Complete Spatial scRNA-seq Workflow

Asad

2023-05-28

Call libraries

```r
library(Seurat)
```

```
## Attaching SeuratObject
```

```r
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
```

```r
library(ggplot2)
library(patchwork)
library(Rfast2)
```

```
## Loading required package: Rcpp

##
## Rfast2:  0.1.4

##  ___ __ __ __ __    __ __ __ __ __ _             _               __ __ __ __ __     __ __ __ __ __ __
## |  __ __ __ __  |  |  __ __ __ __ _/            / \             |  __ __ __ __ /   /__ __ _   _ __ __
## | |           | |  | |                         / _ \            | |                        / /
## | |           | |  | |                        / / \ \           | |                       / /
## | |           | |  | |                       / /   \ \          | |                      / /
## | |__ __ __ __| |  | |__ __ __ __           / /     \ \         | |__ __ __ __ _        / /__/\
## |    __ __ __ __|  |  __ __ __ __|         / /__ _ __\ \        |_ __ __ __ _   |      / ___  /
## |   \              | |                    / _ _ _ _ _ _ \                     | |      \/  / /
## | |\ \             | |                   / /           \ \                    | |         / /
## | | \ \            | |                  / /             \ \                   | |        / /
## | |  \ \           | |                 / /               \ \                  | |       / /
## | |   \ \__ __ _   | |                / /                 \ \     _ __ __ __ _| |      / /
## |_|    \__ __ __\  |_|               /_/                   \_\   /_ __ __ __ ___|      \/
```

```
setwd('E:/scRNA-seq/Spatial scRNAseq/Dataset')
```

Load dataset

```
lung.data<- Load10X_Spatial(data.dir = 'E:/scRNA-seq/Spatial scRNAseq/Dataset',
                            filename = 'raw_feature_bc_matrix.h5',
                            assay = "Spatial", # Name of the assay
                            slice = "Lung", # Name of the test image
                            filter.matrix = TRUE,
                            to.upper = FALSE)
```

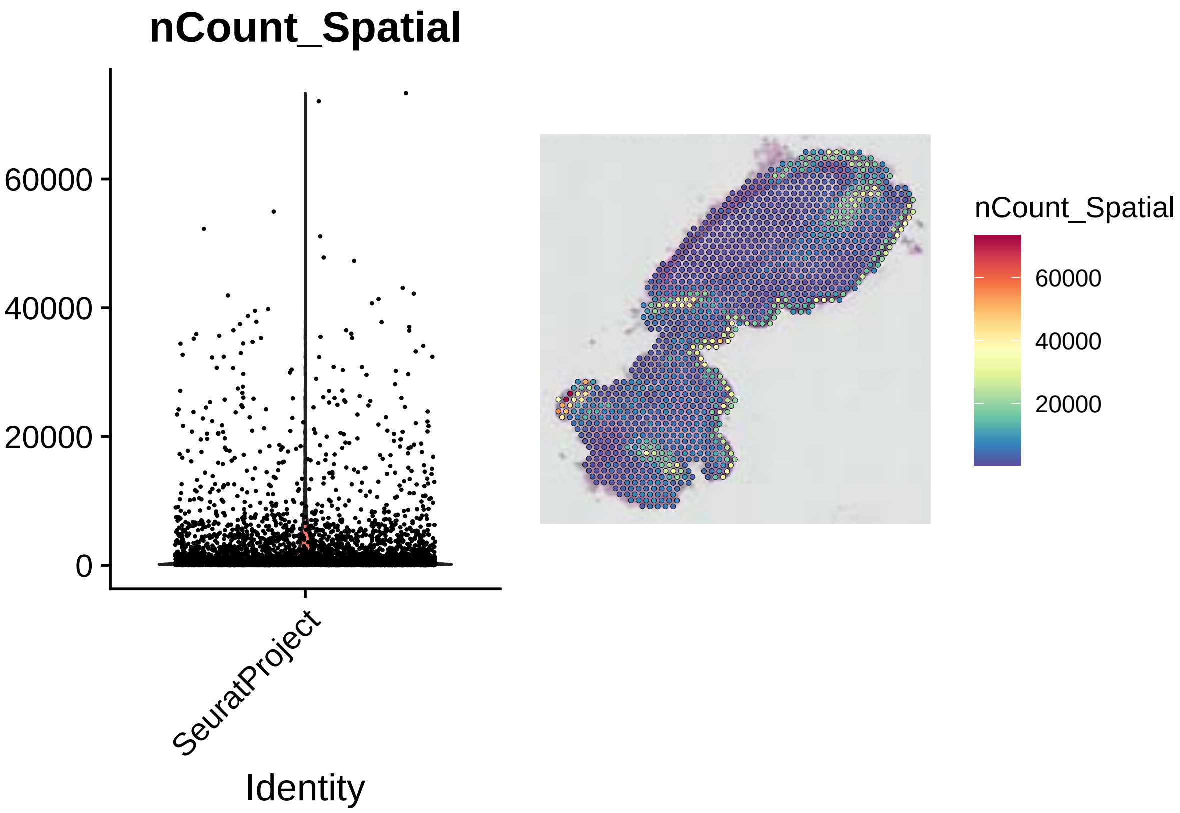
Initial Assessment

#nFeature_Spatial: the number of unique genes in each sample #nCount_Spatial: the total number of detected molecules in each sample lung.data #Inspect metadata View(lung.data@meta.data) #Inspect dimension dim(lung.data) # 33601 features across 4992 samples #Check feature names head(rownames(lung.data), n = 5) lung.data@assays$Spatial@counts[5:10, 1:3]

```

plot1 <- VlnPlot(lung.data, features = "nCount_Spatial", pt.size = 0.1) + NoLegend()
plot2 <- SpatialFeaturePlot(lung.data, features = "nCount_Spatial") +
        theme(legend.position = "right")

wrap_plots(plot1, plot2)
```



Plot

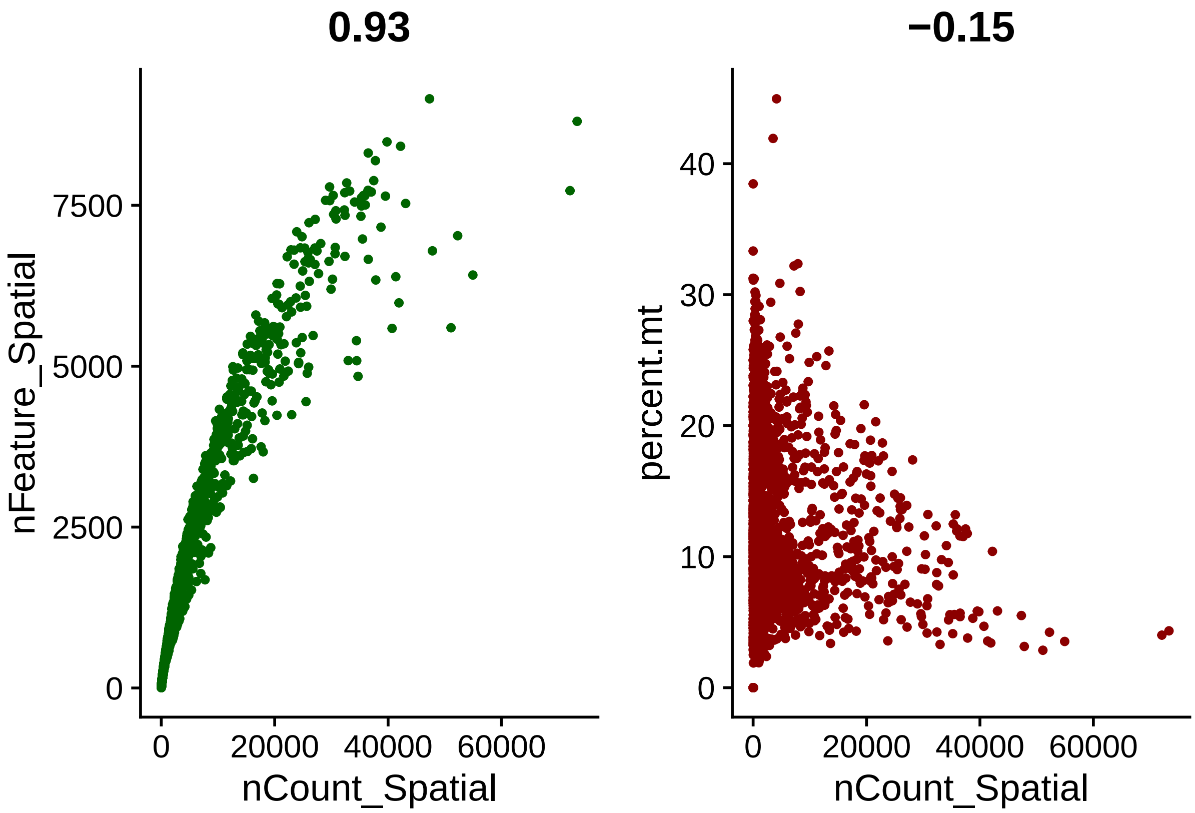
```
#Check mitochondrial RNA percentage
lung.data[["percent.mt"]] <- PercentageFeatureSet(lung.data, pattern = "^MT-")

#Check ribosomal RNA percentage
lung.data[["percent.rb"]] <- PercentageFeatureSet(lung.data, pattern = "^RP[SL]")
View(lung.data@meta.data)
```

