spatial_example_10x_visium

Qingnan Liang

8/22/2022

Introduction

Here is a demonstration of how to use the package 'gsdensity' to perform gene set analysis on single-cell data when we want to investigate the relationship between the pathway and other information of the cells, such as clustering information and spatial information.

Load libraries

```
library(gsdensity)
library(ggplot2) # for plotting
library(ggrepel)
library(reshape2)
library(msigdbr) # for gathering gene sets
library(Seurat)
library(SeuratData)
library(future) # for parallel computing
library(future.apply) # for parallel computing
```

Preparation: Collect gene sets

```


# Collect a single category from the msigdbr database
# For more information please check: https://cran.r-project.org/web/packages/msigdbr/vignettes/msigdbr-

# Use mouse gene sets
mdb_c5 <- msigdbr(species = "Mus musculus", category = "C5")

# If we just want to do biological process:
mdb_c5_bp <- mdb_c5[mdb_c5$gs_subcat == "GO:BP", ]

# convert msigdbr gene sets to a list good for the input
gene.set.list <- list()
for (gene.set.name in unique(mdb_c5_bp$gs_name)){
        gene.set.list[[gene.set.name]] <- mdb_c5_bp[mdb_c5_bp$gs_name %in% gene.set.name, ]$gene_symbol
}
# head(gene.set.list)
```

This is the same as is shown in 'pbmc3k_example'. Here we use gene ontology (GO) biological process gene sets.

Preparation: Single-cell datasets

Here we use the stxBrain data from SeuratData;

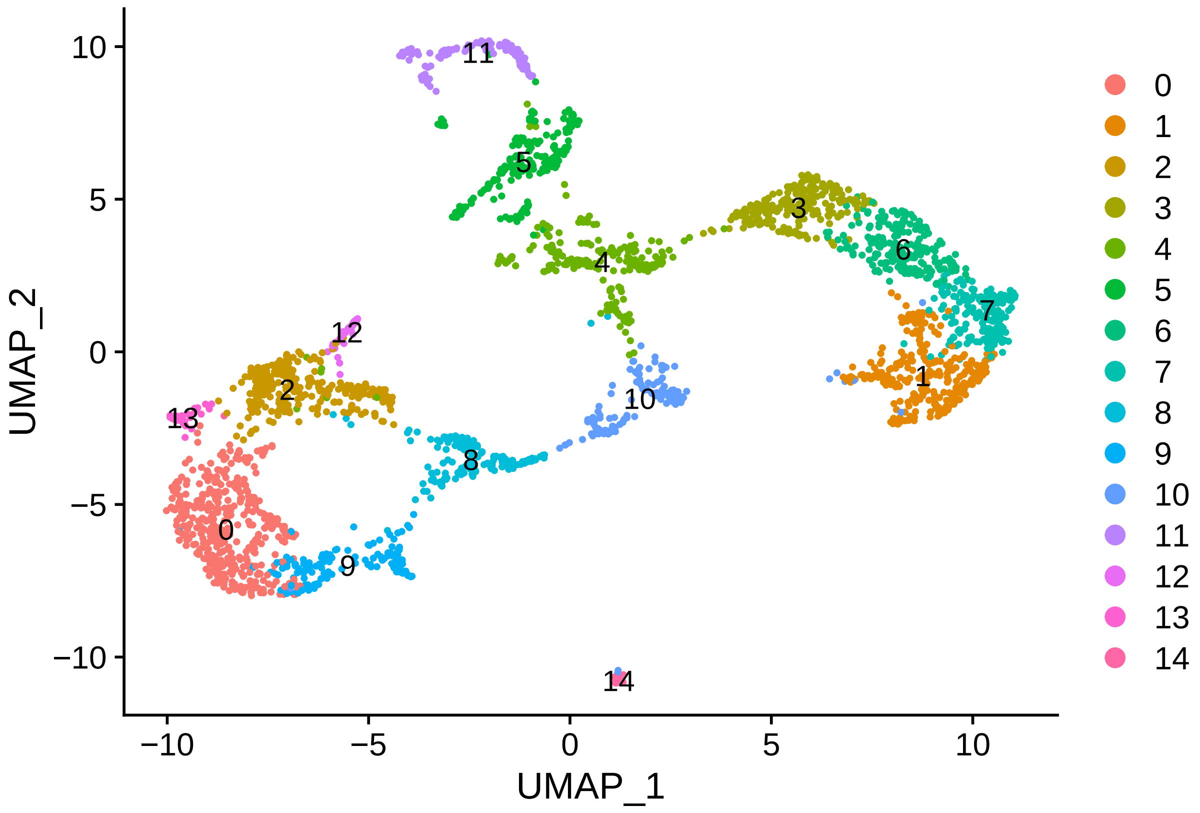
```r
brain <- LoadData("stxBrain", type = "anterior1")
print(brain)
```

We first do the preprocessing following this tutorial https://satijalab.org/seurat/articles/spatial_vignette.html

