UNIVERSITY OF PADUA

DEPARTMENT OF COMPARATIVE BIOMEDICINE AND FOOD SCIENCE

Master's Course in Biotechnologies for Food Sciences

Exploratory Data Analysis of imaging-based spatial transcriptomics: an application to a lung cancer sample

Supervisor: Prof. Davide Risso
Department of Statistical Sciences
University of Padova

Co-Supervisor: Prof. Benedetto Ruperti
Department of Agronomy, Food, Natural Resources, Animals and Environment
University of Padova

Co-Supervisor: Dr. Dario Righelli
Department of Electrical Engineer and Information Technologies
University of Naples "Federico II"

Undergraduate

Sanaz Akbari

Matricola: 2039467

ACADEMIC YEAR 2021/2024

Abstract

exploratory data analysis (EDA) is of the utmost importance in the domain of lung cancer imaging-based spatial transcriptomics, as it enables the comprehension of enormous datasets. Popular EDA techniques such as differential expression analysis, visualization, clustering, and dimensionality reduction were applicable to this study. The maintenance of data structure is facilitated by the dimension reduction process, while the identification of intriguing patterns and regions of gene expression is aided by visualization. To identify clusters of malignant cells in lung tissue samples, the utilization of clustering algorithms such as Louvain can provide significant benefits. A more comprehensive understanding of signalling pathways and biomarkers can be attained by employing differential expression analysis, a technique that quantifies the expression of genes in specific regions. Following workflow analysis, six genes were identified in the current study; of these, MALAT demonstrated the highest level of expression in cluster four, while CD163 exhibited the lowest level of expression in cluster one.

Table of Contents

1.Introduction

1.1. The relationship between cells and transcription

1.1.2. Cellular varieties

The prevalence of cellular heterogeneity within a population of a particular cell line has been demonstrated [1]. Numerous characteristics, functions, and activities of organ tissue are dependent on the presence of highly specialized cell types. Hence, the ability to isolate distinct cell populations from a single tissue and analyse the transcriptomes of individual cell populations from any given tissue is crucial for comprehending the mechanisms by which cells regulate development, growth, and stress adaptation. Different types of cells appear to be present in the bodies of multicellular organisms. It is estimated that the human body contains approximately 210 distinct types of cells [2]. Nevertheless, research indicates that even a solitary cell type is extraordinarily diverse, indicating that such variations may result in alterations in function, regulation, or morphology in response to environmental stimuli and gene expression. Cell types are regarded as the fundamental functional units of various organs, including the brain, lung, and muscle, in molecular biology [3]. With the intention of achieving this, the physiological behaviour, anatomical shape, and functional molecular properties of any given organ are extraordinarily diverse.

1.1.3. The RNA profile.

Gene expression or RNA profiling in molecular biology pertains to the real-time assessment of the collective activity of tens of thousands of genes. RNA profiling experiments frequently entail the assessment of the relative expression of RNAs that vary in expression across distinct developmental stages, organs, and even cell types within a single tissue. It is noteworthy that RNA profiling has evolved into an indispensable instrument for various purposes, including drug discovery. The utilization of expression profiling facilitated the correlation between genomic and omics data, particularly transcriptomic data.

Undoubtedly, the capacity to assess the worldwide expression of tens of thousands of genes in specific cell types within a single experiment will enable scientists to identify not only novel cell types but also differentially expressed genes (DEGs) that are integral to the experimental design. Broadly speaking, RNA profiling techniques have the capacity to surmount inherent constraints that are inherent in bulk cell measurements. In pursuit of this objective, computational frameworks that derive cell type-specific gene expression from RNA profiles are gaining popularity [4].

Notwithstanding the fact that innovative RNA profiling technologies can illuminate the distinction between cell types.

1.1.4. High-throughput omics of single cells

Omics is, in broad terms, the systematic observation and analysis of genes, transcripts, proteins, metabolites, and lipids [5]. The utilization of single-cell omics and other microscale molecular biological tools has enabled the investigation of specific cell types within a population of other cell types from an omics standpoint [6]. Owing to recent developments in single-cell omics data mining, aberrant molecular pathways linked to diseases like cancer cells have come to light. As an illustration, in the field of cancer research, single-cell genomic sequencing has identified uncommon mutations that are linked to the development of tumours [7, 8]. Despite the progress made in early omics technologies over the past few decades, the current focus of omics technologies is single-cell analysis, which involves observing genetic variations between individual cells as opposed to relying on the stochastic average obscured by global bulk measurements [5]. The scholarly community has been significantly transformed regarding the correlation between gene regulation mechanisms and cell type through the utilization of population-level expression measurements. Nevertheless, single-cell experiments have demonstrated that the RNA profile of specific cells can vary significantly from the average of the entire population [9].

1.1.4.1. The study of single-cell transcriptomics

The utilization of single-cell transcriptomics has brought about significant advancements in numerous domains of molecular biology since its initial application [10]. Following the work of Tang and Barbacioru [10], numerous additional protocols for single-cell transcriptomics surfaced, such as SMART-seq/SMART-seq2 [14, 15], STRT-seq [11], CEL-seq [12], MARS-seq [13], and MARS-seq [13]. These protocols varied in terms of amplification technology, transcript coverage, and the degree of liquid handling in plates being automated [16].

While its initial application was limited to mice, single-cell transcriptomic technology subsequently evolved into a valuable tool for comprehending cells and tissues in various model organisms, including zebrafish, nematodes, and others. Following the introduction of single-cell transcriptomic technology by nine years, two research groups issued high-throughput compilations of murine tissues [17, 18]. Research has demonstrated that single-cell transcriptomes offer a distinct advantage in terms of identifying rare cell types, including cancerous cells, and provide a high-resolution measurement of cell identity [19, 20]. Specific events that characterized the major developments in the study of omics data are summarized in Table 1.

Major events in evolution of prequel technique

1969	Radioactive ISH of rRNA
1973	Radioactive ISH of globin mRNA
1977	FISH of rRNA
1982	FISH of action mRNA-immunological FISH with biotin labelled prob
1987	Drosophila Enhancer Trap
1989	WM ISH in Drosophila-ES cell enhancer and gene trap in mice
1991	In situ reporter in C.elagans
Major prequel WM ISH atlases	
1994	Scaling up MW ISH in C.elagans
1995	First mouse MW ISH screening
1998	AXeIDB:1765 clones in Xenopus Laevis
1999	Mouse,GXD
2000	Halocynthia roretzi:MAGEST
2001	C.elegans NEXTDB-Ciona intestinalis:Ghost
2002	Gene paint-Melanogaster BDGP in situ
2003	Zebrafish: FIN- Oryzias Latipes:MEPD
2004	Kitchen: GEISHA
2005	First miRNA atlas
2006	Allen brain atlas-BDTNP: toward single cell resolution
2007	Xenopus Laevis: Xenbase- Fly FISH: Drosophila mRNA at subcellular level
2011	Mouse genitourinary tract: GUDMAP
2017	Hamn and mouse lung: Lung map
2020	T.guttata:ZEBrA
Major events in evolution of current-era technique	
1976	LCM
1989	Single cell cDNA amplification
1987	Ligase SNV detection

1989	FISH with combinatorial barcode
1995	cDNA microarray
1996	Commercial LCM
1998	smFISH- Solexa founded
1999	LCM+microarray
2002	Combinatorial FISH for mRNA
2008	RNA-seq
2012	Tommy-array mouse brain
2013	High throughput RCA+ISS
2014	seqFISH
2015	MERFISH
2016	Spatial transcriptomics
2019	GeoMX DSP

Table 1 Table 1 timeline major events including: Major events in evolution of prequel technique, Major prequel WM ISH atlases, and Major events in evolution of current-era technique[74].

1.1.5. Spatial transcriptomics

The spatial arrangement of a particular cell within a distinct tissue type, in comparison to adjacent cells and extracellular structures, offers valuable insights into the determination of cellular phenotype, tissue function, and cell fate. Novel spatial transcriptomics techniques exhibit significant potential in the concurrent profiling of thousands to hundreds of genes at the subcellular level. While spatial transcriptomics does produce cellular transcriptomes and spatial information pertaining to those transcriptomes within a specific tissue type, it does not produce data on individual cells. To examine the relationship between cell differentiation and morphogenesis, Mantri and Scuderi [21] utilized spatial transcriptomics in conjunction with single-cell technique. Although spatial transcriptomic methods have been in use for nearly a decade, their commercialization has only occurred more recently. Spatial transcriptomics has been rendered more accessible through the availability of several commercial spatial RNA-seq technologies, including the 10X-Visium (10X Genomics), CosMx (Nanostring Technologies), and MERSCOPE (Vizgen).

Spatial transcriptomics has thus far been applied to the characterization of specific cells and cell types. Grauel and Nguyen [22] conducted research on breast cancer by employing droplet-based single-cell RNA-seq to profile cancer sections from diverse clinical subtypes and using Visium. This study's findings regarding gene expression led to the identification of tissue regions that correspond to stromal, immune, and tumor cells. It is noteworthy that the computational data corroborated

pathologist annotations satisfactorily. Thousands of genes were characterized with this data without the need for manual annotation.

Multicellular organisms comprise an extensive variety of cell types that construct organs and tissues in a spatially diverse and well-defined body plan [23]. To gain access to multicellular systems, it is necessary to comprehend the processes that spatial heterogeneity governs [24]. Over the past thirty years, numerous in situ and in vivo techniques have been created with the purpose of deciphering spatial biological information [25]. In the past, subcellular labeling of nucleic acids or proteins required the use of complementary nucleotide probes or antibodies that were enzymatically or fluorescently linked [26]. While these techniques have significantly advanced our comprehension of fine-scale cellular events by elucidating biological processes [25, 26], their restricted throughput and inferior resolution when compared to sequencing-based methods prevent them from revealing the subcellular location of a target [25]. In addition, certain in situ techniques necessitate tissue preparations, and the simultaneous testing of a limited number of molecular markers is feasible [25, 26]. Spatial transcriptomics methods based on next-generation sequencing (NGS) have undergone significant advancements in recent decades [25].

Pioneer technologies label cells with fluorescent markers so that they may be extracted and sequenced [25, 27]. One significant drawback of these approaches, nevertheless, is that the spatial information of cells within a given tissue is not preserved during the bulk photoactivation of all cell types [27]. This complicates the task of precisely locating a particular cell type and deducing its spatial information. Initially, spatial information was captured at the level of individual cells using laser capture microscopy (LCM) in conjunction with next-generation sequencing (NGS) techniques [28, 29]. However, a significant obstacle for LCM technologies is that they necessitate the use of complex and sophisticated equipment in addition to the laborious removal of tissue components [30]. Recent developments in spatial transcriptome sequencing enable scientists to investigate a more extensive region albeit at a reduced level of detail. In lieu of oligonucleotides, spatial transcriptomics was recently enhanced by substituting high density bead arrays with position index barcoded beads [31, 32].

1.1.6. Spatially resolved transcriptomics.

Spatially resolved transcriptomics (SRT) refers to a class of high-throughput technologies that compute the spatial coordinates of gene expression measurements at the transcriptome level [33]. SRT methodologies vary regarding both the quantity of genes assessed and the resolution of their

spatial parts. In general, resolved omics techniques quantify the abundance of transcripts while preserving positional information [25]. At present, there exists an extensive array of proposed applications pertaining to the imaging of single-cell transcriptomes. SRT technologies have been implemented in numerous biological systems to date. Notable among the numerous published studies are those that examine the application of SRT methods to the mouse brain [35], the human brain [34], cancer [36, 37], and mouse embryogenesis [38, 39].