Assessing correlation between assay variables

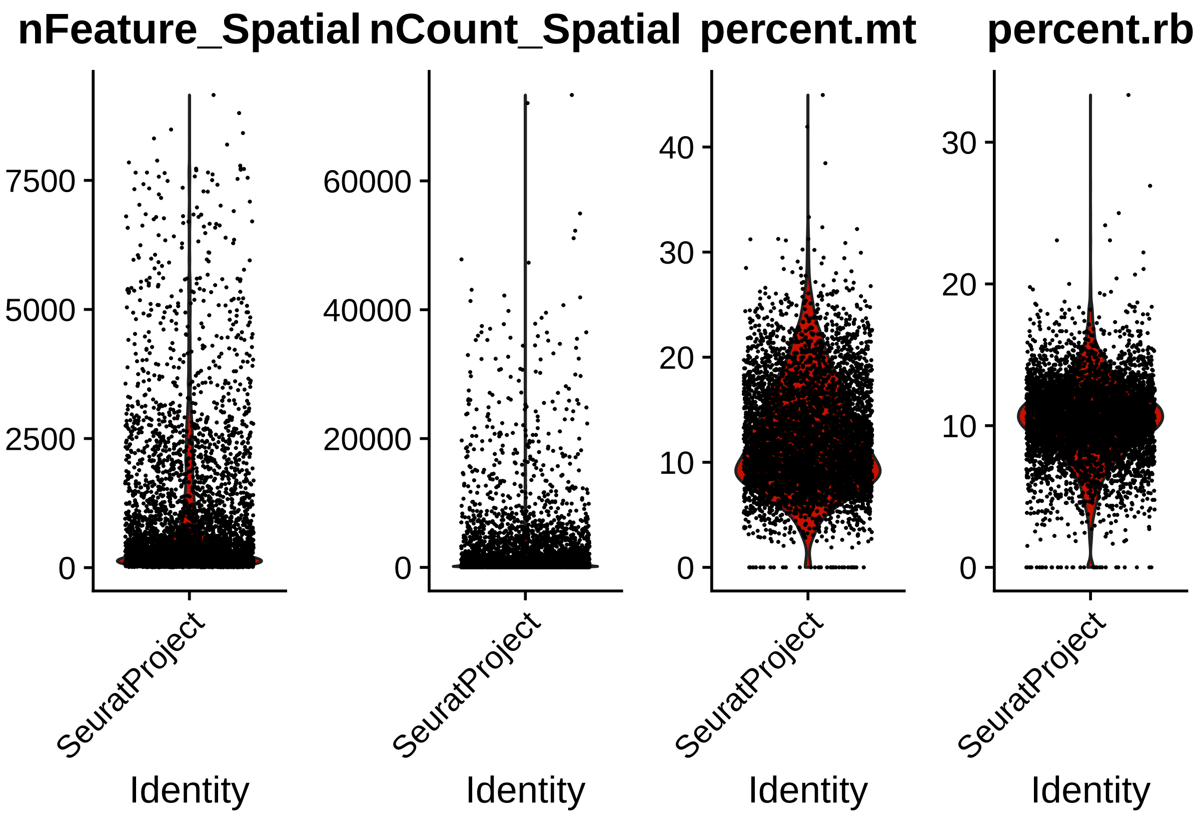
```
count.v.feature <- FeatureScatter(lung.data, feature1 = "nCount_Spatial",
                  cols = 'darkgreen', feature2 = "nFeature_Spatial") +NoLegend()
cout.v.mt <- FeatureScatter(lung.data, feature1 = "nCount_Spatial",
             feature2 = "percent.mt", cols = 'darkred')+ NoLegend()

count.v.feature+cout.v.mt
```

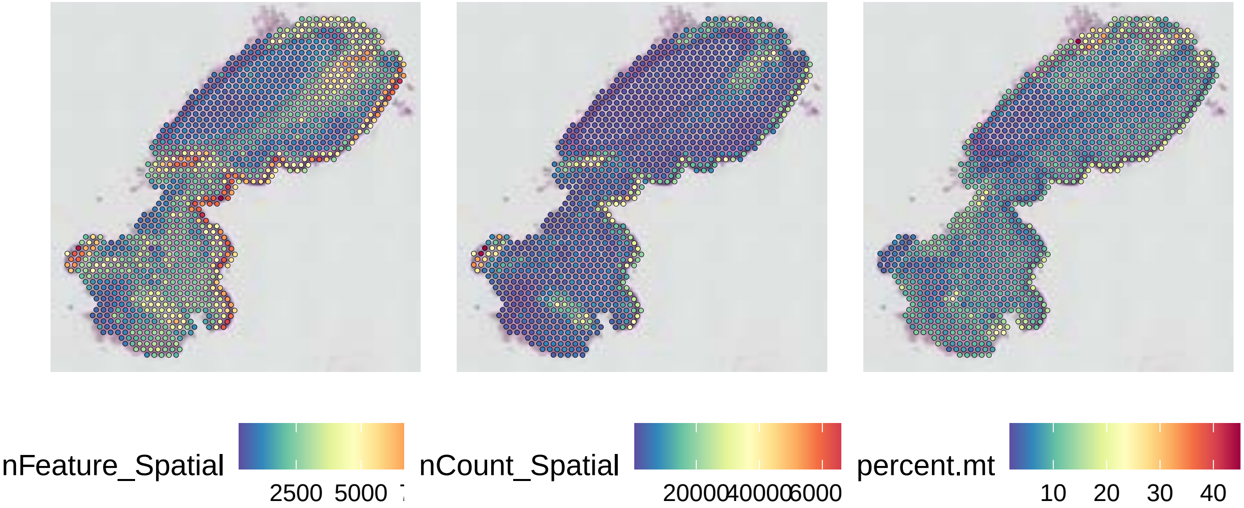


Visual inspection of the distribution of the assay data before QC

```
raw.vp<- VlnPlot(lung.data, features = c("nFeature_Spatial", "nCount_Spatial", "percent.mt",
                            'percent.rb'), ncol = 4,
                             cols = '#C71000')
raw.vp
```



```r
# The % of Rb genes looks fine (i.e., below 45%)
raw.fp<- SpatialFeaturePlot( lung.data, features = c("nFeature_Spatial", "nCount_Spatial",
                "percent.mt")) & theme(legend.position = "bottom")
raw.fp
```

