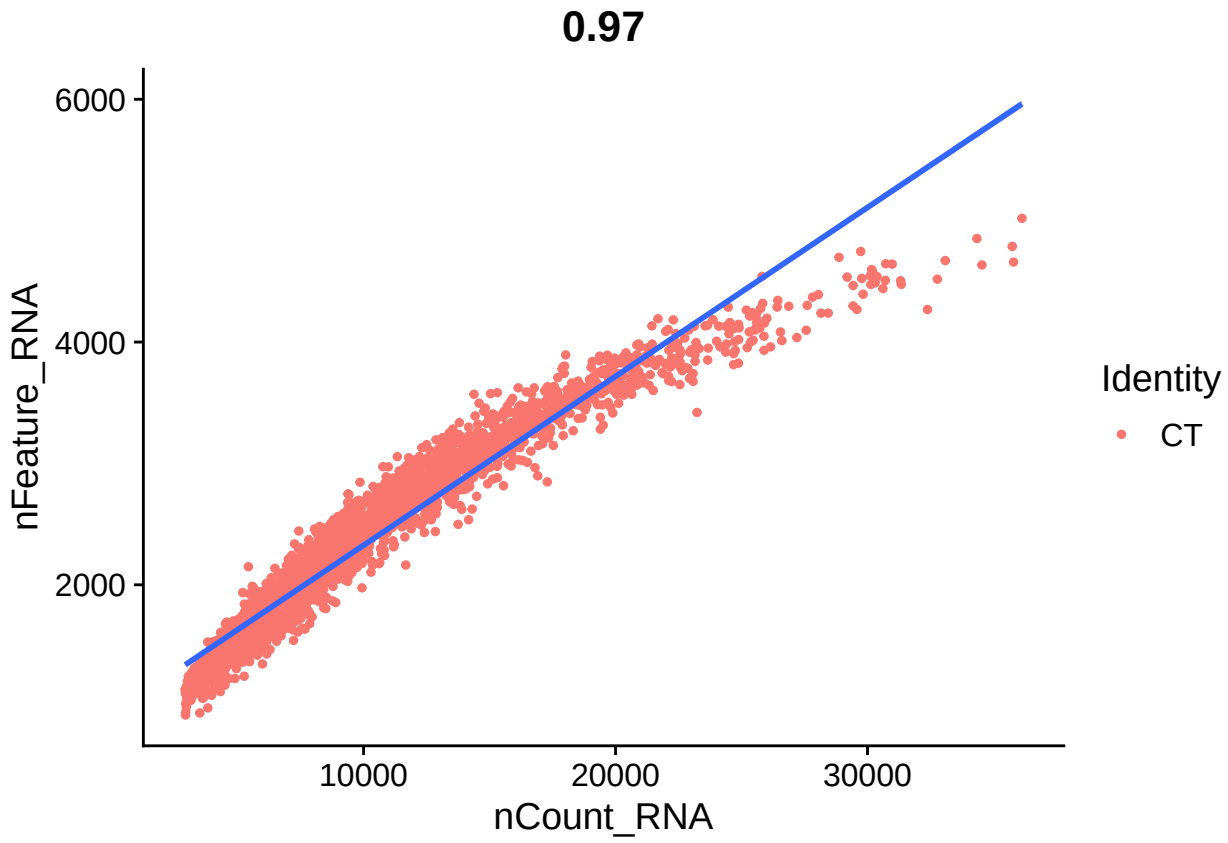
ScRNA-Seq

Chris Doan

2022-07-16

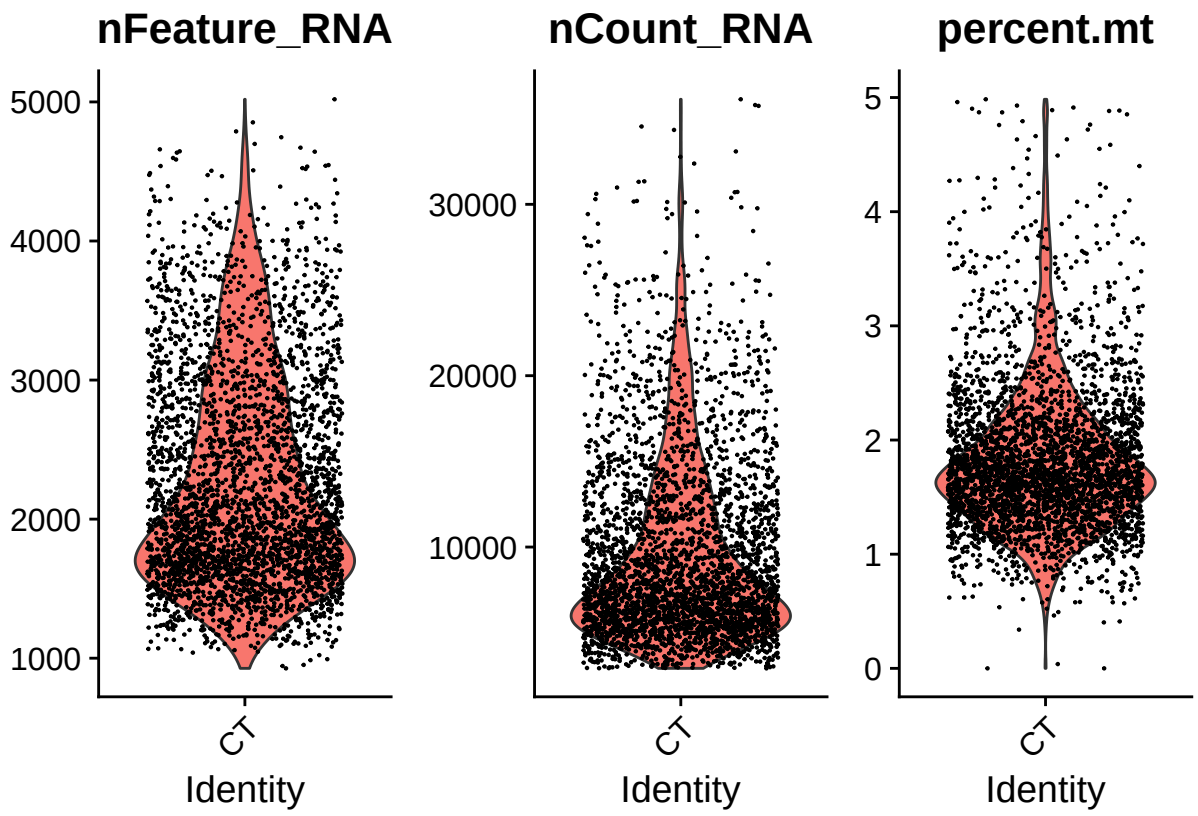
```
dirs <- list.dirs(path = 'intergrate_sc_RNA_Seq_data', recursive = F, full.names = F)

mtx_obj_s5 <- ReadMtx(mtx = 'GSE137015_RAW/S5_filtered_gene_bc_matrices/matrix.mtx',
        features = 'GSE137015_RAW/S5_filtered_gene_bc_matrices/genes.tsv',
        cells = 'GSE137015_RAW/S5_filtered_gene_bc_matrices/barcodes.tsv')
seurat_mtx_s5 <- CreateSeuratObject(counts = mtx_obj_s5, project = 'CT', min.cells = 5)
seurat_mtx_s5[['percent.mt']] <- PercentageFeatureSet(seurat_mtx_s5, pattern = '^mt-')

seurat_mtx_s5 <- subset(seurat_mtx_s5, subset = nCount_RNA < 40000 &
                                    nFeature_RNA > 500 &
                                    percent.mt <5)

seurat_mtx_s5 <- NormalizeData(object = seurat_mtx_s5)
seurat_mtx_s5 <- FindVariableFeatures(object = seurat_mtx_s5)
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
VlnPlot(seurat_mtx_s5, features = c('nFeature_RNA','nCount_RNA','percent.mt'), ncol = 3) +
  geom_smooth(method = 'lm')
```

