:  TYROBP, LAPTM5, SPI1, FCER1G, CD53, ACP5, SDS, C1QC, HLA.DRA, ITGB2
##     LCP1, ALOX5AP, CD300A, LSP1, C1QB, CSF1R, APOE, RGS1, HLA.DMB, FGL2
##     C1QA, TNFRSF1B, SRGN, MYO1F, MPEG1, HCK, HLA.DPA1, PLA2G7, HLA.DRB1, TBXAS1
## Negative:  PXDN, COL4A1, PLVAP, HSPG2, PRSS23, ESAM, FLT1, PODXL, KDR, IGFBP7
##     COL18A1, STC1, VWF, ROBO4, SOX18, PCAT19, SPARC, WWTR1, MCAM, ESM1
##     LAMA4, SERPINE1, ECSCR, ADGRL4, ADGRF5, SEMA3F, ETS1, ANGPTL2, PASK, PPAP2B
```

```r
seurat_baron <- RunUMAP(seurat_baron, dims = 1:10)
```

```
## 12:27:46 UMAP embedding parameters a = 0.9922 b = 1.112

## 12:27:46 Read 8569 rows and found 10 numeric columns

## 12:27:46 Using Annoy for neighbor search, n_neighbors = 30

## 12:27:46 Building Annoy index with metric = cosine, n_trees = 50

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

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

## **************************************************|
## 12:27:47 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc2c7
## 12:27:47 Searching Annoy index using 1 thread, search_k = 3000
## 12:27:49 Annoy recall = 100%
## 12:27:49 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:27:50 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:27:50 Commencing optimization for 500 epochs, with 350842 positive edges
## 12:28:10 Optimization finished
```

```r
seger <- NormalizeData(seger)
```

```
## Normalizing layer: counts
```

```r
seger <- FindVariableFeatures(seger, selection.method = "vst", nfeatures = 2000)
```

```
## Finding variable features for layer counts
```

```r
seger <- ScaleData(seger)
```

```
## Centering and scaling data matrix
```

```
## Warning: Different features in new layer data than already exists for
## scale.data
```