1.1.6.1. Spot-based platforms

Spot-based platforms, such as sci-Space, 10x Genomics Visium, Spatial transcriptomics, and Slide-seqV2, capture transcriptome-wide gene expression in a series of spots on a tissue slide, the positions of which include spatial coordinates. These platforms aimed to quantify the transcriptome's overall gene expression profile across multiple locations. RNA-seq analysis can be carried out at multiple precisely located spots on the surface of a histological slice with any spot-based platform.

1.1.6.2. Molecule-based platforms

This does not apply to spot-based technologies, which are based on molecules like MERFISH, seqFISH, and osmFISH. This is accomplished by combining sequential barcoding and in situ molecular fluorescence probing; additionally, these technologies are based on molecules, and each experiment requires a pre-defined transcript panel. These platforms achieve resolutions down to the subcellular level. These methodologies allow for the simultaneous determination of the spatial coordinates of each transcript with a resolution of micrometers while also identifying thousands of mRNAs. Molecular-based platforms are extremely effective tools that allow researchers to predict the subcellular resolution structures of RNA molecules. These technologies are currently used to represent the spatial distribution of desired gene expression in a variety of biological systems, including cancer cells, brain tissues, and embryonic stages of development.

1.2. Single-cell construction

To analyze data sets derived from the aforementioned technologies, it is critical to devise techniques for collecting, storing, retrieving, and processing transcriptomic data for subsequent applications. There are several distinct approaches available for the analysis of single cell RNA sequencing (scRNA-seq) data. However, these approaches diverge significantly from the methods utilized by bulk RNA-seq analysis platforms. Significant variations in analysis platforms appear to be due to a variety of technical factors, including the prevalence of background noise [40, 41] and a small amount of extracted RNA from a single cell. In recent times, the progress made in multimodal single-cell

technologies has necessitated the creation and refinement of innovative computational algorithms that can effectively integrate data from various types [42].

1.3. Spatial Experiment

High-dimensional spatially resolved omics amass an enormous quantity of data when compared to alternative single-cell "-omics" methodologies. Analyzing such data sets is consequently becoming increasingly difficult. In general, the analysis of spatial datasets necessitates the development of infrastructures and the completion of numerous processing steps. The continuous progress of SRT techniques surpasses the development of bioinformatic algorithms utilized in data analysis by a significant margin.

The spatial resolution is established using the most prevalent SRT method [43] in Visium technology, which employs spots (55 µm in diameter) as individual capture elements. An SRT method is employed to acquire a bright field image of the tissue's characteristics. Following this, morphological data are converted to their corresponding mRNA transcript levels via mapping. As of now, a variety of libraries and applications are accessible for the purpose of examining and investigating Visium data [33, 44-46]. Among these applications, the Seurat R package is the most popular For example, the Seurat [47] and Giotto [48] packages for R, as well as Squidpy [45] and AnnData [49] packages for Python, provide enhanced capabilities for the storage and annotation of data in the form of measurement value tables, in addition to the recording of pertinent spatial and image information. While the Seurat R package provides fundamental functionality for various SRT platforms, it still lacks interoperability with other tools. Numerous efforts have been devoted to addressing the limitations of the analysis.

Within the R/Bioconductor framework, SpatialExperiment, a novel data infrastructure designed to store and retrieve spatially resolved transcriptomics data, is implemented. It offers modularity, interoperability, standardized operations, comprehensive documentation, and additional benefits. SpatialExperiment has been developed autonomously and is compatible with all subsequent analysis packages in Bioconductor that utilize the SingleCellExperiment or SpatialExperiment class [33]. This contrasts with previous infrastructures. Consequently, analysts are readily modifiable using additional packages created by diverse research groups. Additionally, it exhibits compatibility with a wide range of methodologies that utilize single-cell data and have been made available by Bioconductor.

1.4. Cancer of the lung and transcriptomics

Cancer is the second leading cause of death in the United States and a major and progressively significant public health concern worldwide, according to the American Cancer Society [50]. Recent cancer statistics indicate that prostate cancer, breast cancer, lung cancer, bronchus cancer, and colorectal cancer are the most frequently diagnosed malignancies in humans (CRCs). In 2023, lung cancer and colorectal cancer will comprise 52% of all newly diagnosed cases [50]. Lung cancer is the primary cause of mortality for individuals aged 50 years and older, surpassing the combined fatality rates of breast, prostate, and colorectal cancer (CRC) [50].

Utilizing single-cell technologies, the intricate characteristics of tumor immune cell types have been uncovered [51-54]. Prior to recent times, the characterization of tumor subtypes through histopathological methods was the prevailing approach in clinical practice. This approach failed to account for the spatial context of single-cell types within stratified tissues. The correlation between the function of distinct immune cells and their spatial location within a complex tumor has been established [55-57]. In pursuit of this objective, illuminating the spatial distribution of lung cancerous lesions could yield valuable high-resolution data that can be utilized to track the advancement of the disease and aid in the development of innovative therapeutic approaches and biomarker-responsive therapies [58].

The utilization of high-resolution single-cell transcriptomes has enabled the differentiation of various cell types within the tumor microenvironment (TME) [59, 60]. Numerous cancer-related investigations have utilized single cell transcriptome associated analysis: head and neck cancer [66], breast cancer [59], lung cancer [61, 62], liver cancer [63], colorectal cancer [64], and melanoma [65]. For the purpose of gaining insight into the progression of cancer and drug resistance, Karacosta and Anchang [67] employed time-course analysis of mass cytometry. Significant disparities were identified in the trajectories of EMT and MET through the utilization of TRACER, a computational tool.

As demonstrated by single-cell RNA sequencing technologies [68], lung cancer tumors contain a variety of immune cell populations and are heterogeneous. Zhang and Sun [69] demonstrated, via spatially resolved transcriptomics, that small cell lung cancer tumors are heterogeneous both inter and intratumor. An investigation into lung adenocarcinoma using 10Visium SRT technology revealed that the invasive process of lung adenocarcinoma is facilitated by the UBE2C+ cancer cell subpopulation [70]. A different SRT technology was employed by [71] to evaluate non-small cell lung cancer tumors in the same year. It was determined that patients exhibiting CD163+ tumorous cells not only maintain a greater distance from the tumor cells but also experience reduced infiltration levels and a prolonged survival rate. Collectively, these studies demontrate that SRT technologies

have the ability to distinguish between cells containing lung tumors and that it is possible to process data using various spatial approaches.

The storage of gene expression data presents a formidable obstacle for nascent single-cell SRT technologies. Gene expression information is exclusively stored at the cell or spot level in the recently introduced SpatialFeatureExperiment [72], as opposed to SingleCellExperiment and SpatialExperiment. In order to take advantage of the molecule-level resolution capabilities of contemporary SRT technologies, it is crucial to analyze transcripts in their spatial positions, regardless of cellular compartmentalization, and to prevent premature summarization of ST data. The SpatialExperiment data class is the primary [33]. This class facilitates the storage of datasets at either the spot or cell level. For instance, it can be used to aggregate data from molecule-based platforms at the cell level or data from sequencing-based platforms at the spot level. For single-cell RNA sequencing data, SpatialExperiment extends the SingleCellExperiment class [73] by including attributes for storing spatial information, including image files and spatial coordinates.

The following diagram provides a concise overview of the structure of the SpatialExperiment object. I assays, which contain expression counts; II rowData, which contains feature information such as genes; III colData, which contains spot or cell information including nonspatial and spatial metadata; IV spatialCoords, which contains spatial coordinates; and V imgData, which comprises image data. The primary emphasis of this research is on lung cancer CosMx datasets for molecular-based spatial transcriptomics that encompass supplementary data, including the spatial coordinates of individual mRNA molecules as well as the boundaries delineating cells or nuclei. The subsequent Bioconductor classes offer supplementary functionalities for the storage and manipulation of the aforementioned data. These SpatialExperiment-extending classes are applicable to aggregated cell analyses.

1.4.1 CosMx lung cancer data

Those interested in investigating non-small cell lung cancer (NSCLC) using advanced spatial transcriptomics technology will discover the CosMX dataset to be an indispensable resource. The CosMX dataset is generated by employing the CosMx SMI (Spatial Molecular Imager), a sophisticated instrument recognized for its capability to capture images of individual cells within formalin-fixed paraffin-embedded (FFPE) tissues. Tissue samples are frequently preserved and archived utilizing these tissues. This investigation focuses on non-small-cell lung cancer. Although this open-source dataset was acquired from the Nanostring website(https://nanostring.com/inaccessible/), we employ it to examine the intricate intricacies of NSCLC tissue.

Chapter2

Aim of study

When conducting an Exploratory Data Analysis (EDA) on spatial transcriptomics data from imaging in a lung cancer sample, consider the experimental design, highlight cancer-related genes, examine gene expression spatial patterns, integrate clinical annotations into clustering analysis, analyze differential expression compared to normal tissue, integrate with other omics data, and validate and interpret the results. These steps help identify potential disease-associated hallmark genes or biomarkers, identify spatially discrete subpopulations of tumors, and ensure the dependability and reproducibility of the results. It is essential to note that EDA is an exploratory step, and more rigorous analysis techniques like machine learning or statistical modeling should be used to further investigate the identified patterns or biomarkers in the lung cancer sample.

2.1. Quality control

2.1.1. overview

Cellular quality control (QC) procedures are designed to eliminate substandard cells before they are analyzed. We will present the data structure for the 1000-plex CosMx™ RNA assay, which will help develop downstream analysis pipelines. The 1000-plex CosMx Human Universal Cell Characterization panel aims to provide detailed information on cell states, signaling interactions, hormone activity, and microenvironments in both healthy and diseased human organs.

The lung is critical to human health, so this study will use lung CosMx data to examine the diversity and spatial features of individual lung cell types, as well as the molecular and cellular processes that occur in both healthy and diseased tissue.

2.1.2. load data

To start, we load the following libraries:

SpatialExperiment

The Spatial Experiment package provides functionalities for manipulating and analyzing spatially resolved transcriptomics data. It includes features for manipulating spatial transcriptomics data acquired through imaging.

scater

The scater package can help with preprocessing and quality control for single-cell RNA-seq data. Although it is not inherently linked to spatial transcriptomics, it can provide useful data manipulation and analysis capabilities.

scran

The scran package focuses primarily on normalization and batch effect correction methods for single-cell RNA-seq data. It may offer useful functionalities for the preprocessing stages of your analysis.

ggspavis

The ggspavis package's tools enable the exploration and visualization of spatial transcriptomics data. This tool can generate spatial maps and plots to investigate gene expression patterns across a tissue.

Then we load the specified lung cancer data by its name that is in RDS format. Thus, the *head()* function can be used to display the first few rows of the column metadata *(colData)* of a *SpatialExperiment* object *(spe).* (chapter3, table:2)

```
spe <- readRDS("C:/Users/sanaz/OneDrive/Documents/cosmx_lung_5_rep1_SPE.RDS")
```

```
head(colData(spe))
```

2.1.3. Plot data

Plot the spatial coordinates in x-y dimensions on the tissue slide as a preliminary check that the object was loaded correctly and in the expected orientation. Plots are created using the visualization functions provided by the *ggspavis* package. Here's the code:

```
plotSpots(spe, in tissue = NULL)
```

Using the *SpatialExperiment* object *spe*, the *plotSpots()* function is used to show the spatial distribution of cells.