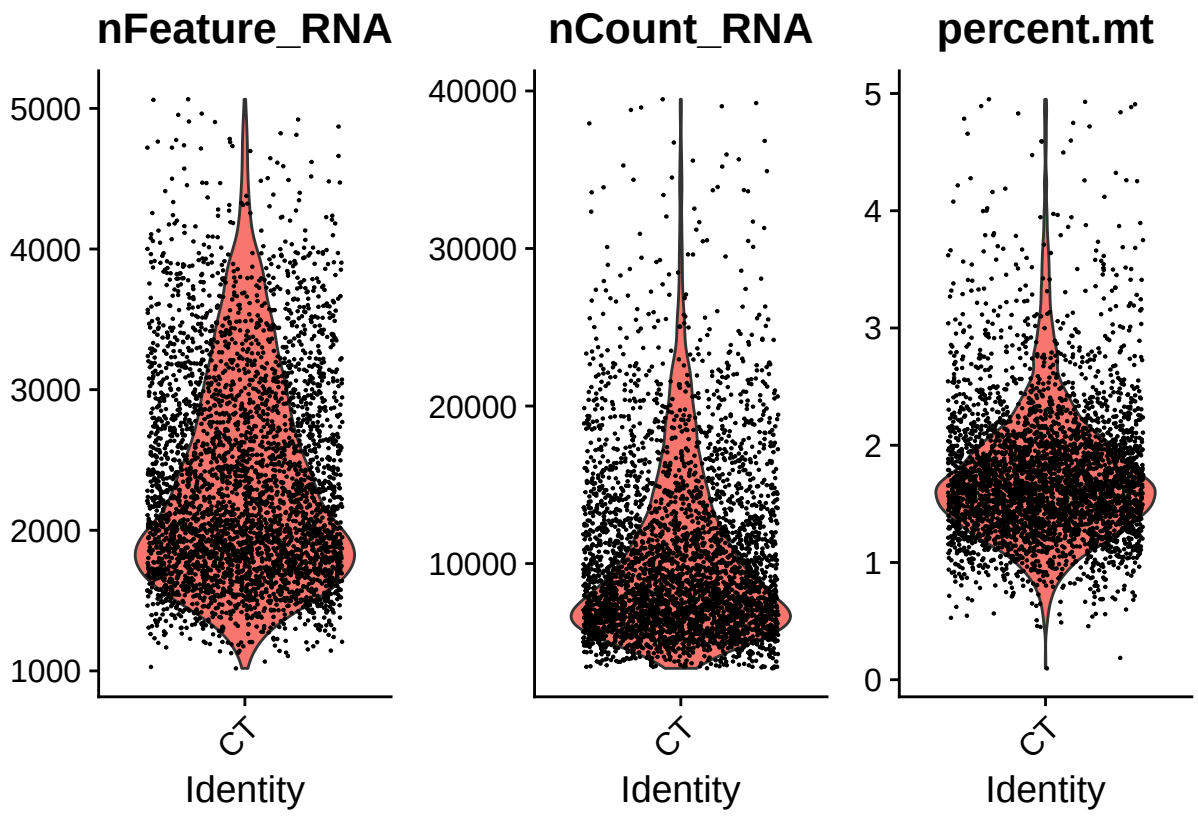
```
## `geom_smooth()` using formula 'y ~ x'
```




```
mtx_obj_s6 <- ReadMtx(mtx = 'GSE137015_RAW/S6_filtered_gene_bc_matrices/matrix.mtx',
                      features = 'GSE137015_RAW/S6_filtered_gene_bc_matrices/genes.tsv',
                      cells = 'GSE137015_RAW/S6_filtered_gene_bc_matrices/barcodes.tsv')
seurat_mtx_s6 <- CreateSeuratObject(counts = mtx_obj_s6, project = 'CT', min.cells = 5)
seurat_mtx_s6[['percent.mt']] <- PercentageFeatureSet(seurat_mtx_s6, pattern = '^mt-')
seurat_mtx_s6 <- subset(seurat_mtx_s6, subset = nCount_RNA < 40000 &
                                    nFeature_RNA > 500 &
                                    percent.mt <5)

seurat_mtx_s6 <- NormalizeData(object = seurat_mtx_s6)
seurat_mtx_s6 <- FindVariableFeatures(object = seurat_mtx_s6)
```

```
VlnPlot(seurat_mtx_s6, features = c('nFeature_RNA','nCount_RNA','percent.mt'), ncol = 3) +
  geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
mtx_obj_s7 <- ReadMtx(mtx = 'GSE137015_RAW/S7_filtered_gene_bc_matrices/matrix.mtx',
                      features = 'GSE137015_RAW/S7_filtered_gene_bc_matrices/genes.tsv',
                      cells = 'GSE137015_RAW/S7_filtered_gene_bc_matrices/barcodes.tsv')
seurat_mtx_s7 <- CreateSeuratObject(counts = mtx_obj_s7, project = 'CT', min.cells = 5)
seurat_mtx_s7[['percent.mt']] <- PercentageFeatureSet(seurat_mtx_s7, pattern = '^mt-')
seurat_mtx_s7 <- subset(seurat_mtx_s7, subset = nCount_RNA < 40000 &
                            nFeature_RNA > 500 &
                            percent.mt <5)

seurat_mtx_s7 <- NormalizeData(object = seurat_mtx_s7)
seurat_mtx_s7 <- FindVariableFeatures(object = seurat_mtx_s7)
```

```
mtx_obj_s8 <- ReadMtx(mtx = 'GSE137015_RAW/S8_filtered_gene_bc_matrices/matrix.mtx',
                      features = 'GSE137015_RAW/S8_filtered_gene_bc_matrices/genes.tsv',
                      cells = 'GSE137015_RAW/S8_filtered_gene_bc_matrices/barcodes.tsv')
seurat_mtx_s8 <- CreateSeuratObject(counts = mtx_obj_s8, project = 'CT', min.cells = 5)
seurat_mtx_s8[['percent.mt']] <- PercentageFeatureSet(seurat_mtx_s8, pattern = '^mt-')
seurat_mtx_s8 <- NormalizeData(object = seurat_mtx_s8)
seurat_mtx_s8 <- FindVariableFeatures(object = seurat_mtx_s8)
```

```
merged_seurat <- merge(seurat_mtx_s5, y = c(seurat_mtx_s6, seurat_mtx_s7, seurat_mtx_s8),
                       add.cell.ids = c('s5','s6','s7','s8'),
                       project = 'CT')
```

```
merged_seurat$sample <- rownames(merged_seurat@meta.data)
merged_seurat@meta.data <- separate(merged_seurat@meta.data, col = 'sample', into = c('cell_types','Bar
                                    sep = '_')
merged_seurat[['percent.mt']] <- PercentageFeatureSet(merged_seurat, pattern = '^mt-')

merged_seurat_filtered <- subset(merged_seurat, subset = nCount_RNA < 40000 &
                           nFeature_RNA > 500 &
                           percent.mt <5)
```