```r
seger <- RunPCA(seger)
```

```
## PC_ 1
## Positive:  IFITM3, SERPING1, CTSH, CDC42EP1, PRSS8, RHOC, DHRS3, KRT7, TACSTD2, ZFP36L1
##     TMSB4X, ANXA4, CD24, CLDN1, S100A11, LGALS3, LCN2, SDC4, IFITM2, SERPINA5
##     LITAF, CLDN10, KRT18, CFB, KRT19, SERPINA3, TM4SF1, PPAP2C, TNFRSF12A, SAT1
## Negative:  G6PC2, CRYBA2, PEMT, HEPACAM2, SCGB2A1, RFX6, PCP4, RGS4, SLC38A4, FAM159B
##     RGS9, PARM1, LOXL4, CRH, FAM105A, PLCE1, EDIL3, RBP4, SST, PCSK1
##     FAP, SERPINI1, NROB1, RASD1, MLLT11, DHRS2, C1QL1, MLIP, SPTSSB, IAPP
## PC_ 2
## Positive:  SPARC, COL4A1, NID1, LRRC32, IFITM1, COL1A2, BGN, PXDN, TIMP3, PDGFRB
##     COL5A1, COL6A2, COL3A1, LAMA4, F2R, COL1A1, SFRP2, COL6A3, TMEM204, THY1
##     PTRF, IGFBP4, ENG, AEBP1, CDH11, MMP2, ANGPTL2, COL15A1, PLXDC1, LHFP
## Negative:  CD24, KRT8, LCN2, CLDN4, GATM, ELF3, PRSS8, SERPINA3, TACSTD2, SDC4
##     CFB, REG1A, KRT7, MUC1, SPINK1, ANPEP, GSTA1, AMBP, RAB11FIP1, CD74
##     KRT18, ANXA4, SERPINA5, LYZ, GGT6, PRSS1, ADH1C, CLDN1, CD44, CYP3A5
## PC_ 3
## Positive:  CTRB1, CTRC, CTRB2, PNLIPRP2, CPA2, PRSS3P2, PNLIPRP1, CPA1, PNLIP, REG1B
##     CELA2A, PRSS1, CELA3A, PRSS3, BCAT1, PLA2G1B, RARRES2, CELA3B, REG3G, REG3A
##     CPB1, PDIA2, SPINK1, MT1H, DPEP1, CEL, GP2, REG1A, KLK1, MT1G
## Negative:  CFTR, TINAGL1, SPP1, CMTM7, CCND1, TSPAN8, AQP1, VTCN1, KRT19, ALDH1A3
##     PPAP2C, CA2, IGFBP7, CEACAM7, MMP7, PMEPA1, SLC3A1, HSD17B2, APCDD1, SERPINA1
##     ONECUT2, SERPINA6, FUT3, ANXA2P2, ANXA2, ANXA9, C6, CA4, SEMA6A, HHEX
## PC_ 4
## Positive:  COL1A2, COL5A1, PDGFRB, COL3A1, COL1A1, SFRP2, COL6A3, PCOLCE, BGN, LUM
##     CDH11, LAMC3, MXRA8, NOTCH3, CYGB, COL6A1, THBS2, COL6A2, FN1, EDNRA
##     AEBP1, DCN, COL5A2, MMP11, MSC, CD248, PLXDC1, ISLR, GPNMB, LAMA2
## Negative:  RGCC, FLT1, PCAT19, PECAM1, PODXL, ELTD1, KDR, PLVAP, ERG, ESAM
##     CLEC14A, S1PR1, MYCT1, VWF, MMP1, FAM101B, CDH5, IFI27, ECSCR, CXCR4
##     GPR4, PRDM1, PASK, ACVRL1, F2RL3, CD36, TM4SF18, DYSF, CALCRL, MMRN2
## PC_ 5
## Positive:  HADH, NPTX2, PCSK1, PFKFB2, C1orf127, FAM159B, RASD1, ADCYAP1, PARM1, G6PC2
##     FAM105A, ELMO1, SLC39A14, CASR, NDRG4, MEG3, DHRS2, PDX1, GSN, C1QL1
##     ENTPD3, IGSF1, HEPACAM2, IAPP, ATP1B1, PEMT, SLC6A6, BMP5, TIMP1, TMEM37
## Negative:  ERCC-468.75:mix1-937.5:mix2, AXL, ERCC-468.75:mix1-468.75:mix2, ERCC-29.296875:mix1-43.945
##     ERCC-117.1875:mix1-175.78125:mix2, ERCC-468.75:mix1-117.1875:mix2, PPY, ERCC-117.1875:mix1-234.37
##     ERCC-58.59375:mix1-58.59375:mix2, OLFM4, CRP, LCN2, SERPINA3, PIGR, DEFB1, SPARC, PLVAP, CEACAM7
```

```r
seger <- RunUMAP(seger, dims = 1:10)
```

```
## 12:28:13 UMAP embedding parameters a = 0.9922 b = 1.112
## 12:28:13 Read 1871 rows and found 10 numeric columns
```

```
## 12:28:13 Using Annoy for neighbor search, n_neighbors = 30
## 12:28:13 Building Annoy index with metric = cosine, n_trees = 50
## 0%   10   20   30   40   50   60   70   80   90   100%
## [----|----|----|----|----|----|----|----|----|----|
## **************************************************|
## 12:28:14 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc1dd
## 12:28:14 Searching Annoy index using 1 thread, search_k = 3000
## 12:28:14 Annoy recall = 100%
## 12:28:14 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:28:15 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:28:15 Commencing optimization for 500 epochs, with 73270 positive edges
## 12:28:19 Optimization finished
```

```r
seger$datasource <- "Segerstolpe (Smart-seq2)"
seurat_baron$datasource <- "Baron (inDrop)"
colnames(seger@meta.data)
```

```
##  [1] "orig.ident"                    "nCount_RNA"
##  [3] "nFeature_RNA"                  "Source Name"
##  [5] "Characteristics [organism]"    "Characteristics [individual]"
##  [7] "Characteristics [sex]"         "Characteristics [age]"
##  [9] "Unit [time unit]"              "percent.mt"
## [11] "individual"                    "predicted.celltype"
## [13] "datasource"
```