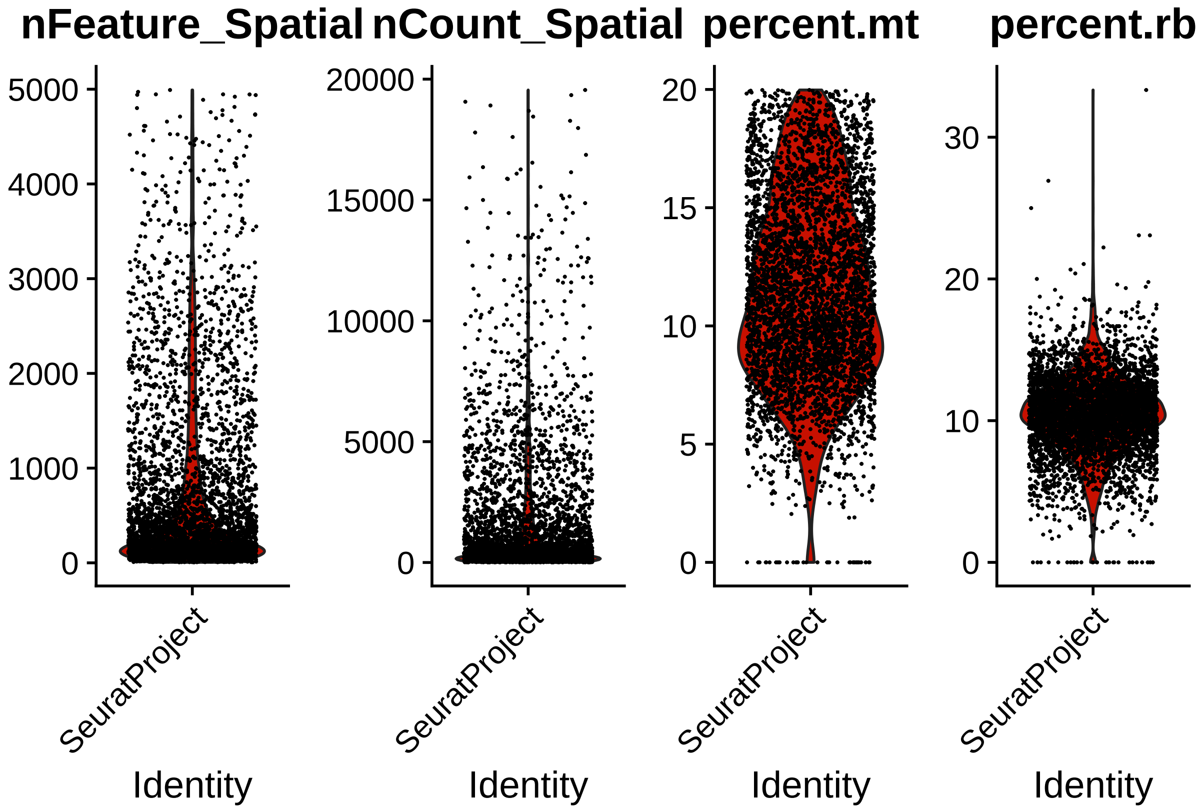


Performing QC

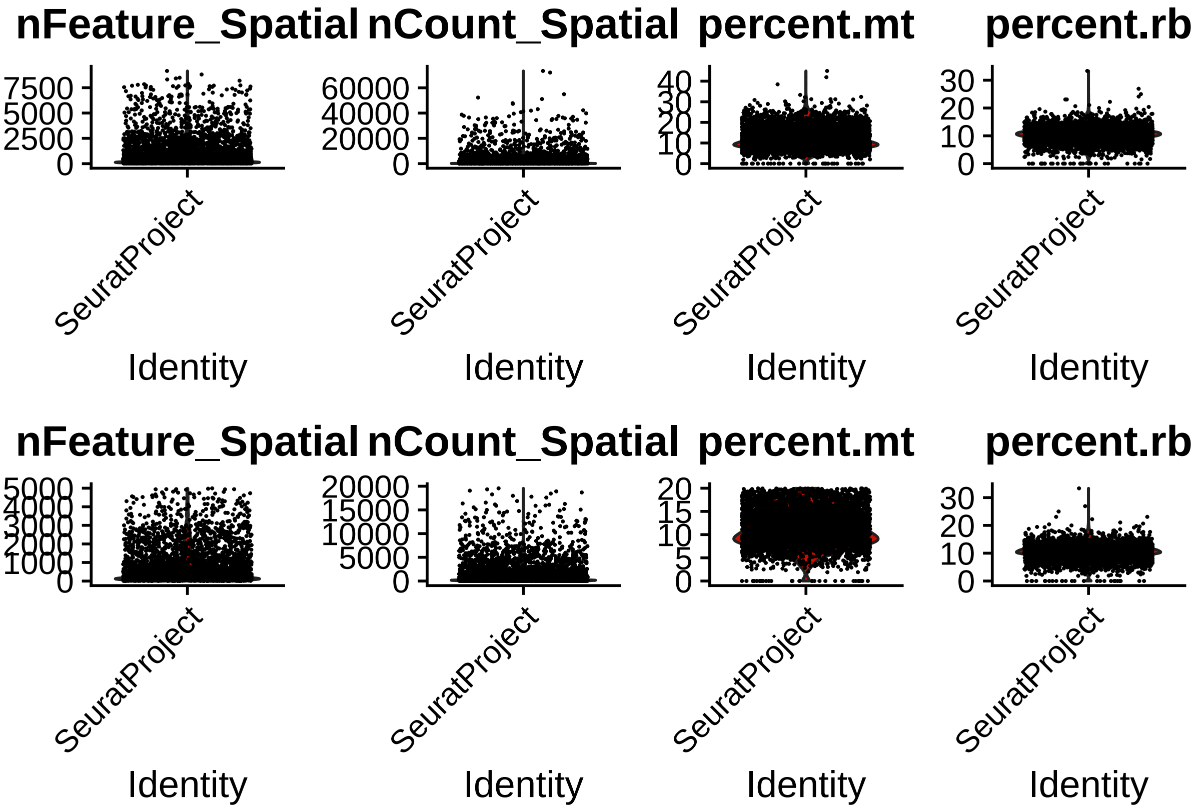
```
lung.qc <- subset(lung.data, subset = nFeature_Spatial < 5000 & percent.mt < 20
               & nCount_Spatial < 20000)
```

Visual inspection of the distribution of the assay data before normalization

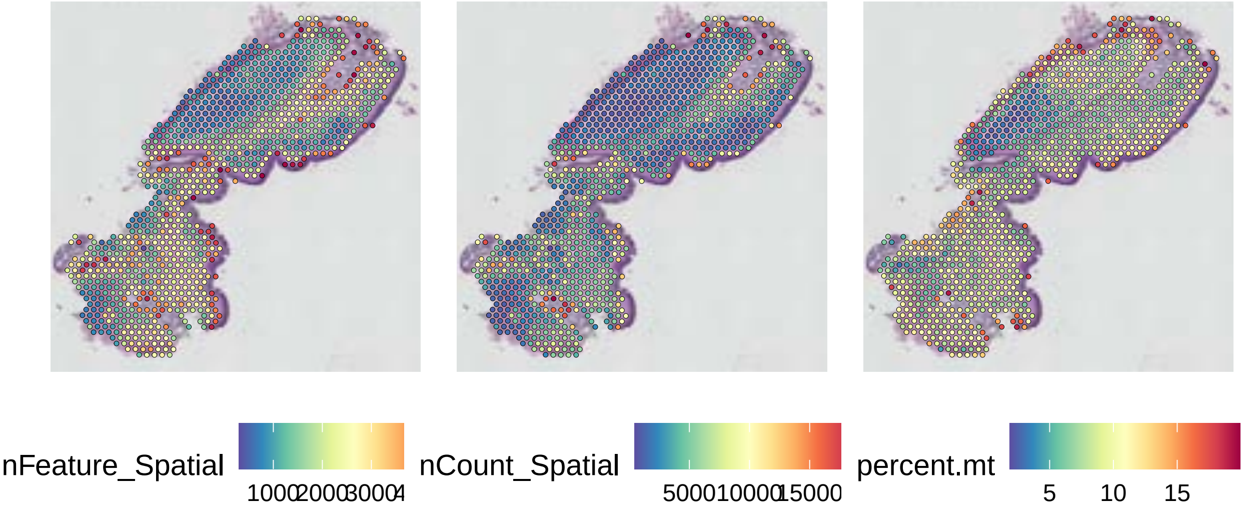
```
qc.vp<- VlnPlot(lung.qc, features = c("nFeature_Spatial", "nCount_Spatial", "percent.mt",
                  'percent.rb'), ncol = 4, cols = '#C71000')
qc.vp
```



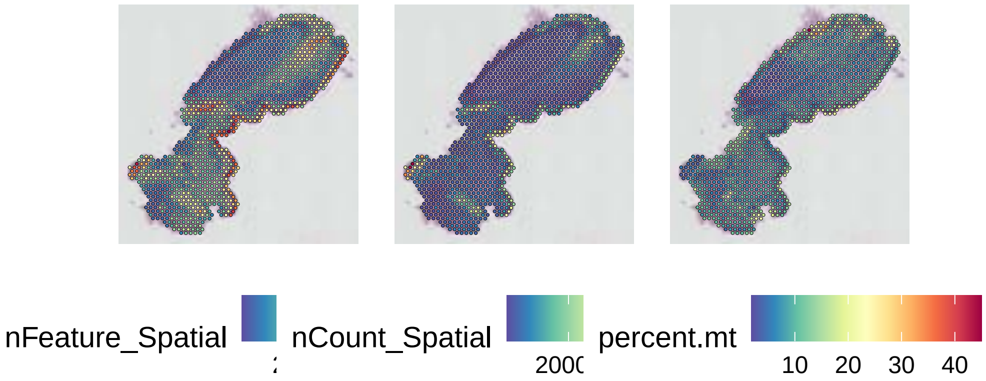
```r
#Combine and compare
wrap_plots(raw.vp, qc.vp, nrow = 2)
```

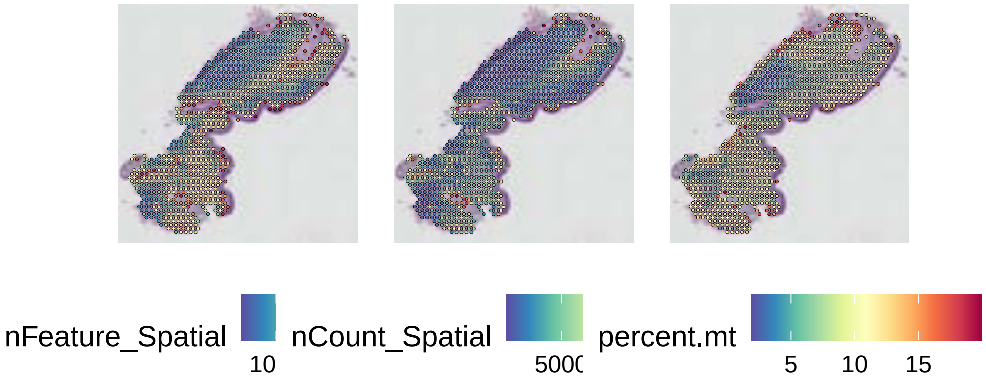


```
qc.fp<- SpatialFeaturePlot( lung.qc, features = c("nFeature_Spatial", "nCount_Spatial",
                           "percent.mt")) & theme(legend.position = "bottom")
qc.fp
```



```
wrap_plots(raw.fp, qc.fp, nrow = 2)
```