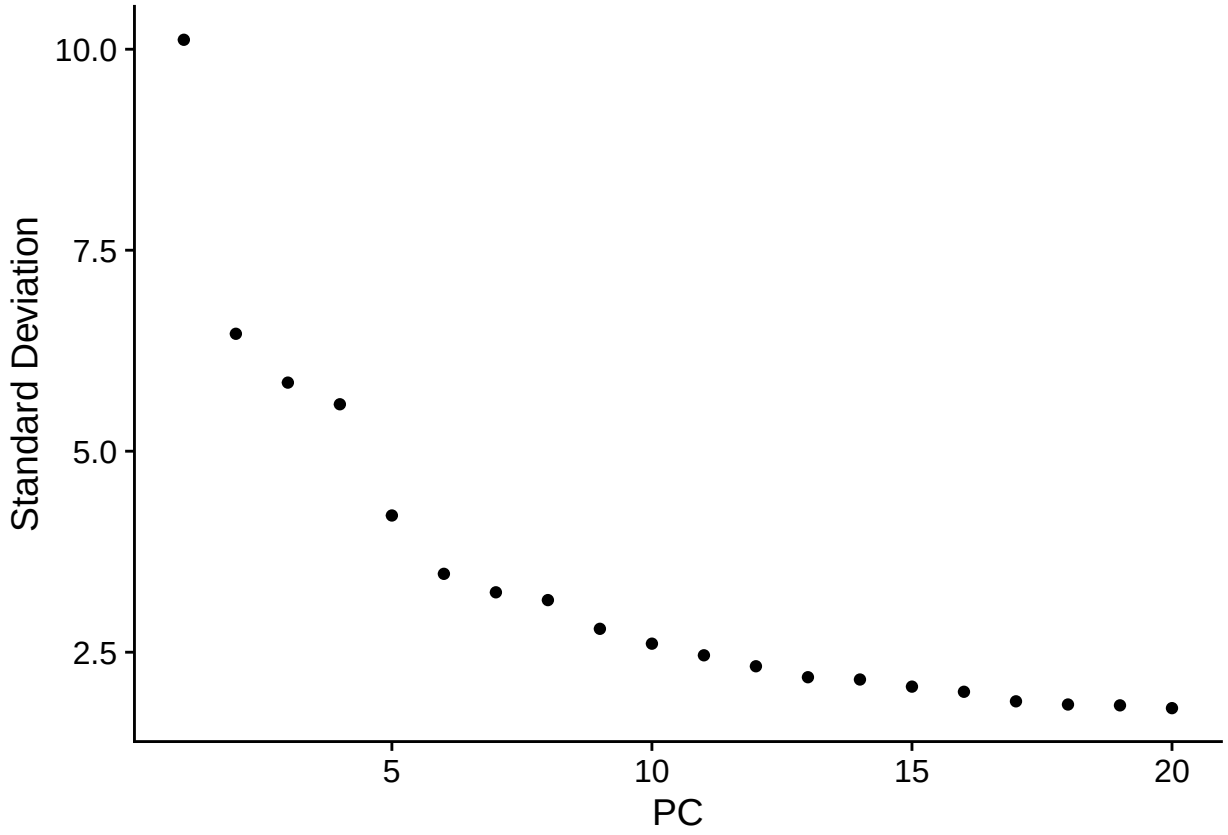
```
merged_seurat_filtered <- NormalizeData(object = merged_seurat_filtered)
merged_seurat_filtered <- FindVariableFeatures(object = merged_seurat_filtered)
merged_seurat_filtered <- ScaleData(object = merged_seurat_filtered)
```

