

BestPracticesST BiocBook

Table of contents

| | |
|--|-----------|
| Welcome | 7 |
| Preamble | 8 |
| Docker image | 9 |
| RStudio Server | 10 |
| Session info | 11 |
| I Introduction | 14 |
| 1 Introduction | 15 |
| 1.1 Overview | 15 |
| 1.2 Contents | 15 |
| 1.3 Scope and who this book is for | 15 |
| 1.4 Bioconductor | 16 |
| 1.5 Additional resources | 16 |
| 1.6 Contributions | 17 |
| References | 17 |
| 2 Spatial transcriptomics | 18 |
| 2.1 Overview | 18 |
| 2.2 Sequencing-based platforms | 18 |
| 2.2.1 10x Genomics Visium | 19 |
| 2.2.2 10x Genomics Visium HD | 20 |
| 2.2.3 Curio Seeker | 20 |
| 2.3 Molecule-based platforms | 20 |
| 2.3.1 10x Genomics Xenium | 20 |
| 2.3.2 Vizgen MERSCOPE | 21 |
| 2.3.3 NanoString CosMx | 21 |
| References | 21 |
| 3 Bioconductor data classes | 22 |
| 3.1 Overview | 22 |

| | | |
|-----------|--|-----------|
| 3.2 | SpatialExperiment class | 22 |
| 3.3 | Molecule-based data | 23 |
| 3.3.1 | MoleculeExperiment | 23 |
| 3.3.2 | SpatialFeatureExperiment | 23 |
| | References | 24 |
| II | Analysis steps | 25 |
| 4 | Analysis steps | 26 |
| 4.1 | Save data objects for re-use in later chapters | 26 |
| 4.1.1 | Human DLPFC dataset | 26 |
| | References | 28 |
| 5 | Load data | 29 |
| 5.1 | Overview | 29 |
| 5.2 | Dataset | 29 |
| 5.3 | Load data | 29 |
| 5.4 | SpatialExperiment object | 30 |
| 5.5 | Build object | 32 |
| 5.6 | Molecule-based data | 33 |
| | References | 33 |
| 6 | Quality control | 34 |
| 6.1 | Overview | 34 |
| 6.2 | Load data from previous steps | 35 |
| 6.3 | Plot data | 35 |
| 6.4 | Calculate QC metrics | 36 |
| 6.5 | Selecting thresholds | 37 |
| 6.5.1 | Library size | 37 |
| 6.5.2 | Number of expressed features | 42 |
| 6.5.3 | Proportion of mitochondrial reads | 46 |
| 6.5.4 | Number of cells per spot | 50 |
| 6.5.5 | Remove low-quality spots | 53 |
| 6.6 | Zero-cell and single-cell spots | 54 |
| 6.7 | Quality control at gene level | 55 |
| | References | 55 |
| 7 | Normalization | 56 |
| 7.1 | Overview | 56 |
| 7.2 | Load data from previous steps | 56 |
| 7.3 | Logcounts | 56 |
| | References | 58 |

| | |
|--|-----------|
| 8 Feature selection | 59 |
| 8.1 Overview | 59 |
| 8.2 Load data from previous steps | 59 |
| 8.3 Highly variable genes (HVGs) | 59 |
| 8.4 Spatially variable genes (SVGs) | 61 |
| 8.4.1 nnSVG | 62 |
| 8.4.2 Downstream analyses | 65 |
| References | 65 |
| 9 Dimensionality reduction | 66 |
| 9.1 Overview | 66 |
| 9.2 Load data from previous steps | 66 |
| 9.3 Principal component analysis (PCA) | 66 |
| 9.4 Uniform Manifold Approximation and Projection (UMAP) | 67 |
| 9.5 Visualizations | 67 |
| References | 69 |
| 10 Clustering | 70 |
| 10.1 Overview | 70 |
| 10.2 Load data from previous steps | 70 |
| 10.3 Non-spatial clustering | 71 |
| 10.3.1 Clustering using HVGs | 71 |
| 10.4 Spatially-aware clustering | 75 |
| 10.4.1 Clustering using SVGs | 75 |
| 10.4.2 Clustering using concatenated features | 77 |
| 10.4.3 Spatially-aware clustering algorithms | 77 |
| References | 78 |
| 11 Spot deconvolution | 79 |
| 11.1 Overview | 79 |
| 11.2 Previous steps | 79 |
| 11.3 Load data from previous steps | 79 |
| 11.4 Number of cells per spot | 79 |
| References | 80 |
| 12 Spatial registration | 81 |
| 12.1 Overview | 81 |
| 13 Differential expression | 82 |
| 13.1 Overview | 82 |
| 13.2 Load data from previous steps | 82 |
| 13.3 Differential expression testing | 82 |
| 13.4 Pseudobulking | 85 |

| | |
|---|------------|
| References | 86 |
| 14 Multiple samples | 87 |
| 14.1 Overview | 87 |
| 15 Spatial co-localization | 88 |
| 15.1 Overview | 88 |
| 15.2 Load data from previous steps | 88 |
| 15.3 Spatial co-localization of cell types within a single sample | 88 |
| References | 88 |
| III Workflows | 89 |
| 16 Workflows | 90 |
| 17 Human DLPFC workflow | 91 |
| 17.1 Overview | 91 |
| 17.2 Description of dataset | 91 |
| 17.3 Load data | 91 |
| 17.4 Plot data | 92 |
| 17.5 Quality control (QC) | 93 |
| 17.6 Normalization | 96 |
| 17.7 Feature selection | 97 |
| 17.8 Spatially-aware feature selection | 99 |
| 17.9 Dimensionality reduction | 102 |
| 17.10 Clustering | 103 |
| 17.11 Differential expression | 106 |
| References | 109 |
| 18 Mouse coronal workflow | 110 |
| 18.1 Overview | 110 |
| 18.2 Description of dataset | 110 |
| 18.3 Load data | 110 |
| 18.4 Plot data | 111 |
| 18.5 Quality control (QC) | 112 |
| 18.6 Normalization | 115 |
| 18.7 Feature selection | 116 |
| 18.8 Spatially-aware feature selection | 118 |
| 18.9 Dimensionality reduction | 121 |
| 18.10 Clustering | 122 |
| 18.11 Differential expression | 124 |
| References | 127 |

| | |
|---|------------|
| 19 spatialLIBD workflow | 128 |
| 19.1 Overview | 128 |
| 19.2 spatialLIBD | 129 |
| 19.2.1 Why use spatialLIBD? | 129 |
| 19.2.2 Want to learn more about spatialLIBD? | 132 |
| 19.3 Code prerequisites | 133 |
| 19.4 Prepare for spatialLIBD | 136 |
| 19.4.1 Basic information | 136 |
| 19.4.2 Gene annotation | 136 |
| 19.4.3 Extra information and filtering | 140 |
| 19.5 Explore the data | 141 |
| 19.6 Sharing your website | 147 |
| 19.7 Wrapping up | 151 |
| R session information | 151 |
| References | 155 |
| Appendices | 156 |
| A Related resources | 156 |
| A.1 Overview | 156 |
| A.2 Data preprocessing procedures for the Visium platform | 156 |
| A.3 Resources for other spatial omics platforms | 156 |
| A.4 Data structures | 157 |
| A.5 Statistical concepts | 157 |
| B Acknowledgments | 158 |
| B.1 Contributors | 158 |
| B.2 Additional acknowledgments | 158 |
| References | 158 |

Welcome

Published: September 26, 2024

This is the website for the online book '**Best Practices for Spatial Transcriptomics Analysis with Bioconductor**'.

This book provides discussion and interactive examples on best practices for computational analysis workflows for spatial transcriptomics data, using the [Bioconductor](#) framework within R. The chapters contain details on individual analysis steps as well as complete example workflows, with interactive example datasets and R code.

The book is organized into several parts, including introductory materials, analysis steps, and example workflows.

Additional details on analysis workflows for non-spatial single-cell data as well as further introductory materials on R and Bioconductor can be found in the related book [Orchestrating Single-Cell Analysis with Bioconductor \(OSCA\)](#).

Preamble

Package: BestPracticesSTBiocBook **Authors:** Lukas M. Weber [aut, cre], Leonardo Collado-Torres [aut], Stephanie C. Hicks [aut] **Compiled:** 2024-09-26 **Package version:** 0.99.0 **R version:** R version 4.4.1 (2024-06-14) **BioC version:** 3.20 **License:** CC BY 4.0

Docker image

A Docker image built from this repository is available here:

ghcr.io/lmweber/bestpracticesstbiocbook

💡 Get started now

You can get access to all the packages used in this book in < 1 minute, using this command in a terminal:

Listing 0.1 bash

```
docker run -it ghcr.io/lmweber/bestpracticesstbiocbook:devel R
```

RStudio Server

An RStudio Server instance can be initiated from the Docker image as follows:

Listing 0.2 bash

```
docker run \
  --volume <local_folder>:<destination_folder> \
  -e PASSWORD=OHCA \
  -p 8787:8787 \
  ghcr.io/lmweber/bestpracticesstbiocbook:devel
```

The initiated RStudio Server instance will be available at <https://localhost:8787>.

Session info

 Click to expand

Part I

Introduction

1 Introduction

1.1 Overview

Bioconductor

1.2 Contents

-
-
-
-

1.3 Scope and who this book is for

Preprocessing

Visium Data

1.4 Bioconductor

[Bioconductor](#)

1.5 Additional resources

- [Orchestrating Single-Cell Analysis with Bioconductor \(OSCA\)](#)
- [R for Data Science](#)
- [Data Carpentry Software Carpentry](#)

- - detailed guide
 - YouTube videos
- Visium Data Preprocessing

1.6 Contributions

[GitHub issues](#)

References

2 Spatial transcriptomics

2.1 Overview

Method
of the Year 2020

2.2 Sequencing-based platforms

2.2.1 10x Genomics Visium

10x Genomics Visium

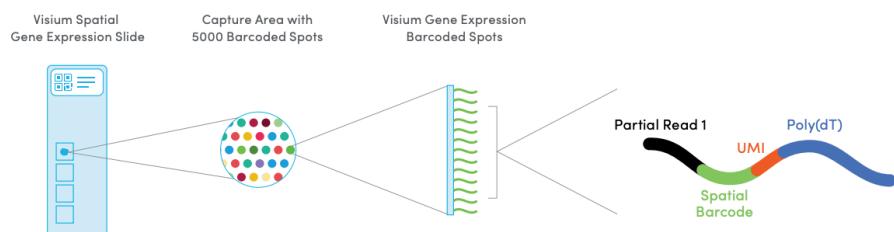


Figure 2.1: Schematic illustrating the 10x Genomics Visium platform. Source: [10x Genomics Visium](#)

2.2.2 10x Genomics Visium HD

[10x Genomics Visium HD](#)

2.2.3 Curio Seeker

[Curio Seeker](#)

2.3 Molecule-based platforms

2.3.1 10x Genomics Xenium

[10x Genomics](#)

2.3.2 Vizgen MERSCOPE

Vizgen

2.3.3 NanoString CosMx

NanoString

References

3 Bioconductor data classes

3.1 Overview

3.2 SpatialExperiment class

[SpatialExperiment](#)

[SingleCellExperiment](#)

[Bioconductor vignette](#)

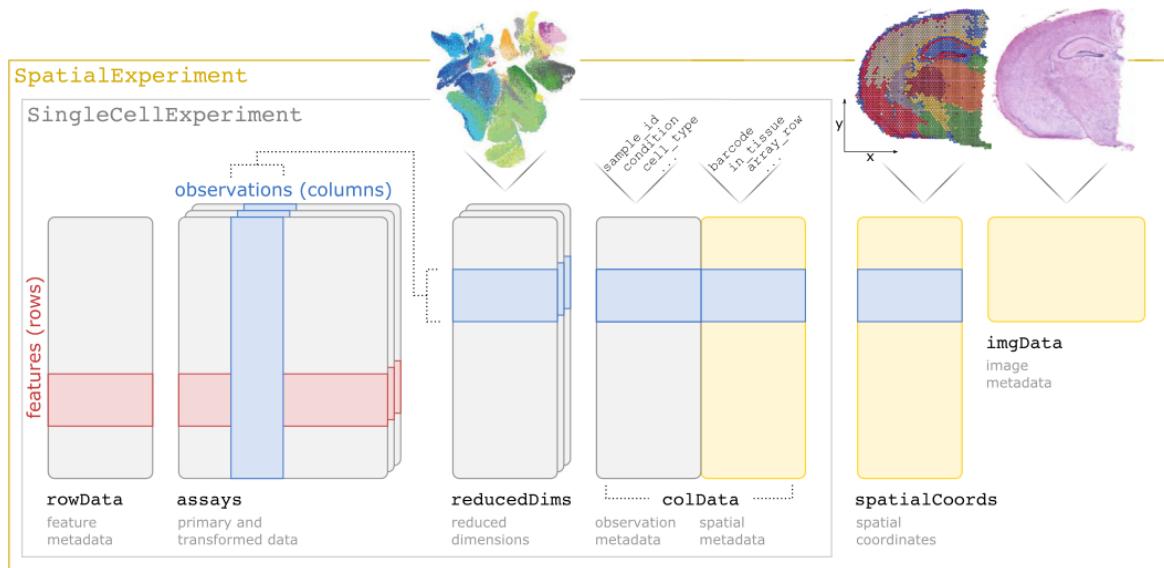


Figure 3.1: Overview of the `SpatialExperiment` data class for storing and manipulating spatial transcriptomics datasets within the Bioconductor framework.

3.3 Molecule-based data

3.3.1 MoleculeExperiment

[Bioconductor package](#)

3.3.2 SpatialFeatureExperiment

[Bioconductor package](#)

References

Part II

Analysis steps

4 Analysis steps

3

16

4.1 Save data objects for re-use in later chapters

4.1.1 Human DLPFC dataset

```
# LOAD DATA  
  
library(SpatialExperiment)  
library(STexampleData)  
spe <- Visium_humanDLPFC()  
  
# save object  
saveRDS(spe, file = "spe_load.rds")
```

```

# QUALITY CONTROL (QC)

library(scater)
# subset to keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]
# identify mitochondrial genes
is_mito <- grep("(^MT-)|(^mt-)", rowData(spe)$gene_name)
# calculate per-spot QC metrics
spe <- addPerCellQC(spe, subsets = list(mito = is_mito))
# select QC thresholds
qc_lib_size <- colData(spe)$sum < 600
qc_detected <- colData(spe)$detected < 400
qc_mito <- colData(spe)$subsets_mito_percent > 28
qc_cell_count <- colData(spe)$cell_count > 10
# combined set of discarded spots
discard <- qc_lib_size | qc_detected | qc_mito | qc_cell_count
colData(spe)$discard <- discard
# filter low-quality spots
spe <- spe[, !colData(spe)$discard]

# save object
saveRDS(spe, file = "spe_qc.rds")

```

```

# NORMALIZATION

library(scran)
# calculate logcounts using library size factors
spe <- logNormCounts(spe)

# save object
saveRDS(spe, file = "spe_logcounts.rds")

```

```

# FEATURE SELECTION

# remove mitochondrial genes
spe <- spe[!is_mito, ]
# fit mean-variance relationship
dec <- modelGeneVar(spe)
# select top HVGs
top_hvgs <- getTopHVGs(dec, prop = 0.1)

# save object

```

```

saveRDS(spe, file = "spe_hvgs.rds")
saveRDS(top_hvgs, file = "top_hvgs.rds")

# DIMENSIONALITY REDUCTION

# compute PCA
set.seed(123)
spe <- runPCA(spe, subset_row = top_hvgs)
# compute UMAP on top 50 PCs
set.seed(123)
spe <- runUMAP(spe, dimred = "PCA")
# update column names
colnames(reducedDim(spe, "UMAP")) <- paste0("UMAP", 1:2)

# save object
saveRDS(spe, file = "spe_reduceddims.rds")

```

```

# CLUSTERING

# graph-based clustering
set.seed(123)
k <- 10
g <- buildSNNGraph(spe, k = k, use.dimred = "PCA")
g_walk <- igraph::cluster_walktrap(g)
clus <- g_walk$membership
colLabels(spe) <- factor(clus)

# save object
saveRDS(spe, file = "spe_cluster.rds")

```

References

5 Load data

5.1 Overview

3

Preprocessing

Visium Data

STexampleData

5.2 Dataset

17

5.3 Load data

STexampleData

29

```

library(SpatialExperiment)
library(STexampleData)

# load object
spe <- Visium_humanDLPFC()

```

5.4 SpatialExperiment object

3

```

# check object
spe
##  class: SpatialExperiment
##  dim: 33538 4992
##  metadata(0):
##  assays(1):
##    counts
##    rownames(33538): ENSG00000243485 ENSG00000237613 ... ENSG00000277475
##    ENSG00000268674
##    rowData names(3): gene_id gene_name feature_type
##    colnames(4992): AAACAACGAATAGTTC-1 AAACAAAGTATCTCCCCA-1 ...
##    TTGTTTGTATTACACG-1 TTGTTTGTGTAAATTC-1
##    colData names(8): barcode_id sample_id ... reference cell_count
##    reducedDimNames(0):
##    mainExpName: NULL
##    altExpNames(0):
##    spatialCoords names(2) : pxl_col_in_fullres pxl_row_in_fullres
##    imgData names(4): sample_id image_id data scaleFactor

# number of genes (rows) and spots (columns)
dim(spe)
## [1] 33538 4992

# names of 'assays'
assayNames(spe)
## [1] "counts"

# row (gene) data
head(rowData(spe))
## DataFrame with 6 rows and 3 columns

```

```

##          gene_id   gene_name    feature_type
## <character> <character>      <character>
## ENSG00000243485 ENSG00000243485 MIR1302-2HG Gene Expression
## ENSG00000237613 ENSG00000237613 FAM138A Gene Expression
## ENSG00000186092 ENSG00000186092 OR4F5 Gene Expression
## ENSG00000238009 ENSG00000238009 AL627309.1 Gene Expression
## ENSG00000239945 ENSG00000239945 AL627309.3 Gene Expression
## ENSG00000239906 ENSG00000239906 AL627309.2 Gene Expression

# column (spot) data
head(colData(spe))
## DataFrame with 6 rows and 8 columns
##          barcode_id sample_id in_tissue array_row
## <character> <character> <integer> <integer>
## AAACAACGAATAGTTC-1 AAACAACGAATAGTTC-1 sample_151673 0 0
## AAACAAGTATCTCCCA-1 AAACAAGTATCTCCCA-1 sample_151673 1 50
## AAACAATCTACTAGCA-1 AAACAATCTACTAGCA-1 sample_151673 1 3
## AACACCCAATAACTGC-1 AACACCCAATAACTGC-1 sample_151673 1 59
## AACAGAGCGACTCCT-1 AACAGAGCGACTCCT-1 sample_151673 1 14
## AACAGCTTCAGAAG-1 AACAGCTTCAGAAG-1 sample_151673 1 43
##          array_col ground_truth reference cell_count
## <integer> <character> <character> <integer>
## AAACAACGAATAGTTC-1 16 NA NA NA
## AAACAAGTATCTCCCA-1 102 Layer3 Layer3 6
## AAACAATCTACTAGCA-1 43 Layer1 Layer1 16
## AACACCCAATAACTGC-1 19 WM WM 5
## AACAGAGCGACTCCT-1 94 Layer3 Layer3 2
## AACAGCTTCAGAAG-1 9 Layer5 Layer5 4

# spatial coordinates
head(spatialCoords(spe))
##          pxl_col_in_fullres pxl_row_in_fullres
## <character> <integer>
## AAACAACGAATAGTTC-1 3913 2435
## AAACAAGTATCTCCCA-1 9791 8468
## AAACAATCTACTAGCA-1 5769 2807
## AACACCCAATAACTGC-1 4068 9505
## AACAGAGCGACTCCT-1 9271 4151
## AACAGCTTCAGAAG-1 3393 7583

# image data
imgData(spe)
## DataFrame with 2 rows and 4 columns

```

```

##           sample_id    image_id   data scaleFactor
##           <character> <character> <list>    <numeric>
## 1 sample_151673      lowres     ##### 0.0450045
## 2 sample_151673      hires     ##### 0.1500150

```

5.5 Build object

SpatialExperiment

```

# create data
n_genes <- 200
n_spots <- 100

counts <- matrix(0, nrow = n_genes, ncol = n_spots)

row_data <- DataFrame(
  gene_name = paste0("gene", sprintf("%03d", seq_len(n_genes)))
)

col_data <- DataFrame(
  sample_id = rep("sample01", n_spots)
)

spatial_coords <- matrix(0, nrow = n_spots, ncol = 2)
colnames(spatial_coords) <- c("x", "y")

# create SpatialExperiment object
spe <- SpatialExperiment(
  assays = list(counts = counts),
  colData = col_data,
  rowData = row_data,
  spatialCoords = spatial_coords
)

```

5.6 Molecule-based data

3

References

6 Quality control

6.1 Overview

-
-
-
-

6.2 Load data from previous steps

4

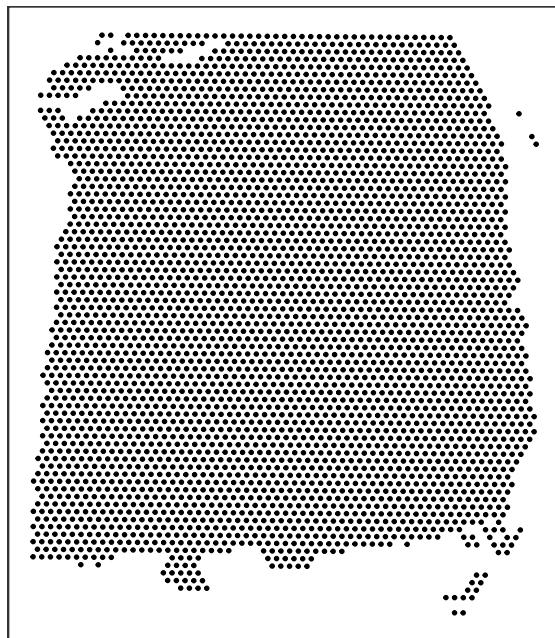
```
library(SpatialExperiment)
spe <- readRDS("spe_load.rds")
```

6.3 Plot data

ggspavis

```
library(ggspavis)
```

```
# plot spatial coordinates (spots)
plotSpots(spe)
```



6.4 Calculate QC metrics

```
library(scater)

# subset to keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]
dim(spe)
## [1] 33538 3639

# identify mitochondrial genes
is_mito <- grep("(^MT-)|(^mt-)", rowData(spe)$gene_name)
table(is_mito)
## is_mito
## FALSE TRUE
## 33525 13
rowData(spe)$gene_name[is_mito]
## [1] "MT-ND1"   "MT-ND2"   "MT-CO1"   "MT-CO2"   "MT-ATP8"   "MT-ATP6"   "MT-CO3"
## [8] "MT-ND3"   "MT-ND4L"  "MT-ND4"   "MT-ND5"   "MT-ND6"   "MT-CYB"

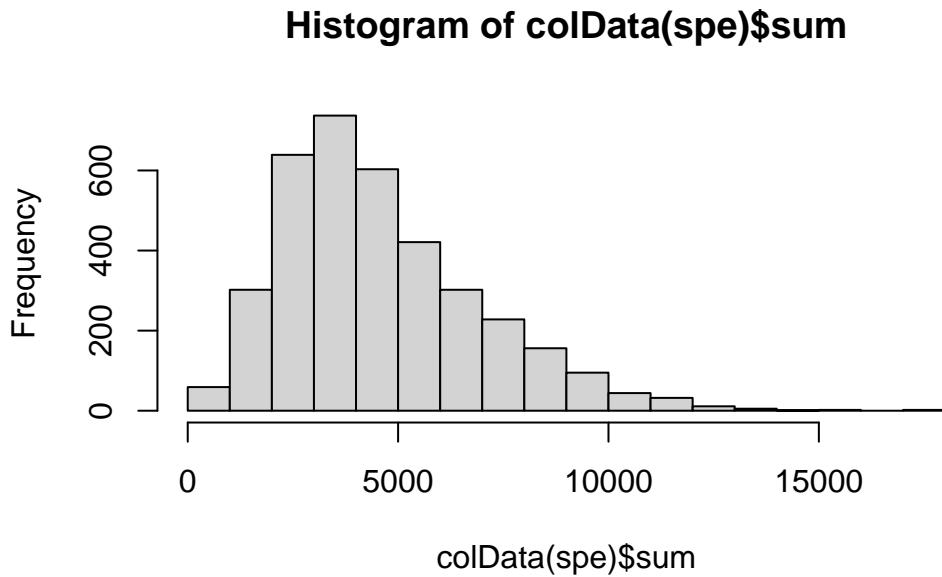
# calculate per-spot QC metrics and store in colData
spe <- addPerCellQC(spe, subsets = list(mito = is_mito))
head(colData(spe))
## DataFrame with 6 rows and 14 columns
##           barcode_id sample_id in_tissue array_row
##           <character> <character> <integer> <integer>
## 1 AAACAAGTATCTCCA-1 sample_151673      1      50
## 2 AAACAATCTACTAGCA-1 sample_151673      1       3
## 3 AACACCCAATAACTGC-1 sample_151673      1      59
## 4 AACAGAGCGACTCCT-1  sample_151673      1      14
## 5 AACAGCTTCAGAAG-1   sample_151673      1      43
## 6 AACAGGGTCTATATT-1 sample_151673      1      47
##           array_col ground_truth reference cell_count
##           <character> <character> <integer> <integer>
```

| | <i><integer></i> | <i><character></i> | <i><character></i> | <i><integer></i> | <i><numeric></i> |
|----|------------------------|--------------------------|--------------------------|------------------------|------------------------|
| ## | | | | | |
| ## | AAACAAGTATCTCCA-1 | 102 | Layer3 | Layer3 | 6 8458 |
| ## | AAACAATCTACTAGCA-1 | 43 | Layer1 | Layer1 | 16 1667 |
| ## | AAACACCAATAACTGC-1 | 19 | WM | WM | 5 3769 |
| ## | AAACAGAGCGACTCCT-1 | 94 | Layer3 | Layer3 | 2 5433 |
| ## | AAACAGCTTCAGAAG-1 | 9 | Layer5 | Layer5 | 4 4278 |
| ## | AAACAGGGTCTATATT-1 | 13 | Layer6 | Layer6 | 6 4004 |
| ## | | detected | subsets_mito_sum | subsets_mito_detected | |
| ## | | <i><numeric></i> | <i><numeric></i> | | <i><numeric></i> |
| ## | AAACAAGTATCTCCA-1 | 3586 | | 1407 | 13 |
| ## | AAACAATCTACTAGCA-1 | 1150 | | 204 | 11 |
| ## | AAACACCAATAACTGC-1 | 1960 | | 430 | 13 |
| ## | AAACAGAGCGACTCCT-1 | 2424 | | 1316 | 13 |
| ## | AAACAGCTTCAGAAG-1 | 2264 | | 651 | 12 |
| ## | AAACAGGGTCTATATT-1 | 2178 | | 621 | 13 |
| ## | | subsets_mito_percent | | total | |
| ## | | <i><numeric></i> | <i><numeric></i> | | |
| ## | AAACAAGTATCTCCA-1 | 16.6351 | | 8458 | |
| ## | AAACAATCTACTAGCA-1 | 12.2376 | | 1667 | |
| ## | AAACACCAATAACTGC-1 | 11.4089 | | 3769 | |
| ## | AAACAGAGCGACTCCT-1 | 24.2223 | | 5433 | |
| ## | AAACAGCTTCAGAAG-1 | 15.2174 | | 4278 | |
| ## | AAACAGGGTCTATATT-1 | 15.5095 | | 4004 | |