```r
colnames(seurat_baron@meta.data)
```

```
## [1] "orig.ident"       "nCount_RNA"       "nFeature_RNA"     "cell"
## [5] "assigned_cluster" "celltype"         "individual"       "percent.mt"
## [9] "datasource"
```

```r
seger$predicted.celltype <- case_when(
  seger@meta.data$`Source Name` %in% c("Beta cells") ~ "Beta",
  seger@meta.data$`Source Name` %in% c("Alpha cells") ~ "Alpha",
  seger@meta.data$`Source Name` %in% c("Delta cells") ~ "Delta",
  TRUE ~ "Unknown"
)

seger$celltype_common <- seger$predicted.celltype
seurat_baron$celltype_common <- seurat_baron$assigned_cluster
merged_naive <- merge(seurat_baron, y = seger, add.cell.ids = c("Baron", "Seger"))
merged_naive <- NormalizeData(merged_naive)
```

```
## Normalizing layer: counts.baron
## Normalizing layer: counts.seger
```

```r
merged_naive <- FindVariableFeatures(merged_naive, selection.method = "vst", nfeatures = 2000)
```

```
## Finding variable features for layer counts.baron
## Finding variable features for layer counts.seger
```

```r
merged_naive <- ScaleData(merged_naive)
```

```
## Centering and scaling data matrix
```

```r
merged_naive <- RunPCA(merged_naive, npcs = 30)
```

```
## PC_ 1
```

```
## Positive:  IFITM3, PMEPA1, COL18A1, IFITM2, CDC42EP1, TMSB10, ZFP36L1, SOX4, SERPING1, RHOC
##     FLNA, MMP14, TMSB4X, YBX3, TACSTD2, MYH9, LGALS3, MSN, S100A16, ITGA5
##     S100A11, LITAF, LGALS1, KRT7, SERPINH1, HSPG2, CYR61, MARCKSL1, TPM1, SDC4
## Negative:  CHGA, CHGB, PEMT, PCSK2, VGF, SLC30A8, GAD2, NEUROD1, ABCC8, MAFB
##     C1QL1, PCP4, STX1A, PCSK1, G6PC2, PAPSS2, C2CD4A, INSM1, ALDH1A1, IAPP
##     STMN2, UCN3, CFC1, RGS9, C2CD4B, PEG10, RASD1, RGS4, SCGB2A1, RFX6
## PC_ 2
## Positive:  CD24, SERPINA3, KRT8, TACSTD2, KRT18, PRSS8, KRT7, LCN2, SDC4, CLDN4
##     PDZK1IP1, CFB, ELF3, CD44, MUC1, KRT19, SERINC2, LAD1, GATM, CLDN1
##     SPINK1, IL32, C3, DEFB1, ANXA4, REG1A, KIAA1522, ABCC3, GDF15, SERPINA5
## Negative:  SPARC, COL4A1, COL6A2, BGN, PDGFRB, IGFBP4, PXDN, COL1A2, COL3A1, ENG
##     COL15A1, COL6A3, AEBP1, COL1A1, CYGB, COL5A1, COL6A1, ITGA1, TIMP3, EMILIN1
##     LRRC32, MMP2, HTRA3, COL5A2, COL5A3, CRLF1, CRISPLD2, LAMA4, SFRP2, MFGE8
## PC_ 3
## Positive:  CTRC, CELA3B, CTRB1, CPA1, CELA3A, CPA2, PLA2G1B, CELA2A, PNLIPRP1, CTRB2
##     CLPS, KLK1, PRSS1, CPB1, SYCN, PRSS2, PNLIP, REG1B, PNLIPRP2, GP2
##     PRSS3, REG1A, SPINK1, CELA2B, CTRL, MGST1, REG1P, FOXD2, PLTP, PDIA2
## Negative:  CFTR, TINAGL1, TSPAN8, SLC4A4, MMP7, AQP1, PPAP2C, ATP1A1, LGALS4, SERPINA5
##     VWA1, CTSH, ANXA4, SPP1, CD74, SERPINA1, AGRN, KRT19, TRPV6, ALDH1A3
##     ONECUT2, SLC3A1, VTCN1, FXYD2, SFRP5, MUC20, CMTM7, CLDN10, HSD17B2, PIGR
## PC_ 4
## Positive:  COL1A1, COL1A2, COL6A3, COL3A1, COL5A1, PDGFRB, BGN, NOTCH3, CRLF1, SFRP2
##     THBS2, COL6A1, FMOD, COL5A2, CYGB, FN1, EMILIN1, VCAN, MXRA8, SPON2
##     LUM, LAMC3, ADAMTS12, C1S, COL5A3, COL6A2, AEBP1, VSTM4, TPM2, CRISPLD2
## Negative:  PLVAP, PECAM1, FLT1, CD93, PODXL, KDR, RGCC, PCAT19, SOX18, ACVRL1
##     VWF, ROBO4, ESM1, TIE1, ESAM, ECSCR, S1PR1, PASK, SEMA3F, CXCR4
##     ERG, CLDN5, DYSF, CDH5, NOTCH4, ANGPT2, IFI27, BCL6B, CLEC14A, PTPRB
## PC_ 5
## Positive:  LAPTM5, TYROBP, FCER1G, CD53, SPI1, ACP5, ALOX5AP, LCP1, ITGB2, C1QC
##     SDS, CD300A, LSP1, RGS1, SRGN, C1QB, CSF1R, APOE, FGL2, TNFRSF1B
##     HCK, C1QA, MPEG1, TBXAS1, PLA2G7, AIF1, RUNX3, SLC15A3, CORO1A, SLCO2B1
## Negative:  PRSS23, COL4A1, HADH, COL18A1, PXDN, IAPP, PCSK1, SPARC, CALD1, HSPG2
##     ANGPTL2, IGFBP4, TSC22D1, ESAM, LAMA4, PLVAP, KDR, WWTR1, IGFBP7, ADCYAP1
##     IGF2, FLT1, PODXL, SOX4, ID1, COL15A1, VWF, MCAM, DLK1, ROBO4
```