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

```
merged_seurat_filtered <- RunPCA(object = merged_seurat_filtered)
```

```
## PC_ 1
## Positive:  2810417H13Rik, Tuba1b, Stmn1, Ptma, Cks1b, Birc5, Hmgb2, Hist1h2ap, Spc24, Tubb5
##     Smc2, Rrm2, Ran, Dut, Ccna2, Tk1, Cdca8, Mki67, Lmnb1, Tubb4b
##     Asf1b, Cdk1, H2afz, Lig1, Mcm5, Nusap1, Rrm1, Hmgb1, Ube2s, Top2a
## Negative:  Tcf7, Ifi27l2a, 2310001H17Rik, Slamf6, Gm26740, Rgs10, Fos, Ms4a4c, Ifit3, Jun
##     Ctss, Emb, Ccnd2, Id3, Klf2, Pou2f2, Cd200, Junb, Btg2, Izumo1r
##     Sell, Irf7, Pde2a, Asap1, Tnfsf8, Actn1, Ly6c2, Sat1, Zfp36l1, Srgn
## PC_ 2
## Positive:  Npm1, Rps8, Tcf7, Rpl41, Rpl14, Slamf6, Rps2, Eef1g, Rps28, Rpl28
##     Rpl22l1, Rps17, Cd200, Gm10073, 2310001H17Rik, Ubac2, Gm10076, Dapl1, Smc4, Id3
##     Izumo1r, Hmgn1, Ncl, Tnfsf8, Nkg7, Adk, Pim2, Ccr7, 1810041H14Rik, Ehd3
## Negative:  Spi1, Alox5ap, Cd74, Lyz2, Ly86, Wfdc17, H2-Aa, Tyrobp, H2-DMb1, Ccl6
##     Mpeg1, Pld4, H2-Ab1, H2-Eb1, Ctsh, Tgfbi, Aif1, Ifi204, Ms4a6c, Fcgr3
##     Cybb, Ifitm3, Plbd1, Csf1r, Ifitm2, Rnase6, Slamf8, Pirb, Clec4a3, Fcgr1
## PC_ 3
## Positive:  Npm1, Rps17, Rps8, Srm, C1qbp, Rpl10, Fbl, Rpl14, Eef1g, Apex1
##     Nop10, Nolc1, Hspe1, Eif5a, Rps2, Ppp1r14b, Nhp2, Ncl, Gnl3, Rpl28
##     Mif, Nme1, Xcl1, Atp5g1, Bzw2, Hspa9, Pa2g4, Mrto4, Mybbp1a, Rrp15
## Negative:  AW112010, Tmsb4x, Ccl5, Nkg7, Arl6ip1, Ube2c, Cxcr6, Ctsd, Cenpf, S100a4
##     Actg1, S100a11, Lgals1, Cdc20, Cdkn3, Fam64a, Glrx, Gzmb, Nusap1, Emp3
##     Hmmr, Cenpe, Vim, S100a6, Lgals3, H2afv, Klrd1, Ccnb2, Crip1, Id2
## PC_ 4
## Positive:  Tcf7, Smc4, Sell, Id3, Dapl1, Ms4a4c, Cdkn2d, Slamf6, Nusap1, Cenpf
##     Cdc25b, Asap1, H2afx, Cd200, Ccna2, Plk1, Emb, Ccr7, Rgs10, Izumo1r
##     Rps28, Ubac2, Klf2, Mxd3, Pdlim1, Kif22, Gm26740, Cdk1, Cenpe, Hmmr
## Negative:  Gzmb, Cxcr6, S100a4, Lgals1, S100a6, S100a11, Actg1, Ccl3, Ccl4, Ctsd
##     Ifng, AW112010, Glrx, Pdcd1, Aldoa, Havcr2, Il13, Sub1, Cdkn2a, Id2
##     Lgals3, Tmsb4x, Tnfrsf9, Nkg7, Epas1, Irak3, 2310031A07Rik, Mt1, Gzma, Bcl2a1d
## PC_ 5
## Positive:  Klrk1, Klrc1, Klrd1, Ly6c2, Ccl5, Plac8, Rbm3, Nkg7, Klrc2, Pfn1
##     Hspa8, Cd7, Ppia, Amica1, Klre1, Calm1, Cfl1, Gm156, Ptma, Prdx1
##     Klra1, Hnrnpa3, Dnajc9, Ifi27l2a, Ccr2, Emp3, Tmsb4x, Lilr4b, Lilrb4a, Itgb1
## Negative:  Pou2f2, Pdzk1ip1, Batf, Icos, Serp1, Ppic, Irak3, Gldc, 2310001H17Rik, Epas1
##     Tnfrsf13b, Gm10076, Cdkn2a, Kirrel3, Uba52, Mt1, Gm10073, Cd226, Siah2, Il13
##     Tmem154, mt-Atp6, Tnfsf8, Furin, Slamf6, Rel, Ociad2, Eea1, Art3, Maf
```

```
ElbowPlot(merged_seurat_filtered)
```




```
merged_seurat_filtered <- FindNeighbors(object = merged_seurat_filtered, dims = 1:20)
```

```
## Computing nearest neighbor graph

## Computing SNN
```