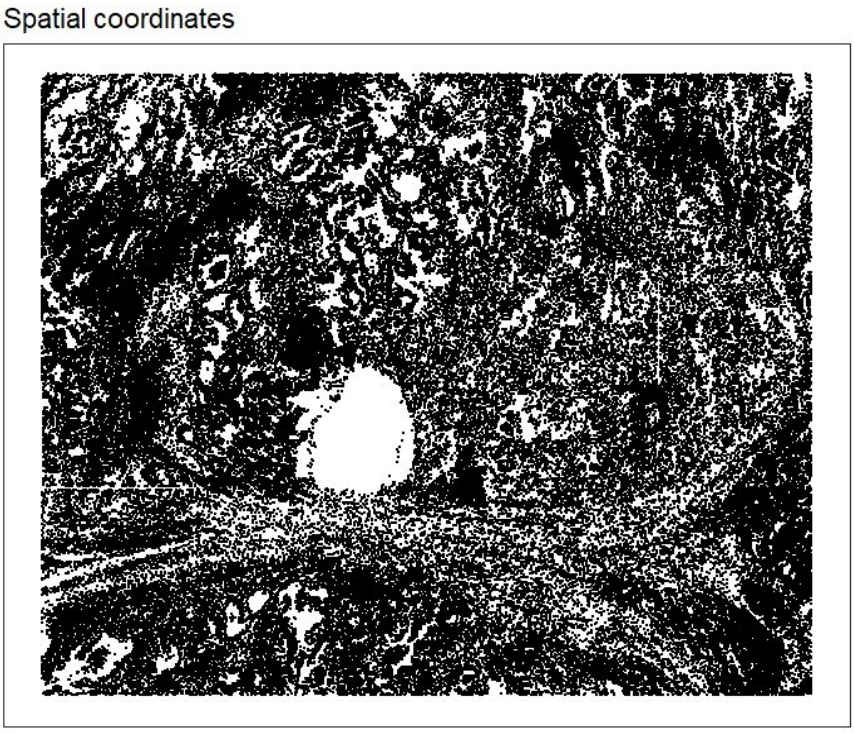

Figure 1 coordinates in the x-y plane relative to the tissue slide

2.1.4. Calculate QC metrics.

The QC metrics described above are calculated using a combination of techniques from the *scater* [75] package. Following that, the QC metrics obtained from *scater* can be calculated and added to the *SpatialExperiment* object. Then compute and store the per-cell QC metrics in *colData* (*"sum"* and *"detected"*). In R, per-cell quality control metrics are calculated by calling the *addPerCellQC()* function on a *SpatialExperiment* object. Per-cell QC metrics can be used to assess the quality of specific cells in a spatial transcriptomics dataset. To update the *SpatialExperiment* object *spe* with the computed QC metrics and use the *addPerCellQC()* function, use the following code:

```
Spe <- addPerCellQC (spe)
```

Following the completion of this code, the *spe* object will receive additional per-cell quality control metrics. The *per.cell.qc* component of the *spe* object's *colData* will provide access to these metrics. Metrics such as total counts, number of detected genes, and other customizable measures can provide useful information about cell quality.

Notably, the *addPerCellQC()* function is part of the *scater* package; therefore, ensure that the *scater* package is loaded before calling this function.

The *head()* function displays the first few rows of a *SpatialExperiment* object's column metadata *(colData) (spe).* The data set now includes the outputs "*Detected*", *"Sum",* and *"Total."*(chapter3, table:3)

```
head(colData(spe))
```

The histograms of QC metrics were as follows:

The provided code creates an illustration consisting of two histograms placed next to each other. The first histogram depicts the distribution of molecules counts per cell, while the second shows the distribution of detected genes per cell. The code goes as follows:

```
par(mfrow = c(1, 2))

hist(colData(spe)$sum, xlab = "sum", main = "molecules per cell")

hist(colData(spe)$detected, xlab = "detected", main = "Genes per cell")
```

The provided code establishes the layout of the plot with one row and two columns *(par(mfrow = c(1, 2)))*. The initial invocation of the *hist()*function generates a histogram of the *''sum ''* column extracted from the *colData* of the *spe* object. The x-axis is annotated as *''sum ''*, and the primary heading is specified as molecules *per cell*. The histogram of the *''detected ''* column from the *colData*

of the *spe* object is generated through the second invocation of the *hist()*function. The main title of the plot is ‘’*Genes per cell*‘’, and the x-axis is annotated as ‘’*detected* ‘’.

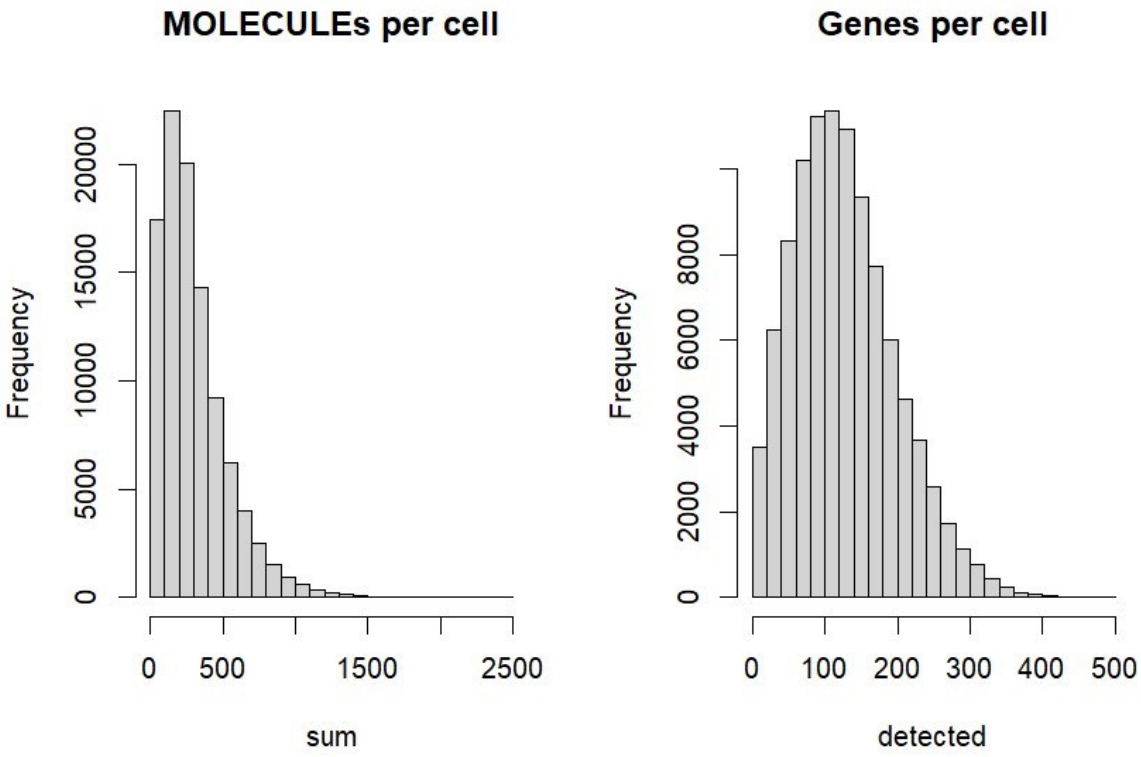

Figure 2 Histograms of molecules per cell and genes per cell with detected information annotated along the x-axis.

2.1.5. Selecting thresholds

The most straightforward way to apply the QC metrics is to set thresholds for each metric and then remove any cells that do not meet the thresholds for one or more metrics. Exploratory visualizations can aid in the selection of appropriate thresholds, which will vary depending on the dataset.

We use visualizations in this section to select thresholds for several QC metrics in our dataset: I the size of the library; (ii) the number of expressed genes (or features).

2.1.5.1. Thresholds for library size (“sum”)

The library size is calculated by summing the molecule counts for each cell. This data is stored in the *‘’sum‘’* column of the *scater* output.

Histogram of library size:

Here's the code:

par(mfrow=c(1,1))

In R, the *par()* function is used to change the graphical parameters of the current plotting device. *Par(mfrow=c(1,1))* in the code you shared configures the plotting device to display only one row and one column of plots. This means that any subsequent plots will be displayed in a single plot region, *hist(colData(spe)$sum, xlab = "sum", breaks=100, main = "Molecule per cell")*

The code you provided uses R's *hist()* function to create a histogram from the values in the ‘’*sum ‘’* column of the *colData(spe)* object. The *xlab* argument specifies the x-axis label as ‘’*sum ‘’*, the breaks argument specifies the number of histogram bins (100 in this case), and the main argument specifies the plot's main title as ‘’*Molecule per cell ‘’*.

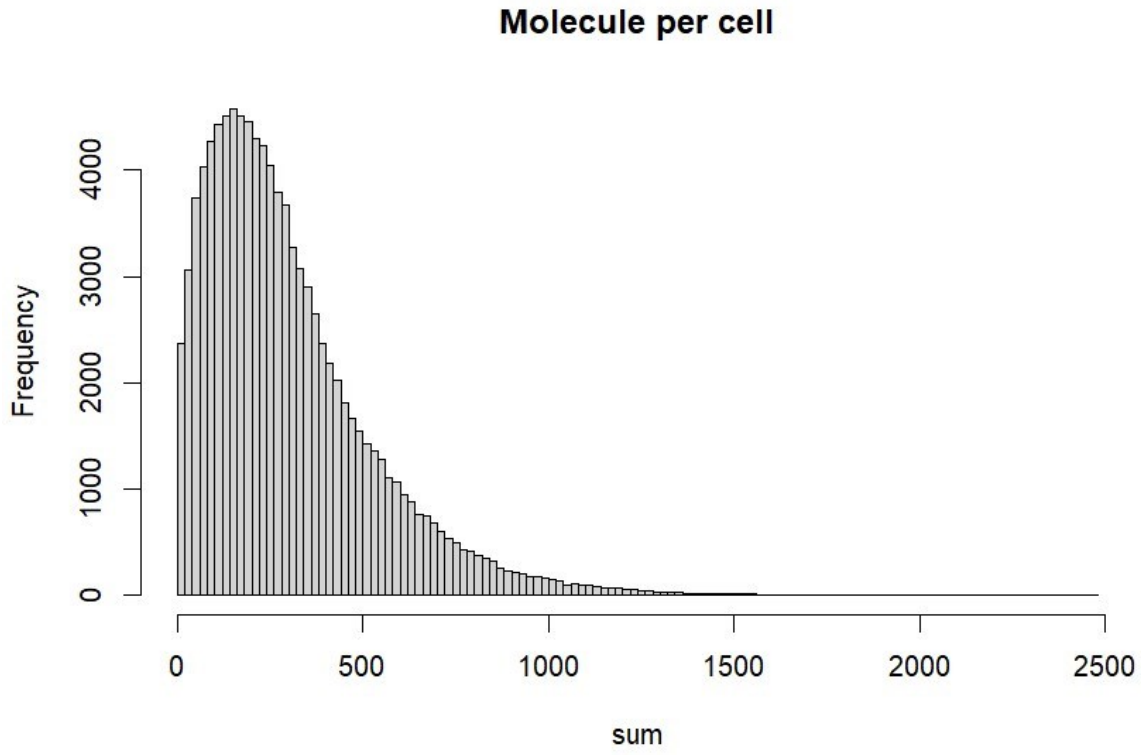

Figure 3 histogram Cellular quantity denoted by the x-axis as a sum.

There are no obvious issues with the distribution, such as a significant increase or decrease at very small library sizes.