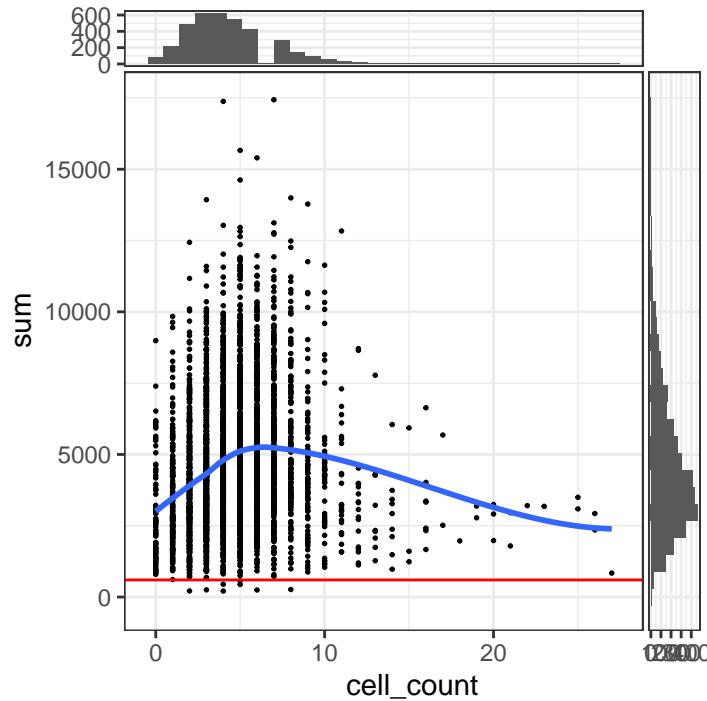
6.5 Selecting thresholds

6.5.1 Library size

```
# histogram of library sizes  
hist(colData(spe)$sum, breaks = 20)
```



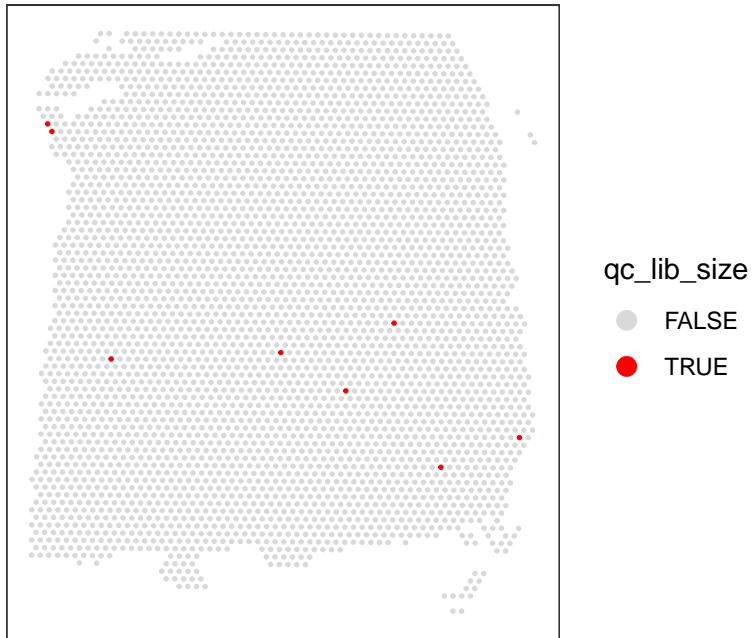
```
# plot library size vs. number of cells per spot  
plotSpotQC(spe, plot_type = "scatter",  
            x_metric = "cell_count", y_metric = "sum",  
            y_threshold = 600)  
## `geom_smooth()` using formula = 'y ~ x'  
## `stat_xsidebin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat_ysidebin()` using `bins = 30`. Pick better value with `binwidth`.
```



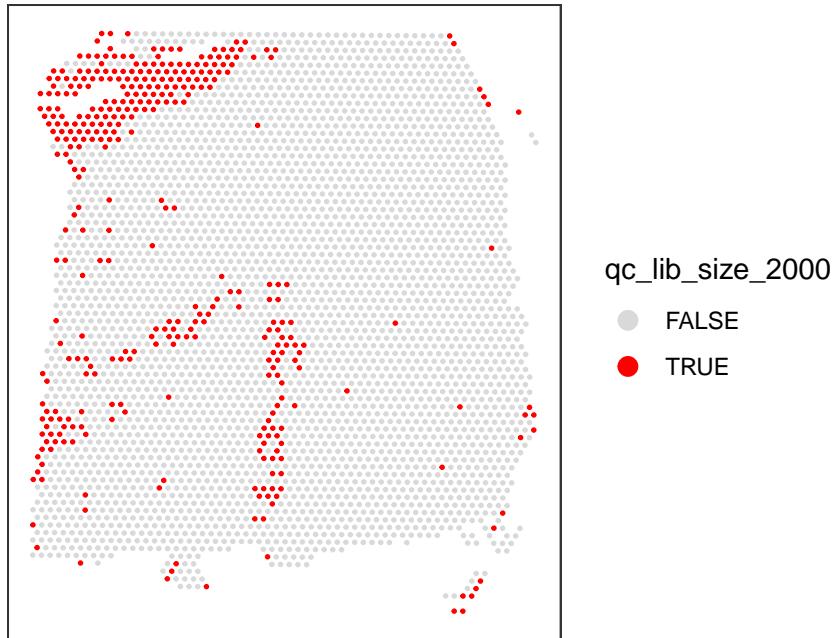
```
# select QC threshold for library size
qc_lib_size <- colData(spe)$sum < 600
table(qc_lib_size)
##   qc_lib_size
##   FALSE  TRUE
##     3631      8

colData(spe)$qc_lib_size <- qc_lib_size
```

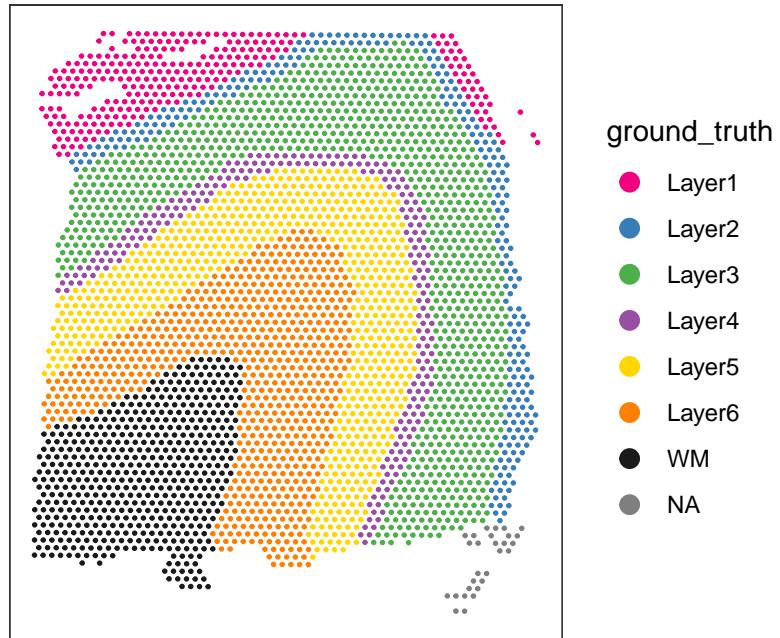
```
# check spatial pattern of discarded spots
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_lib_size")
```



```
# check spatial pattern of discarded spots if threshold is too high
qc_lib_size_2000 <- colData(spe)$sum < 2000
colData(spe)$qc_lib_size_2000 <- qc_lib_size_2000
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_lib_size_2000")
```

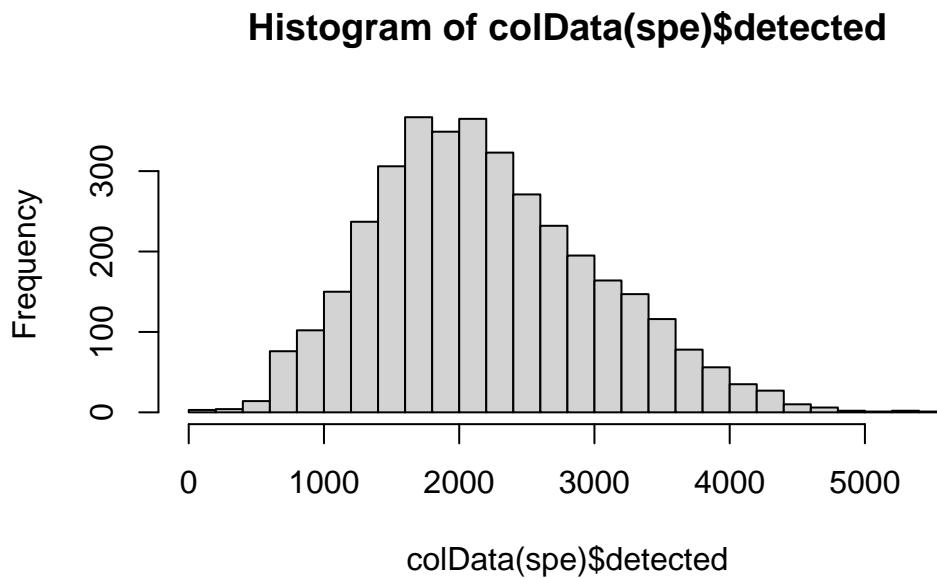


```
# plot ground truth (manually annotated) layers
plotSpots(spe, annotate = "ground_truth",
           pal = "libd_layer_colors")
```

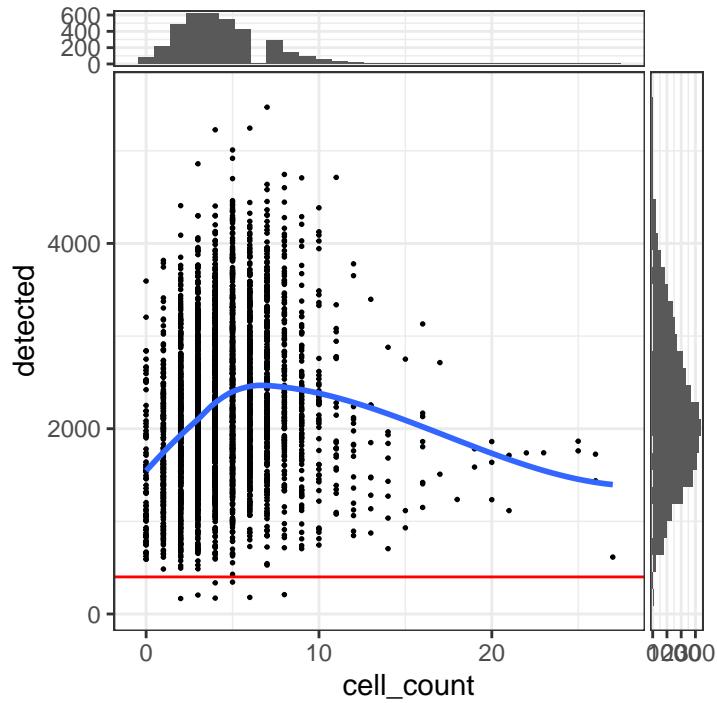


6.5.2 Number of expressed features

```
# histogram of numbers of expressed genes  
hist(colData(spe)$detected, breaks = 20)
```



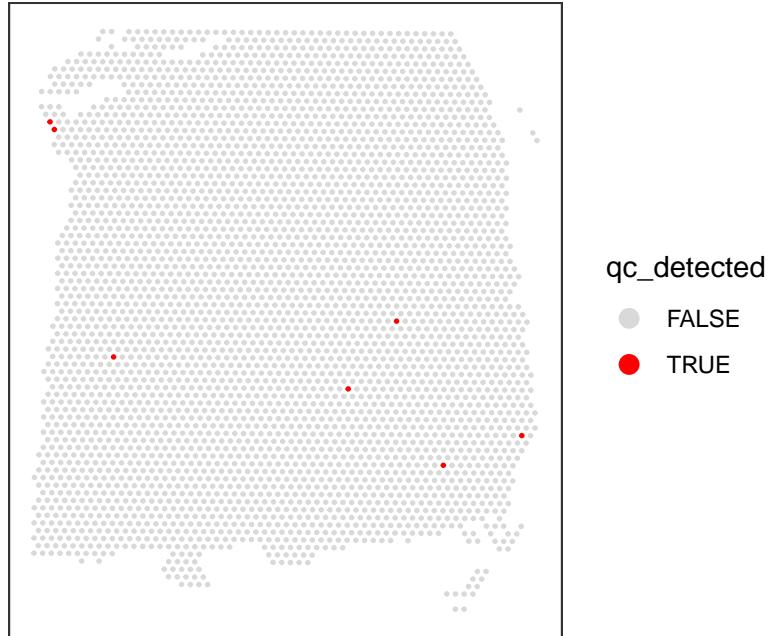
```
# plot number of expressed genes vs. number of cells per spot
plotSpotQC(spe, plot_type = "scatter",
            x_metric = "cell_count", y_metric = "detected",
            y_threshold = 400)
## `geom_smooth()` using formula = 'y ~ x'
## `stat_xsidebin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_ysidebin()` using `bins = 30`. Pick better value with `binwidth`.
```



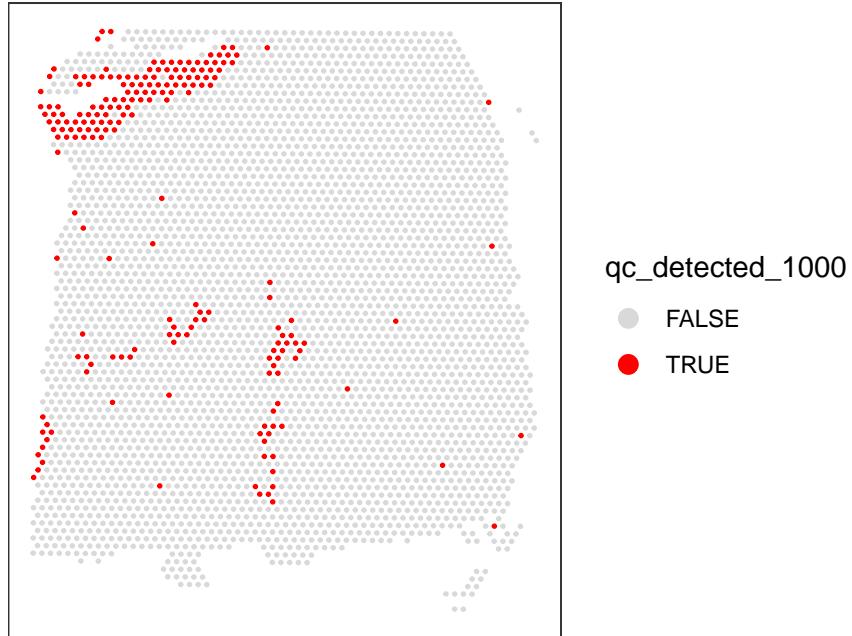
```
# select QC threshold for number of expressed genes
qc_detected <- colData(spe)$detected < 400
table(qc_detected)
##   qc_detected
##   FALSE  TRUE
##   3632     7

colData(spe)$qc_detected <- qc_detected

# check spatial pattern of discarded spots
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_detected")
```



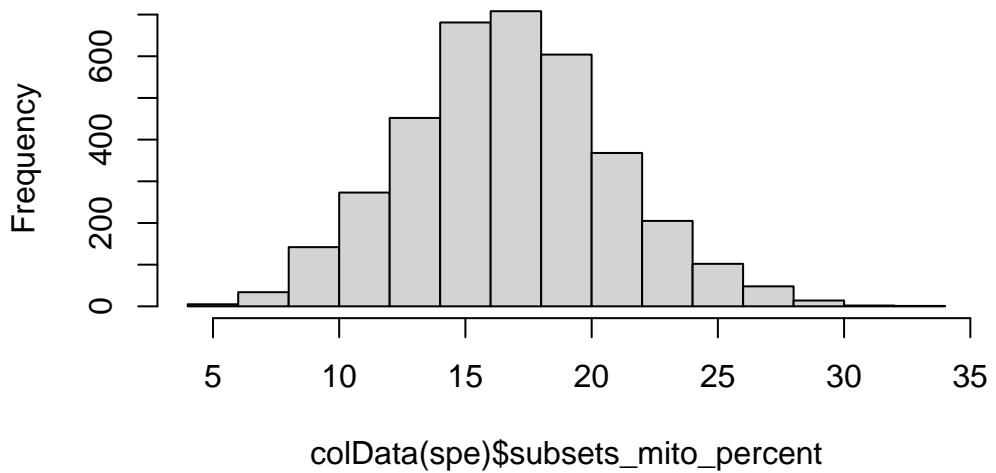
```
# check spatial pattern of discarded spots if threshold is too high
qc_detected_1000 <- colData(spe)$detected < 1000
colData(spe)$qc_detected_1000 <- qc_detected_1000
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_detected_1000")
```



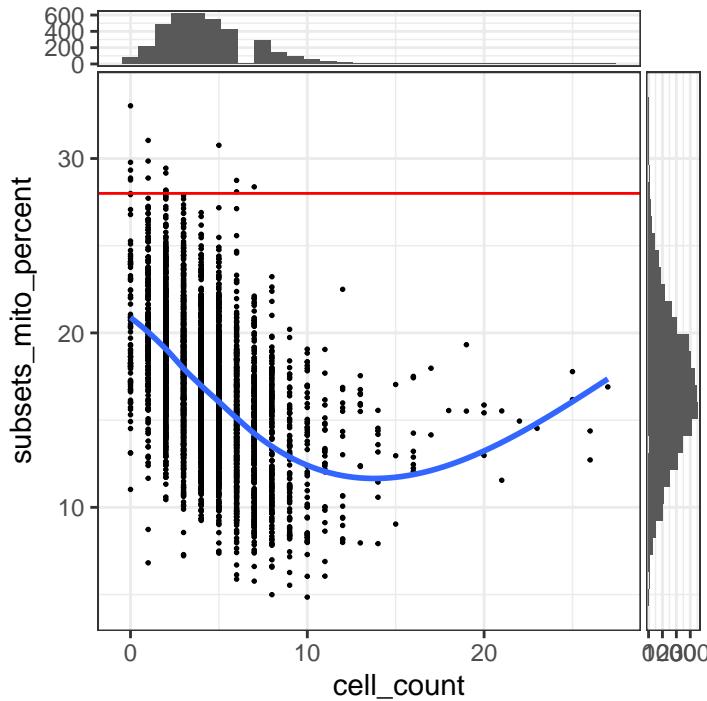
6.5.3 Proportion of mitochondrial reads

```
# histogram of mitochondrial read proportions  
hist(colData(spe)$subsets_mito_percent, breaks = 20)
```

Histogram of colData(spe)\$subsets_mito_percent



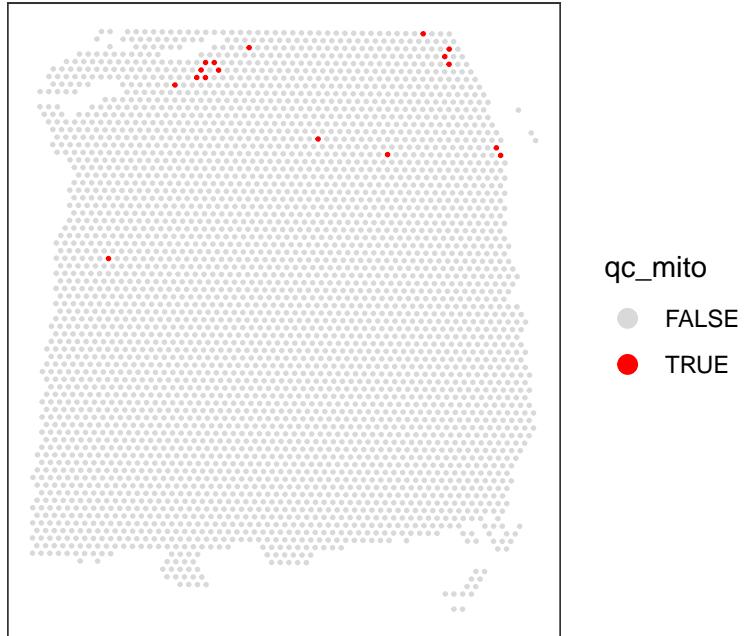
```
# plot mitochondrial read proportion vs. number of cells per spot
plotSpotQC(spe, plot_type = "scatter",
            x_metric = "cell_count", y_metric = "subsets_mito_percent",
            y_threshold = 28)
## `geom_smooth()` using formula = 'y ~ x'
## `stat_xsidebin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_ysidebin()` using `bins = 30`. Pick better value with `binwidth`.
```



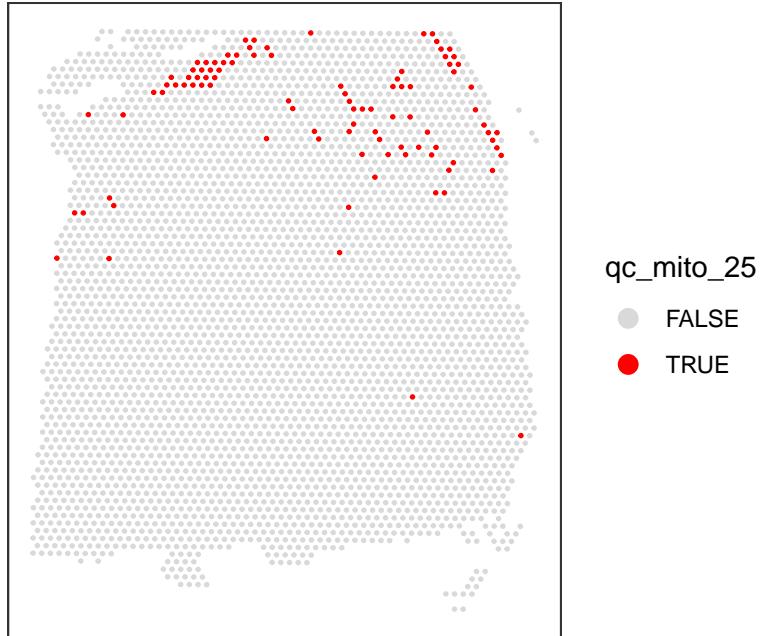
```
# select QC threshold for mitochondrial read proportion
qc_mito <- colData(spe)$subsets_mito_percent > 28
table(qc_mito)
##   qc_mito
## FALSE  TRUE
##    3622    17

colData(spe)$qc_mito <- qc_mito

# check spatial pattern of discarded spots
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_mito")
```



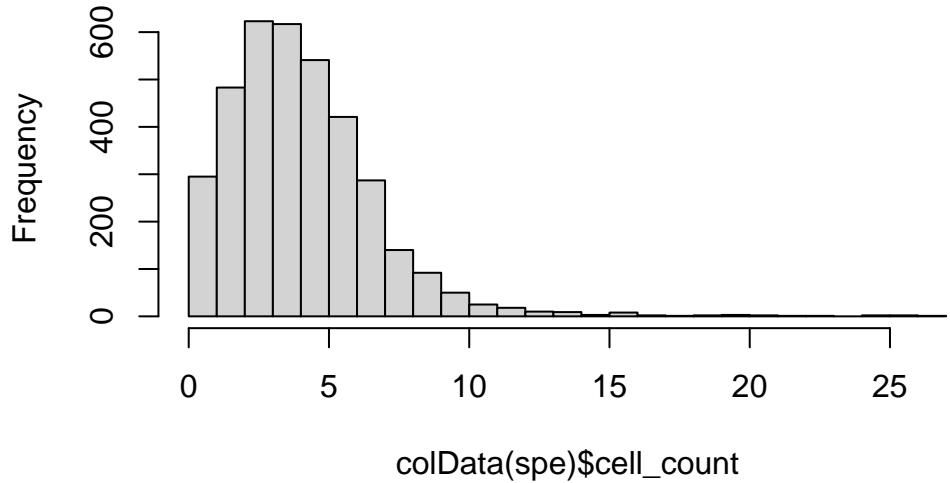
```
# check spatial pattern of discarded spots if threshold is too high
qc_mito_25 <- colData(spe)$subsets_mito_percent > 25
colData(spe)$qc_mito_25 <- qc_mito_25
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_mito_25")
```



6.5.4 Number of cells per spot

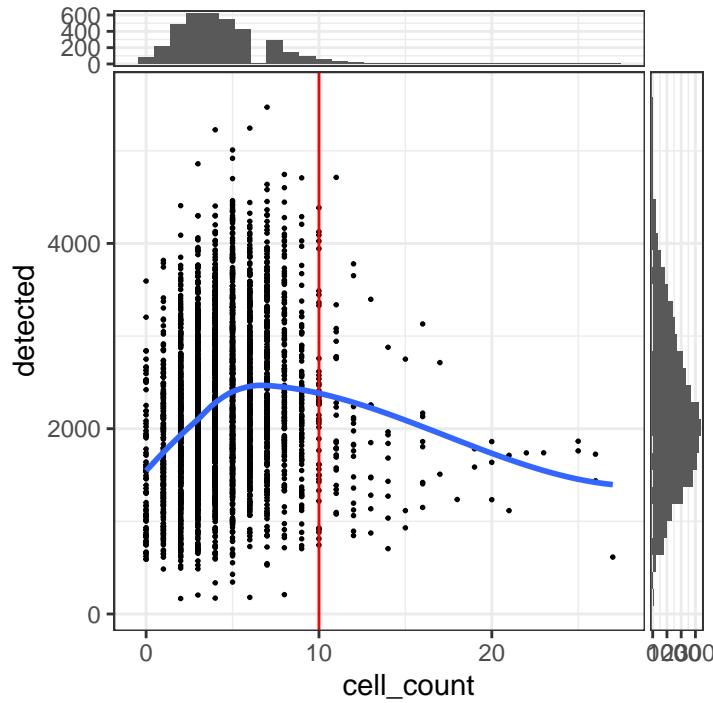
```
# histogram of cell counts  
hist(colData(spe)$cell_count, breaks = 20)
```

Histogram of colData(spe)\$cell_count



```
# distribution of cells per spot
tbl_cells_per_spot <- table(colData(spe)$cell_count)
```

```
# plot number of expressed genes vs. number of cells per spot
plotSpotQC(spe, plot_type = "scatter",
            x_metric = "cell_count", y_metric = "detected",
            x_threshold = 10)
## `geom_smooth()` using formula = 'y ~ x'
## `stat_xsidebin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_ysidebin()` using `bins = 30`. Pick better value with `binwidth`.
```



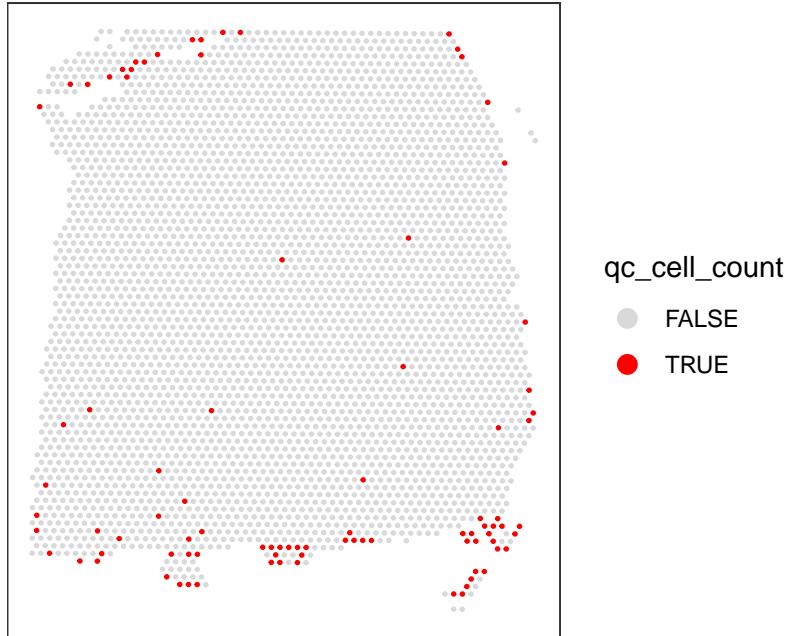
```

# select QC threshold for number of cells per spot
qc_cell_count <- colData(spe)$cell_count > 10
table(qc_cell_count)
##   qc_cell_count
##   FALSE   TRUE
##   3549     90

colData(spe)$qc_cell_count <- qc_cell_count

# check spatial pattern of discarded spots
plotSpotQC(spe, plot_type = "spot",
            annotate = "qc_cell_count")

```



6.5.5 Remove low-quality spots

```
# number of discarded spots for each metric
apply(cbind(qc_lib_size, qc_detected, qc_mito, qc_cell_count), 2, sum)
##      qc_lib_size    qc_detected      qc_mito qc_cell_count
##                 8                  7                 17                 90

# combined set of discarded spots
discard <- qc_lib_size | qc_detected | qc_mito | qc_cell_count
table(discard)
##   discard
```