```r
merged_naive <- RunUMAP(merged_naive, dims = 1:10)
```

```
## 12:28:29 UMAP embedding parameters a = 0.9922 b = 1.112
## 12:28:29 Read 10440 rows and found 10 numeric columns
## 12:28:29 Using Annoy for neighbor search, n_neighbors = 30
## 12:28:29 Building Annoy index with metric = cosine, n_trees = 50
## 0%   10   20   30   40   50   60   70   80   90   100%
## [----|----|----|----|----|----|----|----|----|----|
## **************************************************|
## 12:28:30 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc5043
## 12:28:30 Searching Annoy index using 1 thread, search_k = 3000
## 12:28:32 Annoy recall = 100%
## 12:28:33 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:28:34 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:28:34 Commencing optimization for 200 epochs, with 425042 positive edges
## 12:28:44 Optimization finished
```

```r
library(ggplot2)
p_before <- DimPlot(merged_naive, reduction = "umap", group.by = "datasource", label = TRUE) +
  ggtitle("Before Integration: Clustering by Protocol")
```

```r
anchors <- FindIntegrationAnchors(
  object.list = list(seurat_baron, seger),
  dims = 1:10,  # Increased from 1:10 for better feature alignment
  reduction = "cca"
)
```

```
## Computing 2000 integration features
## Scaling features for provided objects
```

```
## Warning: Different features in new layer data than already exists for
## scale.data
```

```
## Warning: Different features in new layer data than already exists for
## scale.data
```

```
## Finding all pairwise anchors
## Running CCA
## Merging objects
## Finding neighborhoods
## Finding anchors
##  Found 5803 anchors
## Filtering anchors
##  Retained 4348 anchors
```

```r
combined <- IntegrateData(
  anchorset = anchors,
  dims = 1:10,
  k.weight = 20  # Increased from 20 to 50 for better mixing
)
```

```
## Merging dataset 2 into 1
## Extracting anchors for merged samples
## Finding integration vectors
## Finding integration vector weights
## Integrating data
```