We also plot the library size versus the number of expressed genes. This ensures that we do not accidentally remove a biologically significant group of cells. Based on the histogram, the horizontal line (argument threshold) represents our initial guess at a possible filtering threshold for library size.

```
plotQC(spe, type = "scatter",
    metric_x = "detected", metric_y = "sum",
    threshold_y = 10)
```

The *plotQC()* function is most likely a custom function that generates quality control plots based on the arguments passed to it. It's a scatter plot that shows the relationship between the *detected* metric on the x-axis and the *''sum ''* metric on the y-axis. A threshold line is also being added at *y = 10* (chapter 3, figure:16).

Based on the histogram, the horizontal line (argument threshold) represents our initial guess at a possible filtering threshold for library size.

Setting a filtering threshold for library size (for example, at the value shown) does not appear to select for any obvious biologically consistent group of cells, as evidenced by the plot.

A relatively arbitrary threshold of ten molecule counts per cell is established. Use the following code to set the QC threshold based on the library's size:

```
qc_lib_size = colData(spe)$sum < 10
table(qc_lib_size)
```

To generate the logical vector *qc _library* size, run the following code: Qc library *size = colData(spe)$sum < 10*. This vector shows whether the values in the *''sum ''* column of the *colData(spe*) object are less than 10. The resulting vector will have TRUE values if the condition (sum 10) is met and FALSE values otherwise. The code table then generates the frequency table *(qc_ lib_ size)*, which counts the number of times each distinct value appears in the *qc_ lib_ size* vector. The sum of TRUE and FALSE values in the vector will be shown. So, with this description, the result is 99185 FALSE and 1107 TRUE.

Use the following code to assign the contents of the *qc _lib _size* vector to a newly created column named "*qc_ lib _size*" in the *colData(spe)* object: ***colData(spe)$qc_ lib_ size = qc_ lib _size*** adds a column to the *spe* object's metadata, with each row representing a value from the *qc_ lib _size* size vector.

Finally, to ensure that the discarded cells lack any discernible spatial pattern that corresponds to known biological characteristics. If not, the absence of these cells may indicate that the threshold was set too high, causing the loss of biologically significant cells. For that reason, Implementing code Probably a custom function, the *plotQC()* method generates quality control plots based on the

arguments passed to it. The plot denoted by type = "*spots*" will visually represent the dispersion of various metrics across multiple cells; even if we are indicating "spots" in the function type argument the plot still works because we previously summarized the coordinates information by cell. The cells designated by discard = "*qc_ lib _size* " will be excluded from the plot.

```
plotQC(spe, type = "spots",in_tissue = NULL, discard = "qc_lib_size")
```

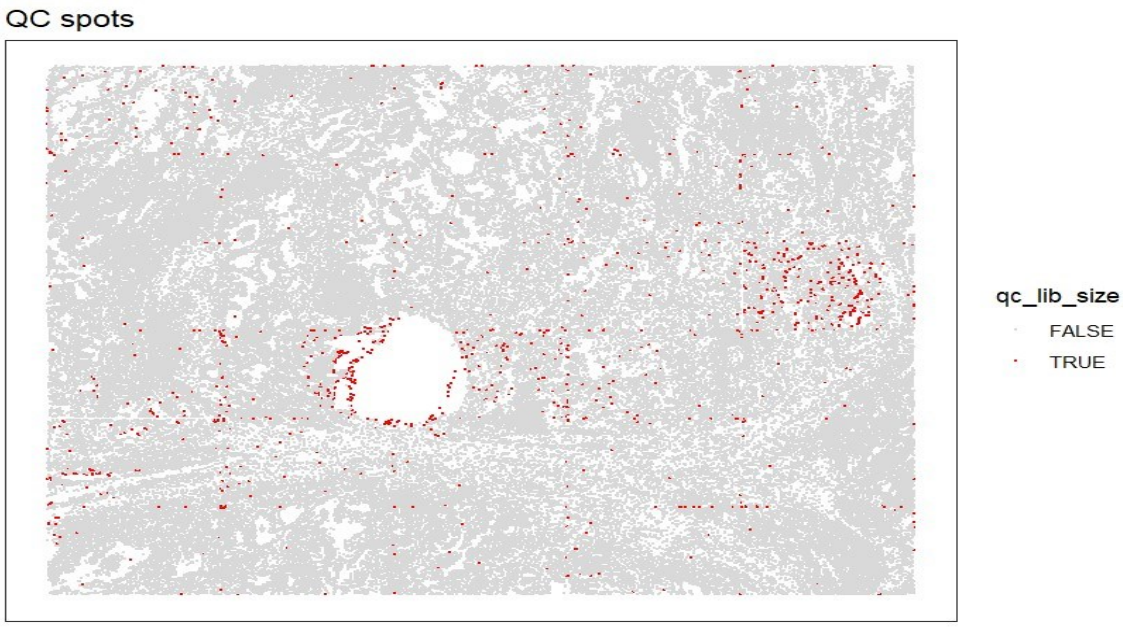

Figure 4 The discarded cells are spatially devoid of any particular pattern.

As illustrated, the discarded cells lack any discernible spatial pattern.

2.1.5.2. Thresholds for Number of expressed genes ("detected")

The number of genes with non-zero molecule counts per cell is the number of expressed genes. This is stored in the column identified in the *scater* output. A comparable set of visual representations is used to establish a threshold for this quality control metric. Initial requirements include a histogram of the detected value.

```
Hist(colData(spe)$detected, xlab = "detected", breaks=100, main = " detected value ")
```

The *hist ()* function creates a histogram of the values in the "*detected*" column of the *colData(spe)* object for further clarification. The *xlab* argument is used to designate the x-axis as *"detected,"* while the breaks argument specifies the number of bins in the histogram (100 in this instance). In contrast, the main argument establishes "*detected value"* as the plot's main title.

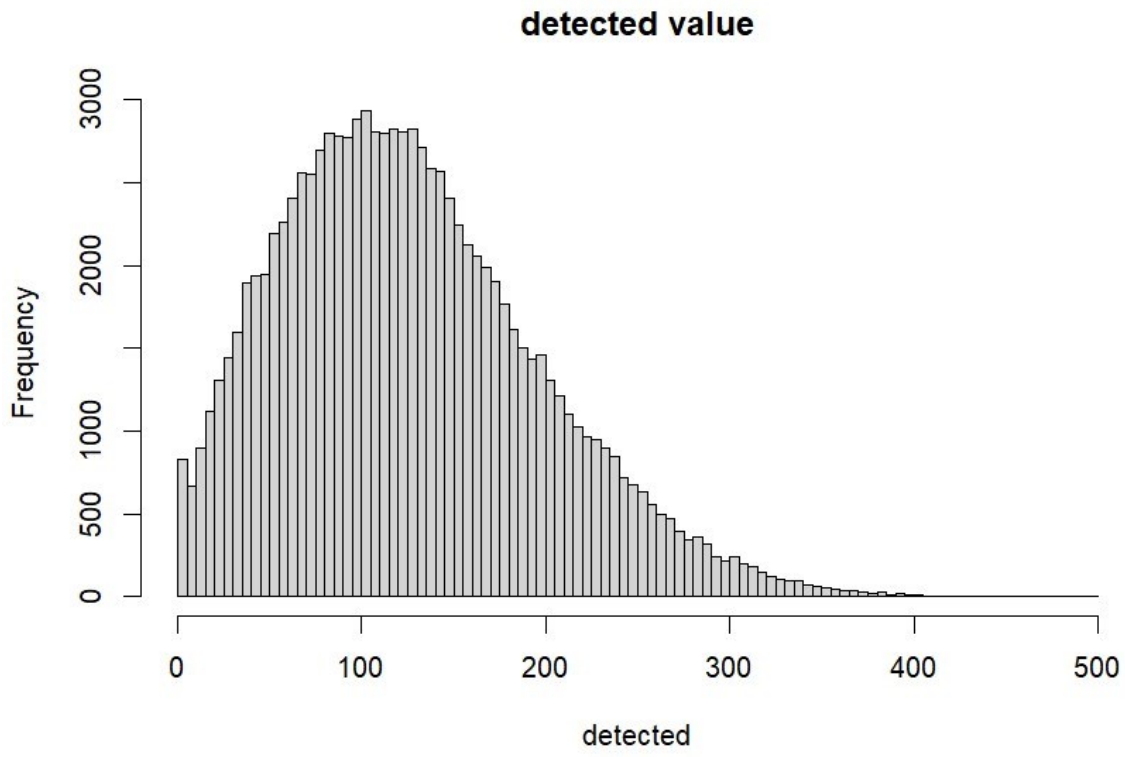

Figure 5 Gene histogram by cell

Second, regarding threshold 10, the following code is used:

```
plotQC(spe, type = "scatter",
    metric_x = "sum", metric_y = "detected",
    threshold_y = 10)
```

The *plotQC()* function generates quality control plots using the arguments passed to it. It's likely that this is a custom function. The scatter plot is used to show the relationship between the *detected* metric (y-axis) and the *sum* metric (x-axis). Furthermore, at *y = 10*, a threshold line is introduced. (chapter 3, figure:17)

To determine the QC threshold for the number of expressed genes:

```
Qc_ detected = colData(spe)$detected < 10
 table(qc_ detected)
```

To check if the values in the *detected* column of the *colData(spe)* object are less than 10, use the code *qc_ detected = colData(spe)$detected < 10*. Creates the logical vector *qc_ detected*. If the condition is met *(detected < 10)*, the vector will include TRUE values. Otherwise, it will return false results. After counting how many times each unique value appears in the *qc_ detected* vector, the code table *(qc_ detected)* generates a frequency table. The function returns the sum of TRUE and FALSE values in the vector. As a result, the answer will be 1353 TRUE and 98939 FALSE.

Finally, calling ***colData(spe)$qc_ detected = qc_ detected*** updates the *colData(spe)* object with the contents of the *qc_ detected* vector into a new column named *qc_ detected.* Each row of this column in the *spe* object's metadata corresponds to a value extracted from the qc detected vector.

```
PlotQC(spe, type = "spots", tissue=NULL, discard="qc _detected")
```

The *plotQC()* method, which is most likely a custom function, generates quality control plots based on the arguments provided. The plot indicated by *type = "spots"* will refer to that part the distribution of different metrics across numerous cells. Despite the absence of these metrics in certain spots, the plot remains applicable to cells. Points that contain the value TRUE in the *qc_ detected* column will be excluded from the graph.

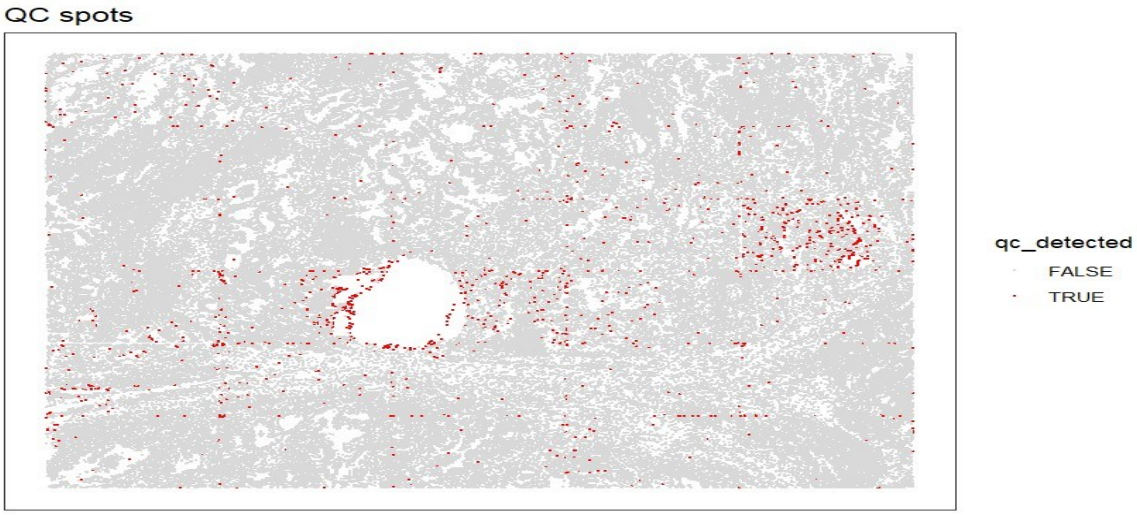

Figure 6 Quality control plots in relation to the given arguments

2.1.5.3. Remove low-quality cells.

Having determined thresholds for each of the calculated QC metrics, we can now combine the sets of low-quality cells and eliminate them from the object. Additionally, we verify once more that the assemblage of discarded cells does not appear to be any discernibly significant group of biological cells.