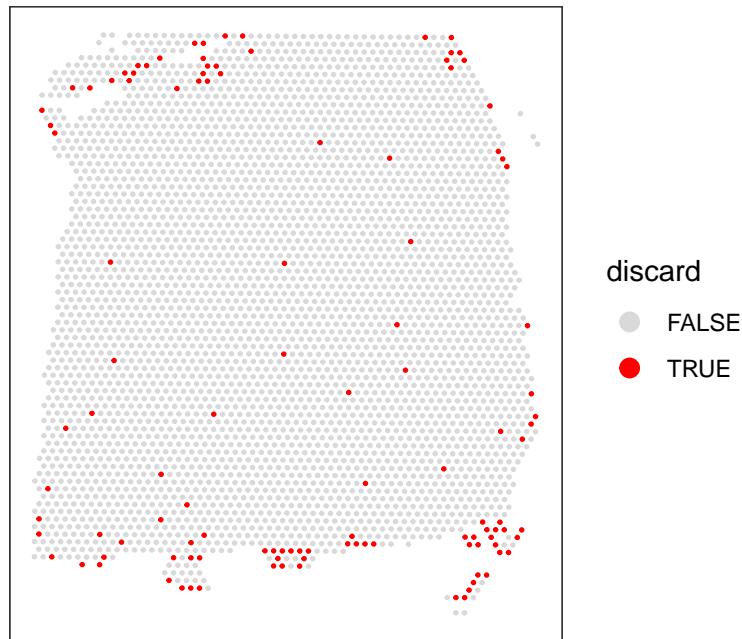
```

## FALSE TRUE
## 3524 115

# store in object
colData(spe)$discard <- discard

# check spatial pattern of combined set of discarded spots
plotSpotQC(spe, plot_type = "spot",
            annotate = "discard")

```



```

# remove combined set of low-quality spots
spe <- spe[, !colData(spe)$discard]
dim(spe)
## [1] 33538 3524

```

6.6 Zero-cell and single-cell spots

```

# distribution of cells per spot
tbl_cells_per_spot[1:13]
##
##    0   1   2   3   4   5   6   7   8   9   10  11  12
##   84 211 483 623 617 541 421 287 140  92  50  25  18

# as proportions
prop_cells_per_spot <- round(tbl_cells_per_spot / sum(tbl_cells_per_spot), 2)
prop_cells_per_spot[1:13]
##
##    0     1     2     3     4     5     6     7     8     9     10    11    12
##  0.02  0.06  0.13  0.17  0.17  0.15  0.12  0.08  0.04  0.03  0.01  0.01  0.00

```

6.7 Quality control at gene level

References

7 Normalization

7.1 Overview

7.2 Load data from previous steps

4

```
library(SpatialExperiment)
spe <- readRDS("spe_qc.rds")
```

7.3 Logcounts

```

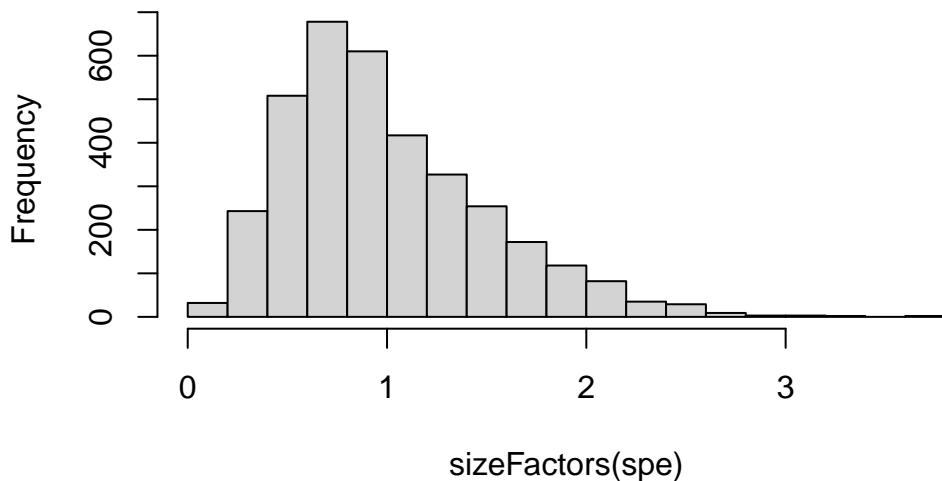
library(scran)

# calculate library size factors
spe <- computeLibraryFactors(spe)

summary(sizeFactors(spe))
##      Min. 1st Qu. Median Mean 3rd Qu. Max.
## 0.1321 0.6312 0.9000 1.0000 1.2849 3.7582
hist(sizeFactors(spe), breaks = 20)

```

Histogram of sizeFactors(spe)



```

# calculate logcounts and store in object
spe <- logNormCounts(spe)

# check
assayNames(spe)
## [1] "counts"      "logcounts"
dim(counts(spe))
## [1] 33538 3524
dim(logcounts(spe))
## [1] 33538 3524

```

References

8 Feature selection

8.1 Overview

8.2 Load data from previous steps

4

```
library(SpatialExperiment)
spe <- readRDS("spe_logcounts.rds")
```

8.3 Highly variable genes (HVGs)

```

# identify mitochondrial genes
is_mito <- grep("(^MT-)|(mt-)", rowData(spe)$gene_name)
table(is_mito)
##  is_mito
## FALSE  TRUE
## 33525    13

# remove mitochondrial genes
spe <- spe[!is_mito, ]
dim(spe)
## [1] 33525 3524

```

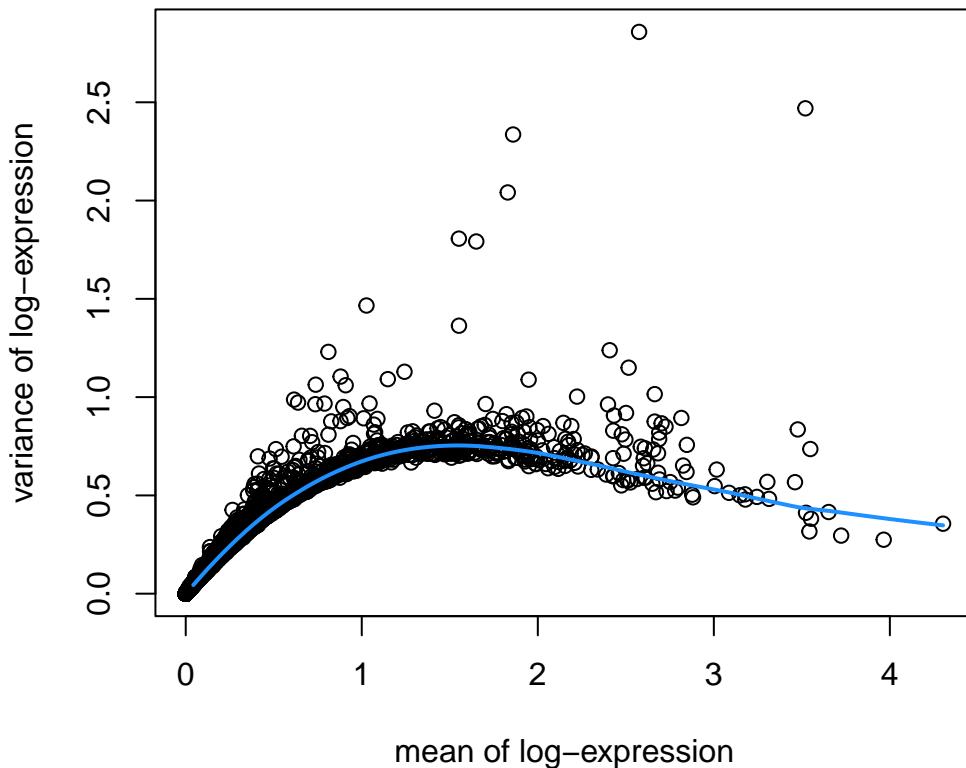
```

library(scran)

# fit mean-variance relationship
dec <- modelGeneVar(spe)

# visualize mean-variance relationship
fit <- metadata(dec)
plot(fit$mean, fit$var,
      xlab = "mean of log-expression", ylab = "variance of log-expression")
curve(fit$trend(x), col = "dodgerblue", add = TRUE, lwd = 2)

```



```
# select top HVGs
top_hvgs <- getTopHVGs(dec, prop = 0.1)
length(top_hvgs)
## [1] 1438
```

8.4 Spatially variable genes (SVGs)

- Bioconductor
 - GitHub
 - GitHub
 - GitHub
- Moran's I Geary's C

8.4.1 nnSVG

[nnSVG](#)

```
library(nnSVG)
```

```
# subsample spots for faster runtime in this example
# note: skip this step in full analysis
n <- 100
set.seed(123)
ix <- sample(seq_len(n), n)
spe_nnSVG <- spe[, ix]

# filter low-expressed and mitochondrial genes
# using stringent filtering for faster runtime in this example
# note: use default filtering in full analysis
spe_nnSVG <- filter_genes(
  spe_nnSVG, filter_genes_ncounts = 10, filter_genes_pcspots = 3
)
## Gene filtering: removing mitochondrial genes
## removed 0 mitochondrial genes
## Gene filtering: retaining genes with at least 10 counts in at least 3% (n = 3) of spatial
```

```

## removed 33353 out of 33525 genes due to low expression

# re-calculate logcounts after filtering
spe_nnSVG <- logNormCounts(spe_nnSVG)

# run nnSVG
set.seed(123)
spe_nnSVG <- nnSVG(spe_nnSVG)

# investigate results

# show results
head(rowData(spe_nnSVG), 3)
## DataFrame with 3 rows and 18 columns
##           gene_id   gene_name feature_type subsets_mito
##           <character> <character> <character> <logical>
## ENSG00000074800 ENSG00000074800      EN01 Gene Expression FALSE
## ENSG00000171603 ENSG00000171603      CLSTN1 Gene Expression FALSE
## ENSG00000162545 ENSG00000162545      CAMK2N1 Gene Expression FALSE
##           sigma.sq    tau.sq      phi    loglik runtime     mean
##           <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000074800 0.00840777 0.518011 15.82454 -109.808 0.016 1.75820
## ENSG00000171603 0.43102027 0.298698 18.12310 -123.212 0.019 2.07433
## ENSG00000162545 0.15373440 0.547736 2.38046 -119.576 0.014 2.63576
##           var     spcov prop_sv loglik_lm LR_stat rank
##           <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000074800 0.530929 0.0521523 0.0159716 -109.735 -0.145454 167
## ENSG00000171603 0.741077 0.3164987 0.5906665 -126.409 6.394009 86
## ENSG00000162545 0.702846 0.1487576 0.2191603 -123.760 8.368951 70
##           pval     padj
##           <numeric> <numeric>
## ENSG00000074800 1.0000000 1.0000000
## ENSG00000171603 0.0408845 0.0817690
## ENSG00000162545 0.0152302 0.0374227

# number of significant SVGs
table(rowData(spe_nnSVG)$padj <= 0.05)
##
## FALSE  TRUE
## 96    76

# show results for top n SVGs

```

```

rowData(spe_nnSVG)[order(rowData(spe_nnSVG)$rank)[1:6], ]
## DataFrame with 6 rows and 18 columns
##           gene_id   gene_name feature_type subsets_mito
##           <character> <character> <character>    <logical>
## ENSG00000197971 ENSG00000197971      MBP Gene Expression FALSE
## ENSG00000123560 ENSG00000123560      PLP1 Gene Expression FALSE
## ENSG00000109846 ENSG00000109846     CRYAB Gene Expression FALSE
## ENSG00000173786 ENSG00000173786      CNP Gene Expression FALSE
## ENSG00000131095 ENSG00000131095     GFAP Gene Expression FALSE
## ENSG00000160307 ENSG00000160307     S100B Gene Expression FALSE
##           sigma.sq  tau.sq      phi    loglik runtime    mean
##           <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000197971 3.92430 0.175336 0.93014 -118.372 0.020 3.78073
## ENSG00000123560 3.23544 0.461590 1.03983 -140.269 0.016 2.86143
## ENSG00000109846 1.89968 0.281795 1.88555 -126.692 0.016 1.86058
## ENSG00000173786 2.28793 0.402524 1.02675 -129.206 0.020 1.79558
## ENSG00000131095 2.23709 0.461297 2.49083 -149.004 0.016 1.94543
## ENSG00000160307 1.24179 0.155737 5.76279 -129.722 0.014 1.82695
##           var    spcov prop_sv loglik_lm LR_stat rank
##           <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000197971 2.76535 0.523969 0.957231 -192.250 147.7556 1
## ENSG00000123560 3.02555 0.628613 0.875146 -196.746 112.9549 2
## ENSG00000109846 1.88123 0.740785 0.870824 -172.988 92.5906 3
## ENSG00000173786 1.97274 0.842396 0.850388 -175.363 92.3137 4
## ENSG00000131095 2.71281 0.768823 0.829047 -191.291 84.5739 5
## ENSG00000160307 1.46946 0.609953 0.888562 -160.636 61.8281 6
##           pval      padj
##           <numeric> <numeric>
## ENSG00000197971 0.00000e+00 0.00000e+00
## ENSG00000123560 0.00000e+00 0.00000e+00
## ENSG00000109846 0.00000e+00 0.00000e+00
## ENSG00000173786 0.00000e+00 0.00000e+00
## ENSG00000131095 0.00000e+00 0.00000e+00
## ENSG00000160307 3.75255e-14 1.07573e-12

# identify top-ranked SVG
rowData(spe_nnSVG)$gene_name[which(rowData(spe_nnSVG)$rank == 1)]
## [1] "MBP"

```

8.4.2 Downstream analyses

10

References

9 Dimensionality reduction

9.1 Overview

9.2 Load data from previous steps

4

```
library(SpatialExperiment)
spe <- readRDS("spe_hvgs.rds")
top_hvgs <- readRDS("top_hvgs.rds")
```

9.3 Principal component analysis (PCA)

```
library(scater)
```

```
# compute PCA
set.seed(123)
spe <- runPCA(spe, subset_row = top_hvgs)

reducedDimNames(spe)
## [1] "PCA"
dim(reducedDim(spe, "PCA"))
## [1] 3524   50
```

9.4 Uniform Manifold Approximation and Projection (UMAP)

```
# compute UMAP on top 50 PCs
set.seed(123)
spe <- runUMAP(spe, dimred = "PCA")

reducedDimNames(spe)
## [1] "PCA"    "UMAP"
dim(reducedDim(spe, "UMAP"))
## [1] 3524    2

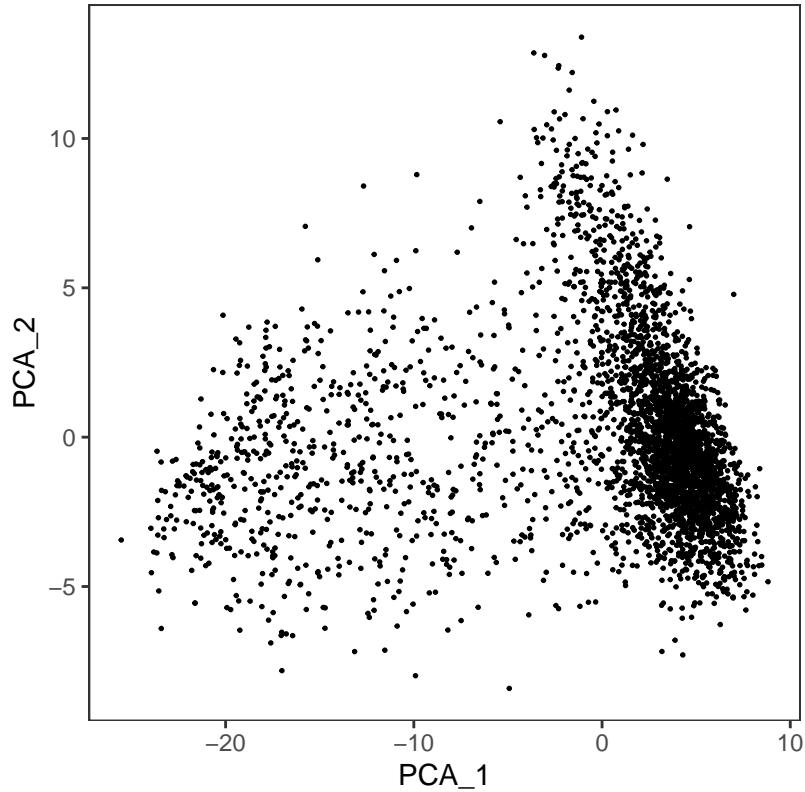
# update column names for easier plotting
colnames(reducedDim(spe, "UMAP")) <- paste0("UMAP", 1:2)
```

9.5 Visualizations

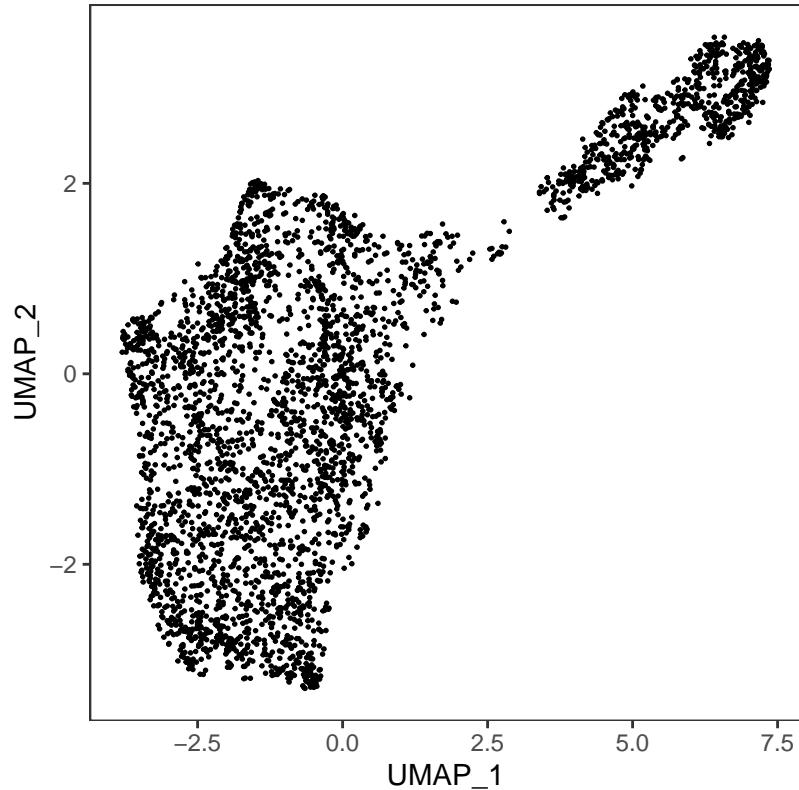
ggspavis

```
library(ggspavis)

# plot top 2 PCA dimensions
plotDimRed(spe, plot_type = "PCA")
```



```
# plot top 2 UMAP dimensions
plotDimRed(spe, plot_type = "UMAP")
```



References

10 Clustering

10.1 Overview

10.2 Load data from previous steps

4

```
library(SpatialExperiment)
spe <- readRDS("spe_reduceddims.rds")
spe_full <- readRDS("spe_logcounts.rds")
```

10.3 Non-spatial clustering

10.3.1 Clustering using HVGs

```
library(scran)

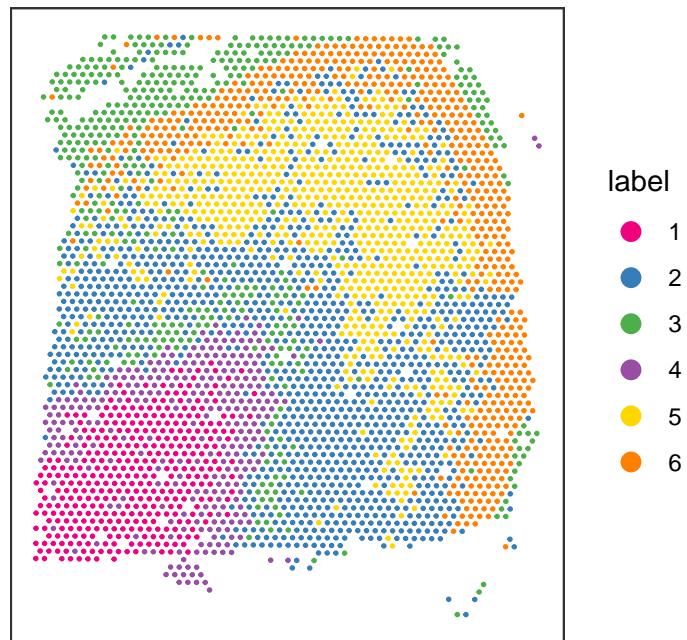
# graph-based clustering
set.seed(123)
k <- 10
g <- buildSNNGraph(spe, k = k, use.dimred = "PCA")
g_walk <- igraph::cluster_walktrap(g)
clus <- g_walk$membership
table(clus)
##   clus
##      1    2    3    4    5    6
##  359 1187  447  291  693  547

# store cluster labels in column 'label' in colData
colLabels(spe) <- factor(clus)
```

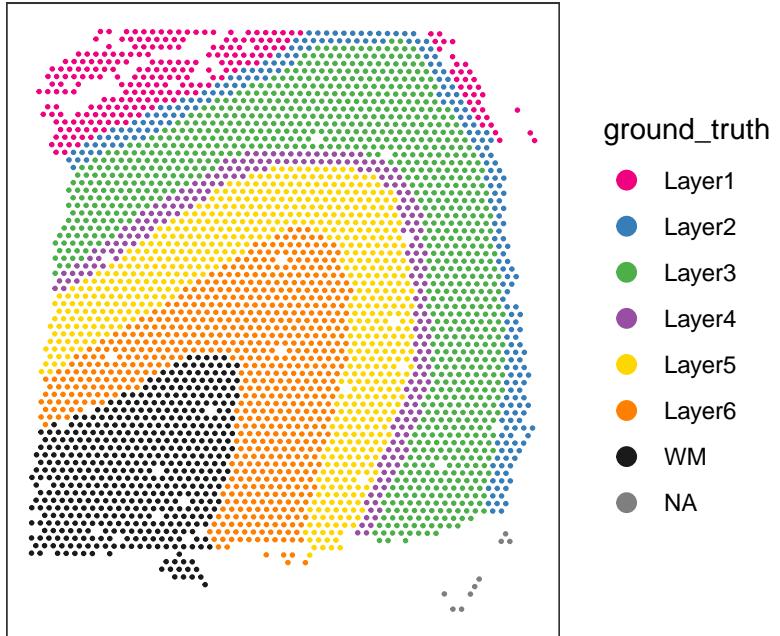
ggspavis

```
library(ggspavis)

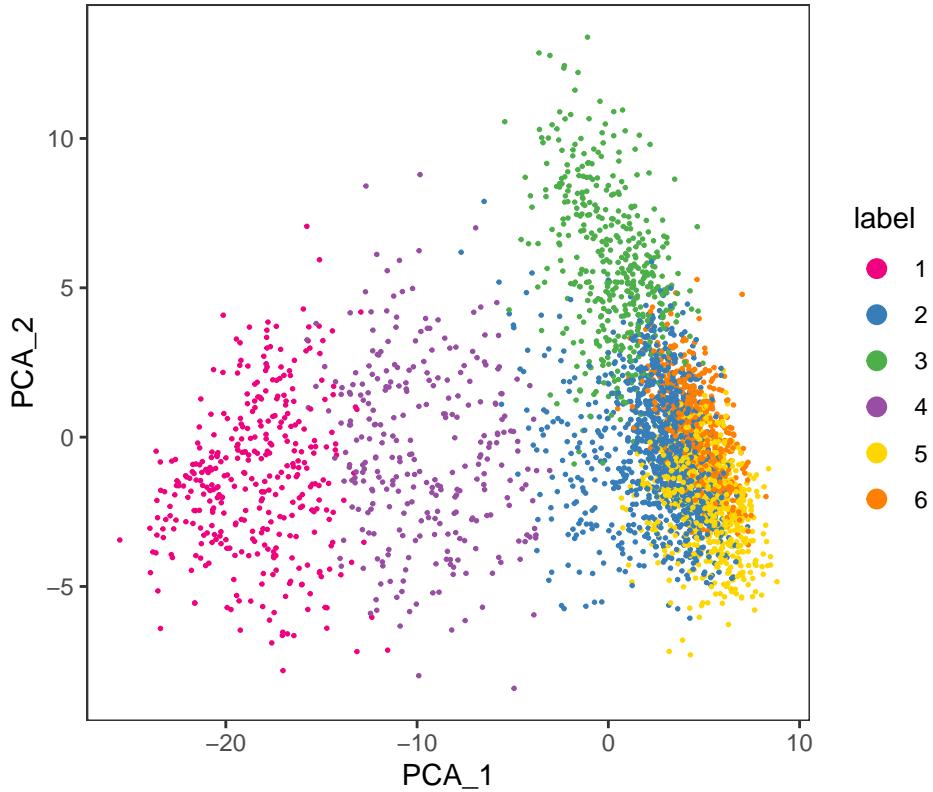
# plot clusters in spatial x-y coordinates
plotSpots(spe, annotate = "label",
           pal = "libd_layer_colors")
```



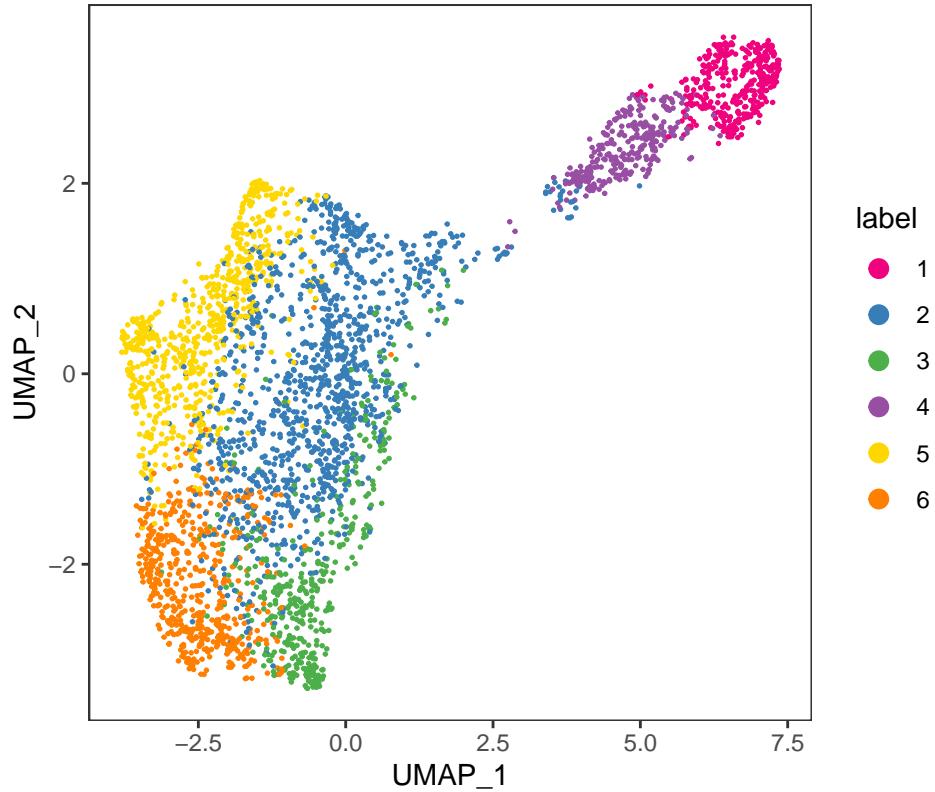
```
# plot ground truth labels in spatial coordinates
plotSpots(spe, annotate = "ground_truth",
           pal = "libd_layer_colors")
```



```
# plot clusters in PCA reduced dimensions
plotDimRed(spe, plot_type = "PCA",
            annotate = "label", pal = "libd_layer_colors")
```



```
# plot clusters in UMAP reduced dimensions
plotDimRed(spe, plot_type = "UMAP",
            annotate = "label", pal = "libd_layer_colors")
```



10.4 Spatially-aware clustering

10.4.1 Clustering using SVGs

8

nnSVG

8

```
library(nnSVG)

# subsample spots for faster runtime in this example
# note: skip this step in full analysis
n <- 100
set.seed(123)
ix <- sample(seq_len(n), n)
spe_nnSVG <- spe_full[, ix]  ## note: using full object from logcounts step

# filter low-expressed and mitochondrial genes
# using stringent filtering for faster runtime in this example
# note: use default filtering in full analysis
spe_nnSVG <- filter_genes(
  spe_nnSVG, filter_genes_ncounts = 10, filter_genes_pcspots = 3
)
## Gene filtering: removing mitochondrial genes
## removed 13 mitochondrial genes
## Gene filtering: retaining genes with at least 10 counts in at least 3% (n = 3) of spatial locations
## removed 33353 out of 33525 genes due to low expression

# re-calculate logcounts after filtering
spe_nnSVG <- logNormCounts(spe_nnSVG)

# run nnSVG
set.seed(123)
spe_nnSVG <- nnSVG(spe_nnSVG)

# select top SVGs
# note: using small subset in this example
# use larger set (e.g. top 1000 genes) in full analysis
n_top <- 50
```

```

ix_top <- order(rowData(spe_nnSVG)$rank) [1:n]
top_svgs <- rowData(spe_nnSVG)[ix_top, "gene_id"]

library(scater)
library(scran)

# dimensionality reduction
# compute PCA
# note: using small number of components in this example
# use larger number (e.g. ncomponents = 50) in full analysis
set.seed(123)
spe_nnSVG <- runPCA(spe_nnSVG, ncomponents = 10, subset_row = top_svgs)

# graph-based clustering
set.seed(123)
k <- 10
g <- buildSNNGraph(spe_nnSVG, k = k, use.dimred = "PCA")
g_walk <- igraph::cluster_walktrap(g)
clus <- g_walk$membership
table(clus)
##  clus
##  1  2  3  4  5
##  6  6 53 24 11

# store cluster labels in column 'label' in colData
colLabels(spe_nnSVG) <- factor(clus)