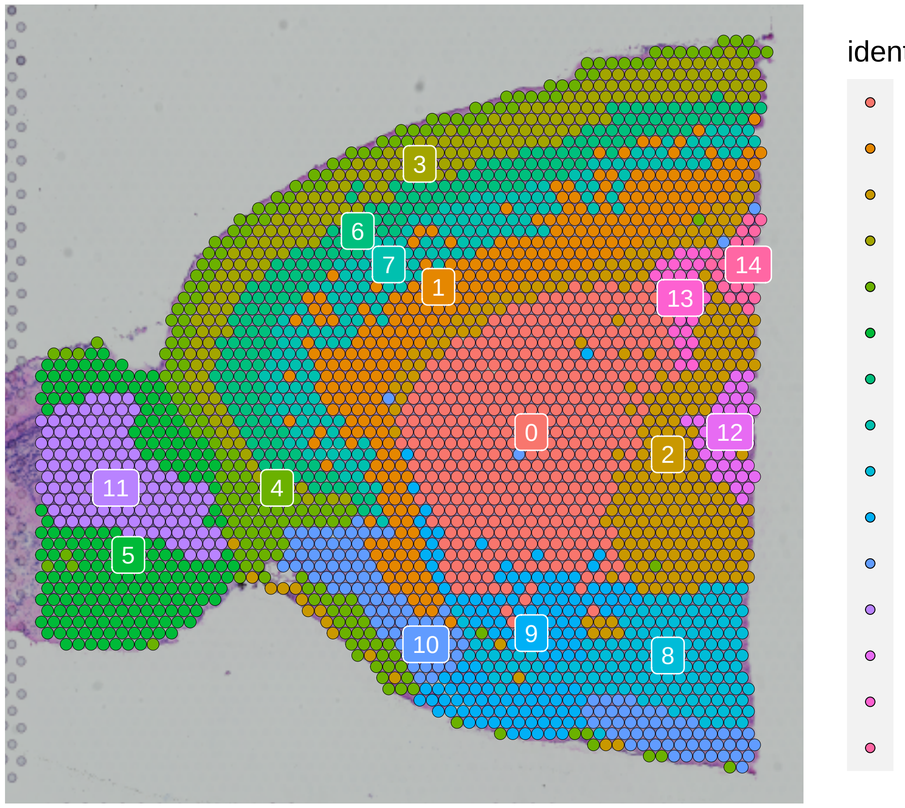
```
## An object of class Seurat
## 31053 features across 2696 samples within 1 assay
## Active assay: Spatial (31053 features, 0 variable features)
```

```r
# Transformation

brain <- SCTransform(brain, assay = "Spatial", verbose = FALSE)
```

```r
# Dimensionality reduction, clustering, and visualization

brain <- RunPCA(brain, assay = "SCT", verbose = FALSE)
brain <- FindNeighbors(brain, reduction = "pca", dims = 1:30)
brain <- FindClusters(brain, verbose = FALSE)
brain <- RunUMAP(brain, reduction = "pca", dims = 1:30)

# plot the clustering information
DimPlot(brain, reduction = "umap", label = TRUE)
```



```
# map the clusters back onto the spatial map
SpatialDimPlot(brain, label = TRUE, label.size = 3)
```



1. Find out spatially related gene sets

```
# Run the gsdensity pipeline; very similar to what is shown in 'pbmc3k_example'

# compute the cell and gene embeddings; we will still refer to each data point as a 'cell' although it

ce <- compute.mca(object = brain)
```

gsdensity starts with gene sets. We first need to calculate the relevance between each cell and the gene sets. Then we use weighted two dimensional kernel density estimation (weighted kde2d) to investigate if the relevant cells show some spatial patterns.

```
## 4.11 sec elapsed
## 123.01 sec elapsed
## 22.87 sec elapsed
```

```
# compute the deviation; we want to use only the gene sets with more than 20 genes
res <- compute.kld(coembed = ce,
                   genes.use = rownames(brain),
                   n.grids = 100,
                   gene.set.list = gene.set.list,
                   gene.set.cutoff = 20,
```

```r
                   n.times = 100)

# we will then focus on th deviated gene sets; here we set a more stringent alpha level cutoff
gene.set.deviated <- res[res$p.adj < 0.001, ]$gene.set
# length(gene.set.deviated)

# compute a nearest neighbor graph (edge list) in the MCA space
cells <- colnames(brain)
el <- compute.nn.edges(coembed = ce, nn.use = 300)

# We then compute the relevance between each cell and the deviated gene sets

cv.df <- run.rwr.list(el = el, gene_set_list = gene.set.list[gene.set.deviated], cells = cells)
cv.df[1:3, 1:3]
```

```
##                      GOBP_ACIDIC_AMINO_ACID_TRANSPORT
## AAACAAGTATCTCCCA-1                       0.0011666124
## AAACACCAATAACTGC-1                       0.0005729724
## AAACAGAGCGACTCCT-1                       0.0000000000
##                      GOBP_ACTIN_CYTOSKELETON_REORGANIZATION
## AAACAAGTATCTCCCA-1                             0.0001623796
## AAACACCAATAACTGC-1                             0.0009500982
## AAACAGAGCGACTCCT-1                             0.0000000000
##                      GOBP_ACTIN_FILAMENT_BASED_MOVEMENT
## AAACAAGTATCTCCCA-1                         0.0003621734
## AAACACCAATAACTGC-1                         0.0001055359
## AAACAGAGCGACTCCT-1                         0.0002774095
```

```r
# We can then binarize the data for each gene get

cl.df <- compute.cell.label.df(cv.df)


# An optional filtering step: we want to only keep the terms with certain numbers of positive cells; he

positive.count <- apply(cl.df, MARGIN = 2, FUN = function(x) {length(x[x == "positive"])})
gene.set.deviated.2 <- names(positive.count[positive.count > 100])
```