```
## Warning: Layer counts isn't present in the assay object; returning NULL
```

```r
DefaultAssay(combined) <- "integrated"
combined <- ScaleData(combined, verbose = FALSE)
combined <- RunPCA(combined, npcs = 30, verbose = FALSE)
combined <- RunUMAP(combined, reduction = "pca", dims = 1:20)
```

```
## 12:29:58 UMAP embedding parameters a = 0.9922 b = 1.112
## 12:29:58 Read 10440 rows and found 20 numeric columns
## 12:29:58 Using Annoy for neighbor search, n_neighbors = 30
## 12:29:58 Building Annoy index with metric = cosine, n_trees = 50
## 0%   10   20   30   40   50   60   70   80   90   100%
## [----|----|----|----|----|----|----|----|----|----|
## **************************************************|
## 12:29:59 Writing NN index file to temp file C:\Users\sushm\AppData\Local\Temp\Rtmp2pp7AL\fileedbc1c5
## 12:29:59 Searching Annoy index using 1 thread, search_k = 3000
## 12:30:01 Annoy recall = 100%
## 12:30:01 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:30:02 Initializing from normalized Laplacian + noise (using RSpectra)
## 12:30:03 Commencing optimization for 200 epochs, with 444672 positive edges
```

```
## 12:30:13 Optimization finished
```

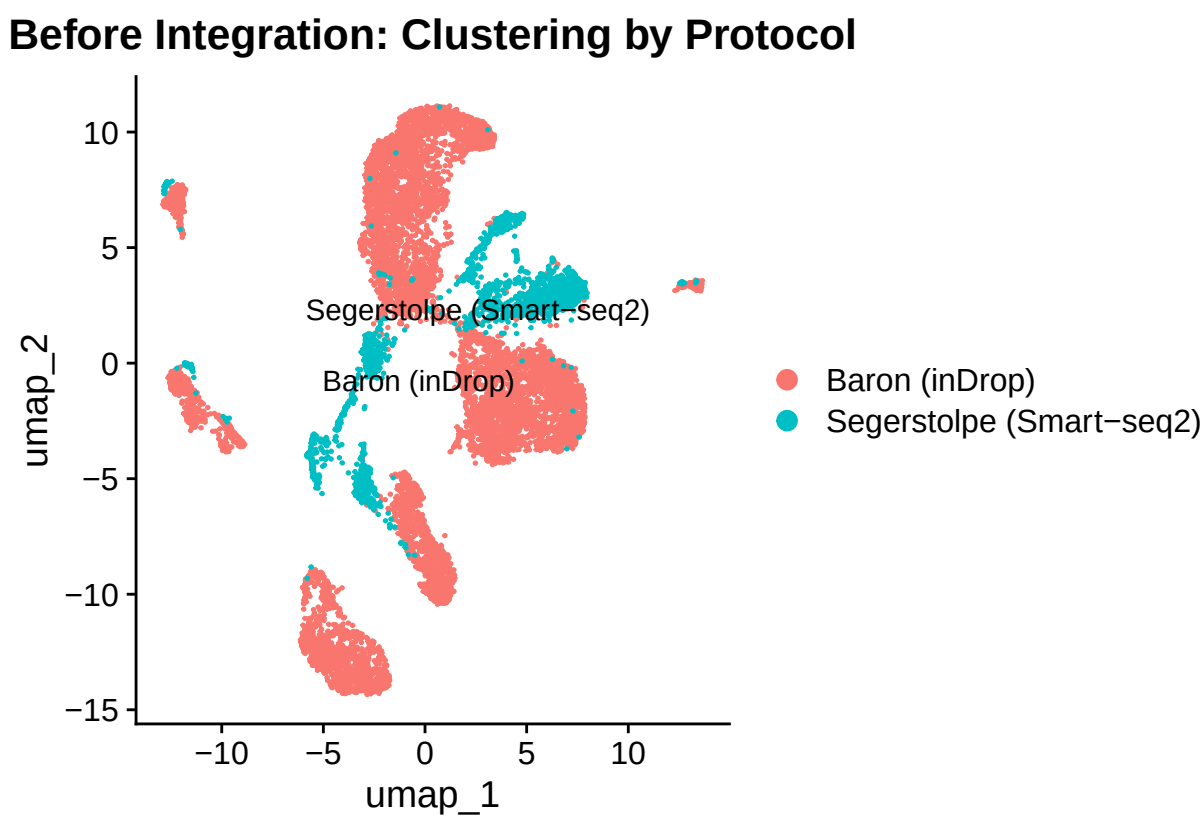
```r
combined <- FindNeighbors(combined, reduction = "pca", dims = 1:20)
```