```

10.4.2 Clustering using concatenated features

10.4.3 Spatially-aware clustering algorithms

- Bioconductor
- version 3.9 onwards
- GitHub
- CRAN

References

11 Spot deconvolution

11.1 Overview

6

10

11.2 Previous steps

11.3 Load data from previous steps

4

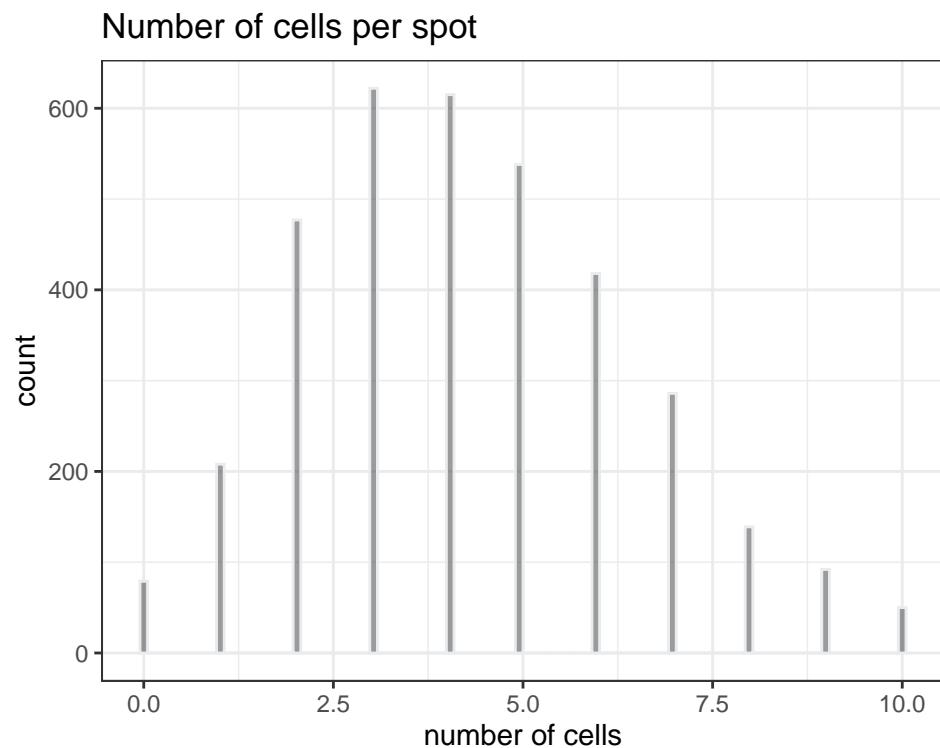
```
library(SpatialExperiment)
spe <- readRDS("spe_cluster.rds")
```

11.4 Number of cells per spot

ggspavis

```
library(ggspavis)

# plot number of cells per spot
plotSpotQC(spe, plot_type = "histogram", x_metric = "cell_count") +
  xlab("number of cells") +
  ggtitle("Number of cells per spot")
```



References

12 Spatial registration

12.1 Overview

11

10

13 Differential expression

13.1 Overview

13.2 Load data from previous steps

4

```
library(SpatialExperiment)
spe <- readRDS("spe_cluster.rds")
```

13.3 Differential expression testing

```
library(scran)
library(scater)
library(pheatmap)
```

```
# set gene names as row names for easier plotting
rownames(spe) <- rowData(spe)$gene_name

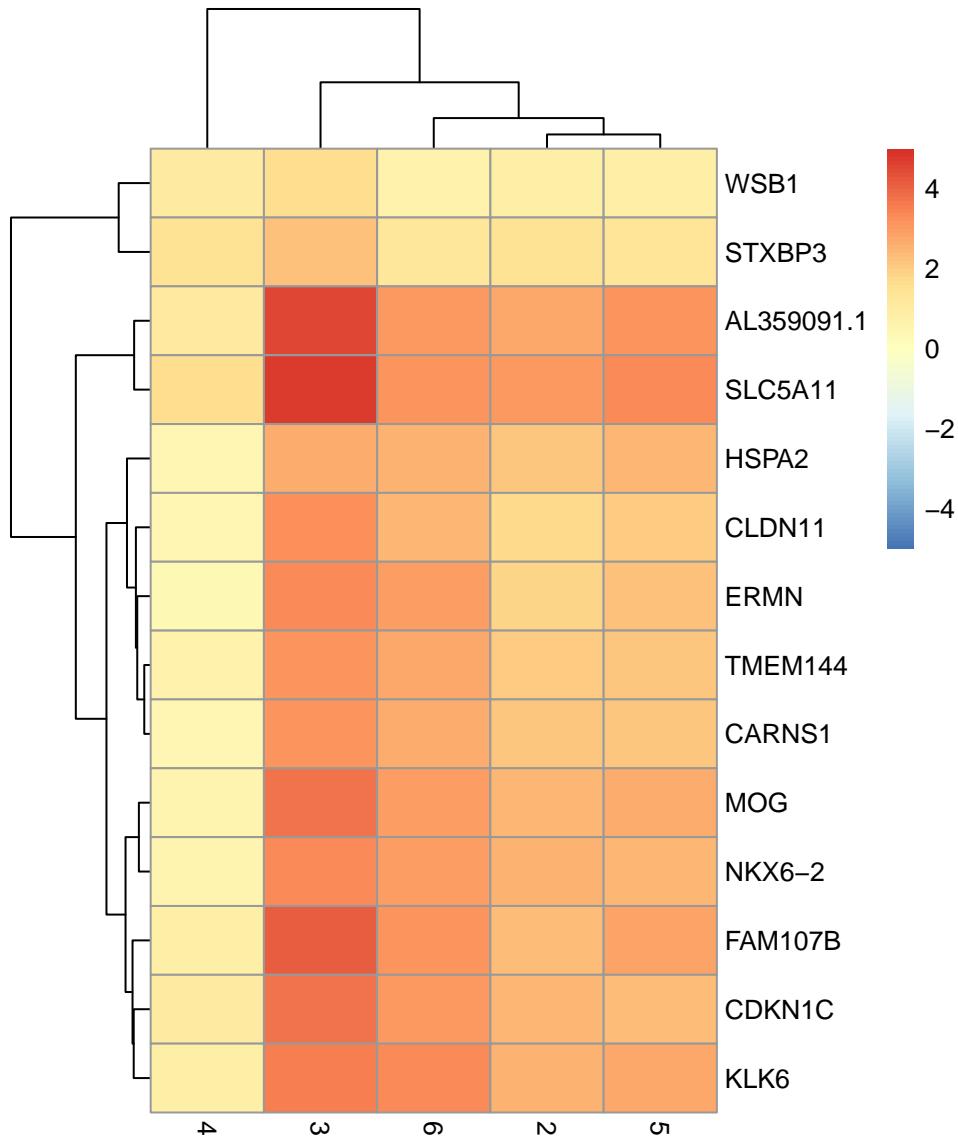
# test for marker genes
```

```
markers <- findMarkers(spe, test = "binom", direction = "up")

# returns a list with one DataFrame per cluster
markers
## List of length 6
## names(6): 1 2 3 4 5 6
```

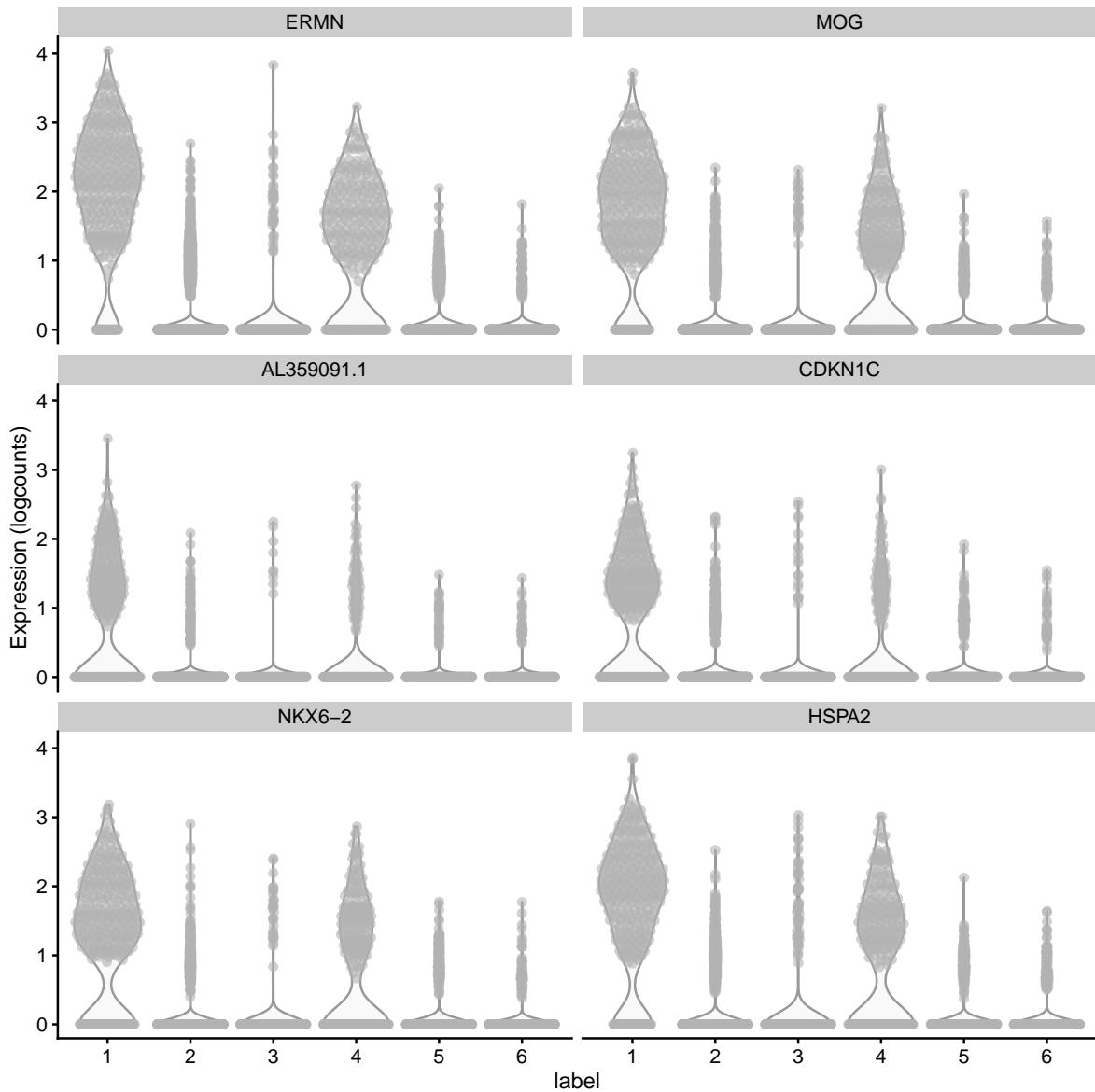
```
# plot log-fold changes for one cluster over all other clusters
# selecting cluster 1
interesting <- markers[[1]]
best_set <- interesting[interesting$Top <= 5, ]
logFCs <- getMarkerEffects(best_set)

pheatmap(logFCs, breaks = seq(-5, 5, length.out = 101))
```



```
# plot log-transformed normalized expression of top genes for one cluster
top_genes <- head(rownames(interesting))

plotExpression(spe, x = "label", features = top_genes)
```



13.4 Pseudobulking

References

14 Multiple samples

14.1 Overview

15 Spatial co-localization

15.1 Overview

15.2 Load data from previous steps

4

15.3 Spatial co-localization of cell types within a single sample

spicyR

References

Part III

Workflows

16 Workflows

4

17 Human DLPFC workflow

17.1 Overview

17.2 Description of dataset

[spatialLIBD](#)
[spatialLIBD Shiny web app](#)

17.3 Load data

[STexampleData](#)

```
library(SpatialExperiment)
library(STexampleData)

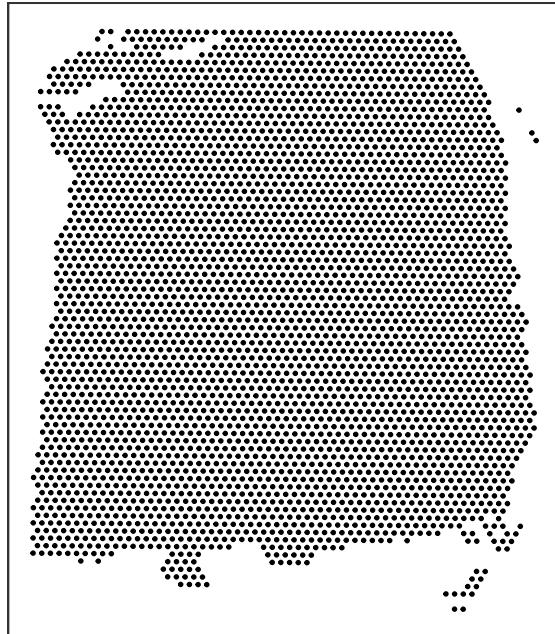
# load object
spe <- Visium_humanDLPFC()
spe
##   class: SpatialExperiment
##   dim: 33538 4992
##   metadata(0):
##   assays(1): counts
##   rownames(33538): ENSG00000243485 ENSG00000237613 ... ENSG00000277475
##     ENSG00000268674
##   rowData names(3): gene_id gene_name feature_type
##   colnames(4992): AAACAACGAATAGTTC-1 AAACAAGTATCTCCA-1 ...
##     TTGTTTGTATTACACG-1 TTGTTTGTGTAATTAC-1
##   colData names(8): barcode_id sample_id ... reference cell_count
##   reducedDimNames(0):
##   mainExpName: NULL
##   altExpNames(0):
##   spatialCoords names(2) : pxl_col_in_fullres pxl_row_in_fullres
##   imgData names(4): sample_id image_id data scaleFactor
```

17.4 Plot data

ggspavis

```
library(ggspavis)
```

```
# plot spatial coordinates (spots)
plotSpots(spe)
```



17.5 Quality control (QC)

```
# subset to keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]
dim(spe)
## [1] 33538 3639
```

6

```
library(scater)

# identify mitochondrial genes
is_mito <- grep("(^MT-)|(mt-)", rowData(spe)$gene_name)
table(is_mito)
## is_mito
## FALSE TRUE
```

```

## 33525    13
rowData(spe)$gene_name[is_mito]
## [1] "MT-ND1"   "MT-ND2"   "MT-CO1"   "MT-CO2"   "MT-ATP8"   "MT-ATP6"   "MT-CO3"
## [8] "MT-ND3"   "MT-ND4L"  "MT-ND4"   "MT-ND5"   "MT-ND6"   "MT-CYB"

# calculate per-spot QC metrics and store in colData
spe <- addPerCellQC(spe, subsets = list(mito = is_mito))
head(colData(spe), 3)
## DataFrame with 3 rows and 14 columns
##           barcode_id      sample_id in_tissue array_row
##           <character>     <character> <integer> <integer>
## AAACAAGTATCTCCA-1 AAACAAGTATCTCCA-1 sample_151673      1      50
## AAACAATCTACTAGCA-1 AAACAATCTACTAGCA-1 sample_151673      1      3
## AACACCCAATAACTGC-1 AACACCCAATAACTGC-1 sample_151673      1      59
##           array_col ground_truth reference cell_count      sum
##           <integer> <character> <character> <integer> <numeric>
## AAACAAGTATCTCCA-1      102      Layer3      Layer3       6     8458
## AAACAATCTACTAGCA-1      43      Layer1      Layer1      16    1667
## AACACCCAATAACTGC-1      19          WM          WM       5    3769
##           detected subsets_mito_sum subsets_mito_detected
##           <numeric> <numeric> <numeric>
## AAACAAGTATCTCCA-1      3586      1407        13
## AAACAATCTACTAGCA-1      1150      204         11
## AACACCCAATAACTGC-1      1960      430         13
##           subsets_mito_percent      total
##           <numeric> <numeric>
## AAACAAGTATCTCCA-1      16.6351    8458
## AAACAATCTACTAGCA-1      12.2376    1667
## AACACCCAATAACTGC-1      11.4089    3769

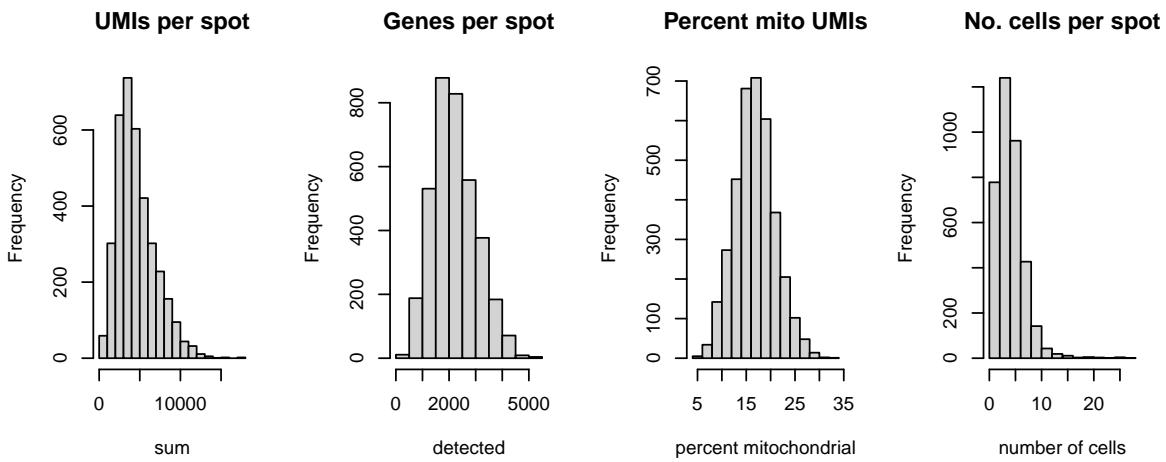
```

6

```

# histograms of QC metrics
par(mfrow = c(1, 4))
hist(colData(spe)$sum, xlab = "sum", main = "UMIs per spot")
hist(colData(spe)$detected, xlab = "detected", main = "Genes per spot")
hist(colData(spe)$subsets_mito_percent, xlab = "percent mitochondrial", main = "Percent mito")
hist(colData(spe)$cell_count, xlab = "number of cells", main = "No. cells per spot")

```



```

par(mfrow = c(1, 1))

# select QC thresholds
qc_lib_size <- colData(spe)$sum < 600
qc_detected <- colData(spe)$detected < 400
qc_mito <- colData(spe)$subsets_mito_percent > 28
qc_cell_count <- colData(spe)$cell_count > 10

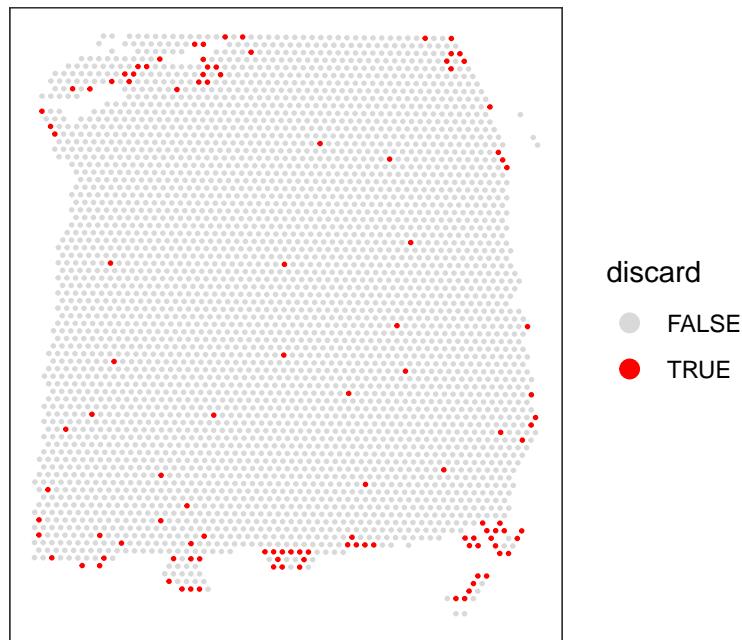
# number of discarded spots for each metric
apply(cbind(qc_lib_size, qc_detected, qc_mito, qc_cell_count), 2, sum)
##      qc_lib_size      qc_detected      qc_mito      qc_cell_count
##                 8                  7                 17                  90

# combined set of discarded spots
discard <- qc_lib_size | qc_detected | qc_mito | qc_cell_count
table(discard)
##   discard
## FALSE  TRUE
## 3524   115

# store in object
colData(spe)$discard <- discard

```

```
# check spatial pattern of discarded spots
plotSpotQC(spe, plot_type = "spot", annotate = "discard")
```



```
# filter low-quality spots
spe <- spe[, !colData(spe)$discard]
dim(spe)
## [1] 33538 3524
```

17.6 Normalization

```

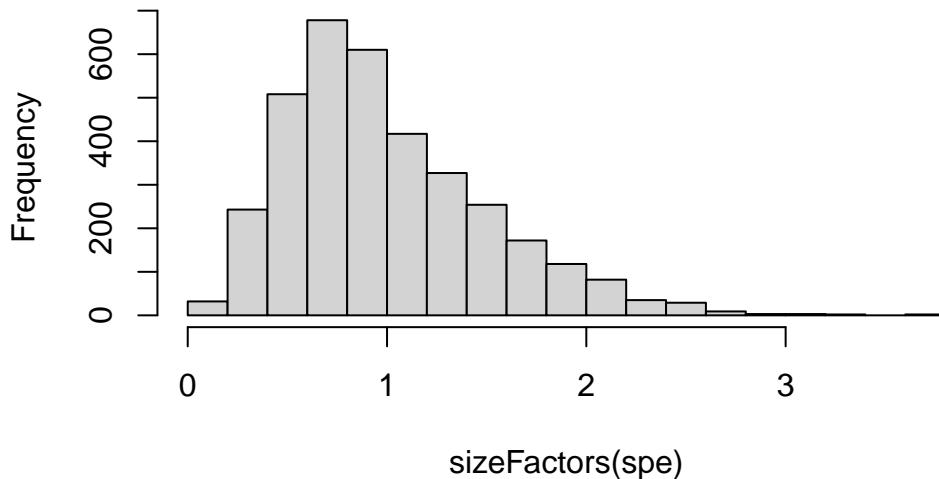
library(scran)

# calculate library size factors
spe <- computeLibraryFactors(spe)

summary(sizeFactors(spe))
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
## 0.1321  0.6312  0.9000  1.0000  1.2849  3.7582
hist(sizeFactors(spe), breaks = 20)

```

Histogram of sizeFactors(spe)



```

# calculate logcounts and store in object
spe <- logNormCounts(spe)

assayNames(spe)
## [1] "counts"      "logcounts"

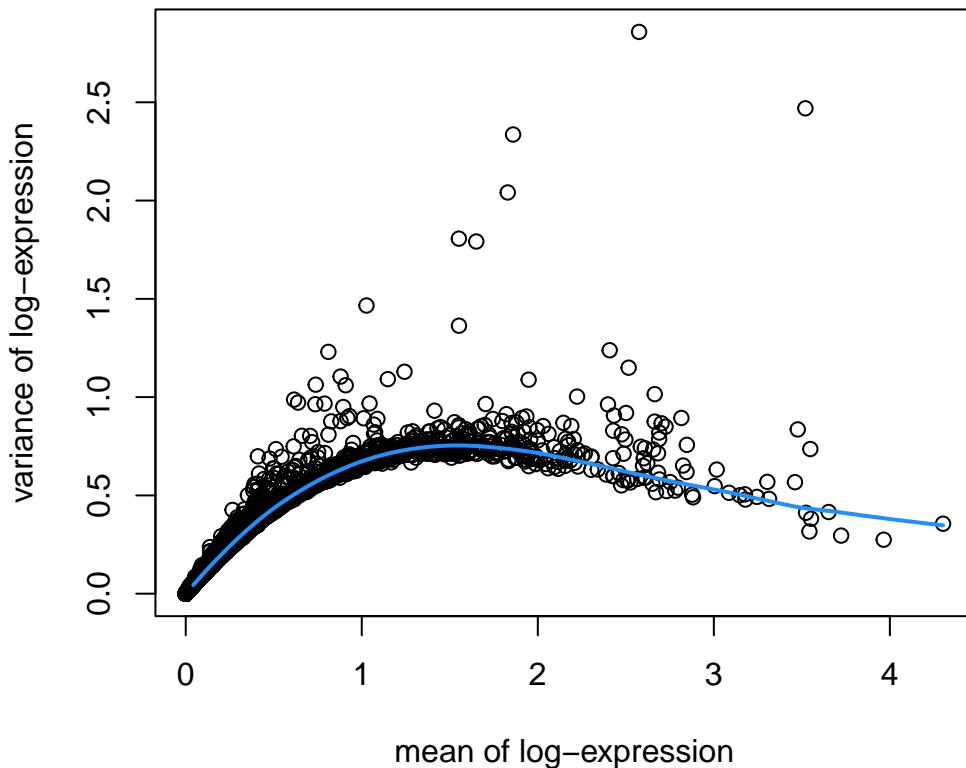
```

17.7 Feature selection

```
# remove mitochondrial genes
spe <- spe[!is_mito, ]
dim(spe)
## [1] 33525 3524

# fit mean-variance relationship
dec <- modelGeneVar(spe)

# visualize mean-variance relationship
fit <- metadata(dec)
plot(fit$mean, fit$var,
      xlab = "mean of log-expression", ylab = "variance of log-expression")
curve(fit$trend(x), col = "dodgerblue", add = TRUE, lwd = 2)
```



```
# select top HVGs
top_hvgs <- getTopHVGs(dec, prop = 0.1)
length(top_hvgs)
## [1] 1438
```

17.8 Spatially-aware feature selection

[nnSVG](#)

```

library(nnSVG)

# subsample spots
n <- 100
set.seed(123)
ix <- sample(seq_len(n), n)

spe_nnSVG <- spe[, ix]

# filter low-expressed and mitochondrial genes
# using very stringent filtering parameters for faster runtime in this example
# note: for a full analysis, use alternative filtering parameters (e.g. defaults)
spe_nnSVG <- filter_genes(
  spe_nnSVG, filter_genes_ncounts = 10, filter_genes_pcspots = 3
)
## Gene filtering: removing mitochondrial genes
## removed 0 mitochondrial genes
## Gene filtering: retaining genes with at least 10 counts in at least 3% (n = 3) of spatial
## removed 33353 out of 33525 genes due to low expression

# re-calculate logcounts after filtering
# using library size factors
spe_nnSVG <- logNormCounts(spe_nnSVG)

# run nnSVG
# using a single core for compatibility on build system
# note: for a full analysis, use multiple cores
set.seed(123)
spe_nnSVG <- nnSVG(spe_nnSVG, n_threads = 1)

# investigate results

# show results
head(rowData(spe_nnSVG), 3)
## DataFrame with 3 rows and 18 columns
##           gene_id   gene_name feature_type subsets_mito
##           <character> <character>    <character>    <logical>
## ENSG00000074800 ENSG00000074800      EN01 Gene Expression FALSE
## ENSG00000171603 ENSG00000171603      CLSTN1 Gene Expression FALSE
## ENSG00000162545 ENSG00000162545      CAMK2N1 Gene Expression FALSE