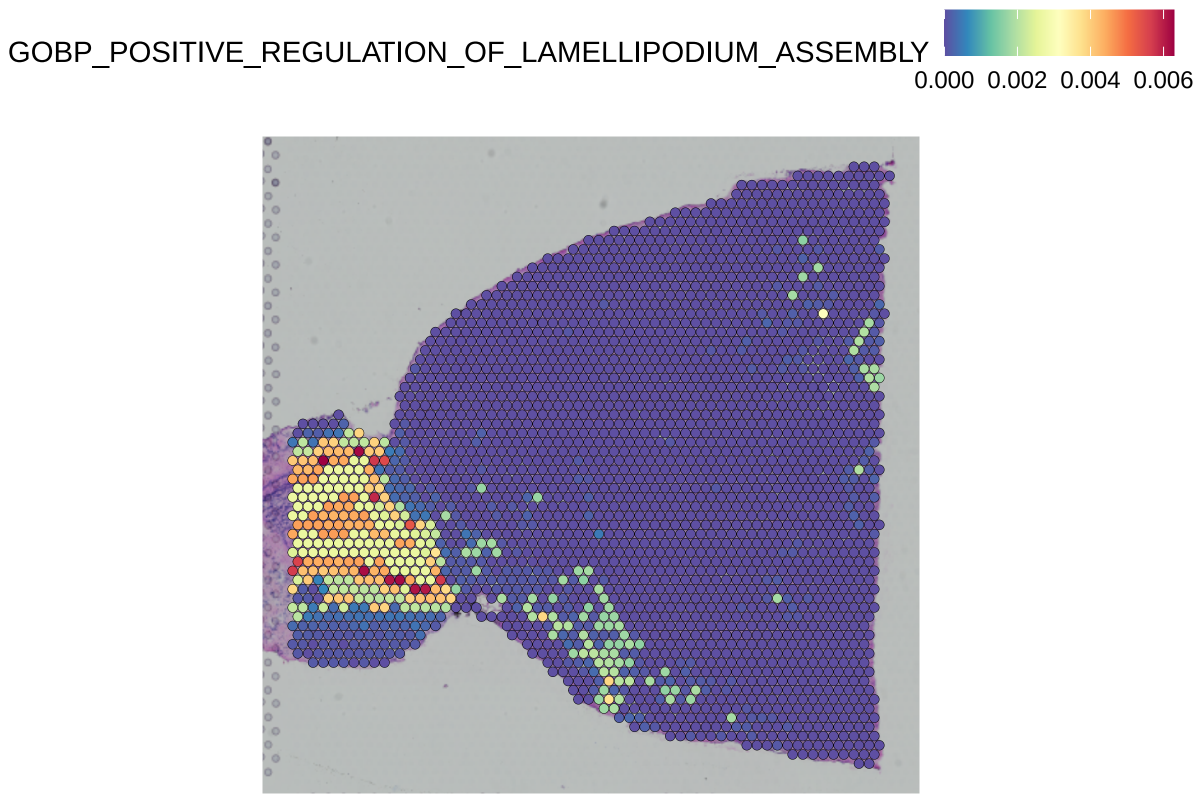
```r
# first we need to find the spatial information of the cells
# this coords.df should be a cell by coordinate matrix/dataframe with cells in rows and coordinates in
# in this dataset, this information can be found as below:
coords.df <- brain@images$anterior1@coordinates[, c("imagerow", "imagecol")]
head(coords.df)
```

```
##                      imagerow imagecol
## AAACAAGTATCTCCCA-1       7475     8501
## AAACACCAATAACTGC-1       8553     2788
## AAACAGAGCGACTCCT-1       3164     7950
## AAACAGCTTTCAGAAG-1       6637     2099
## AAACAGGGTCTATATT-1       7116     2375
## AAACATGGTGAGAGGA-1       8913     1480
```