We establish codes with the following interpretation:

A) Number of discarded cells for each metric:

```
apply(cbind(qc_lib_size, qc_detected), 2, sum)
```

To apply a specified function across rows or columns of a matrix or data frame, use the apply function. In this case, the *cbind()* function combines two columns *(qc_ lib _size and qc_ detected)* to form a single matrix. The sum of each column in the new matrix is then calculated using the apply function. In the apply function, the value 2 indicates that the operation should be carried out column by column. Apply returns the sum of each column in the *qc_ lib _size* and *qc_ detected* columns. The resulting values will be 1107 *qc_ lib _size* and 1353 *qc_ detected*.

B) Combined set of discarded cells:

```
discard = qc_lib_size | qc_detected

table(discard)
```

To generate the discard variable, perform a logical OR (|) operation on the values *qc_ lib_ size* and *qc_ detected*. This will create a new variable with the value FALSE unless either *qc_ lib_ size* or *qc_ detected* is TRUE. Use the table function to create a table that summarizes the number of TRUE and FALSE values in the discard variable. This operation will return a two-row table containing the frequency of TRUE and FALSE values in the discard variable. The discard output would be 98939 false and 1353 true.

C) Store in object:

```
colData(spe)$discard = discard
```

The values of the discard variable will be appended to the discard metadata column of *spe*.

D) Check spatial pattern of combined set of discarded cells:

```
plotQC(spe, type = "spots",in_tissue = NULL,discard = "discard")
```

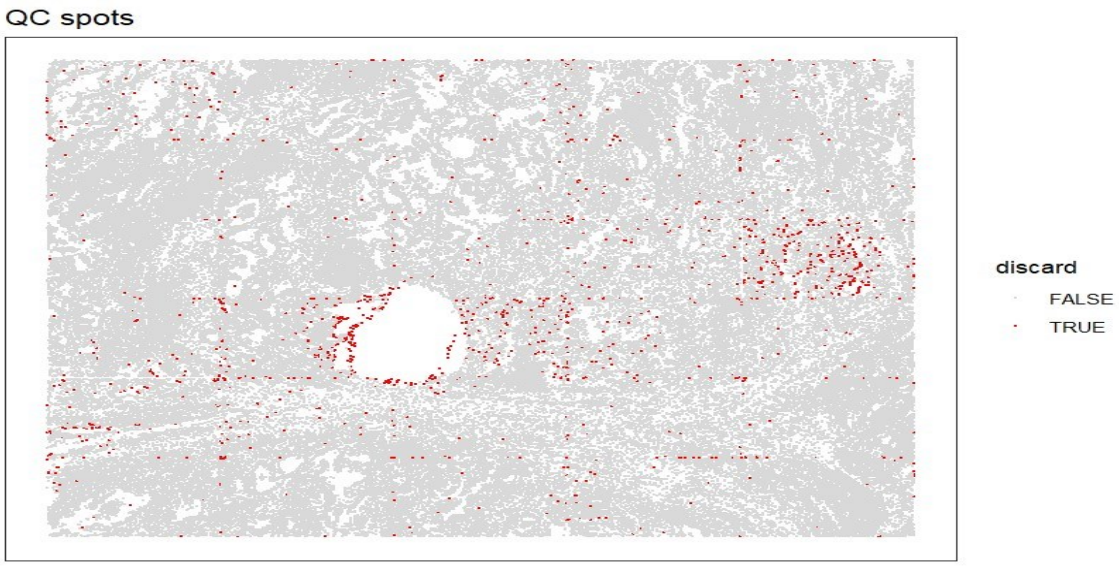

Figure 7 Eliminated cells.

Use the *plotQC* function to create quality control plots for spatial transcriptomics data using images stored in the *spe* object. The term *"spots"* With the exception of the complete absence of spots, the plot retains its applicability to all cells., with *in_ tissue = NULL* (this value can be left blank if we want to plot every cell) and *discard = "discard"*; this parameter specifies that the *spe* object's discard metadata column should be used to filter the cells before plotting. This assumes that the discard metadata column contains boolean values (TRUE or FALSE) and is located in the *spe* object's *colData*.

F) Remove combined set of low-quality cells:

```
spe = spe[, !colData(spe)$discard]

dim(spe)
```

To subset the *spe* object, samples with a TRUE value in the discard metadata column are excluded.

By subsetting the *spe* object, this code only keeps samples for which the discard metadata column returns FALSE. In other words, the! operator nullifies logical values. If discard is set to FALSE, *colData(spe)$discard* returns TRUE for the specified sample. Using the dim function, one can examine the dimensions of the updated *spe* object after subsetting. This function returns the number of cells and genes that remain in the *spe* object after removing samples with TRUE discard. The outpots would be 980 and 98939.

2.2. Normalization

2.2.1. overview

When analyzing gene expression data, normalization is a crucial step that accounts for systematic or technical variations between samples. The objective is to establish comparability in the expression levels of genes among samples, which will facilitate significant comparisons and subsequent analysis.

2.2.2. Logcount

"Logcount" denotes the logarithm of count data, accounted for technical considerations and sample comparability, in gene expression analysis. Variance is stabilized through a logarithmic transformation, which diminishes outliers and produces a more symmetrical distribution. Subsequent analyses that assume normality or equal expression values are facilitated by this. The methods implemented in the *scater* and scran[2] packages are utilized.

To achieve this, begin by eliminating any locations with zero counts. Subsequently, compute the library size and histogram. Finally, compute *logcounts* and store the results in an object. (chapter3, figure:18)

```
spe = spe[, colSums(counts(spe)) > 0]

spe = computeLibraryFactors(spe)

summary(sizeFactors(spe))
```

output:

Min	1st Qu	Median	Mean	3rd Qu	Max
0.03259	0.45293	0.82114	1.00000	1.34251	8.06480

```
hist(sizeFactors(spe), breaks = 100)
```

Logcounts are computed and stored in an object.

```
spe = logNormCounts(spe)
```

for verifying, the function *assayNames(spe)* retrieves the names of different assay types in the *SingleCellExperiment* object *spe*, which typically contains a matrix of data. *dim(counts(spe))* and

dim(logcounts(spe)) retrieve the dimensions of the count's matrix and log-transformed counts matrix respectively.

```
assayNames(spe)

output:[1] "counts"    "logcounts"

dim(counts(spe))

Output:   980/ 98939

dim(logcounts(spe))

Output:   980/ 98939
```

2.3. Feature selection

2.3.1. overview

In this study, feature selection methods are utilized to identify genes that are spatially variable (SVGs) or highly variable (HVGs). These genes can subsequently be examined independently or utilized as inputs for subsequent analyses.

2.3.2. Highly variable genes (HVGs)

Using *scran* methods, we identified a set of top highly variable genes (HVGs) that can be used to classify major cell types.

It should be noted that HVGs are based solely on molecular characteristics (e.g., gene expression) and do not account for spatial information. If the biologically significant spatial information in this dataset primarily represents the spatial distributions of major cell types, then HVGs may be sufficient for subsequent analyses. However, if the dataset contains additional significant spatial features, it may be more fruitful to investigate genes that are spatially variable (SVGs). Scran methods have been implemented. This results in a collection of HVGs that can be used in subsequent analyses. The parameter prop indicates the desired number of HVGs. To illustrate, the value *prop = 0.1* identifies the top 10% of genes.

In first place, Using the *modelGeneVar()* function on the *spe* dataframe, the code *dec = modelGeneVar(spe)* fits a gene variance model and stores the result in the dec object. The code *fit = metadata(dec)* employs the *metadata ()* function to extract metadata from the dec object and assign it to the fit object.

Then, visualize the mean-variance relationship using code (chapter3, figure:19):

```
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)
```

We have ultimately chosen the top 66 HVGs:

```
top_hvgs = getTopHVGs(dec, prop = 0.1)

length(top_hvgs)
```

2.4. Dimensionality reduction

2.4.1. overview

During this segment, dimensionality reduction techniques are implemented to facilitate the visualization of the data and provide inputs for subsequent analyses.

2.4.2. Principal component analysis (PCA)

Utilize principal component analysis (PCA) to decrease the dimensionality of the dataset and preserve the initial 50 principal components (PCs) for subsequent analyses, using the set of highly variable genes (HVGs) as data.

This is achieved for two purposes: (i) to mitigate the impact of noise introduced by the random variation in expression of genes of low biological significance, whose expression patterns are assumed to be unrelated; and (ii) to enhance the computational efficiency of subsequent analyses.

The *scater* package contains a computationally efficient implementation of PCA that we employ. Due to the use of randomization in this implementation, a random seed must be set to ensure reproducibility. To calculate PCA using the subsequent analysis:

A) The code *set.seed(123)* sets the random number generator seed to 123. This ensures that the random number generation will be reproducible in subsequent code executions.

B) Run PCA:

```
spe = runPCA(spe, subset_row = top_hvgs)

reducedDimNames(spe)

dim(reducedDim(spe, "PCA"))
```

The output was reduced *DimNames(spe)* and *dim(reducedDim(spe, "PCA"))* will be "*PCA*" and 98939 / 50, respectively.

To obtain further visualizing, we employ code that appears to be associated with utilizing the *plotSpots* function to visualize the initial three principal components (PC1, PC2, PC3) of the *spe* dataset.

```
spe$PC1 <- reducedDim(spe, "PCA")[, 1]

spe$PC2 <- reducedDim(spe, "PCA")[, 2]

spe$PC3 <- reducedDim(spe, "PCA")[, 3]

plotSpots(spe,annotate ="PC1",in_tissue = NULL )
```

The code seems to utilize the PCA method to extract the PC1, PC2, and PC3 values from the result of the *reducedDim* function. Following that, these values are appended as fresh variables to the *spe* dataset. Following that, using PC1 as an example, the *plotSpots* function is called with PC1 as the *annotate* parameter and *NULL* as the *in_tissue* parameter. This graph most likely represents the spatial arrangement of data points in the dataset in accordance with the designated principal components.