```

```

##          sigma.sq    tau.sq      phi    loglik   runtime     mean
##          <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000074800 0.00840777 0.518011 15.82454 -109.808 0.016 1.75820
## ENSG00000171603 0.43102027 0.298698 18.12310 -123.212 0.021 2.07433
## ENSG00000162545 0.15373440 0.547736 2.38046 -119.576 0.014 2.63576
##          var      spcov prop_sv loglik_lm LR_stat rank
##          <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000074800 0.530929 0.0521523 0.0159716 -109.735 -0.145454 167
## ENSG00000171603 0.741077 0.3164987 0.5906665 -126.409 6.394009 86
## ENSG00000162545 0.702846 0.1487576 0.2191603 -123.760 8.368951 70
##          pval      padj
##          <numeric> <numeric>
## ENSG00000074800 1.0000000 1.0000000
## ENSG00000171603 0.0408845 0.0817690
## ENSG00000162545 0.0152302 0.0374227

# number of significant SVGs
table(rowData(spe_nnSVG)$padj <= 0.05)
##
## FALSE TRUE
## 96 76

# show results for top n SVGs
rowData(spe_nnSVG)[order(rowData(spe_nnSVG)$rank)[1:6], ]
## DataFrame with 6 rows and 18 columns
##          gene_id gene_name feature_type subsets_mito
##          <character> <character> <character> <logical>
## ENSG00000197971 ENSG00000197971 MBP Gene Expression FALSE
## ENSG00000123560 ENSG00000123560 PLP1 Gene Expression FALSE
## ENSG00000109846 ENSG00000109846 CRYAB Gene Expression FALSE
## ENSG00000173786 ENSG00000173786 CNP Gene Expression FALSE
## ENSG00000131095 ENSG00000131095 GFAP Gene Expression FALSE
## ENSG00000160307 ENSG00000160307 S100B Gene Expression FALSE
##          sigma.sq    tau.sq      phi    loglik   runtime     mean
##          <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000197971 3.92430 0.175336 0.93014 -118.372 0.020 3.78073
## ENSG00000123560 3.23544 0.461590 1.03983 -140.269 0.016 2.86143
## ENSG00000109846 1.89968 0.281795 1.88555 -126.692 0.016 1.86058
## ENSG00000173786 2.28793 0.402524 1.02675 -129.206 0.015 1.79558
## ENSG00000131095 2.23709 0.461297 2.49083 -149.004 0.017 1.94543
## ENSG00000160307 1.24179 0.155737 5.76279 -129.722 0.014 1.82695
##          var      spcov prop_sv loglik_lm LR_stat rank

```

```

## <numeric> <numeric> <numeric> <numeric> <numeric> <numeric>
## ENSG00000197971 2.76535 0.523969 0.957231 -192.250 147.7556 1
## ENSG00000123560 3.02555 0.628613 0.875146 -196.746 112.9549 2
## ENSG00000109846 1.88123 0.740785 0.870824 -172.988 92.5906 3
## ENSG00000173786 1.97274 0.842396 0.850388 -175.363 92.3137 4
## ENSG00000131095 2.71281 0.768823 0.829047 -191.291 84.5739 5
## ENSG00000160307 1.46946 0.609953 0.888562 -160.636 61.8281 6
## pval padj
## <numeric> <numeric>
## ENSG00000197971 0.00000e+00 0.00000e+00
## ENSG00000123560 0.00000e+00 0.00000e+00
## ENSG00000109846 0.00000e+00 0.00000e+00
## ENSG00000173786 0.00000e+00 0.00000e+00
## ENSG00000131095 0.00000e+00 0.00000e+00
## ENSG00000160307 3.75255e-14 1.07573e-12

# identify top-ranked SVG
rowData(spe_nnSVG)$gene_name[which(rowData(spe_nnSVG)$rank == 1)]
## [1] "MBP"

```

17.9 Dimensionality reduction

```

# compute PCA
set.seed(123)
spe <- runPCA(spe, subset_row = top_hvgs)

reducedDimNames(spe)
## [1] "PCA"
dim(reducedDim(spe, "PCA"))
## [1] 3524 50

```

```

# compute UMAP on top 50 PCs
set.seed(123)
spe <- runUMAP(spe, dimred = "PCA")

reducedDimNames(spe)
## [1] "PCA"   "UMAP"
dim(reducedDim(spe, "UMAP"))
## [1] 3524    2

# update column names for easier plotting
colnames(reducedDim(spe, "UMAP")) <- paste0("UMAP", 1:2)

```

17.10 Clustering

10

```

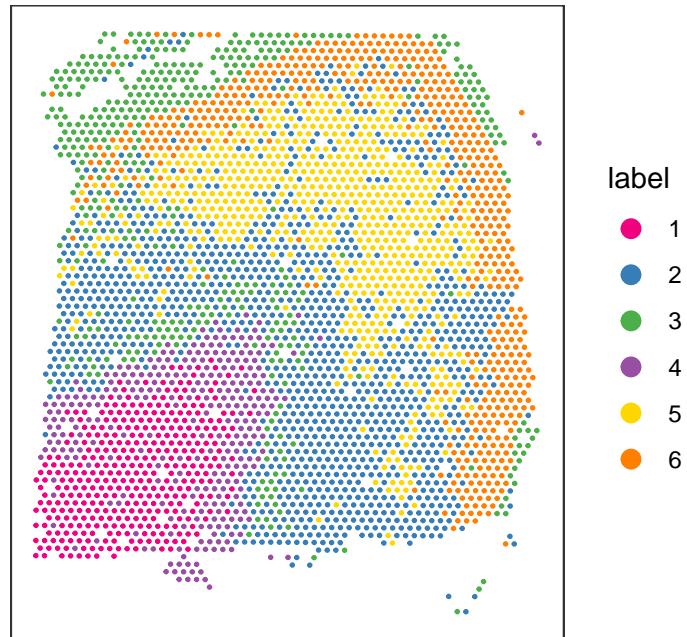
# graph-based clustering
set.seed(123)
k <- 10
g <- buildSNNGraph(spe, k = k, use.dimred = "PCA")
g_walk <- igraph::cluster_walktrap(g)
clus <- g_walk$membership
table(clus)
##  clus
##    1    2    3    4    5    6
##  359 1187 447 291 693 547

# store cluster labels in column 'label' in colData
colLabels(spe) <- factor(clus)

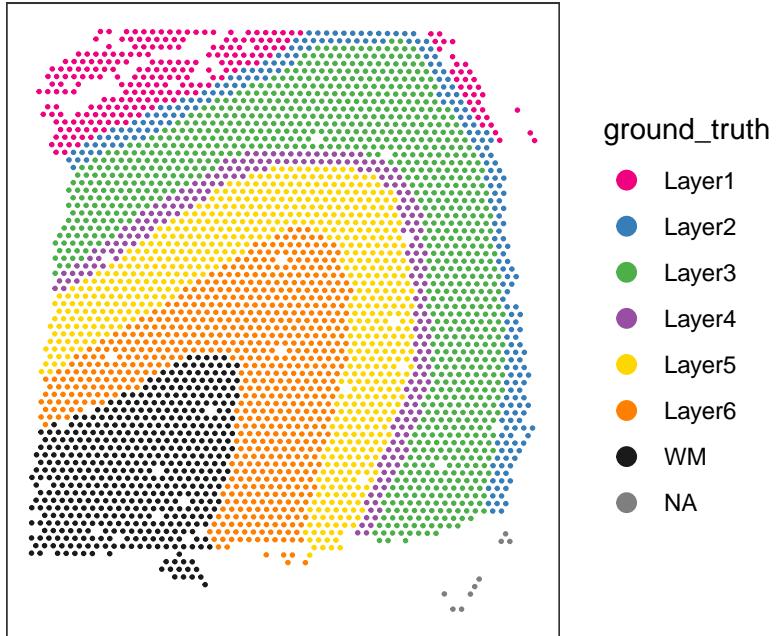
```

103

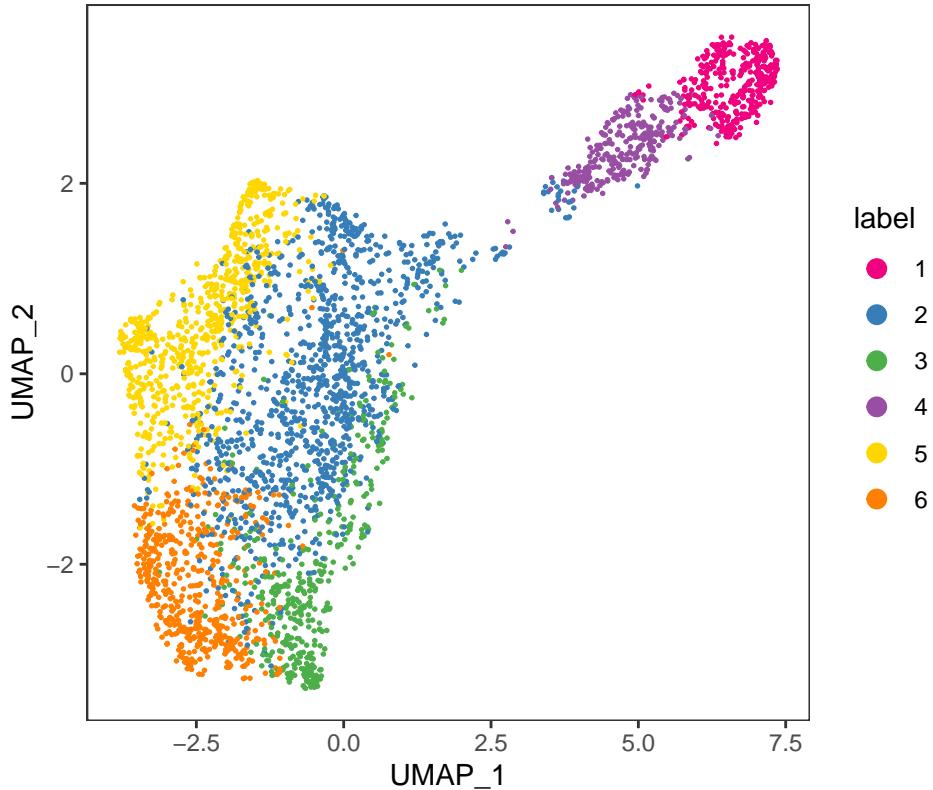
```
# plot clusters in spatial x-y coordinates
plotSpots(spe, annotate = "label",
           pal = "libd_layer_colors")
```



```
# plot ground truth labels in spatial coordinates
plotSpots(spe, annotate = "ground_truth",
           pal = "libd_layer_colors")
```



```
# plot clusters in UMAP reduced dimensions
plotDimRed(spe, plot_type = "UMAP",
            annotate = "label", pal = "libd_layer_colors")
```



17.11 Differential expression

```
# set gene names as row names for easier plotting
rownames(spe) <- rowData(spe)$gene_name

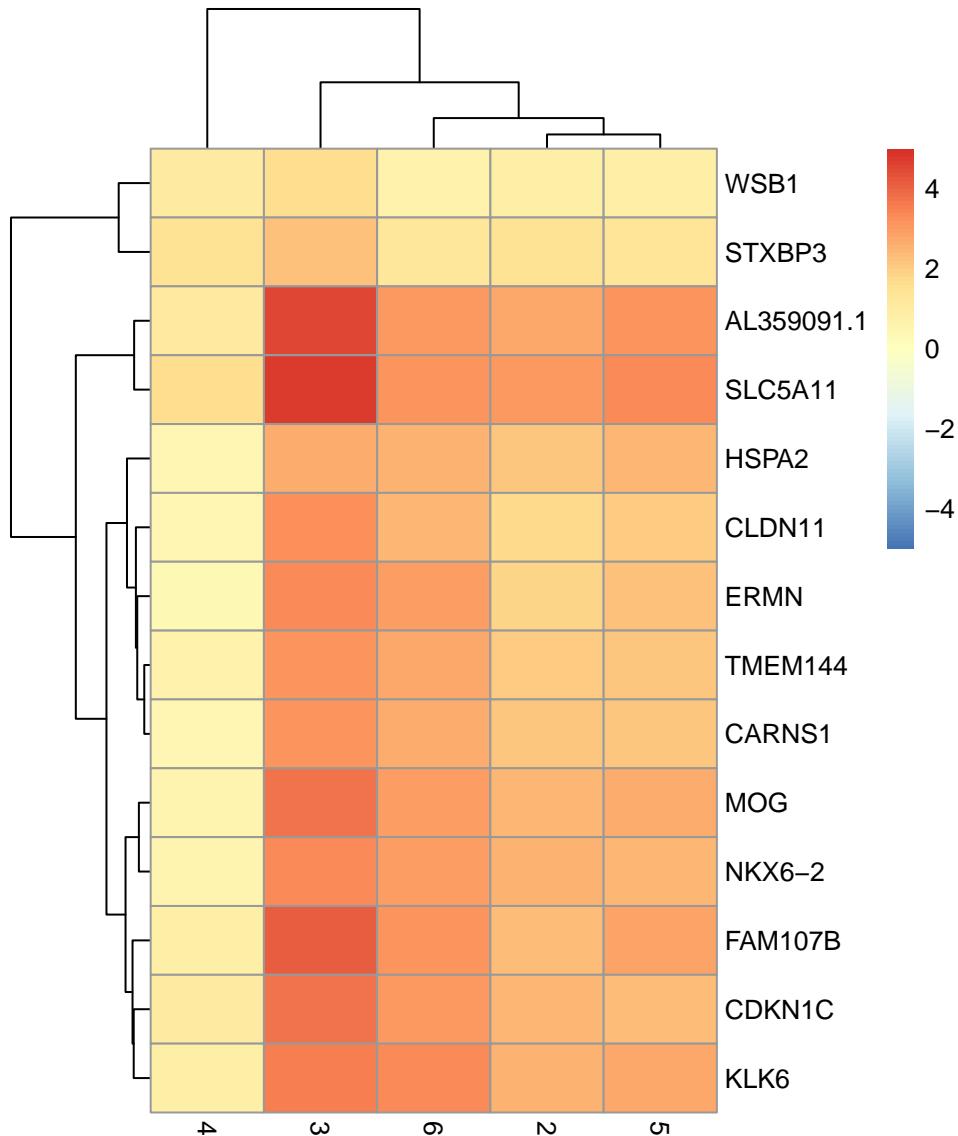
# test for marker genes
markers <- findMarkers(spe, test = "binom", direction = "up")

# returns a list with one DataFrame per cluster
markers
## List of length 6
## names(6): 1 2 3 4 5 6
```

```
library(pheatmap)

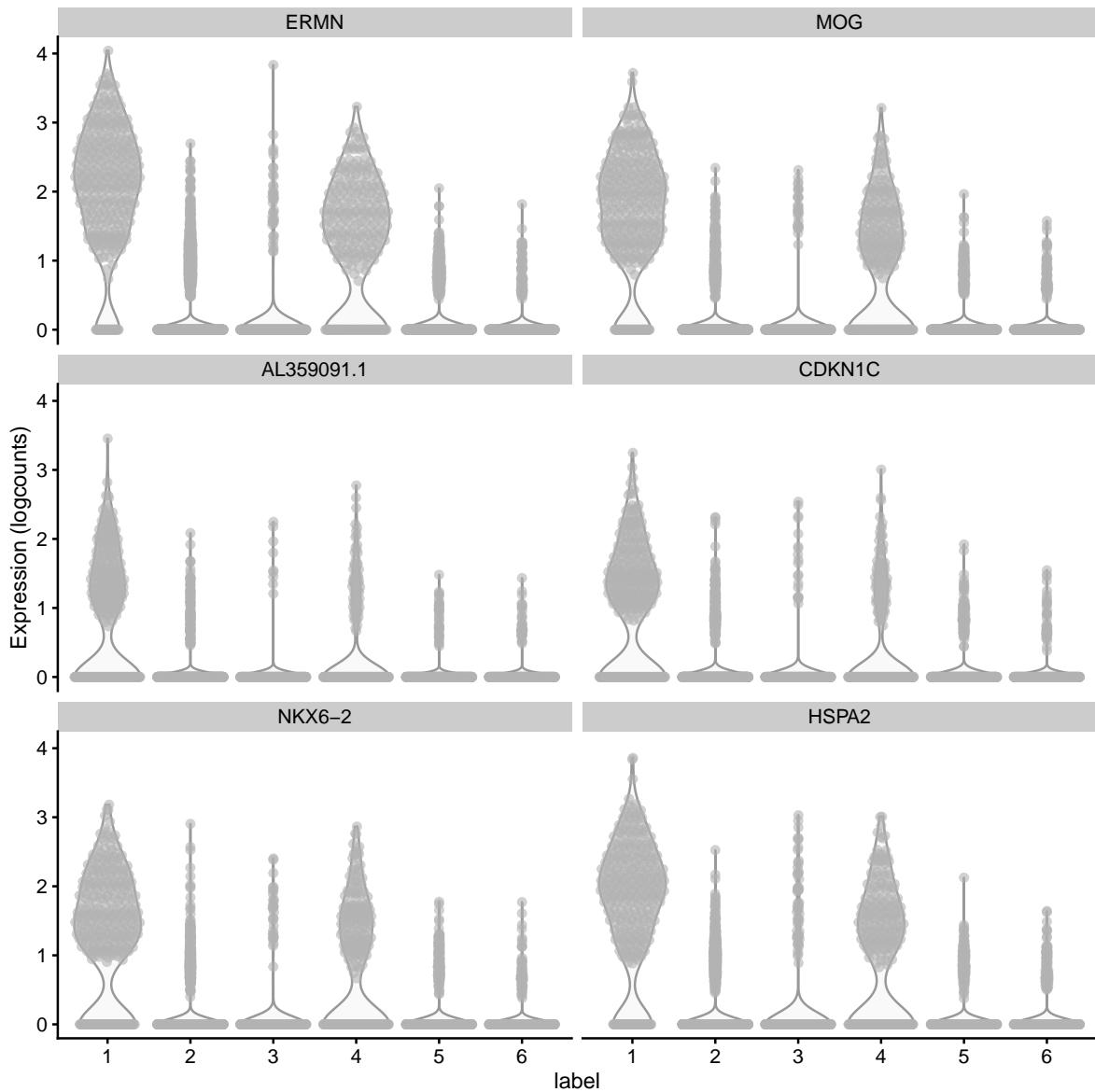
# plot log-fold changes for one cluster over all other clusters
# selecting cluster 1
interesting <- markers[[1]]
best_set <- interesting[interesting$Top <= 5, ]
logFCs <- getMarkerEffects(best_set)

pheatmap(logFCs, breaks = seq(-5, 5, length.out = 101))
```



```
# plot log-transformed normalized expression of top genes for one cluster
top_genes <- head(rownames(interesting))

plotExpression(spe, x = "label", features = top_genes)
```



References

18 Mouse coronal workflow

18.1 Overview

[10x Genomics website](#)

18.2 Description of dataset

[10x Genomics website](#)

18.3 Load data

[STexampleData](#)

```
library(SpatialExperiment)
library(STexampleData)

# load object
spe <- Visium_mouseCoronal()
spe
##  class: SpatialExperiment
##  dim: 32285 4992
##  metadata(0):
##  assays(1): counts
```

```
##  rownames(32285): ENSMUSG00000051951 ENSMUSG00000089699 ...
##    ENSMUSG00000095019 ENSMUSG00000095041
##  rowData names(3): gene_id gene_name feature_type
##  colnames(4992): AAACAACGAATAGTTC-1 AAACAAGTATCTCCCA-1 ...
##    TTGTTTGATTACACG-1 TTGTTTGTTAAATTC-1
##  colData names(5): barcode_id sample_id in_tissue array_row array_col
##  reducedDimNames(0):
##  mainExpName: NULL
##  altExpNames(0):
##  spatialCoords names(2) : pxl_col_in_fullres pxl_row_in_fullres
##  imgData names(4): sample_id image_id data scaleFactor
```

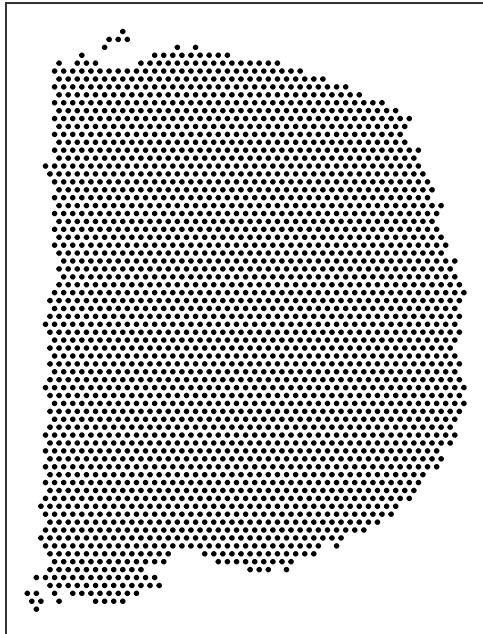
18.4 Plot data

10x Genomics website

ggspavis

```
library(ggspavis)
```

```
# plot spatial coordinates (spots)
plotSpots(spe)
```



18.5 Quality control (QC)

```
# subset to keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]
dim(spe)
## [1] 32285 2702
```

```
library(scater)

# identify mitochondrial genes
is_mito <- grep("(^MT-)|(mt-)", rowData(spe)$gene_name)
table(is_mito)
## is_mito
## FALSE TRUE
## 32272 13
rowData(spe)$gene_name[is_mito]
```

```

## [1] "mt-Nd1"   "mt-Nd2"   "mt-Co1"   "mt-Co2"   "mt-Atp8"  "mt-Atp6"  "mt-Co3"
## [8] "mt-Nd3"   "mt-Nd4l"  "mt-Nd4"   "mt-Nd5"   "mt-Nd6"   "mt-Cytb"

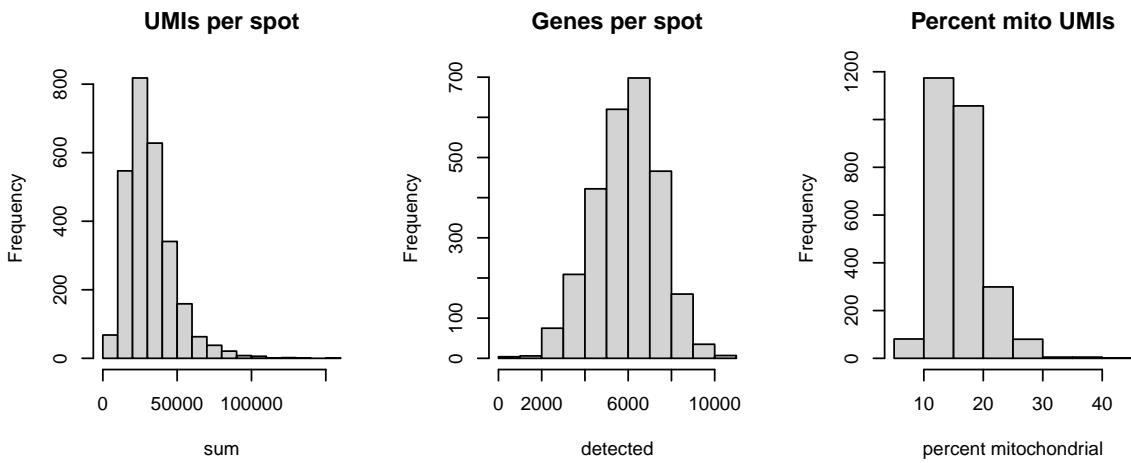
# calculate per-spot QC metrics and store in colData
spe <- addPerCellQC(spe, subsets = list(mito = is_mito))
head(colData(spe), 3)
## DataFrame with 3 rows and 11 columns
##           barcode_id sample_id in_tissue array_row
## <character> <character> <integer> <integer>
## AAACAAGTATCTCCA-1 AAACAAGTATCTCCA-1 sample01      1      50
## AAACAATCTACTAGCA-1 AAACAATCTACTAGCA-1 sample01      1      3
## AACACCCAATAACTGC-1 AACACCCAATAACTGC-1 sample01      1      59
##           array_col     sum detected subsets_mito_sum
## <integer> <numeric> <numeric> <numeric>
## AAACAAGTATCTCCA-1     102    20935      5230        4036
## AAACAATCTACTAGCA-1      43    14789      3646        3419
## AACACCCAATAACTGC-1      19    34646      6272        5068
##           subsets_mito_detected subsets_mito_percent total
## <numeric> <numeric> <numeric>
## AAACAAGTATCTCCA-1          13        19.2787    20935
## AAACAATCTACTAGCA-1          13        23.1185    14789
## AACACCCAATAACTGC-1          13        14.6280    34646

```

```

# histograms of QC metrics
par(mfrow = c(1, 3))
hist(colData(spe)$sum, xlab = "sum", main = "UMIs per spot")
hist(colData(spe)$detected, xlab = "detected", main = "Genes per spot")
hist(colData(spe)$subsets_mito_percent, xlab = "percent mitochondrial", main = "Percent mito"

```



```

par(mfrow = c(1, 1))

# select QC thresholds
qc_lib_size <- colData(spe)$sum < 5000
qc_detected <- colData(spe)$detected < 1000
qc_mito <- colData(spe)$subsets_mito_percent > 30

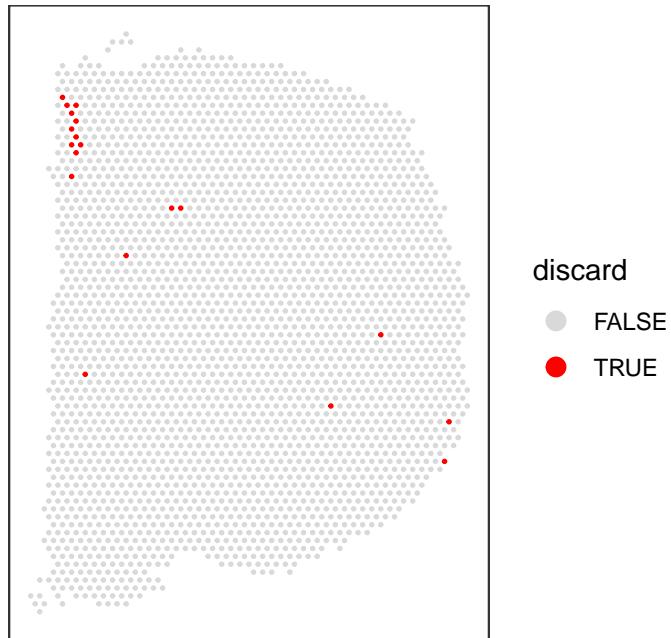
# number of discarded spots for each QC metric
apply(cbind(qc_lib_size, qc_detected, qc_mito), 2, sum)
## qc_lib_size qc_detected qc_mito
##         9           4          11

# combined set of discarded spots
discard <- qc_lib_size | qc_detected | qc_mito
table(discard)
##   discard
## FALSE  TRUE
## 2683    19

# store in object
colData(spe)$discard <- discard

```

```
# check spatial pattern of discarded spots
plotSpotQC(spe, plot_type = "spot", annotate = "discard")
```



```
# filter low-quality spots
spe <- spe[, !colData(spe)$discard]
dim(spe)
## [1] 32285 2683
```

18.6 Normalization

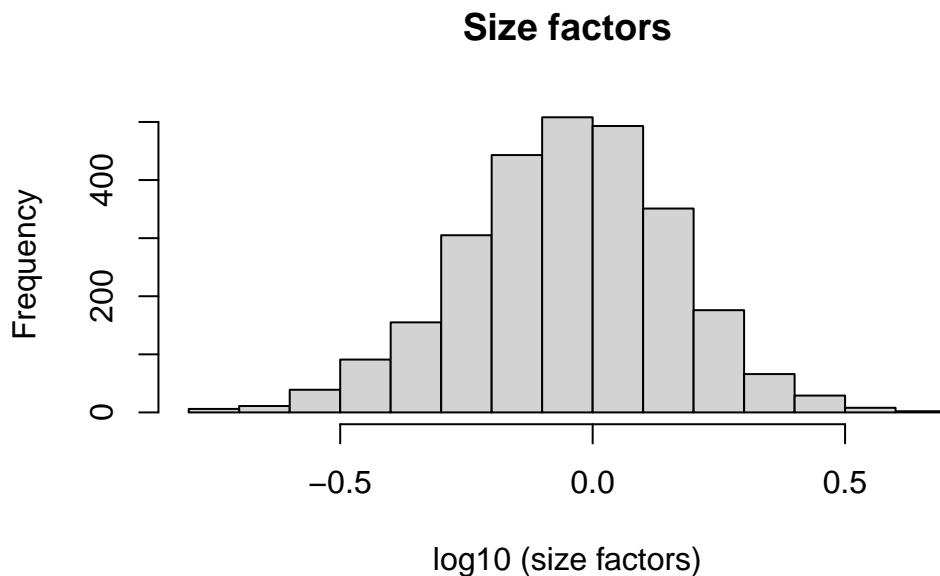
```

library(scran)

# calculate library size factors
spe <- computeLibraryFactors(spe)

summary(sizeFactors(spe))
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
## 0.1615  0.6597  0.9083  1.0000  1.2342  4.8973
hist(log10(sizeFactors(spe)), xlab = "log10 (size factors)", main = "Size factors")

```



```

# calculate logcounts and store in object
spe <- logNormCounts(spe)

assayNames(spe)
## [1] "counts"      "logcounts"