Perform normalization

In spatial scRNA-seq the heterogeneity across cells can't only be attributed to technical issues. Rather a lot of dissimilarities also account for the specific position of the cells. The developer suggests that using SCTransform rather than logNormalize performs well in terms of retaining location(tissue)-specific heterogeneity across the samples

```r
lung.norm<- SCTransform(lung.qc, assay = "Spatial", verbose = FALSE)
names(lung.norm)
```

```
## [1] "Spatial" "SCT"     "Lung"
```

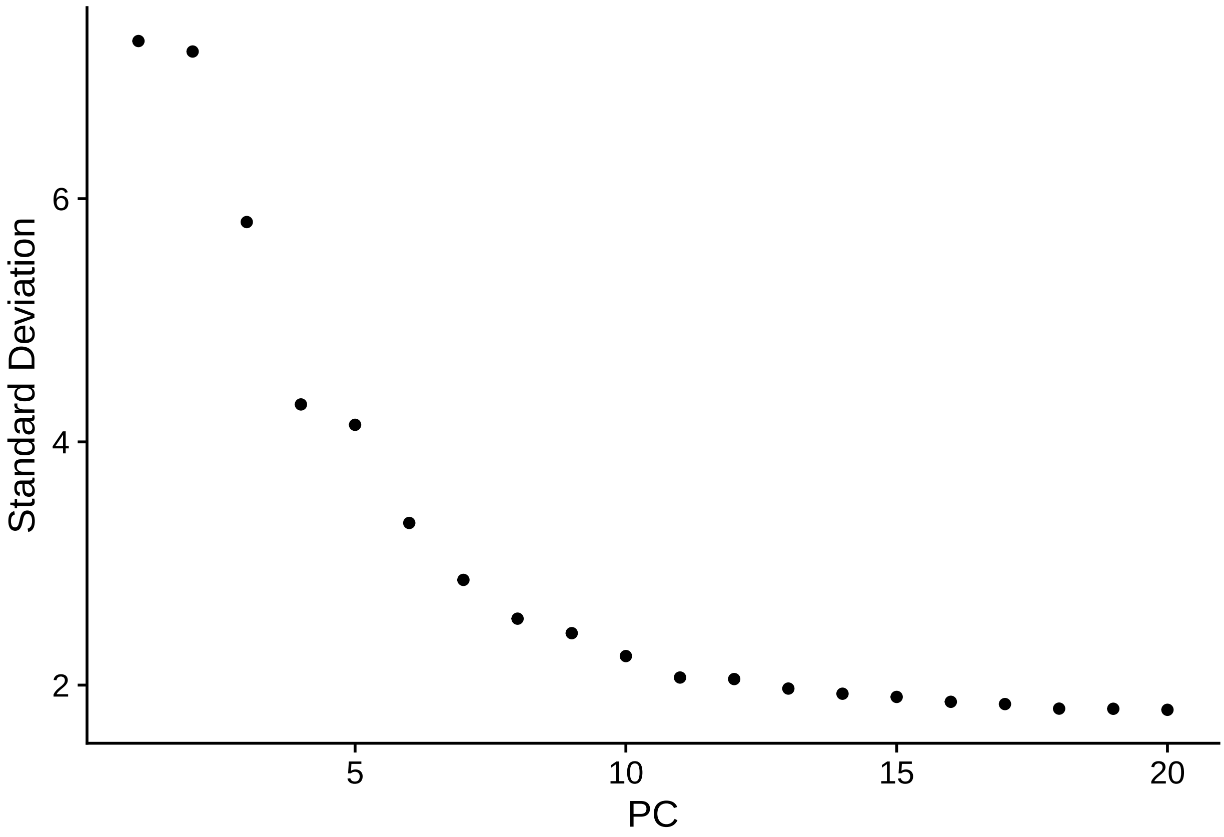
```r
rm(lung.data)
rm(lung.qc) #remove as these are not needed
```

Downstream Analysis

All these steps will follow typical scRNA-seq workflow

```r
# Run dimensionality reduction with PCA
lung.norm <- RunPCA(lung.norm, assay = "SCT", verbose = FALSE)

# Select number of dimensions
ElbowPlot(lung.norm)
```



```
# Compute Shared nearest neighbors (SNN)
lung.norm <- FindNeighbors(lung.norm, reduction = "pca", dims = 1:15)
```

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

```
## Computing SNN
```

```
# Leiden algorithm for community detection
lung.norm <- FindClusters(lung.norm, verbose = FALSE)
View(lung.norm@meta.data)

# RUN UMAP, PCA result is the default UMAP input
lung.final <- RunUMAP(lung.norm, reduction = "pca", dims = 1:15)
```

```
## 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
```

```
## 05:53:52 UMAP embedding parameters a = 0.9922 b = 1.112
```

```
## 05:53:52 Read 4400 rows and found 15 numeric columns
```

```
## 05:53:52 Using Annoy for neighbor search, n_neighbors = 30
## 05:53:52 Building Annoy index with metric = cosine, n_trees = 50
```

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

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

```
## **************************************************|
## 05:53:52 Writing NN index file to temp file C:\Users\HP\AppData\Local\Temp\RtmpIhNBTw\file31e84b9d66
## 05:53:52 Searching Annoy index using 1 thread, search_k = 3000
## 05:53:54 Annoy recall = 100%
## 05:53:54 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 05:53:55 Initializing from normalized Laplacian + noise (using irlba)
## 05:53:55 Commencing optimization for 500 epochs, with 191690 positive edges
## 05:54:05 Optimization finished
```

```r
rm(lung.norm)
```

***N.B.* At this step we need to perform doublet removal which will not be**

performed in this tutorial. Follow standard scRNA-seq workflow

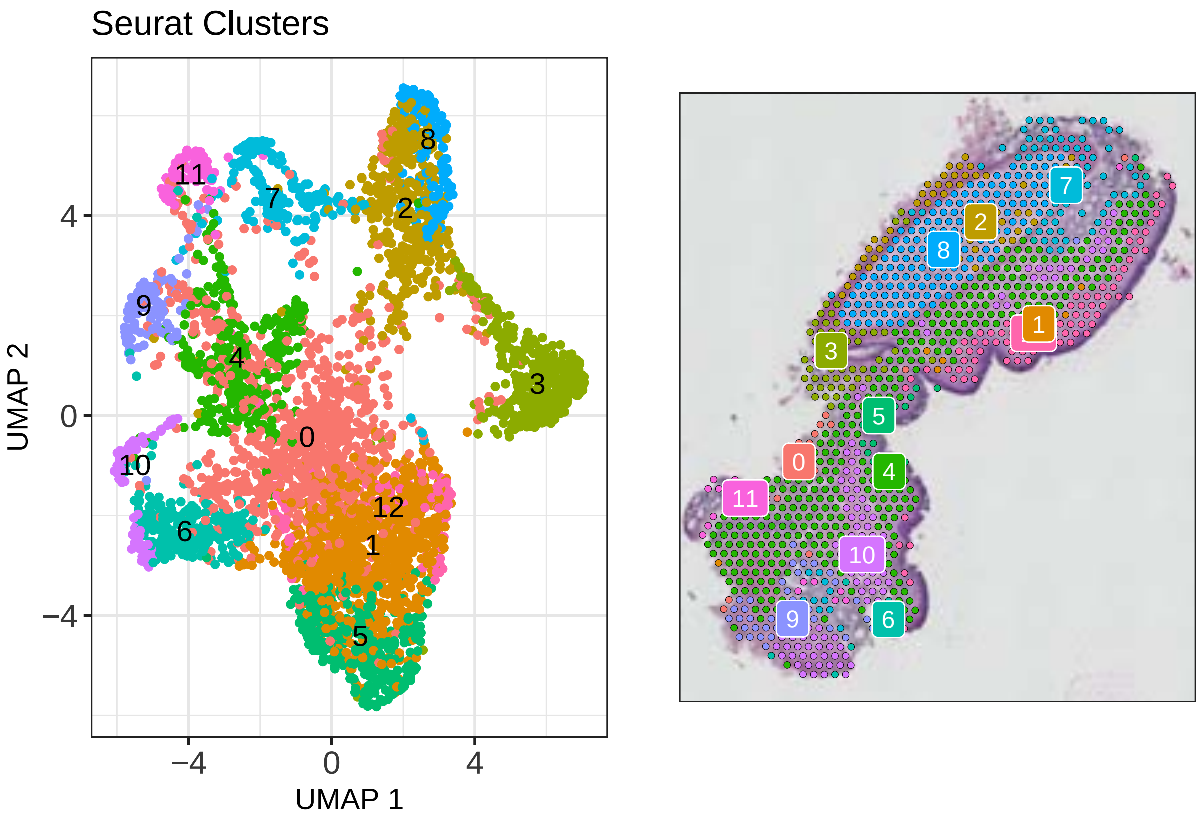
Plotting

```r
plot3 <- DimPlot(lung.final, reduction = "umap", group.by = 'seurat_clusters',
                 label = TRUE, pt.size = 1)
plot3<- plot3+theme_bw()+labs(x='UMAP 1', y='UMAP 2') + ggtitle('Seurat Clusters')+
        theme(axis.text = element_text(size = 12))+ NoLegend()

plot4 <- SpatialDimPlot(lung.final, label = TRUE, label.size = 3)
```