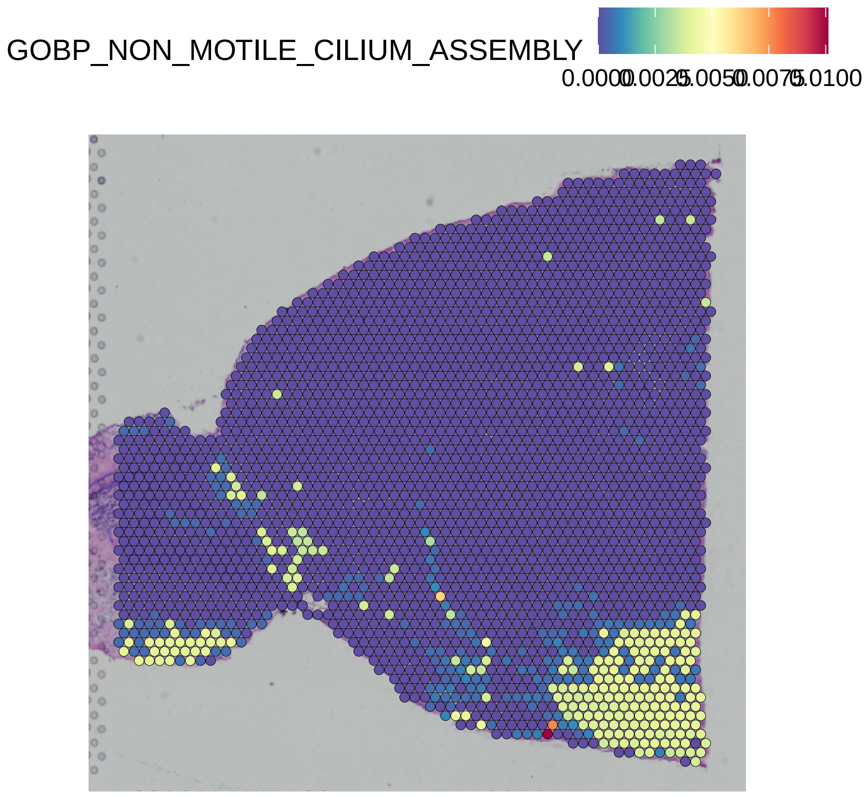
```r
# compute the spatial relevance of gene sets

# the 'weight_df' should have a format as the output of 'run.rwr.list'; here we use the terms with at l
# the parameter 'n' defines how to split the spatial map. n = 10 means that 10 splits are made in each

spatial.klds <- compute.spatial.kld.df(spatial.coords = coords.df,
                                       weight_df = cv.df[, gene.set.deviated.2],
                                       n = 10)

# Then we want to nominate gene sets: here we want to find highly spatially related gene sets

top.spatial.terms <- spatial.klds[spatial.klds > quantile(spatial.klds, 0.99)]
top.spatial.terms
```

```
##                                     GOBP_LAMELLIPODIUM_ASSEMBLY
##                                                    1.385714e-06
##                                 GOBP_NON_MOTILE_CILIUM_ASSEMBLY
##                                                    1.590994e-06
##               GOBP_POSITIVE_REGULATION_OF_LAMELLIPODIUM_ASSEMBLY
##                                                    1.722241e-06
##           GOBP_POSITIVE_REGULATION_OF_LAMELLIPODIUM_ORGANIZATION
##                                                    1.531802e-06
## GOBP_REGULATION_OF_AMYLOID_PRECURSOR_PROTEIN_CATABOLIC_PROCESS
##                                                    1.428714e-06
```