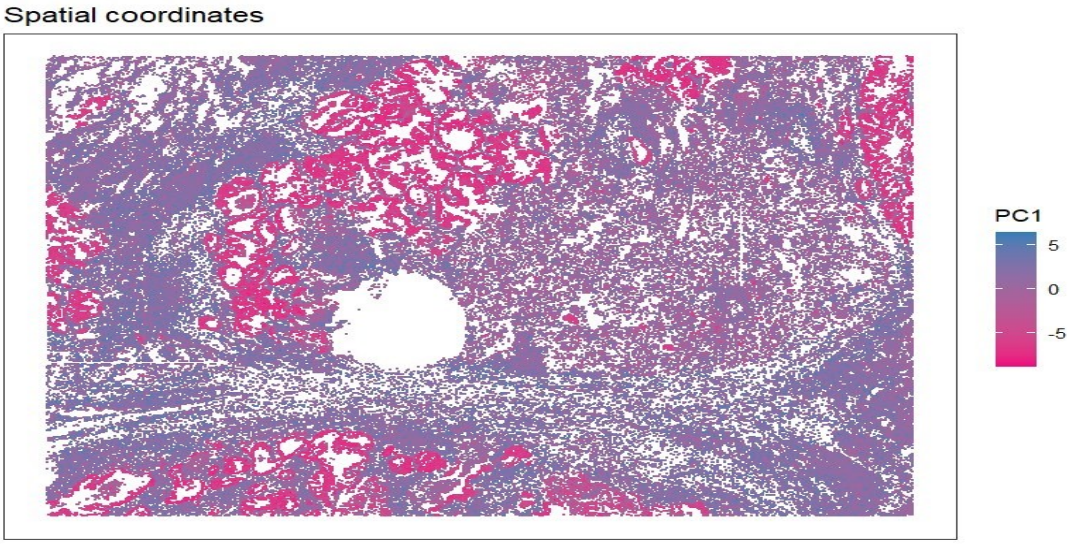

Figure 8 Data points are organized spatially within the dataset using the principal components, denoted as pc1.

2.4.3. Uniform Manifold Approximation and Projection (UMAP)

Additionally, UMAP is executed on the top 50 personal computers, and the two most prominent UMAP components are retained for the purpose of visualization.

```
set.seed(123)

spe = runUMAP(spe, dimred = "PCA")

reducedDimNames(spe)

dim(reducedDim(spe, "UMAP"))
```

The *reducedDimNames(spe)* function returns the available dimensionality reduction methods, which in this case are *PCA* and *UMAP*. The output of *dim(reducedDim(spe, "UMAP"))* indicates that the *UMAP* reduction transformed your data into a matrix with 98939 rows and two columns. This implies that the *UMAP* method reduced your data's dimension to two.

However, the following is done to facilitate plotting:

```
colnames(reducedDim(spe, "UMAP")) = paste0("UMAP", 1:2)
```

The code *colnames(reducedDim(spe, "UMAP")) = paste0("UMAP", 1:2)* assigns column names to the *UMAP* representation of *spe* obtained through *UMAP*. The *reducedDim* function extracts the reduced dimensional representation from an object, and the *paste0* command concatenates *UMAP* with 1 and 2.

2.4.4. Visualizations

Utilize the plotting functions provided by the *ggspavis* package to generate plots. We shall annotate these reduced dimension plots with cluster labels in the subsequent section on clustering.

a) Plot top 2 PCA dimensions:

The code ***plotDimRed(spe, type = "PCA")*** generates a scatter plot of a dimensionally reduced representation of data using Principal Component Analysis (PCA). The plot displays data points in a two-dimensional space, revealing the structure or clustering of the data, enabling visual interpretation and analysis.

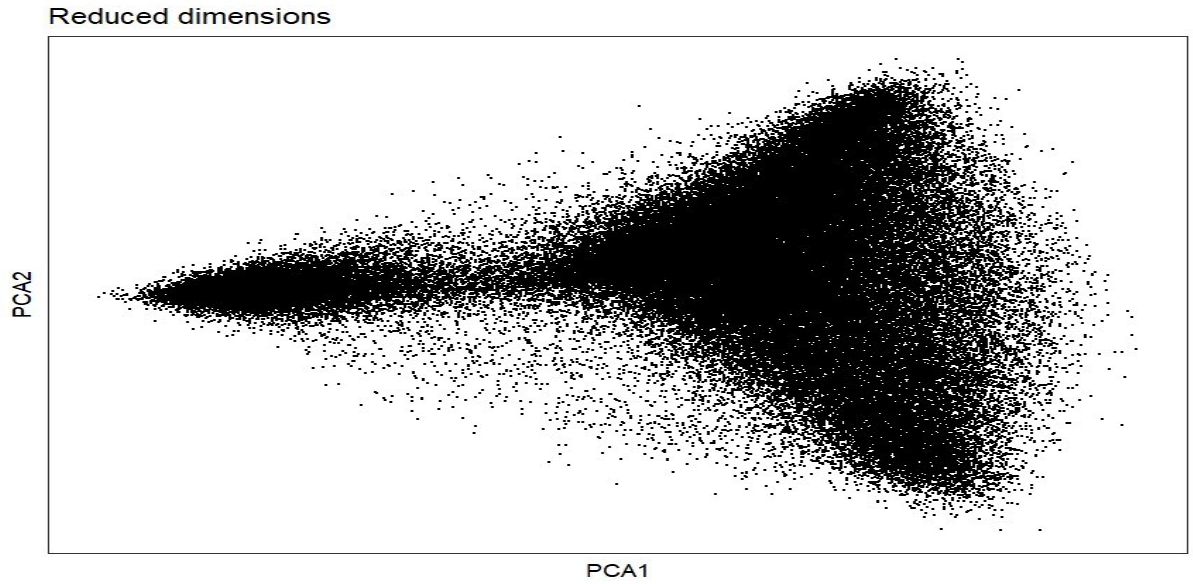

Figure 9 PCA dimensions.

b) Plot top 2 UMAP dimensions:

The code ***plotDimRed(spe, type = "UMAP")*** generates a scatter plot of the dimensionally reduced representation of data using the UMAP technique. This function visualizes high-dimensional data by reducing its dimensionality. The plot displays data points in a two-dimensional space, capturing non-linear relationships and manifold structure. This analysis provides insights into data structure and clustering.

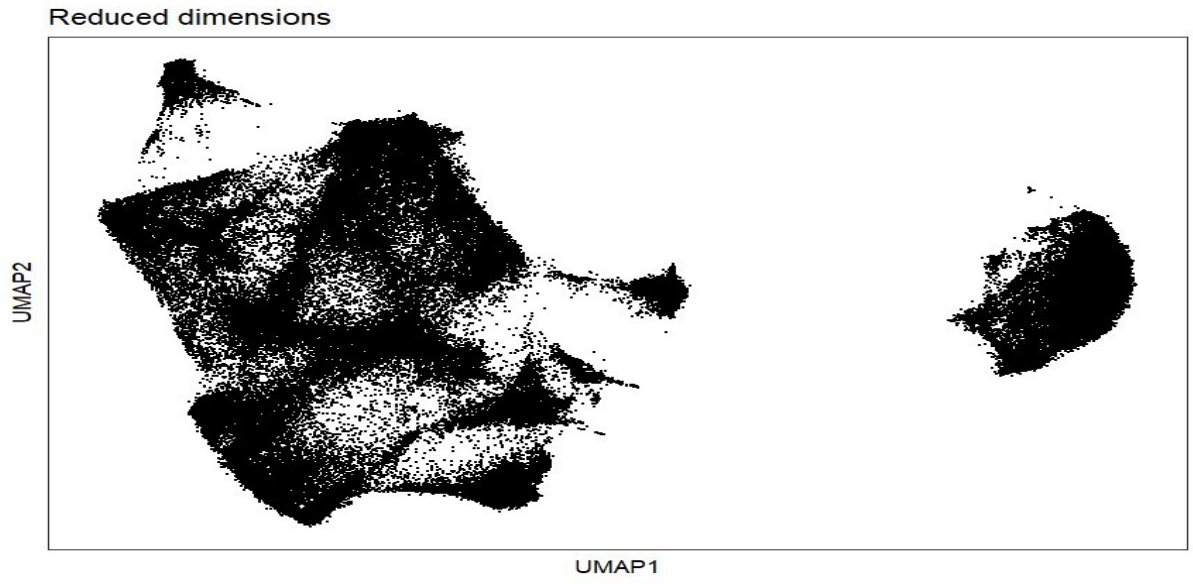

Figure 10 UMAP dimensions

2.5. clustering

2.5.1. overview

By utilizing clustering algorithms on lung cancer data, it is possible to discern "spatial domains," which are regions delineated spatially and comprise consistent gene expression profiles. Spatial domains may comprise regions that exclusively contain a single cell type or a consistent mixture of cell types, although this is not always the case.

Implementing clustering algorithms derived from single-cell workflows onto data, excluding spatial information, is equivalent to utilizing clustering to distinguish cell types within single-cell data.

It is critical to bear in mind that different resolutions can be utilized to define cell types and states. This implies that the optimal number of clusters is context-dependent with respect to clustering. The precise number of clusters is context-dependent and therefore cannot be determined (e.g. major cell populations vs. rare subtypes).

After spatial domains have been identified through clustering or manual analyses, they can be subjected to differential expression testing to identify representative genes.

2.5.2. Non-spatial clustering on HVGs

We apply louvaine clustering method designed for single-cell RNA sequencing data and rely solely on molecular characteristics (gene expression) to perform clustering. We apply graph-based clustering to the top 50 PCs determined from the set of top HVGs utilizing the louvaine method implemented in Scran.

This means that, in the context of spatial data, we assume that molecular characteristics can be used to discern spatial distribution patterns of cell types that are biologically informative (gene expression).

For this purpose, first the code provided executes the following steps:

```
set.seed(123)

k =10

g = buildSNNGraph(spe, k = k, use.dimred = "PCA")

g_walk = igraph::cluster_louvain(g)

clus = g_walk$membership

table(clus)
```

To ensure reproducibility, the code initializes a random seed and parameter k. It then generates a graph with the Shared Nearest Neighbor (SNN) algorithm and the Principal Component Analysis (PCA) representation. The graph is analysed using the Louvain community detection algorithm, which identifies communities and clusters. The code extracts community assignments for each data point and generates a frequency table showing the number of data points in each cluster.

The aforementioned analysis would yield the following results:

1	2	3	4	5	6	7	8	9	10	11	12	13
12889	9458	8660	6123	9875	9771	2158	10461	14753	8591	2561	3238	401

The code ***colLabels(spe) = factor(clus)*** assigns column labels to the *spe* object via the *clus* factor variable, allowing each column to be assigned to a specific cluster. This is useful for tasks like analysis, visualization, and data manipulation that necessitate cluster-based data grouping or referencing.

Additionally, visualize the clusters in (i) reduced dimension space and (ii) spatial coordinates (x-y) on the tissue slide (PCA or UMAP). Plotting functions from the *ggspavis* package are applied.

The visualizations demonstrate that the clustering closely approximates the known biological structure (cortical layers). Again, the clusters are not perfectly separated in UMAP space. The code ***plotSpots(spe, annotate = colData(spe)$label, palette = "libd layer colors", in_ tissue = NULL)*** produces a spatial plot of the *spe* object's cells or data points. It employs the annotate argument to label cells, the palette argument to specify the *color* palette, and the *in-tissue* argument to specify specific regions or masks of the tissue. This function is commonly used to visualize spatial data.