```
## Computing nearest neighbor graph
## Computing SNN
```

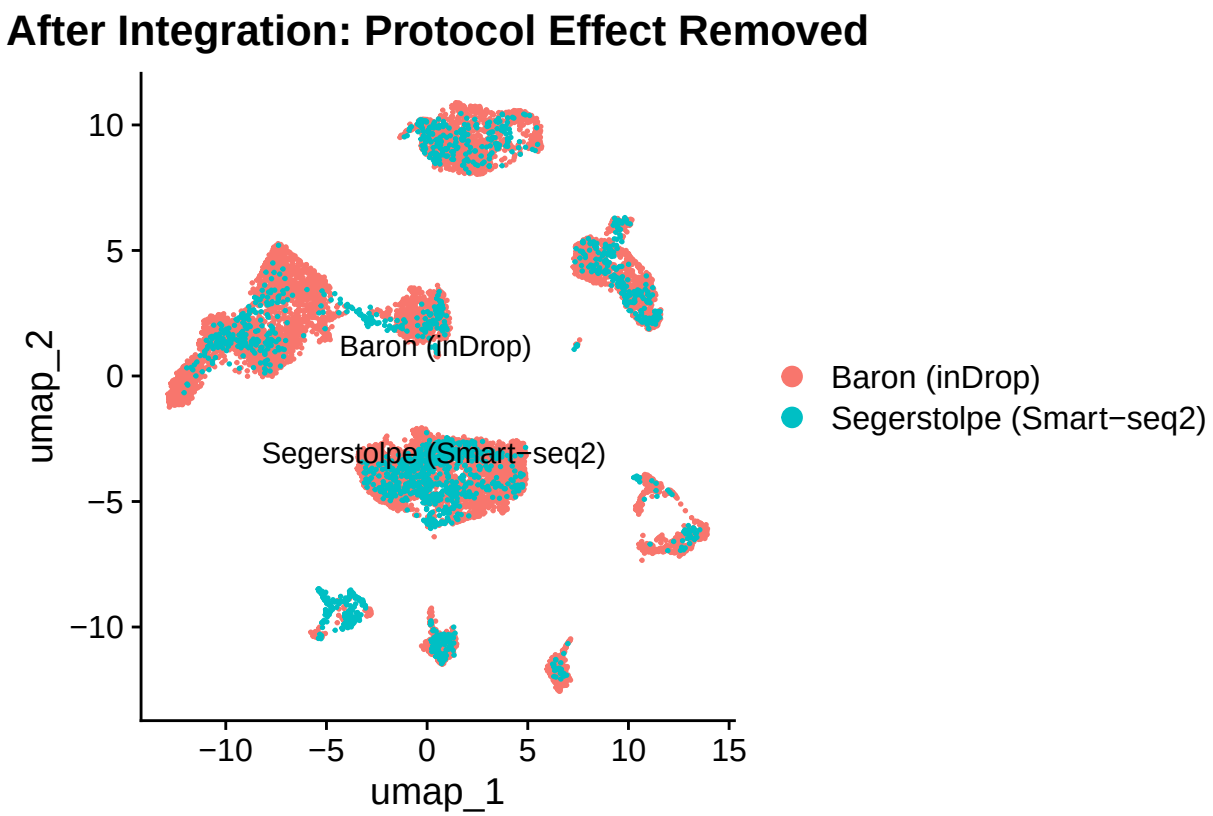
```r
combined <- FindClusters(combined, resolution = 0.8)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 10440
## Number of edges: 397756
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8989
## Number of communities: 21
## Elapsed time: 0 seconds
```