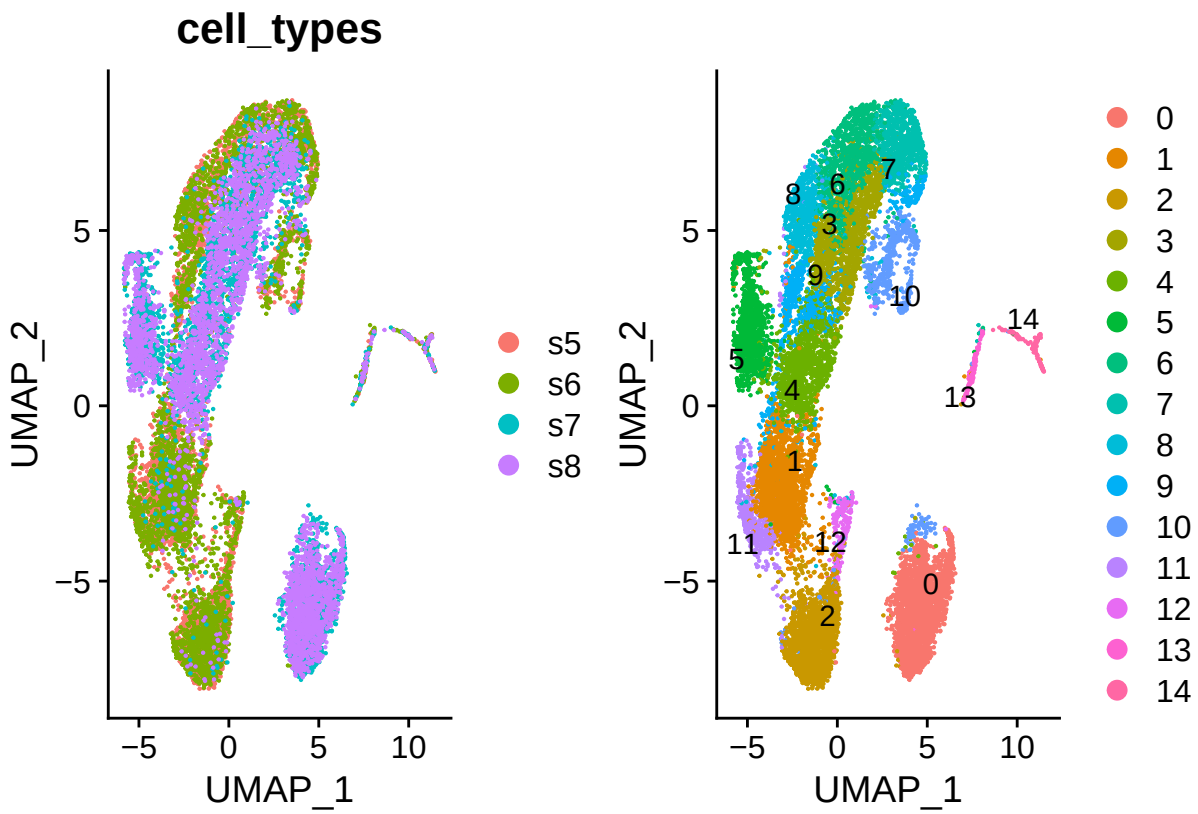
```
merged_seurat_filtered <- FindClusters(object = merged_seurat_filtered)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 14358
## Number of edges: 485792
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8542
## Number of communities: 15
## Elapsed time: 2 seconds
```

```
merged_seurat_filtered <- RunUMAP(object = merged_seurat_filtered, dims = 1:20)
```

```
## 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

## 17:00:43 UMAP embedding parameters a = 0.9922 b = 1.112

## 17:00:43 Read 14358 rows and found 20 numeric columns

## 17:00:43 Using Annoy for neighbor search, n_neighbors = 30

## 17:00:43 Building Annoy index with metric = cosine, n_trees = 50

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

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

## **************************************************|
## 17:00:44 Writing NN index file to temp file /var/folders/07/b7qwj15d1zs41kh2whcqhh0c0000gp/T//Rtmp7l
## 17:00:44 Searching Annoy index using 1 thread, search_k = 3000
## 17:00:47 Annoy recall = 100%
## 17:00:50 Commencing smooth kNN distance calibration using 1 thread
## 17:00:51 Initializing from normalized Laplacian + noise
## 17:00:52 Commencing optimization for 200 epochs, with 606466 positive edges
## 17:01:02 Optimization finished
```

```r
p1 <- DimPlot(merged_seurat_filtered, reduction = 'umap', group.by = 'cell_types')
p2 <- DimPlot(merged_seurat_filtered, reduction = "umap", label = TRUE, repel = TRUE)
p1 + p2
```




```
obj.list1 <- SplitObject(merged_seurat_filtered, split.by = 'cell_types')
```

```
features <- SelectIntegrationFeatures(object.list = obj.list1)

features <- SelectIntegrationFeatures(object.list = c(seurat_mtx_s5,seurat_mtx_s6,seurat_mtx_s7,seurat_
anchors <- FindIntegrationAnchors(object.list = obj.list1,
                        anchor.features = features)
```

```
## Scaling features for provided objects

## Finding all pairwise anchors

## Running CCA

## Merging objects

## Finding neighborhoods

## Finding anchors

##  Found 10860 anchors

## Filtering anchors
```

```
##  Retained 6358 anchors
## Running CCA
## Merging objects
## Finding neighborhoods
## Finding anchors
##  Found 10180 anchors
## Filtering anchors
##  Retained 5328 anchors
## Running CCA
## Merging objects
## Finding neighborhoods
## Finding anchors
##  Found 10846 anchors
## Filtering anchors
##  Retained 5400 anchors
## Running CCA
## Merging objects
## Finding neighborhoods
## Finding anchors
##  Found 10459 anchors
## Filtering anchors
##  Retained 5372 anchors
## Running CCA
## Merging objects
```

```
## Finding neighborhoods
## Finding anchors
##  Found 11081 anchors
## Filtering anchors
##  Retained 5729 anchors
## Running CCA
## Merging objects
## Finding neighborhoods
## Finding anchors
##  Found 10568 anchors
## Filtering anchors
##  Retained 6921 anchors
```

```
seurat.integrated <- IntegrateData(anchorset = anchors)
```

```
## Merging dataset 3 into 4
## Extracting anchors for merged samples
## Finding integration vectors
## Finding integration vector weights
## Integrating data
## Merging dataset 1 into 2
## Extracting anchors for merged samples
## Finding integration vectors
## Finding integration vector weights
## Integrating data
## Merging dataset 4 3 into 2 1
## Extracting anchors for merged samples
## Finding integration vectors
## Finding integration vector weights
## Integrating data
```

```
seurat.integrated <- ScaleData(object = seurat.integrated)
```