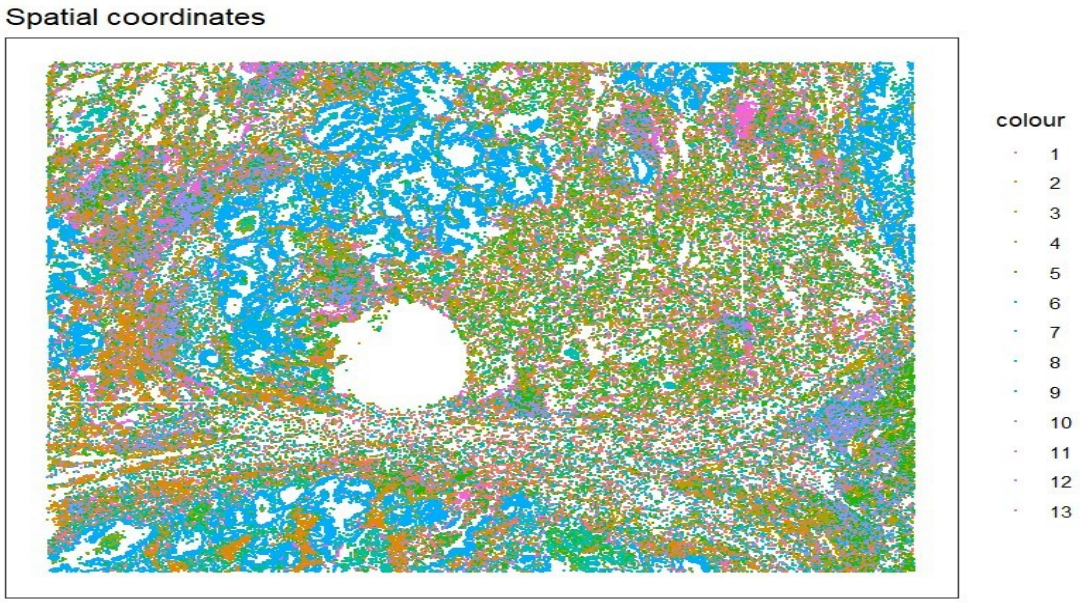

Figure 11 spatial coordinates (x-y) on the tissue slide (PCA or UMAP)

Consider Plot Clusters in PCA Reduced Dimensions *plotDimRed(spe, type = "PCA", annotation = colData(spe)$label, palette = "libd layer colors")*

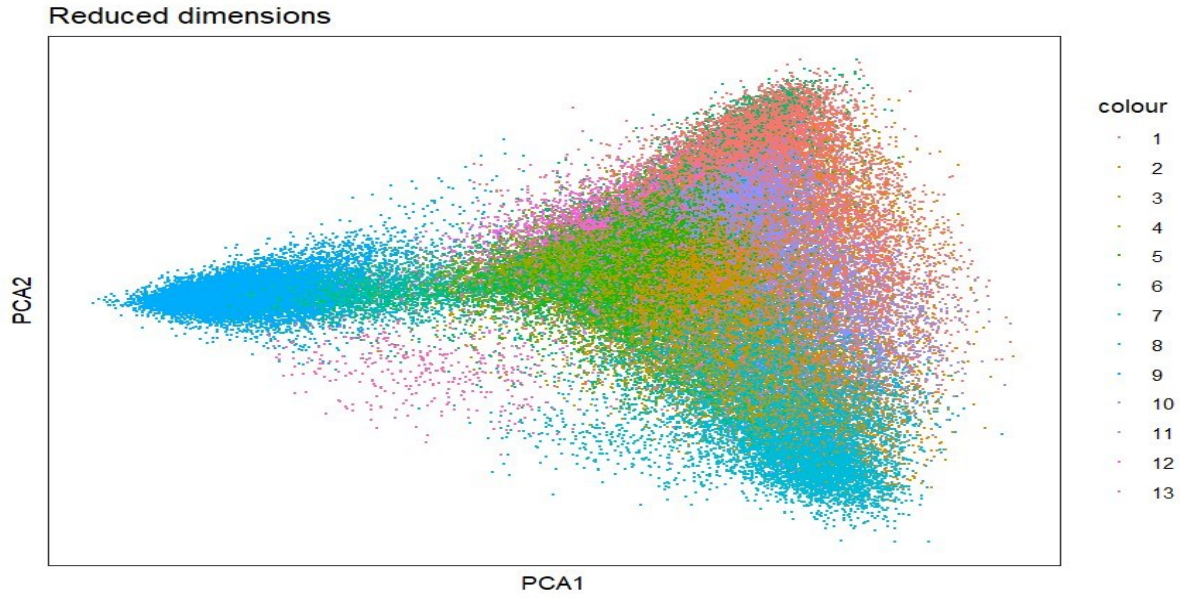

Figure 12 reduced dimension space for PCA.

Additionally, plot clusters in UMAP reduced dimensions, *plotDimRed(spe, type = "UMAP",*

annotate = colData(spe)$label, palette = "libd_layer_colors")

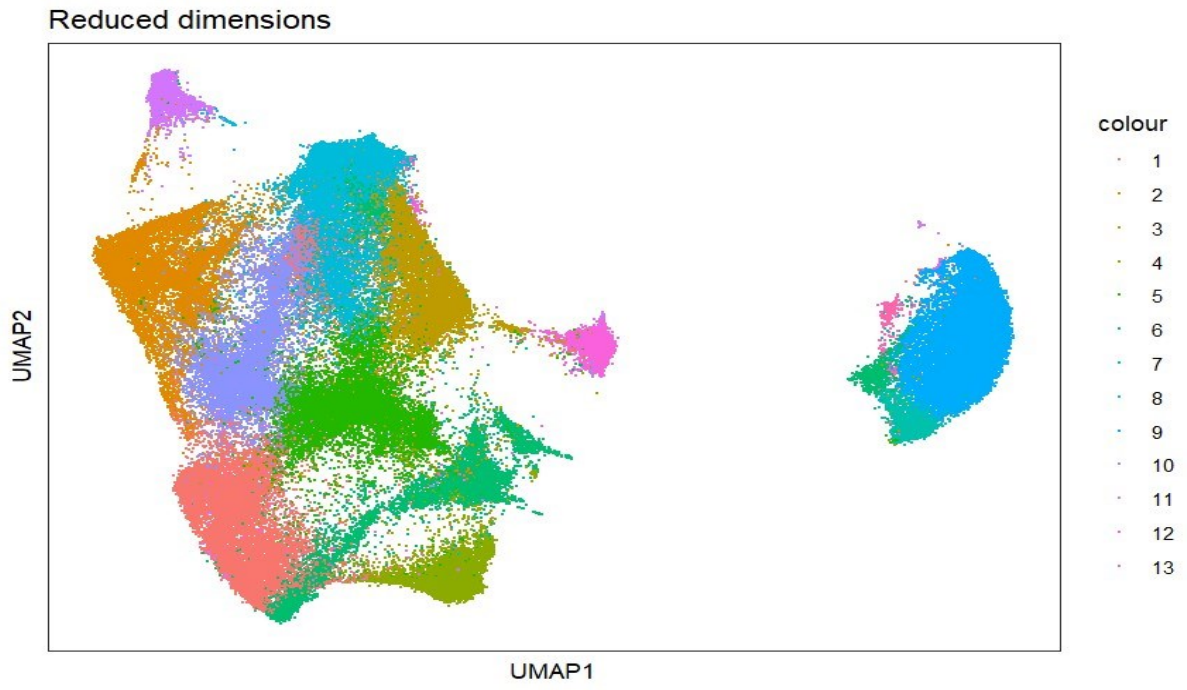

Figure 13 reduced dimension space for UMAP.

6. Marker genes

6.1.overviwe

In this section, we use differential expression testing to determine representative marker genes for each cluster or spatial domain.

6.2. Differential expression testing

Determine representative marker genes for each cluster or spatial domain by looking for differences in gene expression between clusters.

We use the *findMarkers* implementation in scran to perform a binomial test, which looks for genes that differ in the proportion expressed vs. not expressed between clusters. This is a more stringent test than the default t-tests, and it favors genes that are simpler to interpret and validate experimentally.

Firstly, trying to test for marker genes:

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

then, returns a list with one DataFrame per cluster:

```
markers
```

```
library(pheatmap)

interesting = markers[[1]]

best_set = interesting[interesting$Top <= 5, ]

logFCs = getMarkerEffects(best_set)

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

The code generates a *heatmap* using the pheatmap function from the pheatmap package to visualize patterns in large datasets. It uses the *getMarkerEffects* function to obtain the log-fold changes for markers in *best_set*. The heatmap is then generated using *logFCs* as input data and breaks as color levels. The resulting heatmap provides insight into the expression patterns or differences across samples, allowing for better visualization of the selected markers.

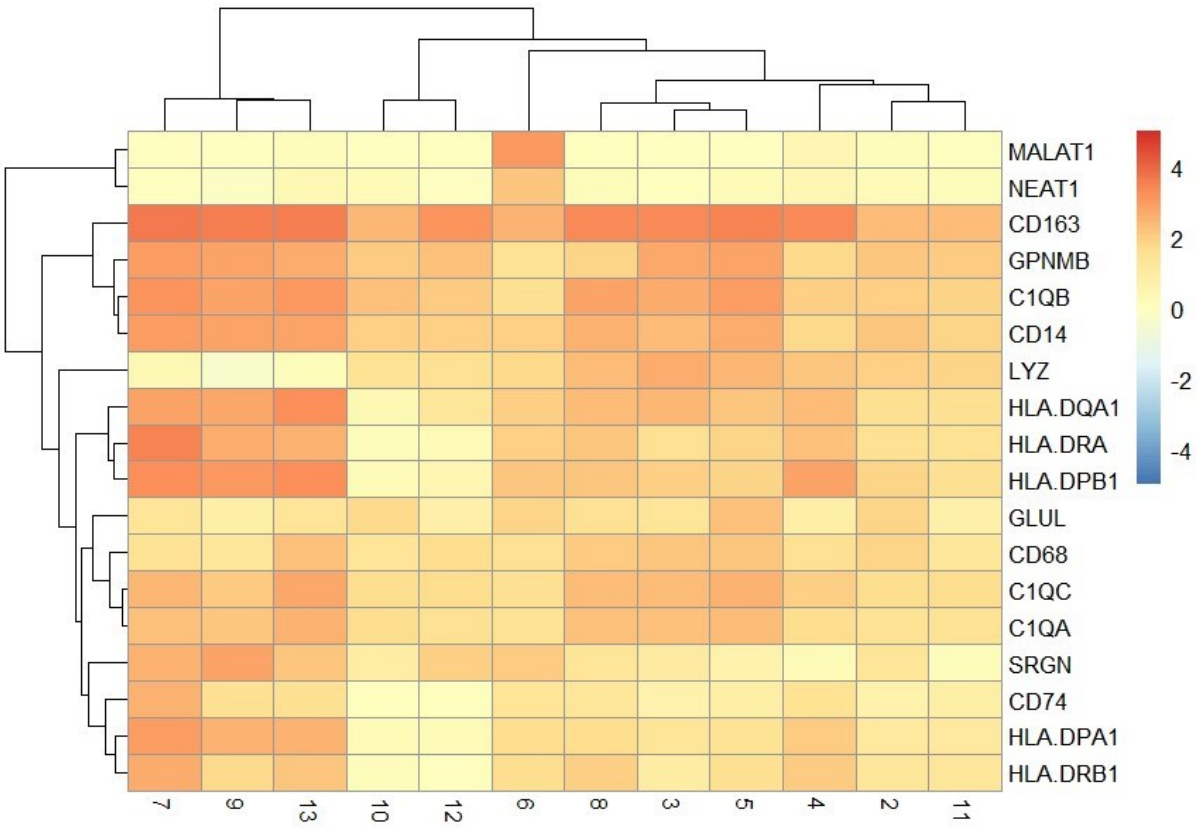

Figure 14 The heatmap offers valuable information regarding the variations or patterns of expression among samples.

Overall, plot log-transformed normalized expression of top genes for one cluster:

```
top_genes = head(rownames(interesting))

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

Furthermore, it appears that the *plotExpression* function is used in spatial experiments to visualize the expression of the top genes *(spe)*. This procedure can help you understand the differences in gene expression patterns between labeled groups. The code *head(rownames(interesting)) = top_ genes* extracts the names of the most important genes from the interesting object, whereas the *head()* function only selects the first few. Additionally, the *plotExpression* function *(spe, x = "label", features = top_ genes)* is used to generate the plot. We're plotting gene expression levels *(features = top_ genes)* across the different groups labeled in our *SpatialExperiment*. The variable *"top_ genes"* in our *SpatialExperiment* refers to the specific genes that will be visualized. The analysis of this plot may reveal useful information about the variation in expression levels of the selected top genes across different labeled groups in our spatial transcriptomics data.