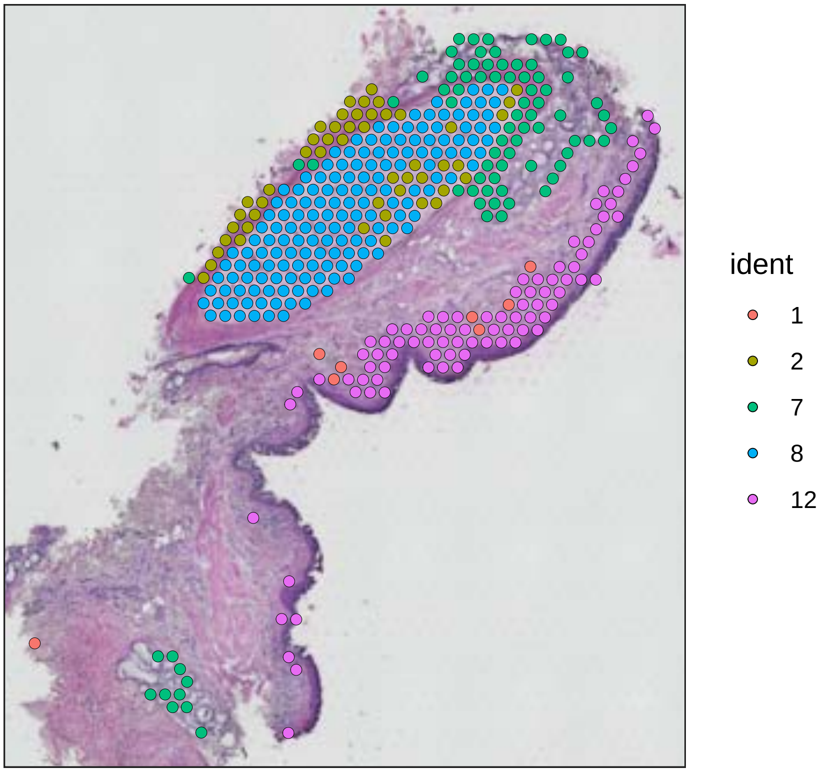
```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

```r
plot4<- plot4+theme_bw()+labs(x='', y='')+ NoLegend()+ theme(axis.text
        = element_blank(), axis.ticks = element_blank())
plot3 + plot4
```



```
subset <- subset(lung.final, idents = c(1, 2, 7, 8, 12))

SpatialDimPlot(subset, pt.size.factor = 2)+theme_bw()+ labs(x='', y='')+
      theme(axis.text = element_blank(), axis.ticks = element_blank())
```



Subset and plot

Find Markers

#Find all markers of cluster 3

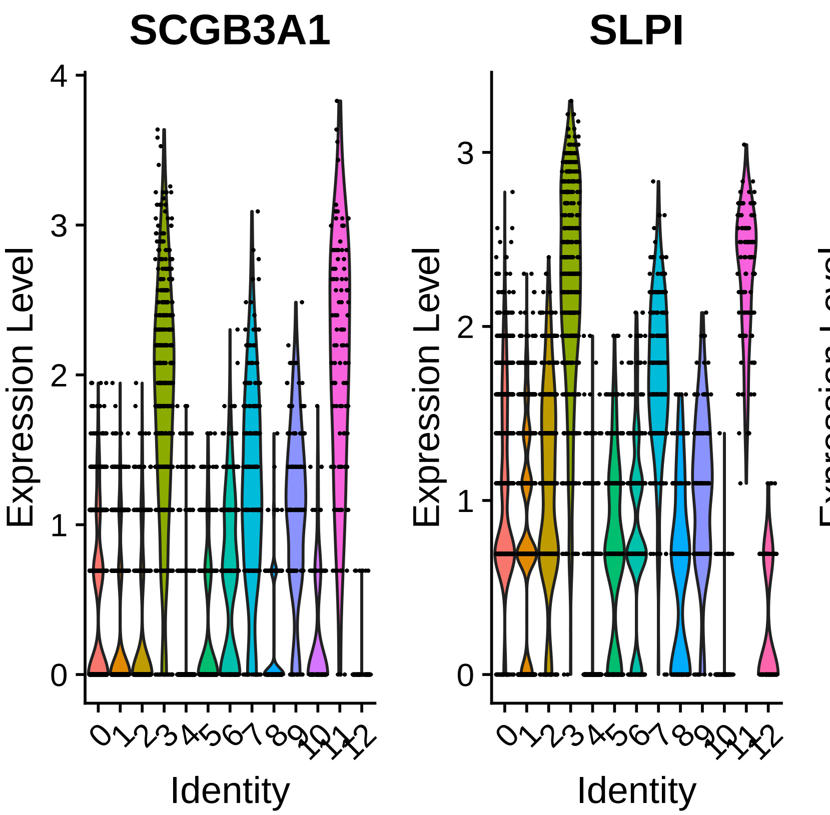
```
cluster3_markers <- FindMarkers(lung.final, ident.1 = 3, min.pct = 0.25)
```

```
## For a more efficient implementation of the Wilcoxon Rank Sum Test,
## (default method for FindMarkers) please install the limma package
## --------------------------------------------
## install.packages('BiocManager')
## BiocManager::install('limma')
## --------------------------------------------
## After installation of limma, Seurat will automatically use the more
## efficient implementation (no further action necessary).
## This message will be shown once per session
```

```
head(cluster3_markers, n = 5)
```

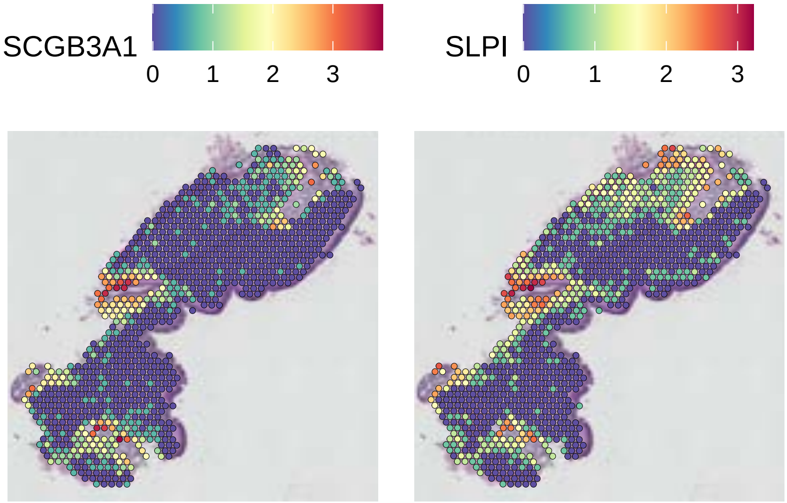
```
##                 p_val avg_log2FC pct.1 pct.2     p_val_adj
## BPIFA1   0.000000e+00  2.9447856 0.980 0.153  0.000000e+00
## SCGB3A1 2.824639e-178  2.0047270 0.937 0.398 4.827591e-174
## SLPI    2.088290e-170  1.8026246 0.995 0.766 3.569096e-166
## MUC5B   9.706018e-150  1.3784627 0.822 0.286 1.658856e-145
## SCGB1A1 8.862652e-147  0.5738154 0.416 0.047 1.514716e-142
```

```
VlnPlot(lung.final, features = c('SCGB3A1', 'SLPI', 'BPIFA1'))
```



Visualize markers in the form of violin plot

```
SpatialFeaturePlot(lung.final, features = c('SCGB3A1', 'SLPI', 'RPS14'), pt.size.factor = 2)
```



Visualize markers in the form of feature plot

```
lung.markers <- FindAllMarkers(lung.final, only.pos = TRUE, min.pct = 0.20,
                                    logfc.threshold = 0.20)
```

Find All Markers