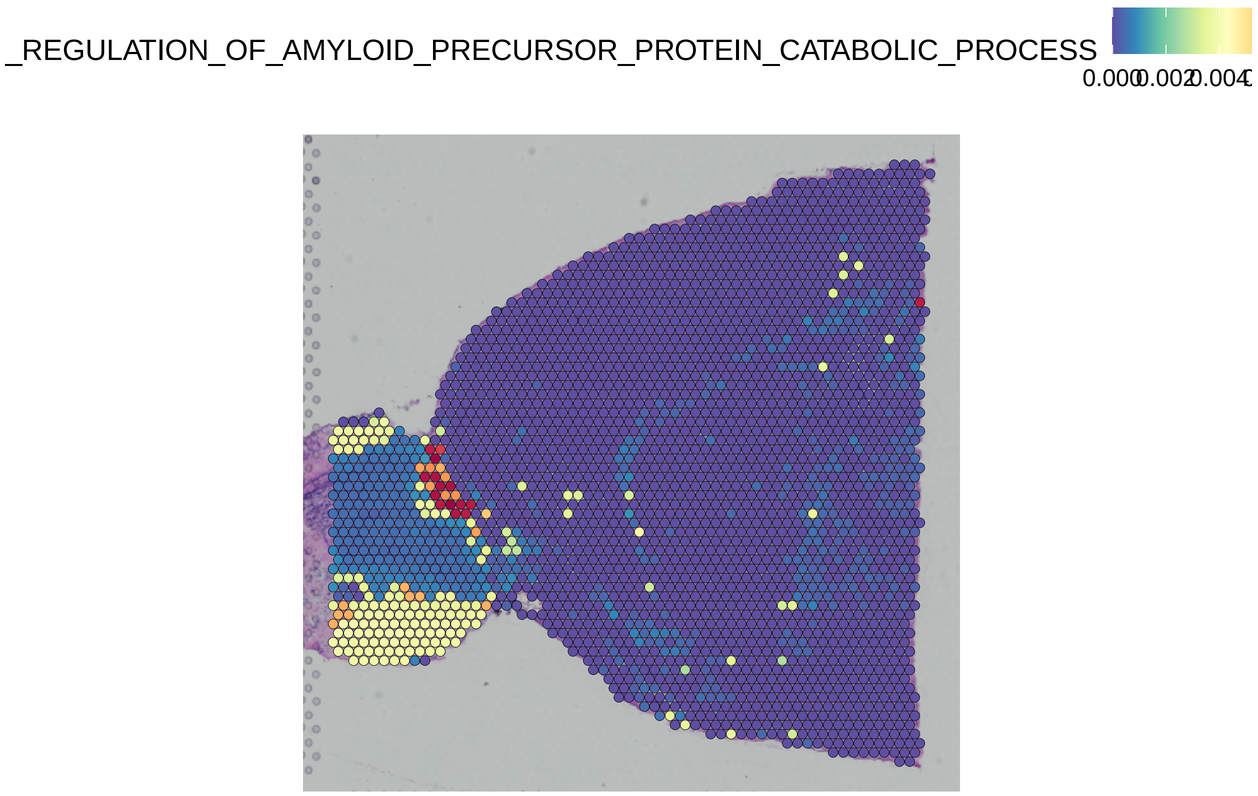
```r
# visualize some of the terms

# add the label propagation probability to metadata
brain@meta.data$GOBP_POSITIVE_REGULATION_OF_LAMELLIPODIUM_ASSEMBLY <- cv.df[rownames(brain@meta.data),
                                                                           "GOBP_POSITIVE_REGULATION_OF_LAMELLIPODIUM_A

SpatialFeaturePlot(brain, features = c("GOBP_POSITIVE_REGULATION_OF_LAMELLIPODIUM_ASSEMBLY")) +
        theme(legend.position = "top")
```



```
brain@meta.data$GOBP_NON_MOTILE_CILIUM_ASSEMBLY <- cv.df[rownames(brain@meta.data),
                                                         "GOBP_NON_MOTILE_CILIUM_ASSEMBLY"]

SpatialFeaturePlot(brain, features = c("GOBP_NON_MOTILE_CILIUM_ASSEMBLY")) +
        theme(legend.position = "top")
```



```
brain@meta.data$GOBP_REGULATION_OF_AMYLOID_PRECURSOR_PROTEIN_CATABOLIC_PROCESS <- cv.df[rownames(brain@m
                                                                             "GOBP_REGULATION_OF_AMYLOID_PRECURSOR_PROTEI

SpatialFeaturePlot(brain, features = c("GOBP_REGULATION_OF_AMYLOID_PRECURSOR_PROTEIN_CATABOLIC_PROCESS")
        theme(legend.position = "top")
```



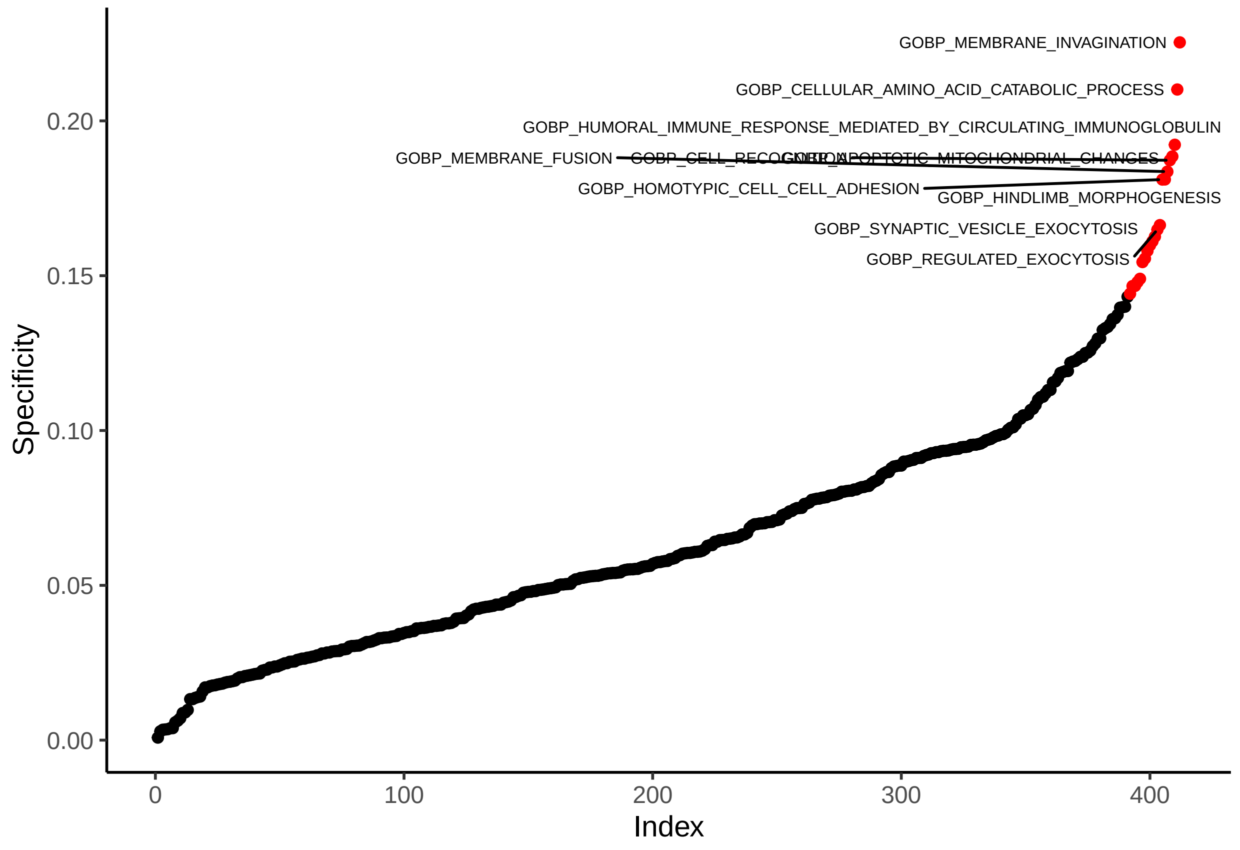
2. Find out cell partition related gene sets

```
# Basically we are computing the Jensen-Shannon distance between the label propagation probability (for
jsd.df <- compute.spec(cell_df = cv.df[, gene.set.deviated.2],
                       metadata = brain@meta.data, # each row is a cell; columns include partition info
                       cell_group = "seurat_clusters" # 'cell_group' should use a column name in the me
                       )
head(jsd.df) # now each column is the specificity for the gene sets to the very cluster
```