```

18.7 Feature selection

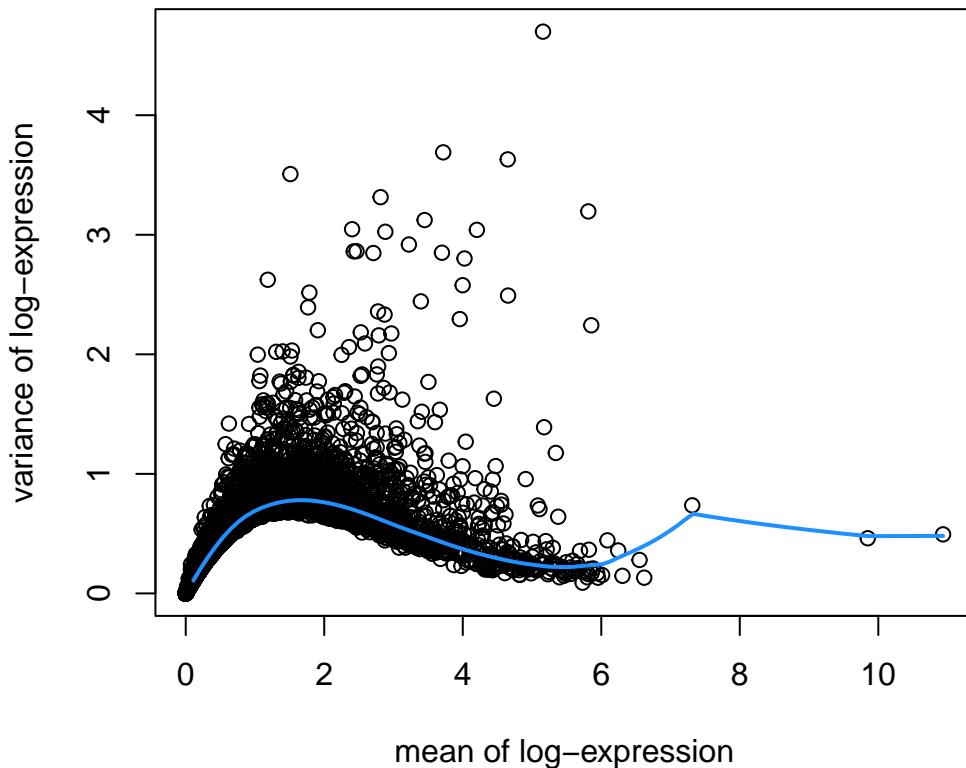
```

# remove mitochondrial genes
spe <- spe[!is_mito, ]
dim(spe)
## [1] 32272 2683

# fit mean-variance relationship
dec <- modelGeneVar(spe)

# visualize mean-variance relationship
fit <- metadata(dec)
plot(fit$mean, fit$var,
      xlab = "mean of log-expression", ylab = "variance of log-expression")
curve(fit$trend(x), col = "dodgerblue", add = TRUE, lwd = 2)

```



```

# select top HVGs
top_hvgs <- getTopHVGs(dec, prop = 0.1)
length(top_hvgs)
## [1] 1274

# identify outlier genes
rev(sort(fit$mean))[1:3]
## ENSMUSG00000115783 ENSMUSG00000098178 ENSMUSG00000024661
##          10.934946          9.847532          7.313407
outlier_ids <- names(rev(sort(fit$mean))[1:3])

rowData(spe)[outlier_ids, ]
## DataFrame with 3 rows and 4 columns
##           gene_id   gene_name feature_type
## <character> <character>    <character>
## ENSMUSG00000115783 ENSMUSG00000115783      Bc1 Gene Expression
## ENSMUSG00000098178 ENSMUSG00000098178      Gm42418 Gene Expression
## ENSMUSG00000024661 ENSMUSG00000024661      Fth1 Gene Expression
##           subsets_mito
## <logical>
## ENSMUSG00000115783      FALSE
## ENSMUSG00000098178      FALSE
## ENSMUSG00000024661      FALSE

```

18.8 Spatially-aware feature selection

[nnSVG](#)

```

library(nnSVG)

# subsample spots
n <- 100
set.seed(123)
ix <- sample(seq_len(n), n)

spe_nnSVG <- spe[, ix]

# filter low-expressed and mitochondrial genes
# using very stringent filtering parameters for faster runtime in this example
# note: for a full analysis, use alternative filtering parameters (e.g. defaults)
spe_nnSVG <- filter_genes(
  spe_nnSVG, filter_genes_ncounts = 50, filter_genes_pcspots = 5
)
## Gene filtering: removing mitochondrial genes
## removed 0 mitochondrial genes
## Gene filtering: retaining genes with at least 50 counts in at least 5% (n = 5) of spatial locations
## removed 32074 out of 32272 genes due to low expression

# re-calculate logcounts after filtering
# using library size factors
spe_nnSVG <- logNormCounts(spe_nnSVG)

# run nnSVG
# using a single core for compatibility on build system
# note: for a full analysis, use multiple cores
set.seed(123)
spe_nnSVG <- nnSVG(spe_nnSVG, n_threads = 1)

# investigate results

# show results
head(rowData(spe_nnSVG), 3)
## DataFrame with 3 rows and 18 columns
##           gene_id   gene_name feature_type
## <character> <character>    <character>
## ENSMUSG00000061518 ENSMUSG00000061518      Cox5b Gene Expression
## ENSMUSG00000073702 ENSMUSG00000073702      Rpl31 Gene Expression
## ENSMUSG00000046330 ENSMUSG00000046330      Rpl37a Gene Expression
## subsets_mito sigma_sq tau_sq phi loglik
## <logical> <numeric> <numeric> <numeric> <numeric>

```

```

##  ENSMUSG00000061518      FALSE 0.0210665 0.1202984 13.53194 -43.6472
##  ENSMUSG00000073702      FALSE 0.0688871 0.0860842 5.84110 -40.3003
##  ENSMUSG00000046330      FALSE 0.0549986 0.1133114 6.47579 -48.2811
##          runtime      mean       var     spcov prop_sv
## <numeric> <numeric> <numeric> <numeric> <numeric>
##  ENSMUSG00000061518      0.026  4.98425 0.142977 0.0291204 0.149022
##  ENSMUSG00000073702      0.012  4.83432 0.143616 0.0542917 0.444515
##  ENSMUSG00000046330      0.015  5.35620 0.166734 0.0437844 0.326770
##          loglik_lm    LR_stat      rank      pval      padj
## <numeric> <numeric> <numeric> <numeric> <numeric>
##  ENSMUSG00000061518 -44.1377  0.981135      184 0.6122789 0.6588653
##  ENSMUSG00000073702 -44.3607  8.120691      147 0.0172431 0.0232254
##  ENSMUSG00000046330 -51.8235  7.084636      151 0.0289462 0.0379559

# number of significant SVGs
table(rowData(spe_nnSVG)$padj <= 0.05)
##
##  FALSE  TRUE
##  46   152

# show results for top n SVGs
rowData(spe_nnSVG)[order(rowData(spe_nnSVG)$rank)[1:6], ]
##  DataFrame with 6 rows and 18 columns
##          gene_id   gene_name feature_type
## <character> <character> <character>
##  ENSMUSG00000046447 ENSMUSG00000046447 Camk2n1 Gene Expression
##  ENSMUSG00000053310 ENSMUSG00000053310 Nrgn Gene Expression
##  ENSMUSG00000018593 ENSMUSG00000018593 Sparc Gene Expression
##  ENSMUSG00000032532 ENSMUSG00000032532 Cck Gene Expression
##  ENSMUSG00000070570 ENSMUSG00000070570 Slc17a7 Gene Expression
##  ENSMUSG00000024617 ENSMUSG00000024617 Camk2a Gene Expression
##          subsets_mito sigma.sq      tau.sq      phi      loglik
## <logical> <numeric> <numeric> <numeric> <numeric>
##  ENSMUSG00000046447      FALSE 1.88342 6.28372e-02 2.05206 -99.021
##  ENSMUSG00000053310      FALSE 3.17506 2.47772e-01 3.26420 -151.225
##  ENSMUSG00000018593      FALSE 2.04163 7.63186e-02 3.68722 -124.793
##  ENSMUSG00000032532      FALSE 2.98145 2.98145e-08 5.07869 -147.464
##  ENSMUSG00000070570      FALSE 2.64081 3.20107e-01 4.15601 -155.620
##  ENSMUSG00000024617      FALSE 1.41249 9.79965e-02 3.74573 -113.269
##          runtime      mean       var     spcov prop_sv
## <numeric> <numeric> <numeric> <numeric> <numeric>
##  ENSMUSG00000046447      0.018  5.35338 1.50820 0.256357 0.967714

```

```

##  ENSMUSG00000053310    0.029  4.85871  3.71912  0.366737  0.927612
##  ENSMUSG00000018593    0.019  3.61331  1.87664  0.395442  0.963966
##  ENSMUSG00000032532    0.016  4.19282  2.72400  0.411820  1.000000
##  ENSMUSG00000070570    0.015  3.62933  3.00604  0.447757  0.891889
##  ENSMUSG00000024617    0.022  4.11652  1.17870  0.288711  0.935123
##          loglik_lm   LR_stat      rank      pval      padj
##          <numeric> <numeric> <numeric> <numeric> <numeric>
##  ENSMUSG00000046447 -161.937 125.8321           1 0.00000e+00 0.00000e+00
##  ENSMUSG00000053310 -207.066 111.6804           2 0.00000e+00 0.00000e+00
##  ENSMUSG00000018593 -172.866 96.1453           3 0.00000e+00 0.00000e+00
##  ENSMUSG00000032532 -191.496 88.0644           4 0.00000e+00 0.00000e+00
##  ENSMUSG00000070570 -196.423 81.6050           5 0.00000e+00 0.00000e+00
##  ENSMUSG00000024617 -149.612 72.6864           6 1.11022e-16 3.66374e-15

# identify top-ranked SVG
rowData(spe_nnSVG)$gene_name[which(rowData(spe_nnSVG)$rank == 1)]
## [1] "Camk2n1"

```

18.9 Dimensionality reduction

```

# compute PCA
set.seed(123)
spe <- runPCA(spe, subset_row = top_hvgs)

reducedDimNames(spe)
## [1] "PCA"
dim(reducedDim(spe, "PCA"))
## [1] 2683   50

```

```

# compute UMAP on top 50 PCs
set.seed(123)
spe <- runUMAP(spe, dimred = "PCA")

reducedDimNames(spe)
## [1] "PCA"   "UMAP"

```

```

dim(reducedDim(spe, "UMAP"))
## [1] 2683    2

# update column names for easier plotting
colnames(reducedDim(spe, "UMAP")) <- paste0("UMAP", 1:2)

```

18.10 Clustering

```

# graph-based clustering
set.seed(123)
k <- 10
g <- buildSNNGraph(spe, k = k, use.dimred = "PCA")
g_walk <- igraph::cluster_walktrap(g)
clus <- g_walk$membership
table(clus)
##   clus
##    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19
## 218 255 176 179   60 127 240 217   83   77 271 100 241 125   57 160   23   22   21
##   20
##   31

# store cluster labels in column 'label' in colData
colLabels(spe) <- factor(clus)

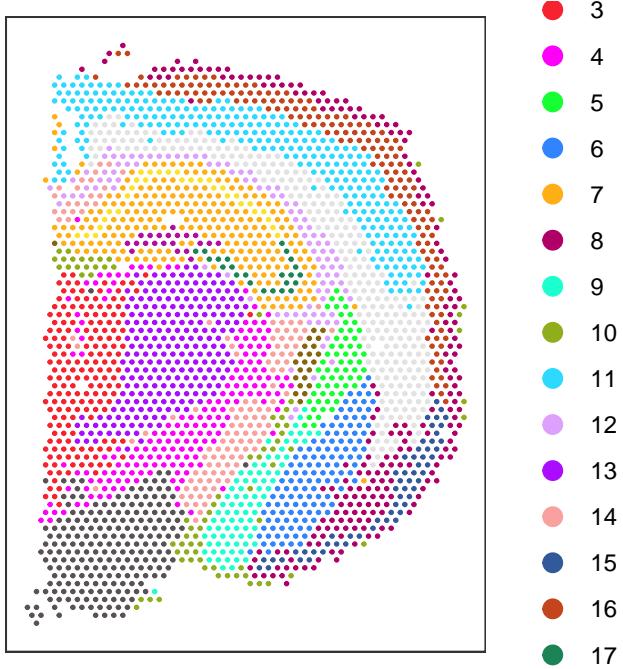
```

```

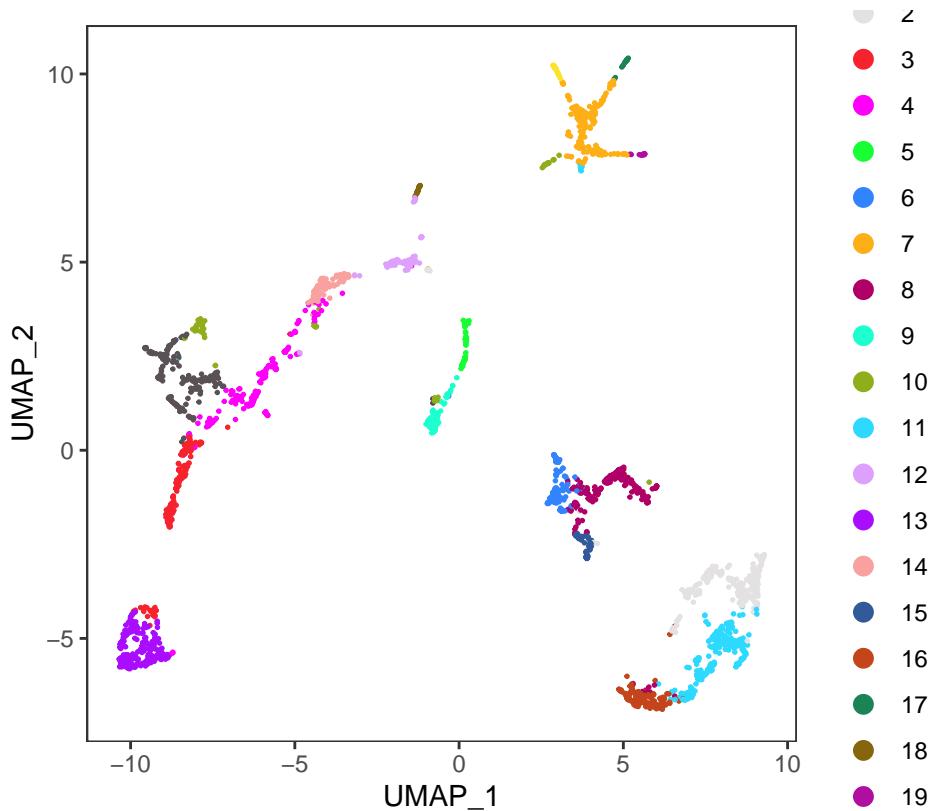
# define custom color palette
colors <- unname(palette.colors(palette = "Polychrome 36"))

# plot clusters in spatial x-y coordinates
plotSpots(spe, annotate = "label",
           pal = colors)

```



```
# plot clusters in UMAP dimensions
plotDimRed(spe, plot_type = "UMAP",
            annotate = "label", pal = colors)
```



18.11 Differential expression

```
# set gene names as row names for easier plotting
rownames(spe) <- rowData(spe)$gene_name

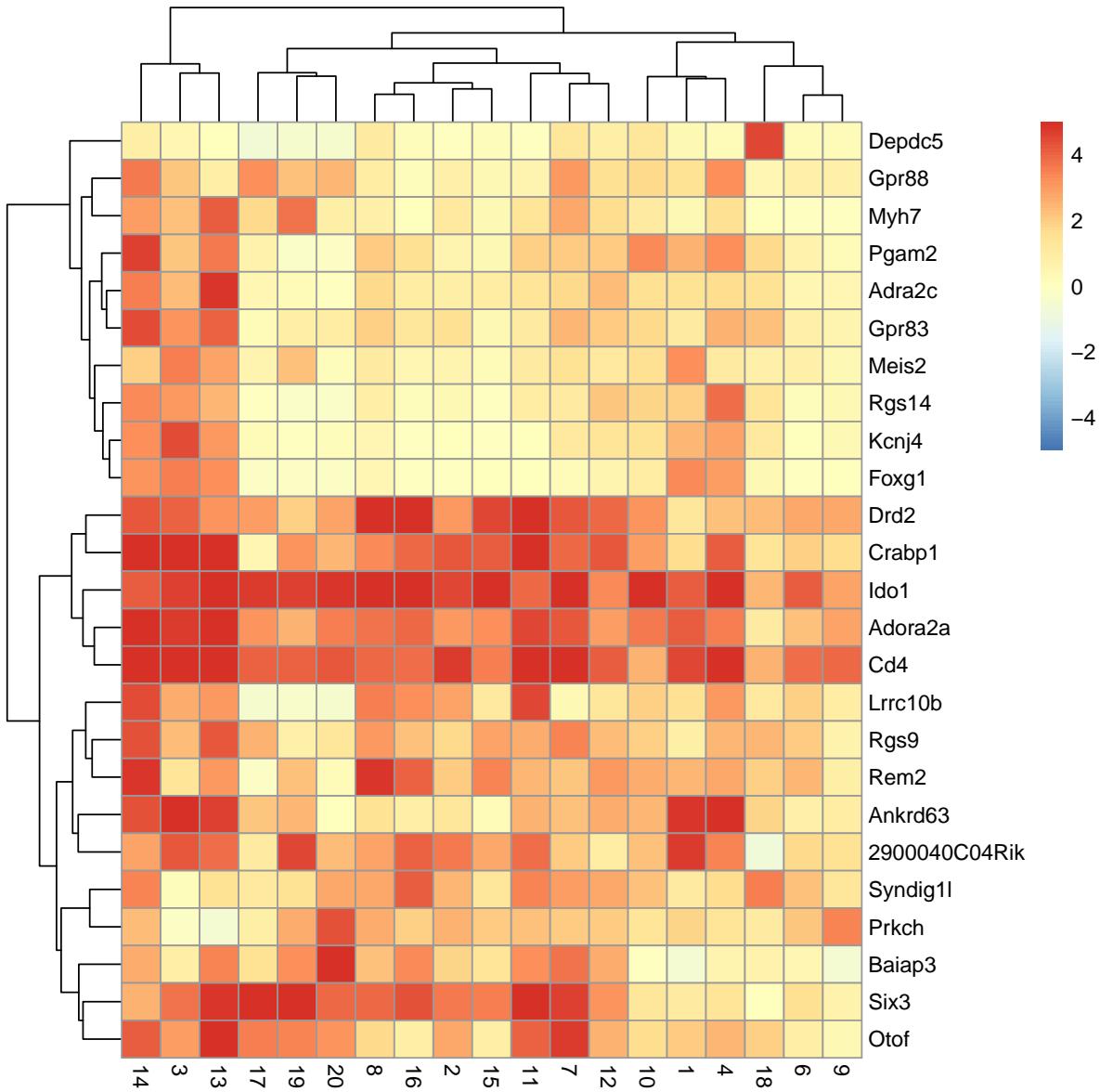
# test for marker genes
markers <- findMarkers(spe, test = "binom", direction = "up")

# returns a list with one DataFrame per cluster
markers
## List of length 20
##  names(20): 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

```
library(pheatmap)

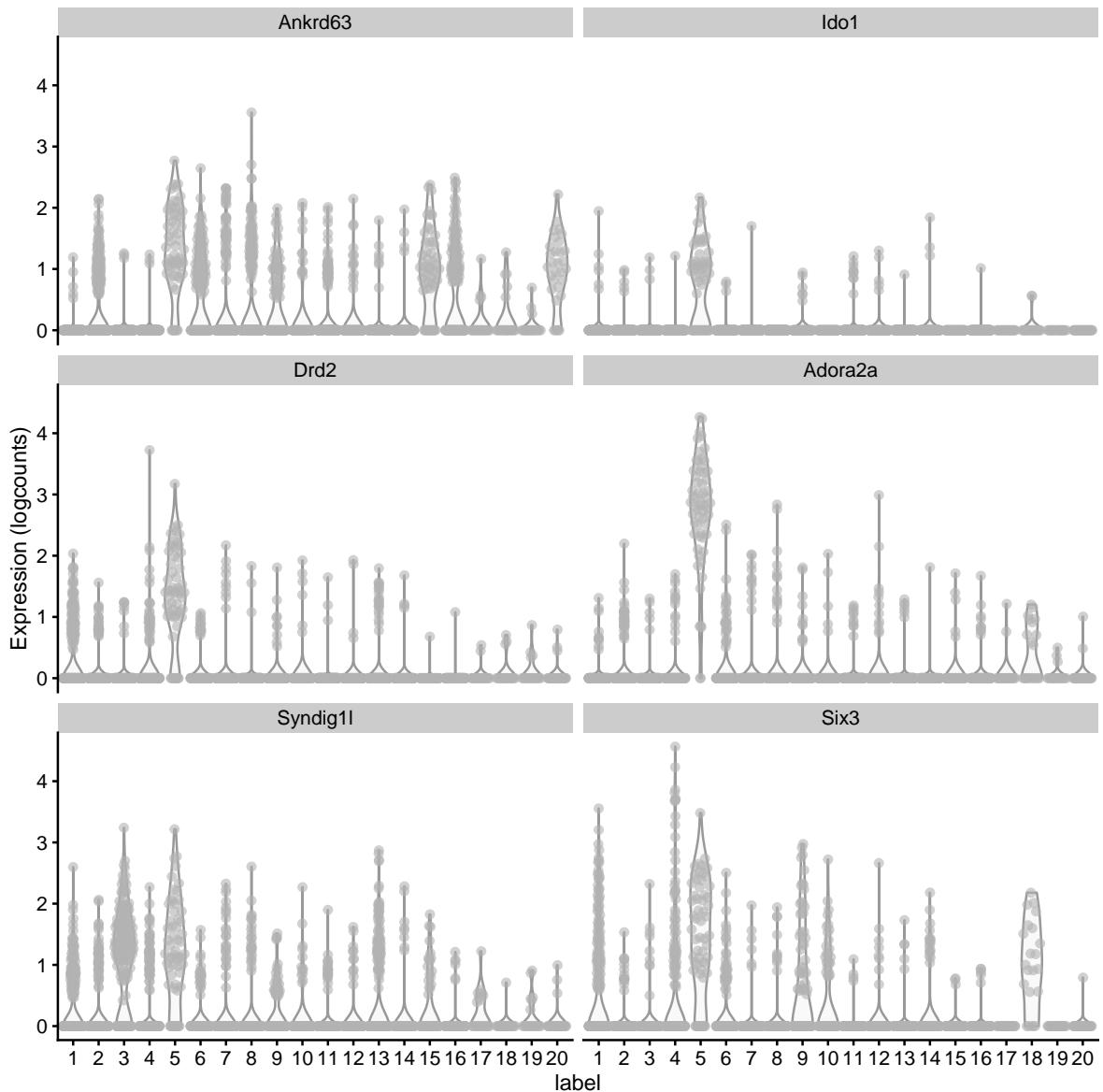
# plot log-fold changes for one cluster over all other clusters
# selecting cluster 5
interesting <- markers[[5]]
best_set <- interesting[interesting$Top <= 5, ]
logFCs <- getMarkerEffects(best_set)

pheatmap(logFCs, breaks = seq(-5, 5, length.out = 101))
```



```
# plot log-transformed normalized expression of top genes for one cluster
top_genes <- head(rownames(interesting))