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

```
seurat.integrated <- RunPCA(object = seurat.integrated)
```

```
## PC_ 1
## Positive:  Tcf7, Ltb, Ms4a4c, Ccnd2, Dapl1, Gm26740, Slamf6, Sell, Ly6c2, Izumo1r
##     Ctss, Jun, Id3, Ifit3, Fos, Sat1, Ccr7, Cd200, Btg2, 2310001H17Rik
##     Rtp4, Junb, Klf2, Zfp36l1, Iigp1, Rgs10, Ifi27l2a, Tespa1, Emb, Ftl1
## Negative:  2810417H13Rik, Stmn1, Tuba1b, Hist1h2ap, Birc5, Spc24, Hmgb2, Ptma, Cks1b, Ccna2
##     Smc2, Tubb5, Rrm2, Cdca8, Mki67, Lmnb1, Nusap1, Asf1b, Ran, Tk1
##     Tubb4b, Top2a, Hmgb1, Cdk1, H2afz, Hmgn2, Ube2s, Dut, Tpx2, Cdca3
## PC_ 2
## Positive:  Cd74, Alox5ap, Spi1, H2-Aa, Lyz2, Mpeg1, H2-DMb1, Ly86, H2-Eb1, H2-Ab1
##     Ctsh, Aif1, Wfdc17, Tyrobp, Plbd1, Tgfbi, Ifi204, Pid1, Ms4a6c, Ccl6
##     Cybb, Pld4, Csf1r, Fcgr3, Rnase6, Fcgr1, AI607873, Ifitm3, Ifitm2, Fcgr2b
## Negative:  Rps18, Nkg7, Rps20, Ltb, Rpl8, Rps8, Rpl41, Rpl14, Rps12, Eef1g
##     Rpl22l1, Rps2, Npm1, Rpl28, Tcf7, Gm10076, Dapl1, Hsp90ab1, Slamf6, Ccnd2
##     Smc4, Izumo1r, Ubac2, Rpl31, Serbp1, Ncl, Ccr7, Bcl2, Id3, Rps17
## PC_ 3
## Positive:  Srm, Npm1, C1qbp, Rps8, Rps17, Rpl14, Nolc1, Rpl8, Eef1g, Eif5a
##     Apex1, Rps2, Fbl, Hspe1, Gnl3, Xcl1, Pa2g4, Rpl31, Nop10, Ranbp1
##     Ung, Mif, Rpl10, Nhp2, Ncl, Nme1, Cdca7, Nop56, Rps18, Hspa9
## Negative:  Arl6ip1, Ube2c, Tmsb4x, Cenpf, AW112010, Cdc20, Hmmr, Cdkn3, Fam64a, Cenpe
##     Ccnb2, Nusap1, Tpx2, Plk1, Aurka, Cdca3, Cenpa, Cdca8, Ccnb1, Knstrn
##     Birc5, Vim, S100a6, Tuba1c, Ccna2, Emp3, Prc1, H2afv, Tubb4b, Racgap1
## PC_ 4
## Positive:  Gzmb, S100a4, Cxcr6, S100a6, S100a11, Ccl5, AW112010, Lgals1, Actg1, Ctsd
##     Pdcd1, Tmsb4x, Nkg7, Lgals3, Glrx, Il2rb, Ccr2, Sub1, Id2, Ccl4
##     Ccl3, Klrd1, Gapdh, Fgl2, Plac8, Cfl1, Pfn1, Txn1, Calm1, Aldoa
## Negative:  Tcf7, Dapl1, Sell, Id3, Slamf6, Rps20, Smc4, Ms4a4c, Ccr7, Rps12
##     Rps18, Rps17, Emb, Cdkn2d, Cd200, Rpl8, Rpl31, Ubac2, Gm26740, Nusap1
##     Myb, Ass1, Plk1, Klf2, Pdlim1, Cenpf, Rpl10, Gm11579, H2afx, Rgs10
## PC_ 5
## Positive:  Cdc20, Xcl1, Ccnb1, Cenpa, Cdkn3, Srgn, Ccnb2, Ube2c, Knstrn, Crtam
##     Cenpf, Cep89, Reep4, Nek2, Bcl2a1b, Hmmr, Tnfrsf9, 2610318N02Rik, Eif5a, Fam64a
##     Utf1, Ckap2, Ccl4, Troap, Cep55, Pfn1, Tuba1c, Sapcd2, Hdgf, Prr11
## Negative:  Lig1, Tcf19, Dhfr, Mcm7, E2f1, Mcm5, Clspn, Mcm3, Rrm2, Prim1
##     Gins2, Cdc6, Slbp, Pcna, Gmnn, Tipin, Rfc3, Chaf1b, Hells, Chaf1a
##     Mcm2, Uhrf1, Tyms, Tk1, Mcm4, Syce2, Rfc2, Orc6, Mcm6, Esco2
```

```
seurat.integrated <- RunUMAP(object = seurat.integrated, dims = 1:50)
```

```
## 17:26:57 UMAP embedding parameters a = 0.9922 b = 1.112
```