gsdensity is a cluster-free approach that does not really require any information of cells. However, when partition information is available for cells, gsdensity can also add onto it. The most common partition of cells is clustering, while others such as 'disease or healthy', 'treatment or control' are possible. Here we just use clustering as an example and find out which gene terms are highly specific for a cluster.

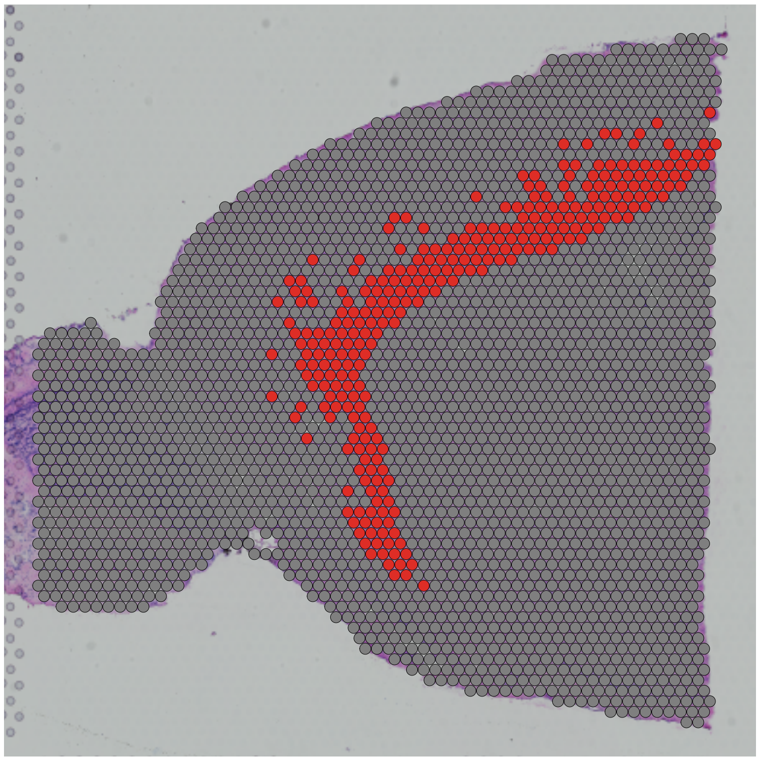
```
##                                                       2         5           3
## GOBP_ACTIN_FILAMENT_BASED_MOVEMENT           0.13062936 0.1115739 0.053368602
## GOBP_ACTIN_FILAMENT_BUNDLE_ORGANIZATION      0.14510497 0.1430864 0.070606985
## GOBP_ACTIN_FILAMENT_ORGANIZATION             0.18893751 0.1144300 0.049777921
## GOBP_ACTIN_FILAMENT_POLYMERIZATION           0.12611237 0.1776660 0.002244348
## GOBP_ACTIN_POLYMERIZATION_OR_DEPOLYMERIZATION 0.11845484 0.1633746 0.002722882
## GOBP_ACTION_POTENTIAL                        0.04668326 0.1299068 0.081720802
##                                                    11          8          4
## GOBP_ACTIN_FILAMENT_BASED_MOVEMENT         0.08010542 0.13021289 0.04383925
## GOBP_ACTIN_FILAMENT_BUNDLE_ORGANIZATION    0.14966735 0.03841777 0.12145915
## GOBP_ACTIN_FILAMENT_ORGANIZATION           0.11927184 0.07642560 0.08833943
## GOBP_ACTIN_FILAMENT_POLYMERIZATION         0.18304320 0.03795145 0.10480016
## GOBP_ACTIN_POLYMERIZATION_OR_DEPOLYMERIZATION 0.16889720 0.03406994 0.11338538
## GOBP_ACTION_POTENTIAL                      0.04198881 0.20764441 0.07262146
##                                                     9          0         10
## GOBP_ACTIN_FILAMENT_BASED_MOVEMENT         0.07991020 0.08490555 0.08143896
## GOBP_ACTIN_FILAMENT_BUNDLE_ORGANIZATION    0.06554492 0.03936212 0.06403408
## GOBP_ACTIN_FILAMENT_ORGANIZATION           0.05962118 0.04646472 0.08521305
## GOBP_ACTIN_FILAMENT_POLYMERIZATION         0.05267140 0.02702752 0.09439317
## GOBP_ACTIN_POLYMERIZATION_OR_DEPOLYMERIZATION 0.07215986 0.04720630 0.08558305
## GOBP_ACTION_POTENTIAL                      0.12640840 0.17800478 0.09740378