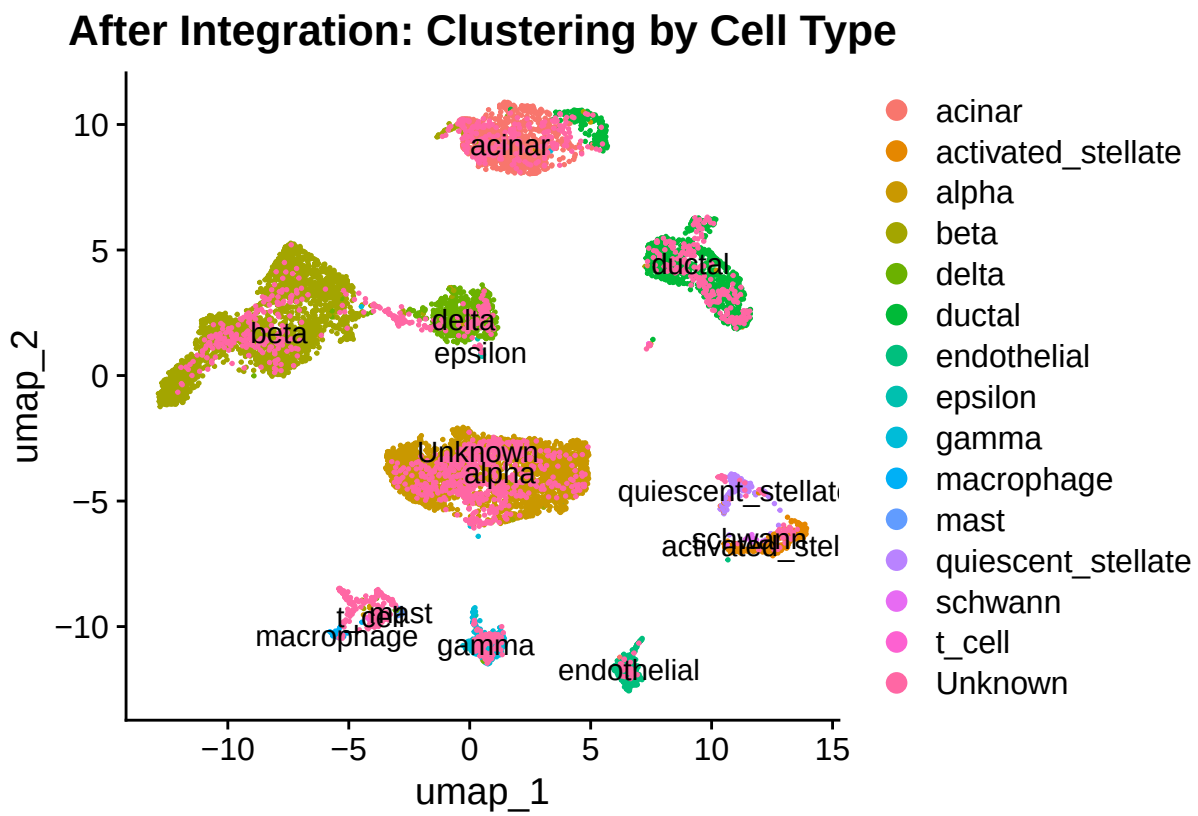
```r
p_after <- DimPlot(combined, reduction = "umap", group.by = "datasource", label = TRUE) +
  ggtitle("After Integration: Protocol Effect Removed")
p_celltype <- DimPlot(combined, reduction = "umap", group.by = "celltype_common", label = TRUE) +
  ggtitle("After Integration: Clustering by Cell Type")
p_before
```