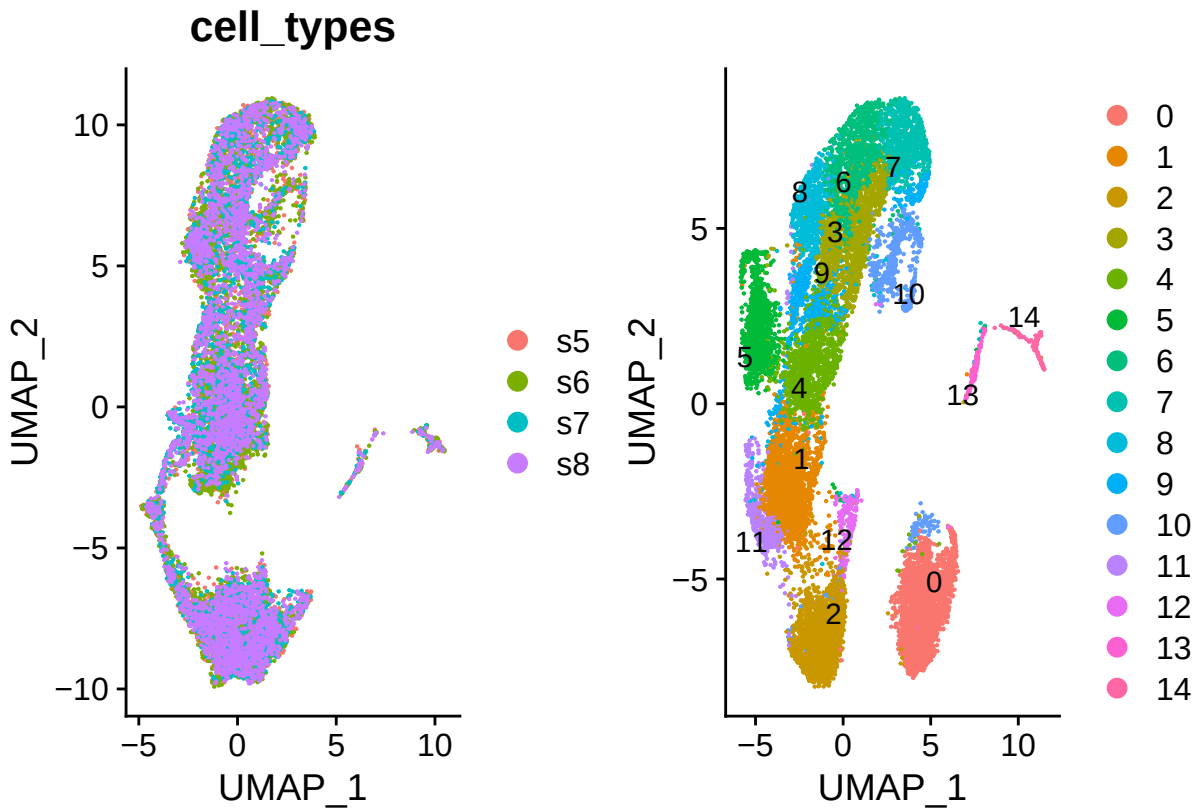
```
## 17:26:57 Read 14358 rows and found 50 numeric columns
```

```
## 17:26:57 Using Annoy for neighbor search, n_neighbors = 30
```

```
## 17:26:57 Building Annoy index with metric = cosine, n_trees = 50
```

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

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

## **************************************************|
## 17:26:59 Writing NN index file to temp file /var/folders/07/b7qwj15d1zs41kh2whcqhh0c0000gp/T//Rtmp7ls
## 17:27:00 Searching Annoy index using 1 thread, search_k = 3000
## 17:27:02 Annoy recall = 100%
## 17:27:03 Commencing smooth kNN distance calibration using 1 thread
## 17:27:05 Initializing from normalized Laplacian + noise
## 17:27:05 Commencing optimization for 200 epochs, with 663672 positive edges
## 17:27:16 Optimization finished
```

```
p5 <- DimPlot(seurat.integrated, reduction = 'umap', group.by = 'cell_types')
p6 <- DimPlot(merged_seurat_filtered, reduction = "umap", label = TRUE, repel = TRUE)
p5+p6
```




```
immune.anchors <- FindIntegrationAnchors(object.list = c(seurat_mtx_s5,seurat_mtx_s6,seurat_mtx_s7,seura
```

```
## Warning in CheckDuplicateCellNames(object.list = object.list): Some cell names
## are duplicated across objects provided. Renaming to enforce unique cell names.

## Scaling features for provided objects
```

```
## Finding all pairwise anchors

## Running CCA

## Merging objects

## Finding neighborhoods

## Finding anchors

##  Found 10860 anchors

## Filtering anchors

##  Retained 6358 anchors

## Running CCA

## Merging objects

## Finding neighborhoods

## Finding anchors

##  Found 10180 anchors

## Filtering anchors

##  Retained 5328 anchors

## Running CCA

## Merging objects

## Finding neighborhoods

## Finding anchors

##  Found 10846 anchors

## Filtering anchors

##  Retained 5400 anchors

## Running CCA

## Merging objects
```

```
## Finding neighborhoods

## Finding anchors

##  Found 10384 anchors

## Filtering anchors

##  Retained 5302 anchors

## Running CCA

## Merging objects

## Finding neighborhoods

## Finding anchors

##  Found 11128 anchors

## Filtering anchors

##  Retained 5424 anchors

## Running CCA

## Merging objects

## Finding neighborhoods

## Finding anchors

##  Found 10679 anchors

## Filtering anchors

##  Retained 6397 anchors
```

```
immune.combined <- IntegrateData(anchorset = anchors)
```

```
## Merging dataset 3 into 4

## Extracting anchors for merged samples

## Finding integration vectors

## Finding integration vector weights
```

```
## Integrating data

## Merging dataset 1 into 2

## Extracting anchors for merged samples

## Finding integration vectors

## Finding integration vector weights

## Integrating data

## Merging dataset 4 3 into 2 1

## Extracting anchors for merged samples

## Finding integration vectors

## Finding integration vector weights

## Integrating data
```

```
immune.combined$sample <- rownames(immune.combined@meta.data)
immune.combined@meta.data <- separate(immune.combined@meta.data, col = 'sample', into = c('cell_types',
                                      sep = '_')
DefaultAssay(immune.combined) <- "integrated"
```