##                                                     1          6         14
## GOBP_ACTIN_FILAMENT_BASED_MOVEMENT         0.15980866 0.13410003 0.02485387
## GOBP_ACTIN_FILAMENT_BUNDLE_ORGANIZATION    0.07937459 0.13520681 0.01408938
## GOBP_ACTIN_FILAMENT_ORGANIZATION           0.12723558 0.13055465 0.01929879
## GOBP_ACTIN_FILAMENT_POLYMERIZATION         0.07759882 0.08427597 0.02593434
## GOBP_ACTIN_POLYMERIZATION_OR_DEPOLYMERIZATION 0.07676870 0.09757087 0.02456959
## GOBP_ACTION_POTENTIAL                      0.04680222 0.11251554 0.01976892
##                                                     7         12         13
## GOBP_ACTIN_FILAMENT_BASED_MOVEMENT         0.11224217 0.03033800 0.03458202
## GOBP_ACTIN_FILAMENT_BUNDLE_ORGANIZATION    0.02858030 0.02554743 0.04139056
## GOBP_ACTIN_FILAMENT_ORGANIZATION           0.08649240 0.03374087 0.03807536
## GOBP_ACTIN_FILAMENT_POLYMERIZATION         0.07057414 0.03535318 0.04365384
## GOBP_ACTIN_POLYMERIZATION_OR_DEPOLYMERIZATION 0.06968356 0.03258304 0.04215792
## GOBP_ACTION_POTENTIAL                      0.03052715 0.00801848 0.02864454
```

```r
# Use cluster 1 as an example

colnames(jsd.df) <- paste0("cluster_", colnames(jsd.df))
coi <- "cluster_1" # cluster of interest

# create data frames for visualization
coi.df <- data.frame(specificity = jsd.df[, coi],
                     term = rownames(jsd.df))

# order the gene sets by specificity
coi.df <- coi.df[order(coi.df$specificity),]

# highlight the top 5% gene sets
highlighted <- coi.df[coi.df$specificity > quantile(coi.df$specificity, 0.95), ]

ggplot(coi.df, aes(x=seq(specificity), y=specificity)) +
  geom_point(color = ifelse(coi.df$term %in% highlighted$term, "red", "black")) +
  geom_text_repel(aes(label = ifelse(term %in% highlighted$term, term, "")),
                  size = 2) +
  xlab("Index") +
  ylab("Specificity") +
  theme_classic()
```

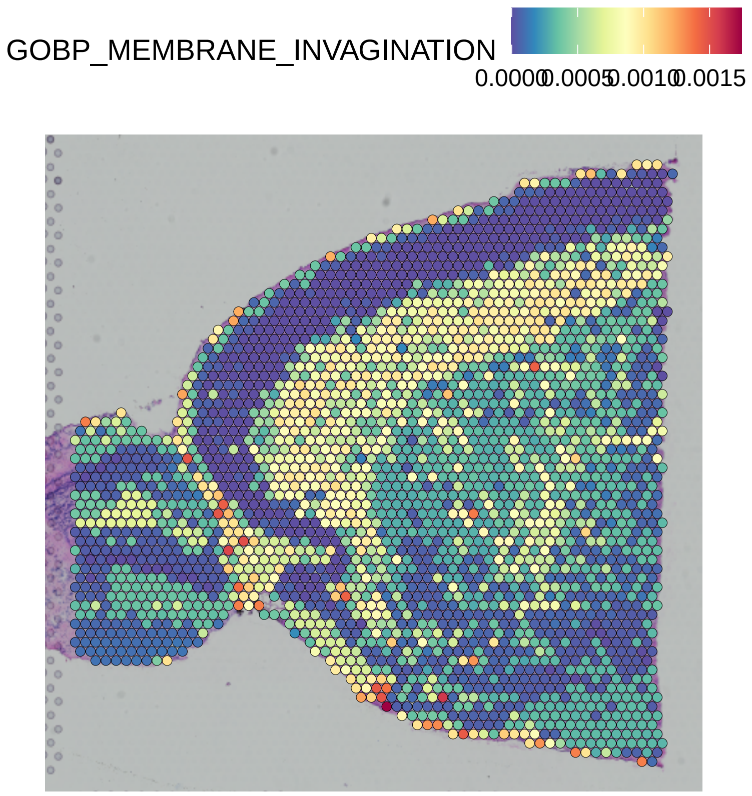


```r
# We can then visualize the gene set on the spatial map
# highlight the cluster 1 in the spatial map
SpatialDimPlot(brain,
               cells.highlight = CellsByIdentities(object = brain, idents = c(1)),
               facet.highlight = TRUE)
```