plotExpression(spe, x = "label", features = top_genes)
```



References

19 spatialLIBD workflow

19.1 Overview

17

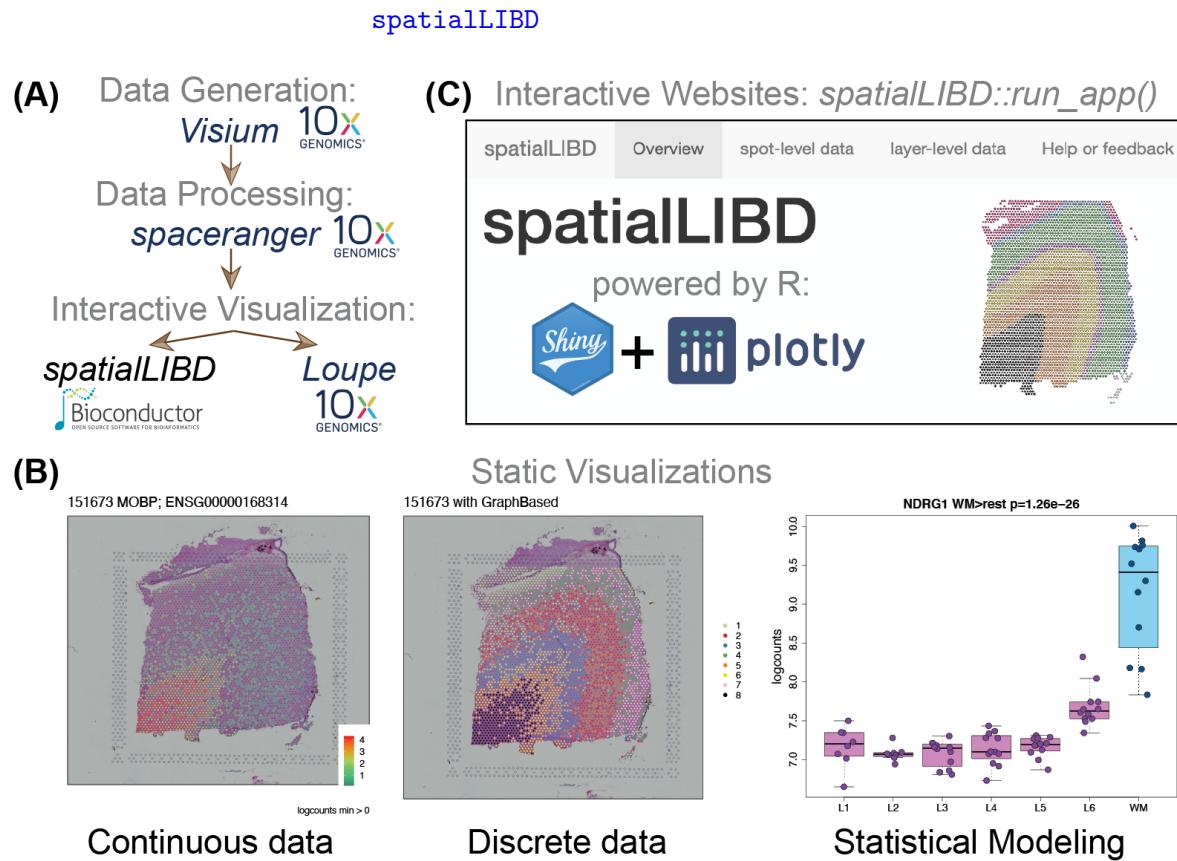


Figure 19.1: spatialLIBD overview. Source: @Pardo2022.

19.2 spatialLIBD

19.2.1 Why use spatialLIBD?

•
•

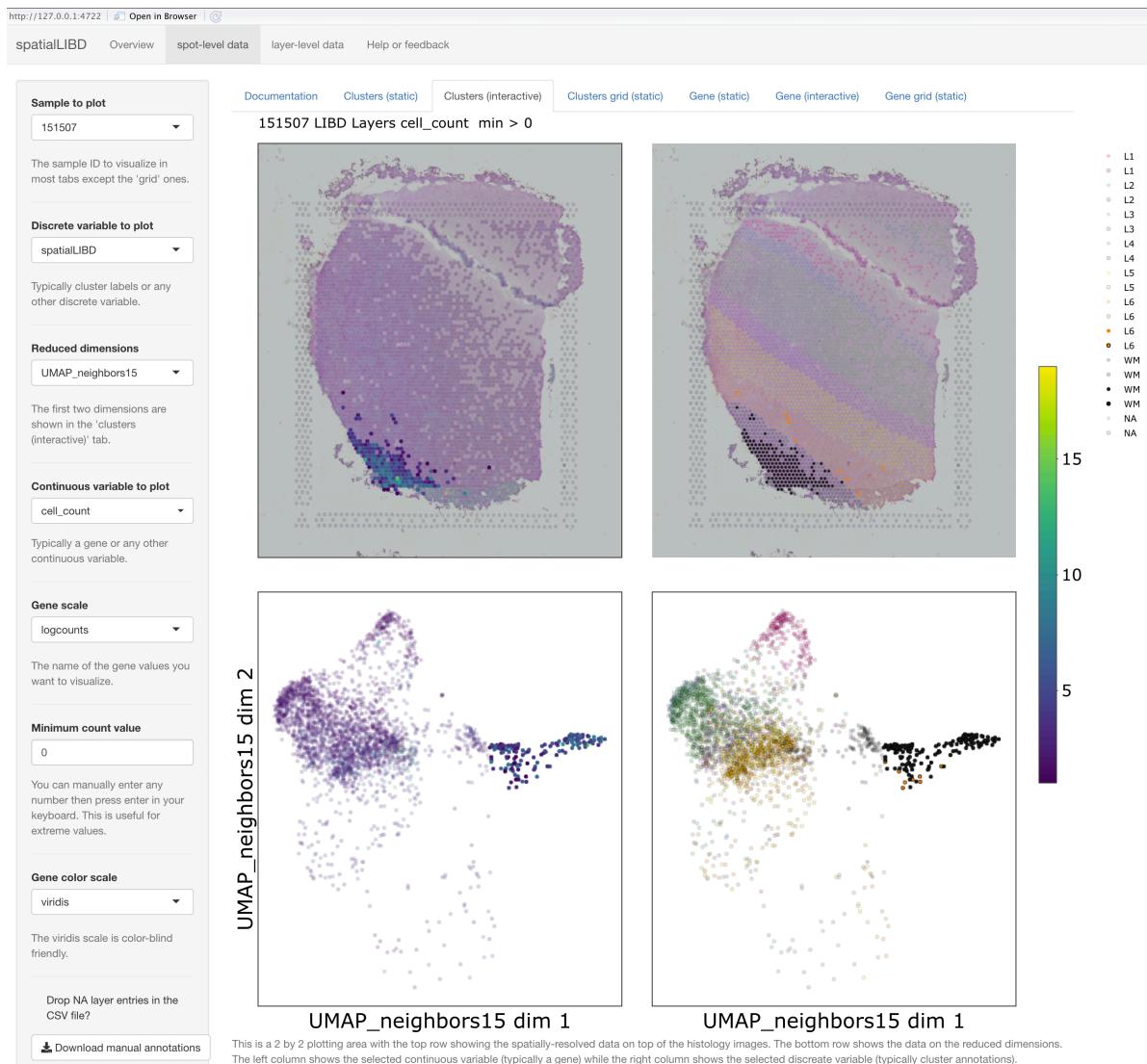


Figure 19.2: Screenshot of the ‘clusters (interactive)’ section of the ‘spot-level data’ panel created with the full spatialLIBD dataset. The website was created with `spatialLIBD::run_app(spatialLIBD::fetch_data('spe'))` version 1.4.0 and then using the lasso selection, we selected a set of spots in the UMAP interactive plot colored by the estimated number of cells per spot (`cell_count`) on the bottom left, which automatically updated the other three plots.

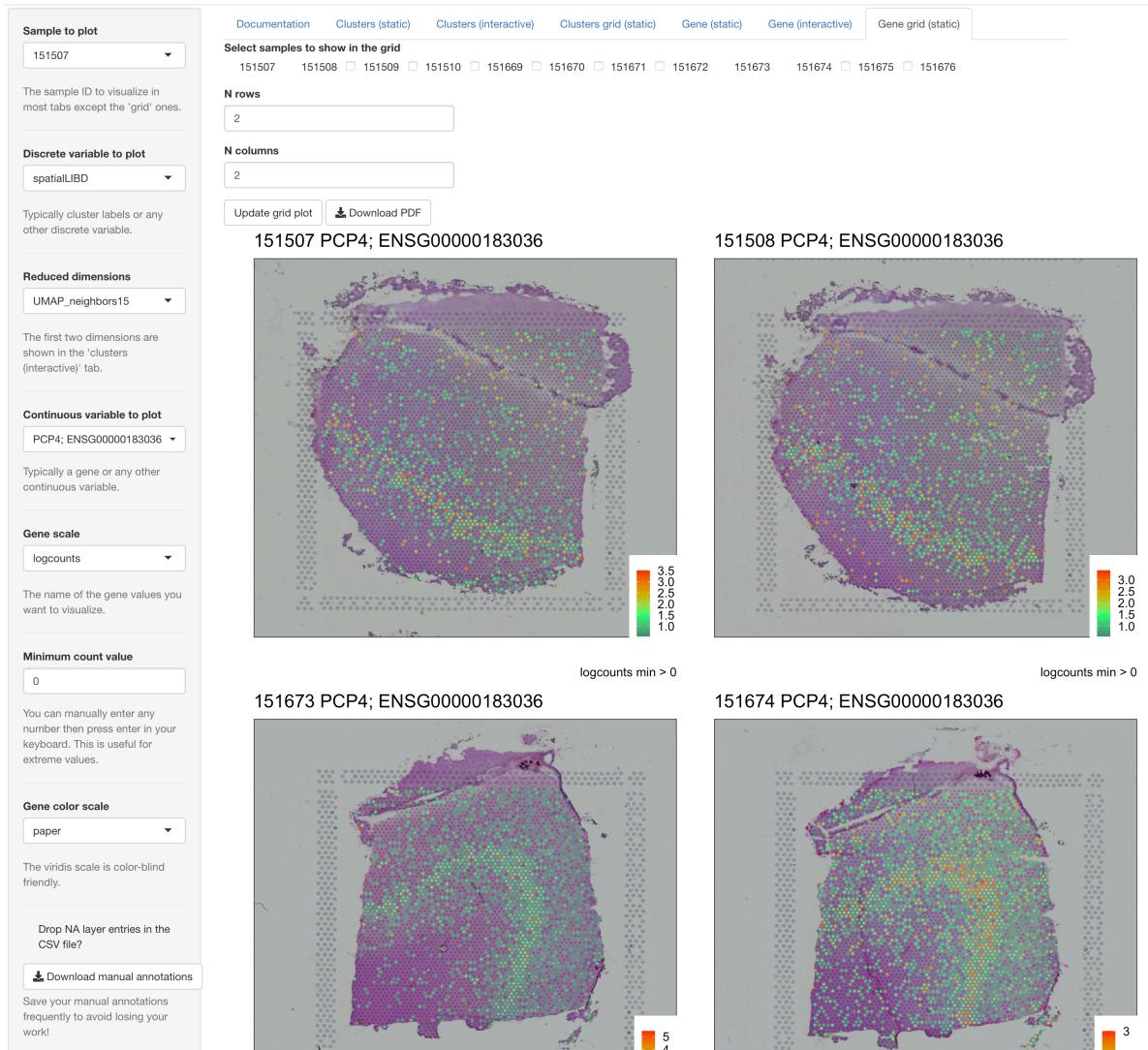


Figure 19.3: Screenshot of the ‘gene grid (static)’ section of the ‘spot-level data’ panel created with the full spatialLIBD dataset. The website was created with `spatialLIBD::run_app(spatialLIBD::fetch_data('spe'))` version 1.4.0, selecting the *PCP4* gene, selecting the *paper* gene color scale, changing the number of rows and columns in the grid 2, selecting two pairs of spatially adjacent replicate samples (151507, 151508, 151673, and 151674), and clicking on the *upgrade grid plot* button. Note that the default *viridis* gene color scale is color-blind friendly.

it on your browser ¹

19.2.2 Want to learn more about spatialLIBD?

Bioconductor landing page [spatialLIBD](#)
pkgdown documentation website

- Introduction to spatialLIBD
- Using spatialLIBD with 10x Genomics public datasets

```
citation("spatialLIBD")[1]
## Pardo B, Spangler A, Weber LM, Hicks SC, Jaffe AE, Martinowich K,
## Maynard KR, Collado-Torres L (2022). "spatialLIBD: an R/Bioconductor
## package to visualize spatially-resolved transcriptomics data." _BMC
## Genomics_. doi:10.1186/s12864-022-08601-w
## <https://doi.org/10.1186/s12864-022-08601-w>,
## <https://doi.org/10.1186/s12864-022-08601-w>.
##
## A BibTeX entry for LaTeX users is
##
## @Article{,
##   title = {spatialLIBD: an R/Bioconductor package to visualize spatially-resolved trans-
##   author = {Brenda Pardo and Abby Spangler and Lukas M. Weber and Stephanie C. Hicks and
##   year = {2022},
##   journal = {BMC Genomics},
##   doi = {10.1186/s12864-022-08601-w},
##   url = {https://doi.org/10.1186/s12864-022-08601-w},
## }
```

19.2.2.1 Recordings

²

¹Check <https://github.com/LieberInstitute/spatialLIBD#shiny-website-mirrors> in case you need to use a mirror. shiny-powered websites work best on browsers such as Google Chrome and Mozilla Firefox, among others.

²Originally available at https://github.com/LieberInstitute/spatialLIBD/blob/master/inst/app/www/documentation_spe.md#slides-and-videos.

19.3 Code prerequisites

17

- `SpatialExperiment`
- `STexampleData`
- `scater`
- `scran`
- `igraph`
- `BiocFileCache`
- `rtracklayer`
- `lobstr`

3

documented by Bioconductor

Bioconductor docker images

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

## Check that you have a valid installation
BiocManager::valid()

## Install the required R packages for this workflow
BiocManager::install(c(
  "SpatialExperiment",
  "STexampleData",
  "scater",
```

³This book is under development, so right now you actually need to use `bioc-devel`. Once this book is submitted to Bioconductor, then it'll work with `bioc-release`. TODO: remove this comment when the book is available on `bioc-release`.

```

  "scran",
  "igraph",
  "BiocFileCache",
  "rtracklayer",
  "lobstr",
  "spatialLIBD"
))

```

17

```

## Load packages required for the
## "Visium human DLPFC workflow"
library("SpatialExperiment")
library("STexampleData")
library("scater")
library("scran")
library("igraph")
library("BiocFileCache")
library("rtracklayer")
library("lobstr")
library("spatialLIBD")

## Start tracking time
time_start <- Sys.time()

# load object
spe <- Visium_humanDLPFC()

# subset to keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]

# identify mitochondrial genes
is_mito <- grep("(^MT-)|(mt-)", rowData(spe)$gene_name)

# calculate per-spot QC metrics and store in colData
spe <- addPerCellQC(spe, subsets = list(mito = is_mito))

# select QC thresholds
qc_lib_size <- colData(spe)$sum < 600
qc_detected <- colData(spe)$detected < 400
qc_mito <- colData(spe)$subsets_mito_percent > 28

```

```

qc_cell_count <- colData(spe)$cell_count > 10

# combined set of discarded spots
discard <- qc_lib_size | qc_detected | qc_mito | qc_cell_count

# store in object
colData(spe)$discard <- discard

# filter low-quality spots
spe <- spe[, !colData(spe)$discard]

# calculate logcounts using library size factors
spe <- logNormCounts(spe)

# remove mitochondrial genes
spe <- spe[!is_mito, ]

# fit mean-variance relationship
dec <- modelGeneVar(spe)

# select top HVGs
top_hvgs <- getTopHVGs(dec, prop = 0.1)

# compute PCA
set.seed(123)
spe <- runPCA(spe, subset_row = top_hvgs)

# compute UMAP on top 50 PCs
set.seed(123)
spe <- runUMAP(spe, dimred = "PCA")

# update column names for easier plotting
colnames(reducedDim(spe, "UMAP")) <- paste0("UMAP", 1:2)

# graph-based clustering
set.seed(123)
k <- 10
g <- buildSNNGraph(spe, k = k, use.dimred = "PCA")
g_walk <- igraph::cluster_walktrap(g)
clus <- g_walk$membership

# store cluster labels in column 'label' in colData

```

```

colLabels(spe) <- factor(clus)

# set gene names as row names for easier plotting
rownames(spe) <- rowData(spe)$gene_name

# test for marker genes
markers <- findMarkers(spe, test = "binom", direction = "up")

## Find the interesting markers for each cluster
interesting <- sapply(markers, function(x) x$Top <= 5)
colnames(interesting) <- paste0("gene_interest_", seq_len(length(markers)))
rowData(spe) <- cbind(rowData(spe), interesting)

## How long this code took to run
time_prereqs <- Sys.time()
time_prereqs - time_start
## Time difference of 1.54969 mins

```

19.4 Prepare for spatialLIBD

using spatialLIBD with 10x Genomics public datasets

19.4.1 Basic information

```

## Add some information used by spatialLIBD
spe$key <- paste0(spe$sample_id, "_", colnames(spe))
spe$sum_umi <- colSums(counts(spe))
spe$sum_gene <- colSums(counts(spe) > 0)

```

19.4.2 Gene annotation

add the gene annotation data from Gencode

```

## Download the Gencode v32 GTF file and cache it
bfc <- BiocFileCache::BiocFileCache()
gtf_cache <- BiocFileCache::bfcrpath(
  bfc,
  paste0(
    "ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/",
    "release_32/gencode.v32.annotation.gtf.gz"
  )
)

## Show the GTF cache location
gtf_cache
##                                     BFC1
## "/root/.cache/R/BiocFileCache/6c65f14425f_gencode.v32.annotation.gtf.gz"

## Import into R (takes ~1 min)
gtf <- rtracklayer::import(gtf_cache)

## Subset to genes only
gtf <- gtf[gtf$type == "gene"]

## Remove the .x part of the gene IDs
gtf$gene_id <- gsub("\\..*", "", gtf$gene_id)

## Set the names to be the gene IDs
names(gtf) <- gtf$gene_id

## Match the genes
match_genes <- match(rowData(spe)$gene_id, gtf$gene_id)
table(is.na(match_genes))
##
## FALSE  TRUE
## 33267  258

## Drop the few genes for which we don't have information
spe <- spe[!is.na(match_genes), ]
match_genes <- match_genes[!is.na(match_genes)]

## Keep only some columns from the gtf
mcols(gtf) <- mcols(gtf)[, c("source", "type", "gene_id", "gene_name", "gene_type")]

## Save the "interesting" columns from our original spe object

```

```

interesting <- rowData(spe) [, grep("interest", colnames(rowData(spe)))]  
  

## Add the gene info to our SPE object  

rowRanges(spe) <- gtf[match_genes]  
  

## Add back the "interest" coolumns  

rowData(spe) <- cbind(rowData(spe), interesting)  
  

## Inspect the gene annotation data we added  

rowRanges(spe)  

## GRanges object with 33267 ranges and 11 metadata columns:  

##           seqnames      ranges strand / source      type  

##           <Rle>       <IRanges> <Rle> / <factor> <factor>  

## ENSG00000243485    chr1     29554-31109    + / HAVANA   gene  

## ENSG00000237613    chr1     34554-36081    - / HAVANA   gene  

## ENSG00000186092    chr1     65419-71585    + / HAVANA   gene  

## ENSG00000238009    chr1     89295-133723   - / HAVANA   gene  

## ENSG00000239945    chr1     89551-91105    - / HAVANA   gene  

##           ...       ...       ...       .       ...       ...  

## ENSG00000160298    chr21   46300181-46323875  - / HAVANA   gene  

## ENSG00000160299    chr21   46324141-46445769  + / HAVANA   gene  

## ENSG00000160305    chr21   46458891-46570015  + / HAVANA   gene  

## ENSG00000160307    chr21   46598604-46605208  - / HAVANA   gene  

## ENSG00000160310    chr21   46635595-46665124  + / HAVANA   gene  

##           gene_id    gene_name      gene_type gene_interest_1  

##           <character> <character> <character> <logical>  

## ENSG00000243485 ENSG00000243485 MIR1302-2HG lncRNA      TRUE  

## ENSG00000237613 ENSG00000237613 FAM138A lncRNA      TRUE  

## ENSG00000186092 ENSG00000186092 OR4F5 protein_coding TRUE  

## ENSG00000238009 ENSG00000238009 AL627309.1 lncRNA      TRUE  

## ENSG00000239945 ENSG00000239945 AL627309.3 lncRNA      TRUE  

##           ...       ...       ...       .       ...  

## ENSG00000160298 ENSG00000160298 C21orf58 protein_coding FALSE  

## ENSG00000160299 ENSG00000160299 PCNT protein_coding FALSE  

## ENSG00000160305 ENSG00000160305 DIP2A protein_coding FALSE  

## ENSG00000160307 ENSG00000160307 S100B protein_coding FALSE  

## ENSG00000160310 ENSG00000160310 PRMT2 protein_coding FALSE  

##           gene_interest_2 gene_interest_3 gene_interest_4  

##           <logical>      <logical>      <logical>  

## ENSG00000243485      TRUE        TRUE        TRUE  

## ENSG00000237613      TRUE        TRUE        TRUE  

## ENSG00000186092      TRUE        TRUE        TRUE

```

```

##      ENSG00000238009      TRUE      TRUE      TRUE
##      ENSG00000239945      TRUE      TRUE      TRUE
##      ...
##      ENSG00000160298     FALSE     FALSE     FALSE
##      ENSG00000160299     FALSE     FALSE     FALSE
##      ENSG00000160305     FALSE     FALSE     FALSE
##      ENSG00000160307     FALSE     FALSE     FALSE
##      ENSG00000160310     FALSE     FALSE     FALSE
##      gene_interest_5 gene_interest_6
##                  <logical>      <logical>
##      ENSG00000243485      TRUE      TRUE
##      ENSG00000237613      TRUE      TRUE
##      ENSG00000186092      TRUE      TRUE
##      ENSG00000238009      TRUE      TRUE
##      ENSG00000239945      TRUE      TRUE
##      ...
##      ENSG00000160298     FALSE     FALSE
##      ENSG00000160299     FALSE     FALSE
##      ENSG00000160305     FALSE     FALSE
##      ENSG00000160307     FALSE     FALSE
##      ENSG00000160310     FALSE     FALSE
## -----
##      seqinfo: 25 sequences from an unspecified genome; no seqlengths

```

```

## Add information used by spatialLIBD
rowData(spe)$gene_search <- paste0(
  rowData(spe)$gene_name, " ; ", rowData(spe)$gene_id
)

## Compute chrM expression and chrM expression ratio
is_mito <- which(seqnames(spe) == "chrM")
spe$expr_chrM <- colSums(counts(spe)[is_mito, , drop = FALSE])
spe$expr_chrM_ratio <- spe$expr_chrM / spe$sum_umi

```

19.4.3 Extra information and filtering

filter the object

```
## Add a variable for saving the manual annotations
spe$ManualAnnotation <- "NA"

## Remove genes with no data
no_expr <- which(rowSums(counts(spe)) == 0)

## Number of genes with no counts
length(no_expr)
## [1] 11596

## Compute the percent of genes with no counts
length(no_expr) / nrow(spe) * 100
## [1] 34.85737
spe <- spe[-no_expr, , drop = FALSE]

## Remove spots without counts
summary(spe$sum_umi)
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
##      537    2414   3466    3870   4938   15862

## If we had spots with no counts, we would remove them
if (any(spe$sum_umi == 0)) {
  spots_no_counts <- which(spe$sum_umi == 0)
  ## Number of spots with no counts
  print(length(spots_no_counts))
  ## Percent of spots with no counts
  print(length(spots_no_counts) / ncol(spe) * 100)
  spe <- spe[, -spots_no_counts, drop = FALSE]
}
```

```
## Run check_spe() function
spatialLIBD::check_spe(spe)
```

```

##  class: SpatialExperiment
##  dim: 21671 3524
##  metadata(0):
##  assays(2): counts logcounts
##  rownames(21671): ENSG00000243485 ENSG00000238009 ... ENSG00000160307
##    ENSG00000160310
##  rowData names(12): source type ... gene_interest_6 gene_search
##  colnames(3524): AAACAAGTATCTCCCA-1 AAACACCAATAACTGC-1 ...
##    TTGTTTCCATACAAC-1 TTGTTTGTGTAAATTG-1
##  colData names(23): barcode_id sample_id ... expr_chrm_ratio
##  ManualAnnotation
##  reducedDimNames(2): PCA UMAP
##  mainExpName: NULL
##  altExpNames(0):
##  spatialCoords names(2) : pxl_col_in_fullres pxl_row_in_fullres
##  imgData names(4): sample_id image_id data scaleFactor

## End tracking time
time_end <- Sys.time()

## How long this code took to run
time_end - time_prereqs
## Time difference of 32.08765 secs

```

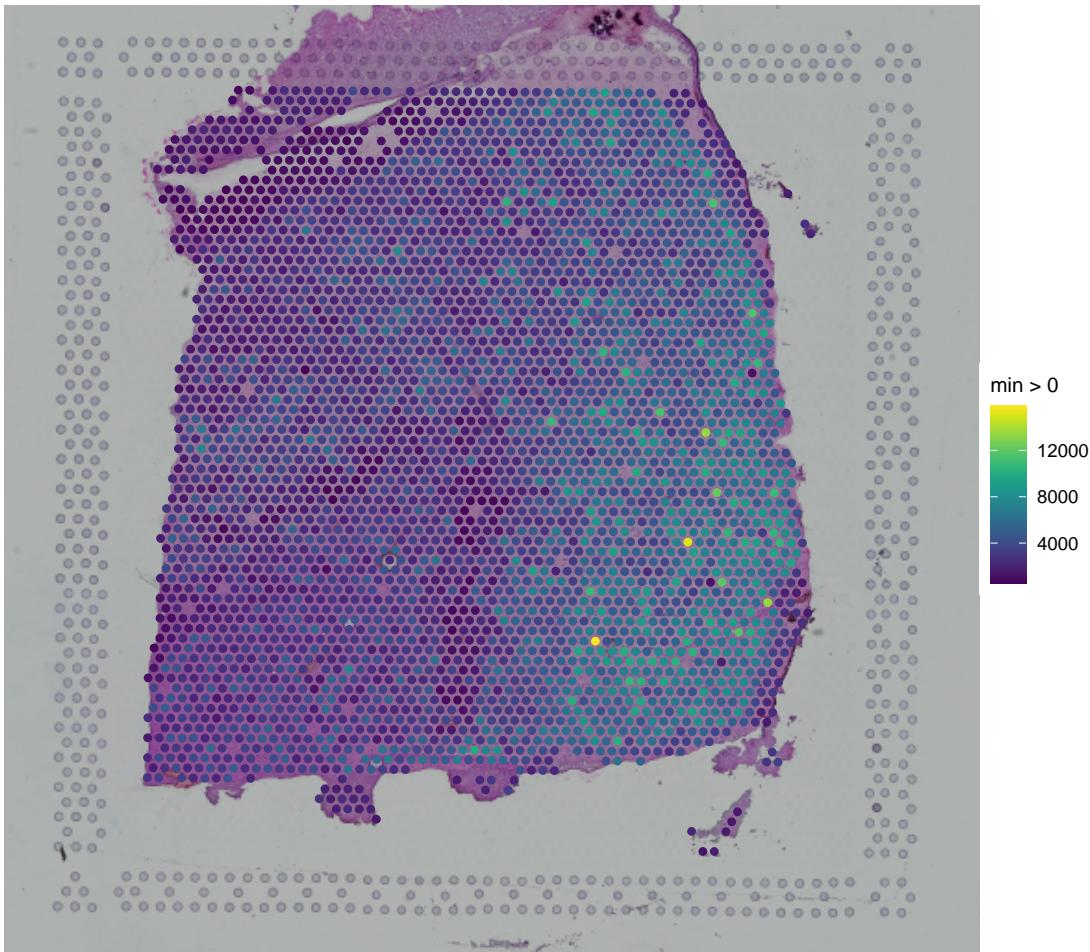
```
saveRDS(spe, file = "spe_workflow_Visium_spatialLIBD.rds")
```

```
spe <- readRDS("spe_workflow_Visium_spatialLIBD.rds")
```

19.5 Explore the data

```
## Sum of UMI
spatialLIBD::vis_gene(
  spe = spe,
  sampleid = "sample_151673",
  geneid = "sum_umi"
)
```

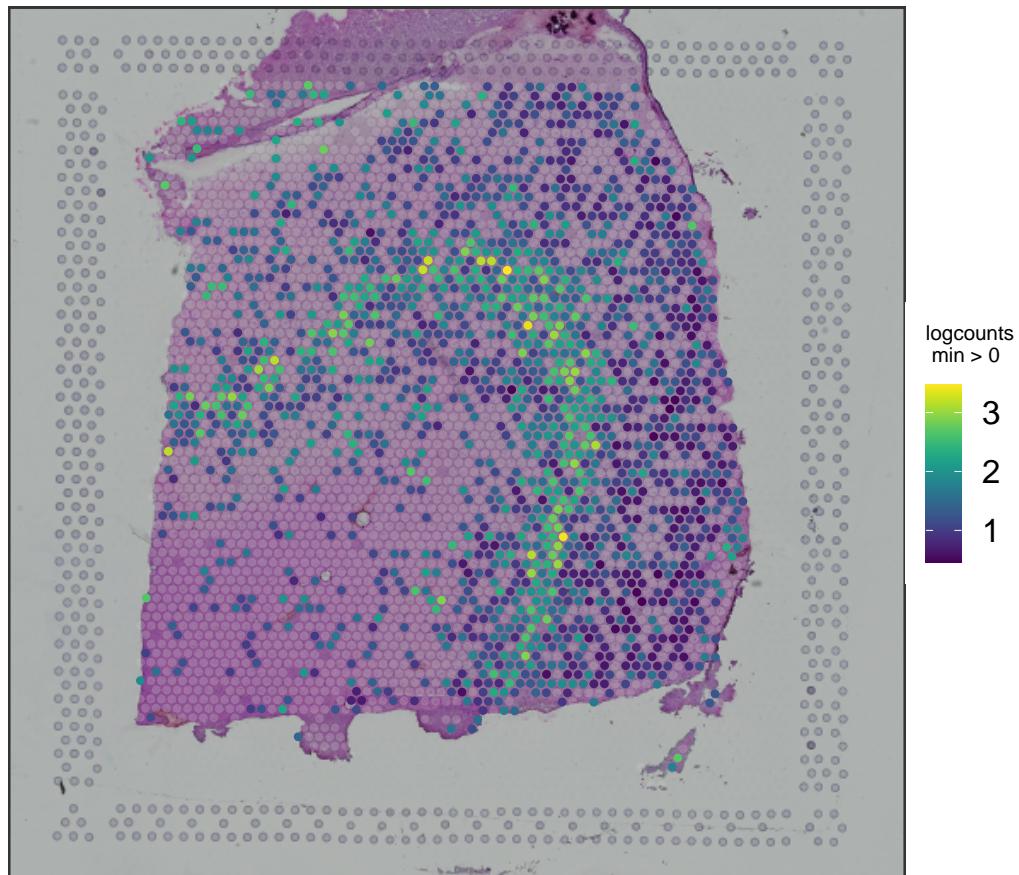
sample_151673 sum_umi



```
## PCP4, a layer 5 marker gene
spatialLIBD::vis_gene(
  spe = spe,
  sampleid = "sample_151673",
```

```
    geneid = rowData(spe)$gene_search[which(rowData(spe)$gene_name == "PCP4")]
}
```

sample_151673 PCP4; ENSG00000183036



[visualize the data interactively](#)

```
## Explore all the variables we can use
colData(spe)
## DataFrame with 3524 rows and 23 columns
##          barcode_id      sample_id  in_tissue array_row
```