```
## Calculating cluster 0

## Calculating cluster 1

## Calculating cluster 2

## Calculating cluster 3

## Calculating cluster 4

## Calculating cluster 5

## Calculating cluster 6

## Calculating cluster 7
## Calculating cluster 8
```

```
## Calculating cluster 9
```

```
## Calculating cluster 10
```

```
## Calculating cluster 11
```

```
## Calculating cluster 12
```

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

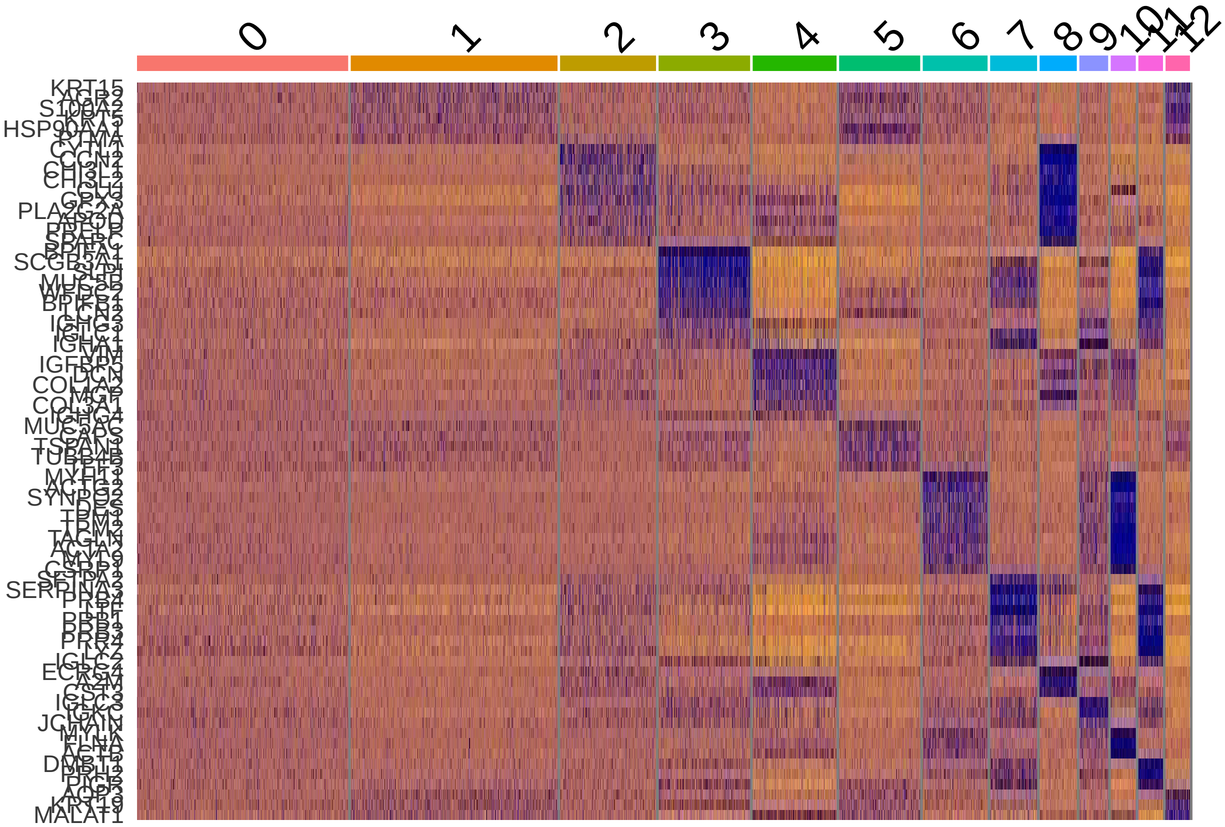
```
## # A tibble: 25 x 7
## # Groups:   cluster [13]
##        p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##        <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 1.85e-  6      0.221 0.963 0.904 3.16e-  2 0       EEF1A1
##  2 5.97e- 30      0.313 0.993 0.96  1.02e- 25 1       MT-ND1
##  3 4.11e- 32      0.299 0.999 0.986 7.02e- 28 1       MT-ATP6
##  4 3.90e-102      0.743 0.562 0.144 6.67e- 98 2       CCN2
##  5 3.86e- 45      0.670 0.581 0.29  6.60e- 41 2       GPX3
##  6 0              2.94  0.98  0.153 0         3       BPIFA1
##  7 2.82e-178      2.00  0.937 0.398 4.83e-174 3       SCGB3A1
##  8 3.62e-134      1.26  0.842 0.277 6.19e-130 4       DCN
##  9 2.11e-193      1.25  0.898 0.24  3.61e-189 4       VIM
## 10 6.10e-152      0.914 0.774 0.18  1.04e-147 5       HSP90AA1
## # i 15 more rows
```

```
top10 <- lung.markers %>%
  group_by(cluster) %>%
  top_n(n = 10, wt = avg_log2FC)

DoHeatmap(lung.final, features = top10$gene) + NoLegend() +
           scale_fill_gradient(high = 'darkblue', low = '#FFB900')
```

```
## Warning in DoHeatmap(lung.final, features = top10$gene): The following features
## were omitted as they were not found in the scale.data slot for the SCT assay:
## GSTP1, PABPC1, SAT1, FBLN1, MT-ND2, MT-CO3, MT-ND1, MT-ATP6, EEF1A1
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```



Identify spatially variable genes

Method I: Moran's I Moran's I is a global statistic that assesses the relationship between local observed gene expression values and the average of nearby values. Moran's I can be seen as a spatial autocorrelation metric, analogous to the Pearson correlation coefficient in spatial statistics.

```
#Perform Moran's I
lung.moransi <- FindSpatiallyVariableFeatures(
  lung.final, assay = "SCT",
  features = VariableFeatures(lung.final)[1:10], #We are considering only 10 genes
  #Since running against 17000 features is time consuming
  selection.method = "moransi")
```

```
## Computing Moran's I
```

```
#Get dataframe and inspect
lung.moransi.result <- lung.moransi@assays$SCT@meta.features %>%
   na.exclude #NA value due to calculating moransi only for 10 genes
head(lung.moransi.result[order(lung.moransi.result$MoransI_observed, decreasing = T),])
```

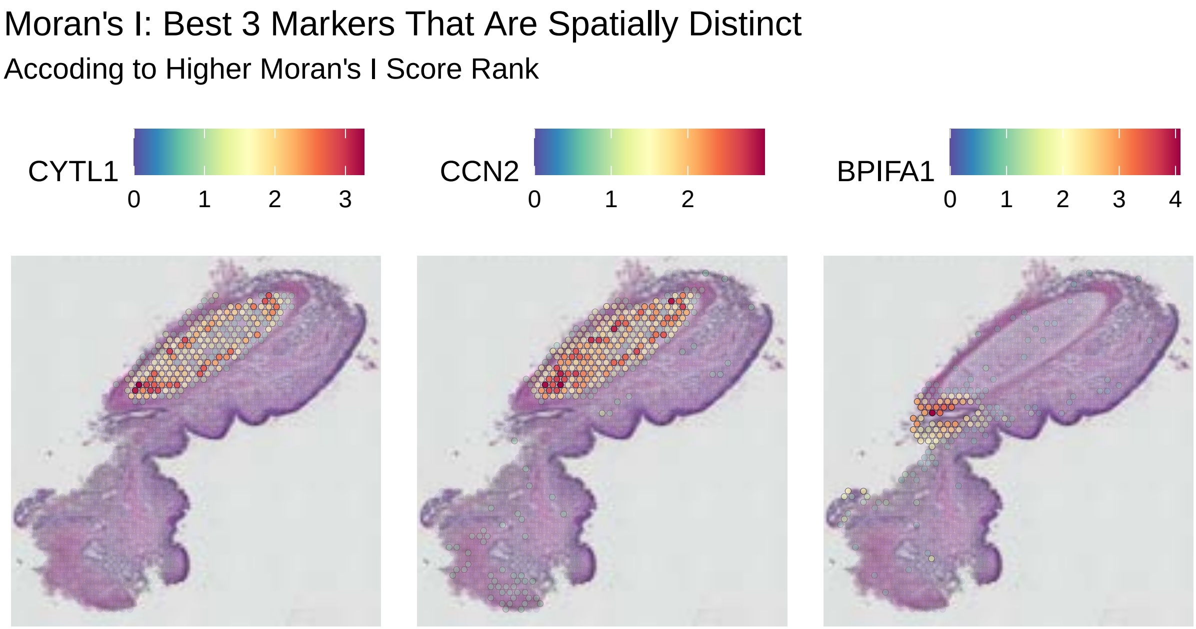
```
##          MoransI_observed MoransI_p.value moransi.spatially.variable
## CYTL1           0.5647301    0.0009756098                       TRUE
## CCN2            0.5324512    0.0009756098                       TRUE
## BPIFA1          0.4939050    0.0009756098                       TRUE
## SERPINA3        0.4320394    0.0009756098                       TRUE
## MATN1           0.4259535    0.0009756098                       TRUE
## LTF             0.4148645    0.0009756098                       TRUE
##          moransi.spatially.variable.rank
## CYTL1                                  1
## CCN2                                   2
## BPIFA1                                 3
## SERPINA3                               4
## MATN1                                  5
## LTF                                    6
```

```
#According to higher Moran's I Value
Best_Moransi_Marker <- head(
  SpatiallyVariableFeatures(lung.moransi,
                            selection.method = "moransi"), 3)
```