```
immune.combined <- ScaleData(immune.combined, verbose = FALSE)
immune.combined <- RunPCA(immune.combined, npcs = 30, verbose = FALSE)
immune.combined <- RunUMAP(immune.combined, reduction = "pca", dims = 1:30)
```

```
## 17:50:47 UMAP embedding parameters a = 0.9922 b = 1.112

## 17:50:48 Read 14358 rows and found 30 numeric columns

## 17:50:48 Using Annoy for neighbor search, n_neighbors = 30

## 17:50:48 Building Annoy index with metric = cosine, n_trees = 50

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

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

## **************************************************|
## 17:50:49 Writing NN index file to temp file /var/folders/07/b7qwj15d1zs41kh2whcqhh0c0000gp/T//Rtmp7ls
## 17:50:49 Searching Annoy index using 1 thread, search_k = 3000
## 17:50:52 Annoy recall = 100%
## 17:50:52 Commencing smooth kNN distance calibration using 1 thread
## 17:50:54 Initializing from normalized Laplacian + noise
## 17:50:54 Commencing optimization for 200 epochs, with 638636 positive edges
## 17:51:04 Optimization finished
```

```
immune.combined <- FindNeighbors(immune.combined, reduction = "pca", dims = 1:30)
```

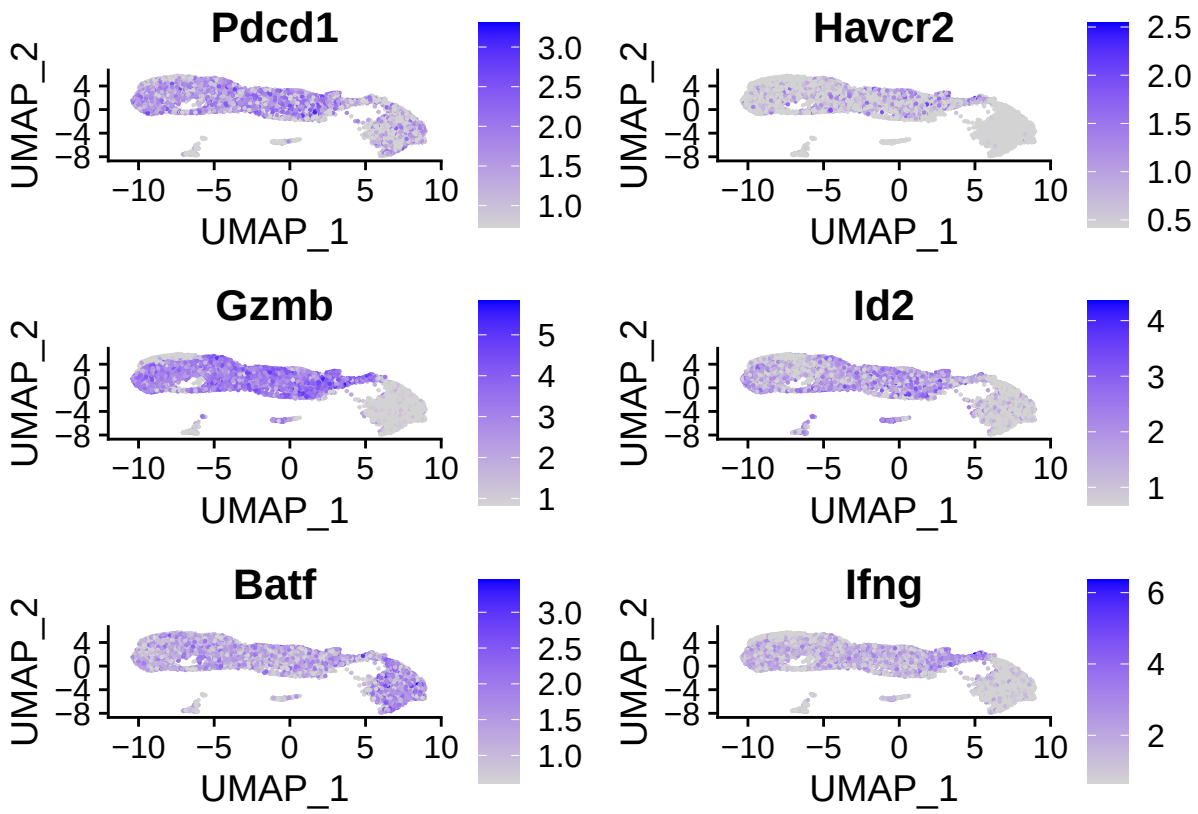
```
## Computing nearest neighbor graph
## Computing SNN
```

```
immune.combined <- FindClusters(immune.combined, resolution = 0.5)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 14358
## Number of edges: 611493
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8716
## Number of communities: 12
## Elapsed time: 2 seconds
```

```
DefaultAssay(immune.combined) <- "RNA"
```

```
FeaturePlot(immune.combined, features = c("Pdcd1", "Havcr2", "Gzmb", "Id2", "Batf", "Ifng"
                                          ), min.cutoff = "q9")
```



Biological conclusions:

After checking quality control, all samples are good for downstream analysis. We have samples from wild-type and knock-out cells so I perform scRNA-Seq integration.

Cells from s5 and s6 have similar sources of variation, and cells from s7 and s8 have similar sources of variation. The algorithms identify a total of 15 clusters of cells. Cells from s8 samples have the highest sources of variation. Many cells from s7 and s8 samples separate from s5 and s6 samples.

From FeaturePlot genes of interest (Pdcd1, Havcr2, Gzmb, Id2, Batf, Ifng), we see different expression levels of these marker genes in all cells from wild-type and knockouts cells. Gzmb gene has the expression level highest.