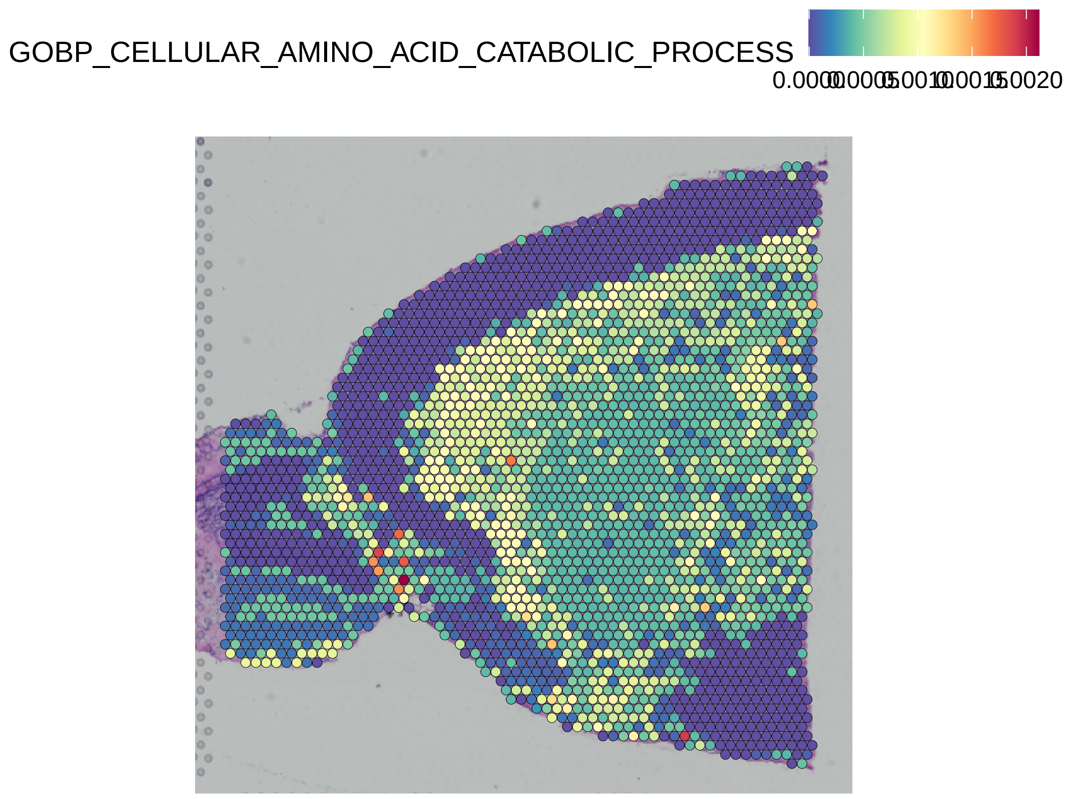
1



```
# add the label propagation probability to metadata for cluster_1 specific terms
brain@meta.data$GOBP_MEMBRANE_INVAGINATION <- cv.df[rownames(brain@meta.data),
                                                    "GOBP_MEMBRANE_INVAGINATION"]

SpatialFeaturePlot(brain, features = c("GOBP_MEMBRANE_INVAGINATION")) +
        theme(legend.position = "top")
```



```
# add the label propagation probability to metadata for cluster_1 specific terms
brain@meta.data$GOBP_CELLULAR_AMINO_ACID_CATABOLIC_PROCESS <- cv.df[rownames(brain@meta.data),
                                                           "GOBP_CELLULAR_AMINO_ACID_CATABOLIC_PROCESS"]

SpatialFeaturePlot(brain, features = c("GOBP_CELLULAR_AMINO_ACID_CATABOLIC_PROCESS")) +
        theme(legend.position = "top")
```


GOBP_CELLULAR_AMINO_ACID_CATABOLIC_PROCESS
0.0000
0.0005
0.0010
0.0015
0.0020