Plot 3 most variable genes according to Moran's I experiment

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

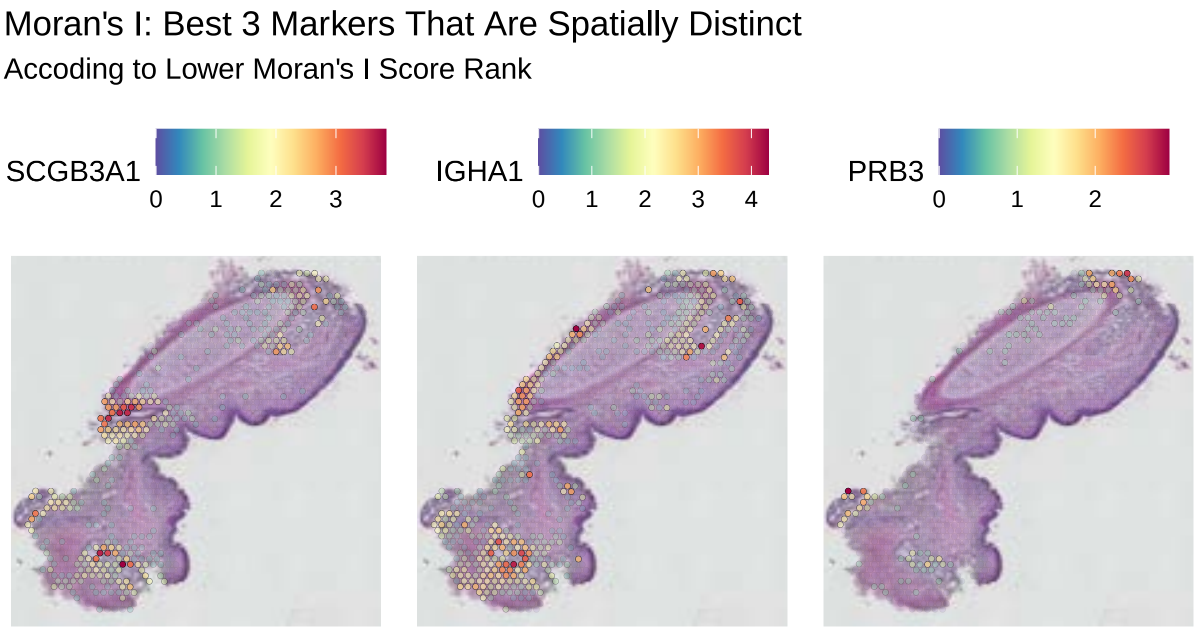
```
SpatialFeaturePlot(lung.moransi,  features = Best_Moransi_Marker, ncol = 3, alpha
                   = c(0.1, 1), pt.size.factor = 2) +
  plot_annotation(
    title = "Moran's I: Best 3 Markers That Are Spatially Distinct",
    subtitle = "Accoding to Higher Moran's I Score Rank")
```


Moran's I: Best 3 Markers That Are Spatially Distinct
Accoding to Higher Moran's I Score Rank
CYTL1
0 1 2 3
CCN2
0 1 2
BPIFA1
0 1 2 3 4

```
#According to lower Moran's I Value
Least_Moransi_Marker <- tail(
  SpatiallyVariableFeatures(lung.moransi,
                            selection.method = "moransi"), 3)
```

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
SpatialFeaturePlot(lung.moransi,  features = Least_Moransi_Marker, ncol = 3, alpha
                   = c(0.1, 1), pt.size.factor = 2) +
  plot_annotation(
    title = "Moran's I: Best 3 Markers That Are Spatially Distinct",
    subtitle = "Accoding to Lower Moran's I Score Rank")
```



Method II: Variogram Variogram analyzes spatial transcriptomics data as a mark point process and computes a ‘variogram’ that finds genes whose expression level is affected by their spatial position. More specifically, this procedure computes gamma(r) values, which measure the dependence between two points separated by a “r” distance. To save time, we utilize an r-value of ‘5’ by default in these studies and only compute these values for variable genes (variation calculated independently of spatial location).

```
#Perform variogram
lung.variogram <- FindSpatiallyVariableFeatures(
  lung.final, assay = "SCT",
  features = VariableFeatures(lung.final)[1:10], #Same 10 marker as MoransI
  selection.method = "markvariogram")
```

```
## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
```

```
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
```

```
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density

## Warning in (function (x, bw = "nrd0", adjust = 1, kernel = c("gaussian", :
## sum(weights) != 1 -- will not get true density
```

```r
#Get result
lung.variogram.reuslt<- lung.variogram @assays$SCT@meta.features %>%
  na.exclude
head(lung.variogram.reuslt[order(lung.variogram.reuslt$r.metric.5), ])
```

```
##         r.metric.5 markvariogram.spatially.variable
## BPIFA1   0.1373911                             TRUE
## PRR4     0.1834831                             TRUE
## CYTL1    0.1892281                             TRUE
## LTF      0.2019019                             TRUE
## CCN2     0.2132435                             TRUE
## SCGB3A1  0.2162279                             TRUE
##         markvariogram.spatially.variable.rank
## BPIFA1                                      1
## PRR4                                        2
## CYTL1                                       3
## LTF                                         4
## CCN2                                        5
## SCGB3A1                                     6
```

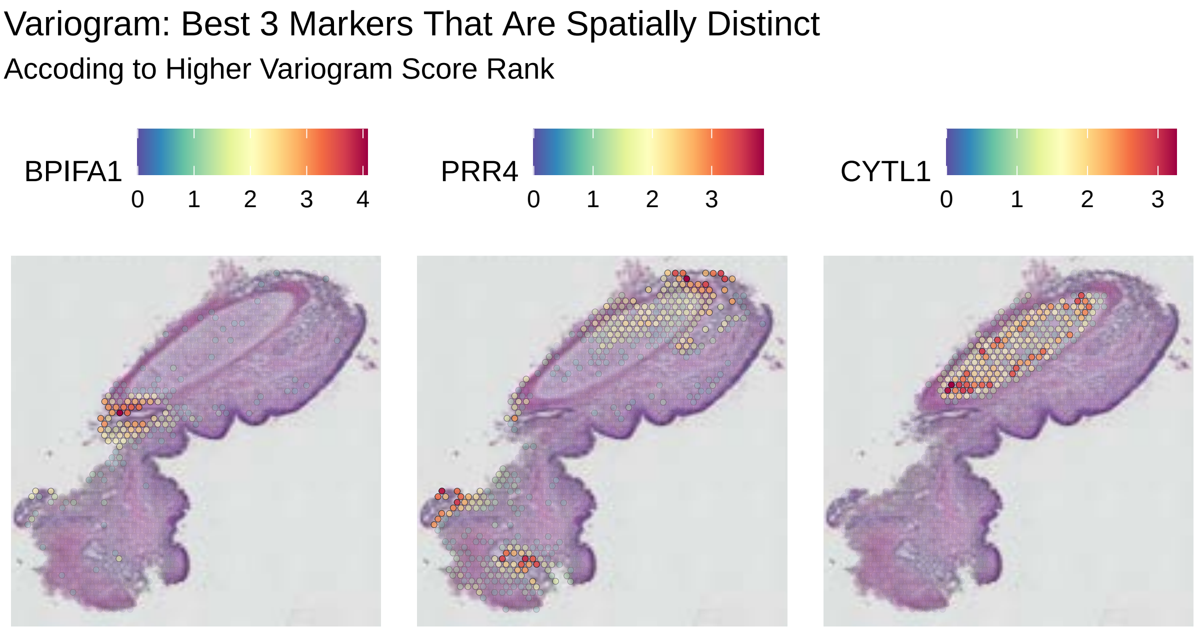
```r