```
p_after
```




```
p_celltype
```




```
anchors_transfer <- FindTransferAnchors(
  reference = combined,
  query = seger,
  dims = 1:10,
  reference.reduction = "pca",
  recompute.residuals = FALSE
)
```

```
## Projecting cell embeddings
## Finding neighborhoods
## Finding anchors
##  Found 4606 anchors
```

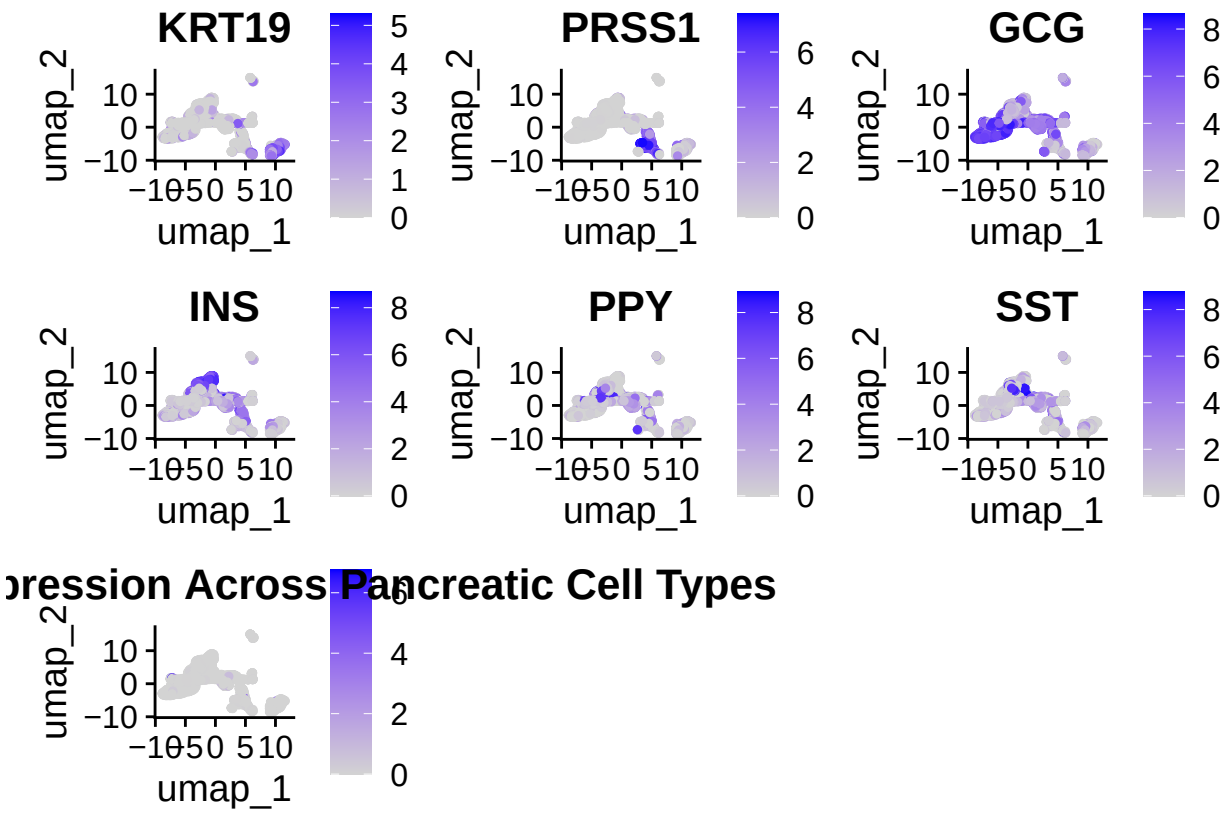
```
seger <- MapQuery(
  anchorset = anchors_transfer,
  query = seger,
  reference = combined,
  refdata = list(predicted.celltype = "celltype_common"),
  reference.reduction = "pca"
)
```

```
## Finding integration vectors
## Finding integration vector weights
## Predicting cell labels
```

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

```
## Warning: Layer counts isn't present in the assay object; returning NULL
## Warning: Layer counts isn't present in the assay object; returning NULL
## Warning: Layer counts isn't present in the assay object; returning NULL

##
## Integrating dataset 2 with reference dataset
## Finding integration vectors
## Integrating data
```

```
Idents(seger) <- "individual"
marker_genes <- c("KRT19", "PRSS1", "GCG", "INS", "PPY", "SST", "GHRL")
FeaturePlot(seger, features = marker_genes, reduction = "umap", ncol = 3) +
  ggtitle("Marker Gene Expression Across Pancreatic Cell Types")
```



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.