| | <character> | <character> | <integer> | <integer> |
|----|--------------------|----------------------|------------------|-----------------------|
| ## | | | | |
| ## | AAACAAGTATCTCCA-1 | AAACAAGTATCTCCA-1 | sample_151673 | 1 |
| ## | AAACACCAATAACTGC-1 | AAACACCAATAACTGC-1 | sample_151673 | 1 |
| ## | AAACAGAGCGACTCCT-1 | AAACAGAGCGACTCCT-1 | sample_151673 | 1 |
| ## | AAACAGCTTCAGAAG-1 | AAACAGCTTCAGAAG-1 | sample_151673 | 1 |
| ## | AAACAGGGTCTATATT-1 | AAACAGGGTCTATATT-1 | sample_151673 | 1 |
| ## | ... | ... | ... | ... |
| ## | TTGTTGTGTCAAGA-1 | TTGTTGTGTCAAGA-1 | sample_151673 | 1 |
| ## | TTGTTCACATCCAGG-1 | TTGTTCACATCCAGG-1 | sample_151673 | 1 |
| ## | TTGTTCATAGTCTA-1 | TTGTTCATAGTCTA-1 | sample_151673 | 1 |
| ## | TTGTTCCATACAACT-1 | TTGTTCCATACAACT-1 | sample_151673 | 1 |
| ## | TTGTTGTGTAAATTC-1 | TTGTTGTGTAAATTC-1 | sample_151673 | 1 |
| ## | | array_col | ground_truth | reference |
| ## | | ground_truth | reference | cell_count |
| ## | | reference | cell_count | sum |
| ## | | <integer> | <character> | <character> |
| ## | AAACAAGTATCTCCA-1 | 102 | Layer3 | Layer3 |
| ## | AAACACCAATAACTGC-1 | 19 | WM | WM |
| ## | AAACAGAGCGACTCCT-1 | 94 | Layer3 | Layer3 |
| ## | AAACAGCTTCAGAAG-1 | 9 | Layer5 | Layer5 |
| ## | AAACAGGGTCTATATT-1 | 13 | Layer6 | Layer6 |
| ## | ... | ... | ... | ... |
| ## | TTGTTGTGTCAAGA-1 | 77 | Layer5 | Layer5 |
| ## | TTGTTCACATCCAGG-1 | 42 | WM | WM |
| ## | TTGTTCATAGTCTA-1 | 30 | WM | WM |
| ## | TTGTTCCATACAACT-1 | 27 | Layer6 | Layer6 |
| ## | TTGTTGTGTAAATTC-1 | 51 | Layer2 | Layer2 |
| ## | | detected | subsets_mito_sum | subsets_mito_detected |
| ## | | <numeric> | <numeric> | <numeric> |
| ## | AAACAAGTATCTCCA-1 | 3586 | 1407 | 13 |
| ## | AAACACCAATAACTGC-1 | 1960 | 430 | 13 |
| ## | AAACAGAGCGACTCCT-1 | 2424 | 1316 | 13 |
| ## | AAACAGCTTCAGAAG-1 | 2264 | 651 | 12 |
| ## | AAACAGGGTCTATATT-1 | 2178 | 621 | 13 |
| ## | ... | ... | ... | ... |
| ## | TTGTTGTGTCAAGA-1 | 1982 | 789 | 13 |
| ## | TTGTTCACATCCAGG-1 | 2170 | 370 | 12 |
| ## | TTGTTCATAGTCTA-1 | 1560 | 314 | 12 |
| ## | TTGTTCCATACAACT-1 | 1343 | 476 | 13 |
| ## | TTGTTGTGTAAATTC-1 | 2927 | 991 | 13 |
| ## | | subsets_mito_percent | total | discard |
| ## | | <numeric> | <numeric> | <logical> |
| ## | AAACAAGTATCTCCA-1 | 16.6351 | 8458 | FALSE |
| ## | AAACACCAATAACTGC-1 | 11.4089 | 3769 | FALSE |
| ## | | | | sizeFactor |

```

##  AAACAGAGCGACTCCT-1      24.2223    5433   FALSE  1.170902
##  AAACAGCTTCAGAAG-1      15.2174    4278   FALSE  0.921980
##  AAACAGGGTCTATATT-1     15.5095    4004   FALSE  0.862928
## ...
## ...
##  TTGTTGTGTCAAGA-1      19.89410   3966   FALSE  0.854739
##  TTGTTCACATCCAGG-1      8.55689    4324   FALSE  0.931894
##  TTGTTCATTAGTCTA-1      11.37269   2761   FALSE  0.595041
##  TTGTTCCATACAAC-1       20.49957   2322   FALSE  0.500429
##  TTGTTGTAAATT-1         15.77774   6281   FALSE  1.353660
##          label           key   sum_umi sum_gene
## <factor> <character> <numeric> <integer>
##  AAACAAGTATCTCCA-1     2 sample_151673_AAACAA.. 7051   3573
##  AACACCAATAACTGC-1     1 sample_151673_AAACAC.. 3339   1947
##  AAACAGAGCGACTCCT-1     6 sample_151673_AAACAG.. 4117   2411
##  AAACAGCTTCAGAAG-1     2 sample_151673_AAACAG.. 3627   2252
##  AAACAGGGTCTATATT-1     2 sample_151673_AAACAG.. 3383   2165
## ...
## ...
##  TTGTTGTGTCAAGA-1      2 sample_151673_TTGTTG.. 3177   1969
##  TTGTTCACATCCAGG-1     1 sample_151673_TTGTTT.. 3954   2158
##  TTGTTCATTAGTCTA-1     1 sample_151673_TTGTTT.. 2447   1548
##  TTGTTCCATACAAC-1      4 sample_151673_TTGTTT.. 1846   1330
##  TTGTTGTAAATT-1        6 sample_151673_TTGTTT.. 5290   2914
##          expr_chrM expr_chrM_ratio ManualAnnotation
## <numeric> <numeric> <character>
##  AAACAAGTATCTCCA-1     0          0          NA
##  AACACCAATAACTGC-1     0          0          NA
##  AAACAGAGCGACTCCT-1     0          0          NA
##  AAACAGCTTCAGAAG-1     0          0          NA
##  AAACAGGGTCTATATT-1     0          0          NA
## ...
## ...
##  TTGTTGTGTCAAGA-1      0          0          NA
##  TTGTTCACATCCAGG-1     0          0          NA
##  TTGTTCATTAGTCTA-1     0          0          NA
##  TTGTTCCATACAAC-1      0          0          NA
##  TTGTTGTAAATT-1        0          0          NA

## Run our shiny app
if (interactive()) {
  spatialLIBD::run_app(
    spe,
    sce_layer = NULL,
    modeling_results = NULL,

```

```
    sig_genes = NULL,
    title = "OSTA spatialLIBD workflow example",
    spe_discrete_vars = c("ground_truth", "label", "ManualAnnotation"),
    spe_continuous_vars = c(
      "cell_count",
      "sum_umi",
      "sum_gene",
      "expr_chrM",
      "expr_chrM_ratio",
      "sum",
      "detected",
      "subsets_mito_sum",
      "subsets_mito_detected",
      "subsets_mito_percent",
      "total",
      "sizeFactor"
    ),
    default_cluster = "label"
  )
}
```

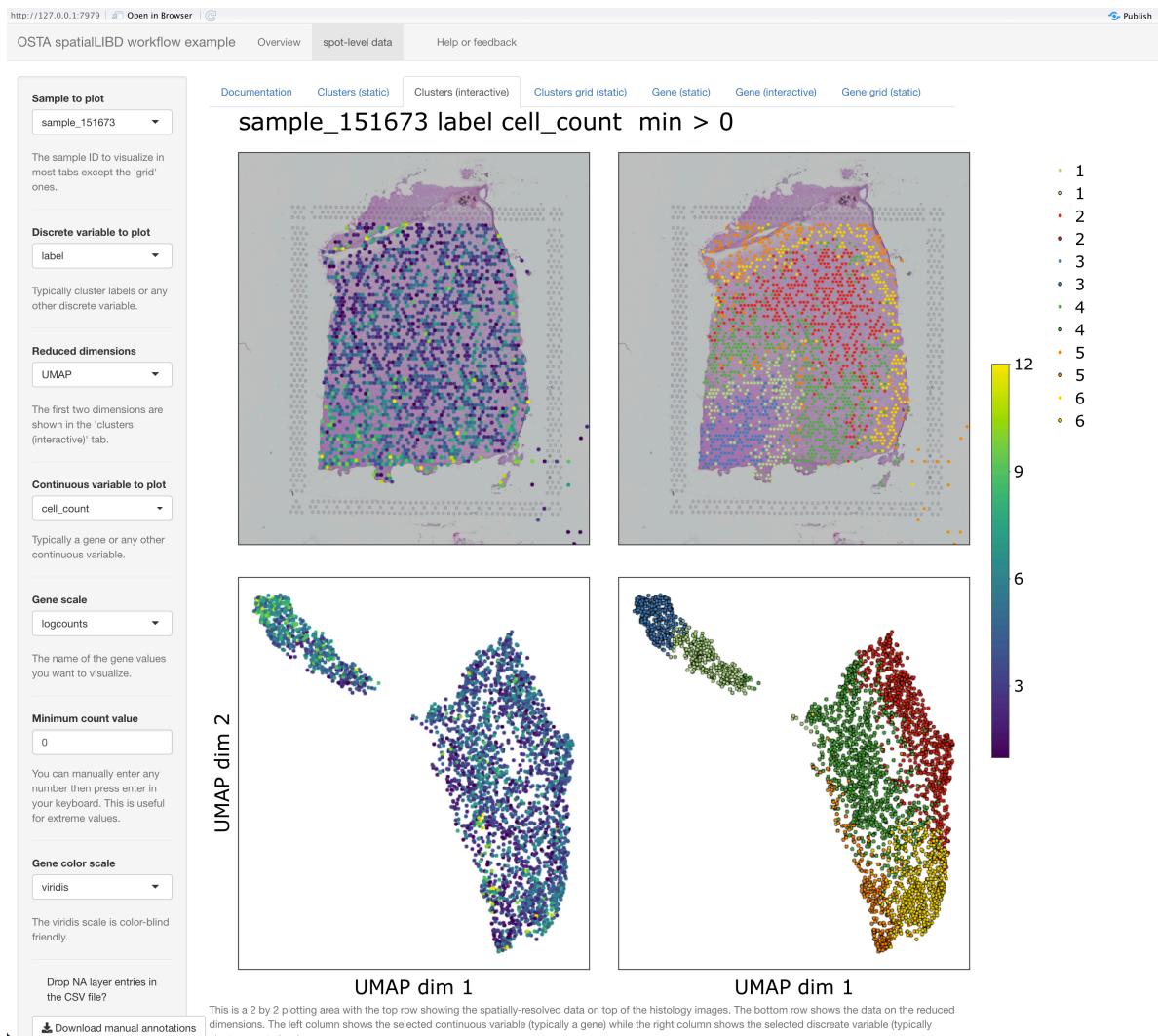


Figure 19.4: Screenshot of the ‘clusters (interactive)’ section of the ‘spot-level data’ panel created with the data from this workflow.

19.6 Sharing your website

```
## Object size
lobstr::obj_size(spe) / 1024^2 ## Convert to MB
## 293.07 B
```

shinyapps.io

on the [spatialLIBD_demo](#) directory

http://127.0.0.1:6509 | [Open in Browser](#) | [Report](#)

OSTA spatialLIBD workflow example

[Overview](#) [spot-level data](#) [Help or feedback](#)

OSTA spatialLIBD demo

This website is part of the demonstration of [spatialLIBD](#) for *Orchestrating Spatial Transcriptomics Analyses* (OSTA) with Bioconductor. Please check the *Visium spatialLIBD workflow* chapter for more details about the materials presented here.

tab-7114-3

This [shiny](#) application was developed by the [R/Bioconductor-powered Team Data Science](#) group at the [Lieber Institute for Brain Development](#) and is hosted by LIBD. It is powered by the [spatialLIBD](#) R/Bioconductor package which you can find described in [its documentation website](#) and you can use to run locally this shiny application by running the command `spatialLIBD::run_app()`.

If you tweet about this website, the data or the R package please use the `#OSTA` hashtag. You can find previous tweets that way as shown [here](#). Thank you! [Tweet #OSTA](#)

[Publish to Server](#)

LIEBER BRAIN MALTZ RE tab-7114-6

Publish Files From: .../OSTA-book/spatialLIBD_demo

app.R
 spe_workflow_Visium_spatialLIBD.rds
 MD www/documentation_sce_layer.md
 MD www/documentation_spe.md
 www/favicon.ico
 www/footer.html
 MD www/README.md

Publish To Account:

libd: shinyapps.io

Update: Create New
[OSTA_spatialLIBD_demo](#)
https://libd.shinyapps.io/OSTA_spatialLIBD_demo/

Uncheck All

Launch browser

[Publish](#) [Cancel](#)

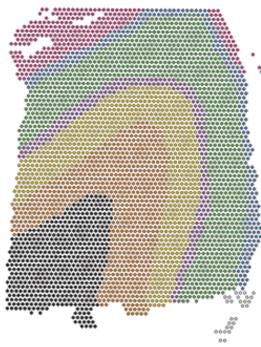


Figure 19.5: Screenshot of the RStudio window for publishing your spatialLIBD-powered website to shinyapps.io

www directory files from the spatialLIBD repository
adapted them to our liking

just like this one

•
•
•

19.7 Wrapping up

-
-
-
-
-

17

R session information

```
options(width = 120)
sessioninfo::session_info()
## - Session info -----
##   setting  value
##   version  R version 4.4.1 (2024-06-14)
##   os        Ubuntu 22.04.4 LTS
##   system   x86_64, linux-gnu
##   ui        X11
##   language (EN)
##   collate   C
##   ctype     en_US.UTF-8
##   tz        Etc/UTC
##   date      2024-09-26
##   pandoc   3.3 @ /usr/bin/ (via rmarkdown)
##
```

| ## - Packages - | | | | |
|-----------------|-----------------|-----------|----------------|-----------------------------|
| | package | * version | date (UTC) | lib source |
| ## | abind | 1.4-8 | 2024-09-12 [2] | RSPM (R 4.4.0) |
| ## | AnnotationDbi | 1.67.0 | 2024-05-01 [2] | Bioconductor 3.20 (R 4.4.0) |
| ## | AnnotationHub | * 3.13.3 | 2024-08-19 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | attempt | 0.3.1 | 2020-05-03 [2] | RSPM (R 4.4.0) |
| ## | beachmat | 2.21.6 | 2024-09-05 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | beeswarm | 0.4.0 | 2021-06-01 [2] | RSPM (R 4.4.0) |
| ## | benchmarkme | 1.0.8 | 2022-06-12 [2] | RSPM (R 4.4.0) |
| ## | benchmarkmeData | 1.0.4 | 2020-04-23 [2] | RSPM (R 4.4.0) |
| ## | Biobase | * 2.65.1 | 2024-08-28 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | BiocFileCache | * 2.13.0 | 2024-05-01 [2] | Bioconductor 3.20 (R 4.4.0) |
| ## | BiocGenerics | * 0.51.1 | 2024-09-03 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | BiocIO | 1.15.2 | 2024-08-23 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | BiocManager | 1.30.25 | 2024-08-28 [2] | CRAN (R 4.4.1) |
| ## | BiocNeighbors | 1.99.1 | 2024-09-22 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | BiocParallel | 1.39.0 | 2024-05-01 [2] | Bioconductor 3.20 (R 4.4.0) |
| ## | BiocSingular | 1.21.4 | 2024-09-22 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | BiocVersion | 3.20.0 | 2024-05-01 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | Biostings | 2.73.1 | 2024-06-02 [2] | Bioconductor 3.20 (R 4.4.0) |
| ## | bit | 4.5.0 | 2024-09-20 [2] | RSPM (R 4.4.0) |
| ## | bit64 | 4.5.2 | 2024-09-22 [2] | RSPM (R 4.4.0) |
| ## | bitops | 1.0-8 | 2024-07-29 [2] | RSPM (R 4.4.0) |
| ## | blob | 1.2.4 | 2023-03-17 [2] | RSPM (R 4.4.0) |
| ## | bluster | 1.15.1 | 2024-09-06 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | bslib | 0.8.0 | 2024-07-29 [2] | RSPM (R 4.4.0) |
| ## | cachem | 1.1.0 | 2024-05-16 [2] | RSPM (R 4.4.0) |
| ## | cli | 3.6.3 | 2024-06-21 [2] | RSPM (R 4.4.0) |
| ## | cluster | 2.1.6 | 2023-12-01 [3] | CRAN (R 4.4.1) |
| ## | codetools | 0.2-20 | 2024-03-31 [3] | CRAN (R 4.4.1) |
| ## | colorspace | 2.1-1 | 2024-07-26 [2] | RSPM (R 4.4.0) |
| ## | config | 0.3.2 | 2023-08-30 [2] | RSPM (R 4.4.0) |
| ## | cowplot | 1.1.3 | 2024-01-22 [2] | RSPM (R 4.4.0) |
| ## | crayon | 1.5.3 | 2024-06-20 [2] | RSPM (R 4.4.0) |
| ## | curl | 5.2.3 | 2024-09-20 [2] | RSPM (R 4.4.0) |
| ## | data.table | 1.16.0 | 2024-08-27 [2] | RSPM (R 4.4.0) |
| ## | DBI | 1.2.3 | 2024-06-02 [2] | RSPM (R 4.4.0) |
| ## | dbplyr | * 2.5.0 | 2024-03-19 [2] | RSPM (R 4.4.0) |
| ## | DelayedArray | 0.31.11 | 2024-08-04 [2] | Bioconductor 3.20 (R 4.4.1) |
| ## | digest | 0.6.37 | 2024-08-19 [2] | RSPM (R 4.4.0) |
| ## | doParallel | 1.0.17 | 2022-02-07 [2] | RSPM (R 4.4.0) |
| ## | dotCall64 | 1.1-1 | 2023-11-28 [2] | RSPM (R 4.4.0) |

| | | | |
|----|--------------------------|----------|--|
| ## | <i>dplyr</i> | 1.1.4 | 2023-11-17 [2] RSPM (R 4.4.0) |
| ## | <i>dqrng</i> | 0.4.1 | 2024-05-28 [2] RSPM (R 4.4.0) |
| ## | <i>DT</i> | 0.33 | 2024-04-04 [2] RSPM (R 4.4.0) |
| ## | <i>edgeR</i> | 4.3.16 | 2024-09-22 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>evaluate</i> | 1.0.0 | 2024-09-17 [2] RSPM (R 4.4.0) |
| ## | <i>ExperimentHub</i> | * 2.13.1 | 2024-07-31 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>fansi</i> | 1.0.6 | 2023-12-08 [2] RSPM (R 4.4.0) |
| ## | <i>farver</i> | 2.1.2 | 2024-05-13 [2] RSPM (R 4.4.0) |
| ## | <i>fastmap</i> | 1.2.0 | 2024-05-15 [2] RSPM (R 4.4.0) |
| ## | <i>fields</i> | 16.2 | 2024-06-27 [2] RSPM (R 4.4.0) |
| ## | <i>filelock</i> | 1.0.3 | 2023-12-11 [2] RSPM (R 4.4.0) |
| ## | <i>FNN</i> | 1.1.4.1 | 2024-09-22 [2] RSPM (R 4.4.0) |
| ## | <i>foreach</i> | 1.5.2 | 2022-02-02 [2] RSPM (R 4.4.0) |
| ## | <i>generics</i> | 0.1.3 | 2022-07-05 [2] RSPM (R 4.4.0) |
| ## | <i>GenomeInfoDb</i> | * 1.41.1 | 2024-05-24 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>GenomeInfoDbData</i> | 1.2.12 | 2024-06-24 [2] Bioconductor |
| ## | <i>GenomicAlignments</i> | 1.41.0 | 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>GenomicRanges</i> | * 1.57.1 | 2024-06-12 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>ggbeeswarm</i> | 0.7.2 | 2023-04-29 [2] RSPM (R 4.4.0) |
| ## | <i>ggplot2</i> | * 3.5.1 | 2024-04-23 [2] RSPM (R 4.4.0) |
| ## | <i>ggrepel</i> | 0.9.6 | 2024-09-07 [2] RSPM (R 4.4.0) |
| ## | <i>glue</i> | 1.7.0 | 2024-01-09 [2] RSPM (R 4.4.0) |
| ## | <i>golem</i> | 0.5.1 | 2024-08-27 [2] RSPM (R 4.4.0) |
| ## | <i>gridExtra</i> | 2.3 | 2017-09-09 [2] RSPM (R 4.4.0) |
| ## | <i>gtable</i> | 0.3.5 | 2024-04-22 [2] RSPM (R 4.4.0) |
| ## | <i>htmltools</i> | 0.5.8.1 | 2024-04-04 [2] RSPM (R 4.4.0) |
| ## | <i>htmlwidgets</i> | 1.6.4 | 2023-12-06 [2] RSPM (R 4.4.0) |
| ## | <i>httpuv</i> | 1.6.15 | 2024-03-26 [2] RSPM (R 4.4.0) |
| ## | <i>httr</i> | 1.4.7 | 2023-08-15 [2] RSPM (R 4.4.0) |
| ## | <i>igraph</i> | * 2.0.3 | 2024-03-13 [2] RSPM (R 4.4.0) |
| ## | <i>IRanges</i> | * 2.39.2 | 2024-07-17 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>irlba</i> | 2.3.5.1 | 2022-10-03 [2] RSPM (R 4.4.0) |
| ## | <i>iterators</i> | 1.0.14 | 2022-02-05 [2] RSPM (R 4.4.0) |
| ## | <i>jquerylib</i> | 0.1.4 | 2021-04-26 [2] RSPM (R 4.4.0) |
| ## | <i>jsonlite</i> | 1.8.9 | 2024-09-20 [2] RSPM (R 4.4.0) |
| ## | <i>KEGGREST</i> | 1.45.1 | 2024-06-17 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>knitr</i> | 1.48 | 2024-07-07 [2] RSPM (R 4.4.0) |
| ## | <i>labeling</i> | 0.4.3 | 2023-08-29 [2] RSPM (R 4.4.0) |
| ## | <i>later</i> | 1.3.2 | 2023-12-06 [2] RSPM (R 4.4.0) |
| ## | <i>lattice</i> | 0.22-6 | 2024-03-20 [3] CRAN (R 4.4.1) |
| ## | <i>lazyeval</i> | 0.2.2 | 2019-03-15 [2] RSPM (R 4.4.0) |
| ## | <i>lifecycle</i> | 1.0.4 | 2023-11-07 [2] RSPM (R 4.4.0) |

| | | | |
|----|-----------------------|-----------|--|
| ## | <i>limma</i> | 3.61.10 | 2024-09-22 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>lobstr</i> | * 1.1.2 | 2022-06-22 [2] RSPM (R 4.4.0) |
| ## | <i>locfit</i> | 1.5-9.10 | 2024-06-24 [2] RSPM (R 4.4.0) |
| ## | <i>magick</i> | 2.8.5 | 2024-09-20 [2] RSPM (R 4.4.0) |
| ## | <i>magrittr</i> | 2.0.3 | 2022-03-30 [2] RSPM (R 4.4.0) |
| ## | <i>maps</i> | 3.4.2 | 2023-12-15 [2] RSPM (R 4.4.0) |
| ## | <i>Matrix</i> | 1.7-0 | 2024-04-26 [3] CRAN (R 4.4.1) |
| ## | <i>MatrixGenerics</i> | * 1.17.0 | 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>matrixStats</i> | * 1.4.1 | 2024-09-08 [2] RSPM (R 4.4.0) |
| ## | <i>memoise</i> | 2.0.1 | 2021-11-26 [2] RSPM (R 4.4.0) |
| ## | <i>metapod</i> | 1.13.0 | 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>mime</i> | 0.12 | 2021-09-28 [2] RSPM (R 4.4.0) |
| ## | <i>msnsell</i> | 0.5.1 | 2024-04-01 [2] RSPM (R 4.4.0) |
| ## | <i>paletteer</i> | 1.6.0 | 2024-01-21 [2] RSPM (R 4.4.0) |
| ## | <i>pillar</i> | 1.9.0 | 2023-03-22 [2] RSPM (R 4.4.0) |
| ## | <i>pkgconfig</i> | 2.0.3 | 2019-09-22 [2] RSPM (R 4.4.0) |
| ## | <i>plotly</i> | 4.10.4 | 2024-01-13 [2] RSPM (R 4.4.0) |
| ## | <i>png</i> | 0.1-8 | 2022-11-29 [2] RSPM (R 4.4.0) |
| ## | <i>prettyunits</i> | 1.2.0 | 2023-09-24 [2] RSPM (R 4.4.0) |
| ## | <i>promises</i> | 1.3.0 | 2024-04-05 [2] RSPM (R 4.4.0) |
| ## | <i>purrrr</i> | 1.0.2 | 2023-08-10 [2] RSPM (R 4.4.0) |
| ## | <i>R6</i> | 2.5.1 | 2021-08-19 [2] RSPM (R 4.4.0) |
| ## | <i>rappdirs</i> | 0.3.3 | 2021-01-31 [2] RSPM (R 4.4.0) |
| ## | <i>RColorBrewer</i> | 1.1-3 | 2022-04-03 [2] RSPM (R 4.4.0) |
| ## | <i>Rcpp</i> | 1.0.13 | 2024-07-17 [2] RSPM (R 4.4.0) |
| ## | <i>RCurl</i> | 1.98-1.16 | 2024-07-11 [2] RSPM (R 4.4.0) |
| ## | <i>rematch2</i> | 2.1.2 | 2020-05-01 [2] RSPM (R 4.4.0) |
| ## | <i>restfulr</i> | 0.0.15 | 2022-06-16 [2] RSPM (R 4.4.0) |
| ## | <i>rjson</i> | 0.2.23 | 2024-09-16 [2] RSPM (R 4.4.0) |
| ## | <i>rlang</i> | 1.1.4 | 2024-06-04 [2] RSPM (R 4.4.0) |
| ## | <i>rmarkdown</i> | 2.28 | 2024-08-17 [2] RSPM (R 4.4.0) |
| ## | <i>Rsamtools</i> | 2.21.1 | 2024-08-16 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>RSQLite</i> | 2.3.7 | 2024-05-27 [2] RSPM (R 4.4.0) |
| ## | <i>rsvud</i> | 1.0.5 | 2021-04-16 [2] RSPM (R 4.4.0) |
| ## | <i>rtracklayer</i> | * 1.65.0 | 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>S4Arrays</i> | 1.5.8 | 2024-09-22 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>S4Vectors</i> | * 0.43.2 | 2024-07-17 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>sass</i> | 0.4.9 | 2024-03-15 [2] RSPM (R 4.4.0) |
| ## | <i>ScaledMatrix</i> | 1.13.0 | 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0) |
| ## | <i>scales</i> | 1.3.0 | 2023-11-28 [2] RSPM (R 4.4.0) |
| ## | <i>scater</i> | * 1.33.4 | 2024-07-21 [2] Bioconductor 3.20 (R 4.4.1) |
| ## | <i>scranc</i> | * 1.33.2 | 2024-09-06 [2] Bioconductor 3.20 (R 4.4.1) |

```

## scuttle * 1.15.4 2024-08-14 [2] Bioconductor 3.20 (R 4.4.1)
## sessioninfo 1.2.2 2021-12-06 [2] RSPM (R 4.4.0)
## shiny 1.9.1 2024-08-01 [2] RSPM (R 4.4.0)
## shinyWidgets 0.8.7 2024-09-23 [2] RSPM (R 4.4.0)
## SingleCellExperiment * 1.27.2 2024-05-24 [2] Bioconductor 3.20 (R 4.4.0)
## spam 2.10-0 2023-10-23 [2] RSPM (R 4.4.0)
## SparseArray 1.5.39 2024-09-23 [2] Bioconductor 3.20 (R 4.4.1)
## SpatialExperiment * 1.15.1 2024-06-20 [2] Bioconductor 3.20 (R 4.4.0)
## spatialLIBD * 1.17.8 2024-07-25 [2] Bioconductor 3.20 (R 4.4.1)
## statmod 1.5.0 2023-01-06 [2] RSPM (R 4.4.0)
## STexampleData * 1.13.3 2024-05-21 [2] Bioconductor 3.20 (R 4.4.0)
## SummarizedExperiment * 1.35.1 2024-06-28 [2] Bioconductor 3.20 (R 4.4.1)
## tibble 3.2.1 2023-03-20 [2] RSPM (R 4.4.0)
## tidyverse 1.3.1 2024-01-24 [2] RSPM (R 4.4.0)
## tidyselect 1.2.1 2024-03-11 [2] RSPM (R 4.4.0)
## UCSC.utils 1.1.0 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0)
## utf8 1.2.4 2023-10-22 [2] RSPM (R 4.4.0)
## uwot 0.2.2 2024-04-21 [2] RSPM (R 4.4.0)
## vctrs 0.6.5 2023-12-01 [2] RSPM (R 4.4.0)
## viror 0.4.7 2023-12-18 [2] RSPM (R 4.4.0)
## viridis 0.6.5 2024-01-29 [2] RSPM (R 4.4.0)
## viridisLite 0.4.2 2023-05-02 [2] RSPM (R 4.4.0)
## withr 3.0.1 2024-07-31 [2] RSPM (R 4.4.0)
## xfun 0.47 2024-08-17 [2] RSPM (R 4.4.0)
## XML 3.99-0.17 2024-06-25 [2] RSPM (R 4.4.0)
## xtable 1.8-4 2019-04-21 [2] RSPM (R 4.4.0)
## XVector 0.45.0 2024-05-01 [2] Bioconductor 3.20 (R 4.4.0)
## yaml 2.3.10 2024-07-26 [2] RSPM (R 4.4.0)
## zlibbioc 1.51.1 2024-06-05 [2] Bioconductor 3.20 (R 4.4.0)
##
## [1] /tmp/RtmpnrKRIw/Rinst8c95bc845
## [2] /usr/local/lib/R/site-library
## [3] /usr/local/lib/R/library
##
## -----

```

References

A Related resources

A.1 Overview

A.2 Data preprocessing procedures for the Visium platform

- [Visium data preprocessing](#)

A.3 Resources for other spatial omics platforms

- [Analysis workflow for IMC data](#)
- [VectraPolarisData](#)

A.4 Data structures

3

- [AnnData](#) scverse
- [SpatialData](#)

A.5 Statistical concepts

- [Modern Statistics for Modern Biology](#) chapter on image
data and spatial statistics

B Acknowledgments

B.1 Contributors

- Lukas M. Weber
- Leonardo Collado-Torres
- Stephanie C. Hicks

[GitHub contributors](#)

B.2 Additional acknowledgments

- [Orchestrating Single-Cell Analysis with Bioconductor \(OSCA\)](#)
- [Bioconductor Team](#) [Bioconductor](#) [Bioconductor Core](#)

References