#According to higher Variogra, Value
Best_Variogram_Marker <- head(
  SpatiallyVariableFeatures(lung.variogram,
                            selection.method = "markvariogram"), 3)
```

Plot

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

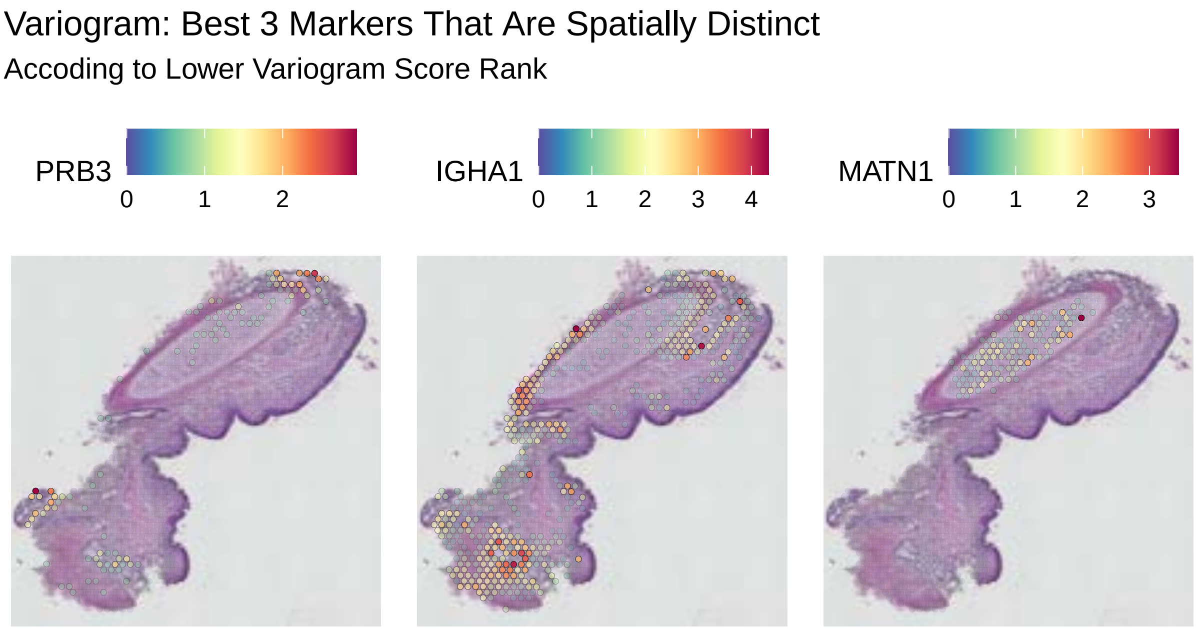
```r
SpatialFeaturePlot(lung.variogram,  features = Best_Variogram_Marker, ncol = 3, alpha
                   = c(0.1, 1), pt.size.factor = 2) +
  plot_annotation(
    title = "Variogram: Best 3 Markers That Are Spatially Distinct",
    subtitle = "Accoding to Higher Variogram Score Rank")
```


Variogram: Best 3 Markers That Are Spatially Distinct
Accoding to Higher Variogram Score Rank
BPIFA1
0 1 2 3 4
PRR4
0 1 2 3
CYTL1
0 1 2 3

```
#According to lower Variogram Value
Least_Ranked_Marker <- tail(
  SpatiallyVariableFeatures(lung.variogram,
                            selection.method = "markvariogram"), 3)
```

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
SpatialFeaturePlot(lung.variogram,  features = Least_Ranked_Marker, ncol = 3, alpha
                   = c(0.1, 1), pt.size.factor = 2) +
  plot_annotation(
    title = "Variogram: Best 3 Markers That Are Spatially Distinct",
    subtitle = "Accoding to Lower Variogram Score Rank")
```



Variogram: Best 3 Markers That Are Spatially Distinct

Accoding to Lower Variogram Score Rank

Annotation and Visualization

We performed manual annotation from PanglaoDB and CellMarker 2.0 due to shortage of memory. The annotation procedure with SingleR is available in original scRNA-seq script.

```
#Renaming clusters
new.names<- c('0'='Immune Cell', '1'='Basal Cell', '2'='Immune Cell',
              '3'='Airway Secretory Cell', '4'='Fibroblast', '5'='Immune Cell',
              '6'='Smooth Muscle Cell', '7'='Airway Secretory Cell', '8'=
                'Fibroblast', '9'='Smooth Muscle Cell', '10'='Fibroblast',
              '11'='Airway Secretory Cell', '12'='Epithelial Cell')
#Create annotation column
lung.final@meta.data$Annotation<- new.names[lung.final@meta.data$seurat_clusters]
table(lung.final@meta.data$Annotation)
```

```
##
## Airway Secretory Cell            Basal Cell       Epithelial Cell
##                   702                   890                   105
##            Fibroblast           Immune Cell    Smooth Muscle Cell
##                   629                  1671                   403
```

Final plotting

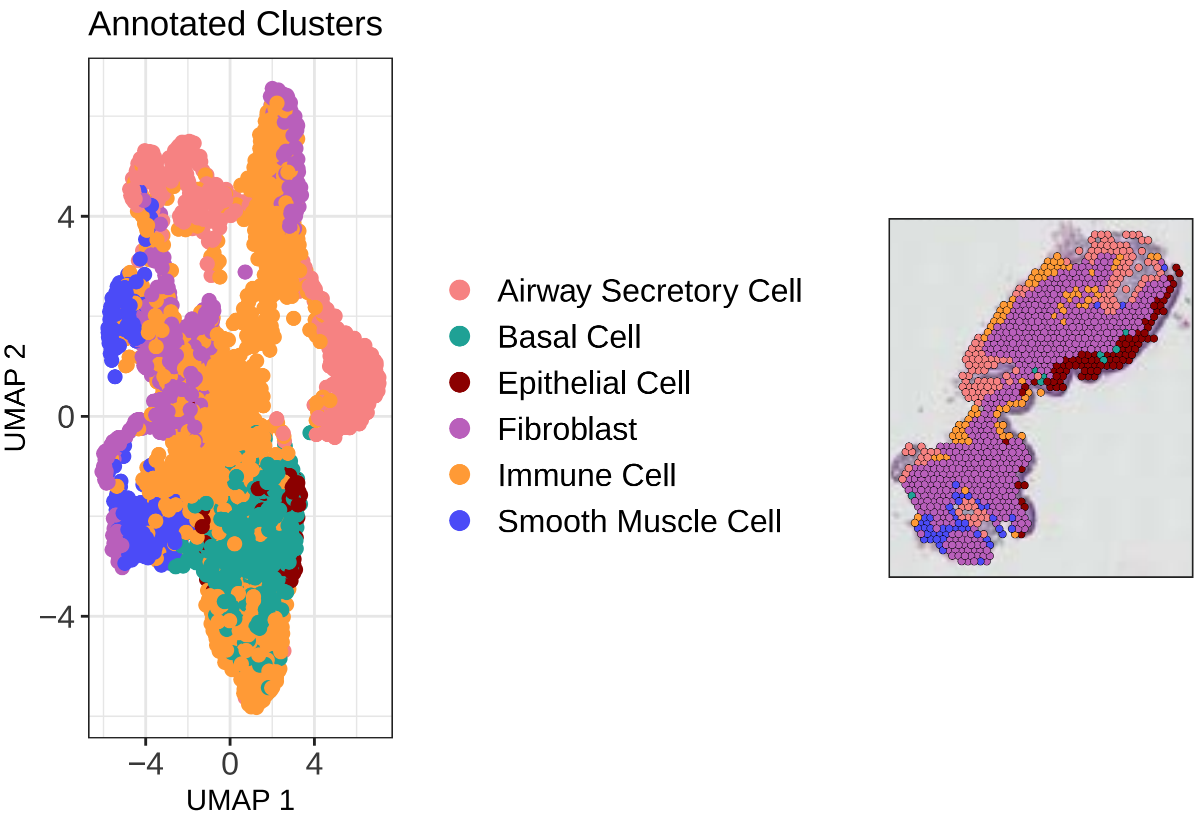
```
#Adding color
cols2 <- c('Airway Secretory Cell'='#F68282','Basal Cell'='#1FA195','Fibroblast'='#B95FBB',
           'Immune Cell'='#ff9a36','Smooth Muscle Cell'='#4B4BF7','Epithelial Cell'='darkred')

#Dimension plot
Final.dim<-DimPlot(lung.final, reduction = 'umap', group.by = 'Annotation', label = F, cols = cols2, pt

Final.dim<- Final.dim+ theme_bw()+labs(x='UMAP 1', y='UMAP 2') + ggtitle('Annotated Clusters')+
  theme(axis.text = element_text(size = 12), legend.text = element_text(size=12))

#Spatial feature plot
Final.fet <- SpatialDimPlot(lung.final, group.by = 'Annotation', label = F,
                            pt.size.factor  = 3, cols = cols2)
Final.fet<- Final.fet+theme_bw()+labs(x='', y='')+ theme(axis.text
                       = element_blank(), axis.ticks = element_blank())+ NoLegend()

Final.dim+Final.fet
```


Annotated Clusters
UMAP 2
UMAP 1
● Airway Secretory Cell
● Basal Cell
● Epithelial Cell
● Fibroblast
● Immune Cell
● Smooth Muscle Cell