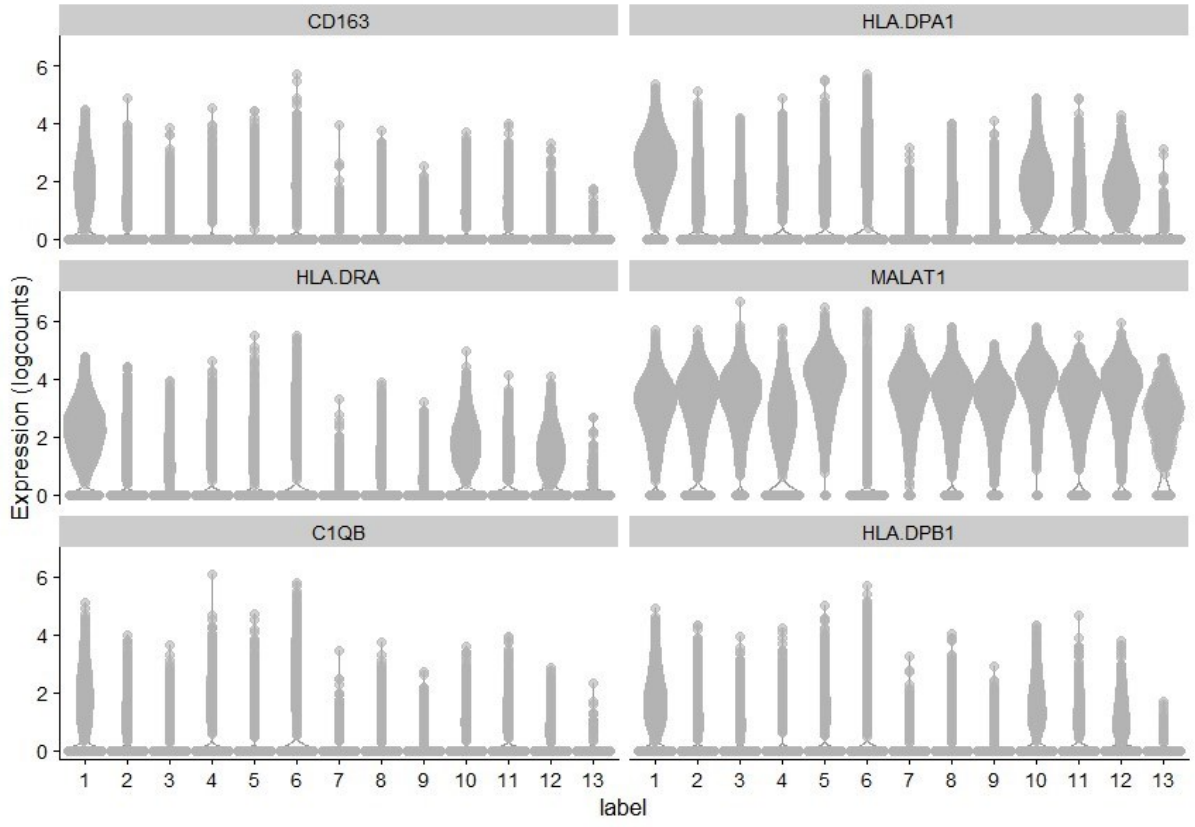

Figure 15top genes

Chapter 3

Conclusion

Spatial transcriptomics (ST) data has enabled the accumulation of significant knowledge in a variety of fields, including but not limited to cancer, neuroscience, and various diseases. This facilitates the molecular differentiation of normal and tumor tissues, as well as the spatial identification of specific cell types [76]. ST technologies are classified into two types: imaging-based and sequencing-based technologies [77,78]. This investigation involved exploratory data analysis (EDA) of a spatial transcriptomics dataset based on images, specifically the CosMX lung cancer dataset, which is the focus of this study. What we did with this survey is Conduct quality control inspections to identify areas or genes with poor quality. Use PCA or UMAP to see how cells are distributed spatially. Use spatial clustering analysis to identify spatially coherent cell groups, as well as genes whose expression varies according to the condition or cell type. Finally, we looked at the expression levels of six genes: HLA.DPA1, MALAT1, C1QB, and HLA.DPA1. MALAT1 has the highest expression, while CD163 has the lowest. ST, on the other hand, is a technological innovation that advances our understanding of complex biological systems and diseases, creating new opportunities for advancement in these fields. Nonetheless, the technology faces a number of challenges, including tissue complexity, spatial resolution limitations, integration issues, substandard data, data noise, sampling bias, and technological diversity. Addressing these challenges through interdisciplinary collaborations, technological advancements, and ethical structures is critical for maximizing ST capabilities and accelerating biological and medical research[79].

Chapter4

supplementary element

4.1 Quality control

```
DataFrame with 6 rows and 19 columns
        fov   cell_ID      Area AspectRatio CenterX_local_px CenterY_local_px      Width
  <integer> <integer> <integer>   <numeric>        <integer>        <integer> <integer>
1         1         1      1259        1.34             1027             3631        47
2         1        10      2550        1.68              923             3626        74
3         1       100      3867        1.49             1846             3516        94
4         1      1000      3234        0.68              941             1952        57
5         1      1001      1179        0.48             2231             1972        29
6         1      1002      2914        1.11             3944             1963        68
     Height Mean.MembraneStain Max.MembraneStain Mean.PanCK Max.PanCK Mean.CD45  Max.CD45
  <integer>          <integer>          <integer>  <integer> <integer> <integer> <integer>
1        35               3473               7354        715      5755       361       845
2        44               4062              10027        534       970       453      1499
3        63               3503              38825        634      2032       597      2198
4        84               3916              15686       1047     11014       348      4406
5        60               1976               5692        549      1405       153       581
6        61               2627               6660        560       843       298       650
   Mean.CD3   Max.CD3 Mean.DAPI  Max.DAPI   sample_id
  <integer> <integer> <integer> <integer> <character>
1        22       731      4979     26374    sample01
2        19       619      5917     28946    sample01
3         0       193     13226     43964    sample01
4         0        59      4730     23181    sample01
5        38       641      5083     23733    sample01
6         0        68     12780     36322    sample01
```

Table 2 exhibit the result obtained from the data loading operation.

```
DataFrame with 6 rows and 22 columns
        fov   cell_ID      Area AspectRatio CenterX_local_px CenterY_local_px      Width
  <integer> <integer> <integer>   <numeric>        <integer>        <integer> <integer>
1         1         1      1259        1.34             1027             3631        47
2         1        10      2550        1.68              923             3626        74
3         1       100      3867        1.49             1846             3516        94
4         1      1000      3234        0.68              941             1952        57
5         1      1001      1179        0.48             2231             1972        29
6         1      1002      2914        1.11             3944             1963        68
     Height Mean.MembraneStain Max.MembraneStain Mean.PanCK Max.PanCK Mean.CD45  Max.CD45
  <integer>          <integer>          <integer>  <integer> <integer> <integer> <integer>
1        35               3473               7354        715      5755       361       845
2        44               4062              10027        534       970       453      1499
3        63               3503              38825        634      2032       597      2198
4        84               3916              15686       1047     11014       348      4406
5        60               1976               5692        549      1405       153       581
6        61               2627               6660        560       843       298       650
   Mean.CD3   Max.CD3 Mean.DAPI  Max.DAPI   sample_id       sum  detected     total
  <integer> <integer> <integer> <integer> <character> <numeric> <numeric> <numeric>
1        22       731      4979     26374    sample01        23        19        23
2        19       619      5917     28946    sample01        99        52        99
3         0       193     13226     43964    sample01       243       131       243
4         0        59      4730     23181    sample01       196       115       196
5        38       641      5083     23733    sample01        15        11        15
6         0        68     12780     36322    sample01       128        86       128
```

Table 3 outcome of the compute QC metrics

4.1.1 Thresholds for library size (“sum”)

The *plotQC()* function is most likely a custom function that generates quality control plots based on the arguments passed to it. It's a scatter plot that shows the relationship between the "detected" metric on the x-axis and the "sum" metric on the y-axis. A threshold line is also being added at y = 10.

According to the histogram, the horizontal line (argument threshold) represents a potential filtering threshold for library size.

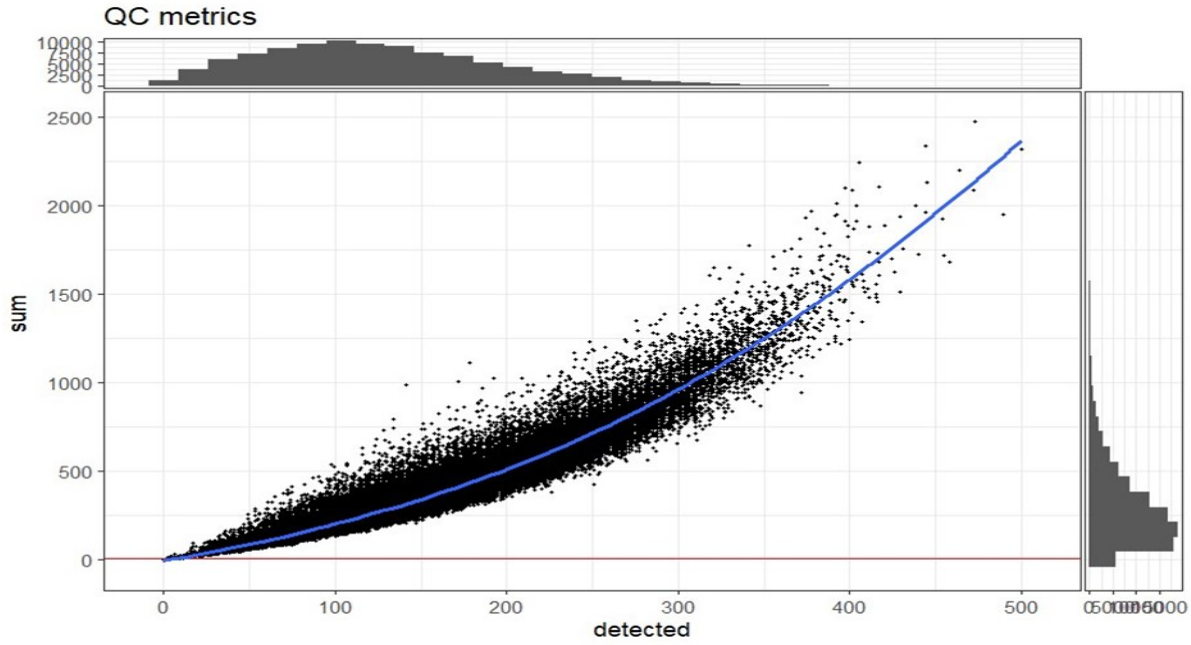

Figure 16 Thresholds for library size

4.1.2. Thresholds for Number of expressed genes (“detected”)

The plotQC() function generates quality control plots using the arguments passed to it. It's likely that this is a custom function. The scatter plot is used to show the relationship between the "detected" metric (y-axis) and the "sum" metric (x-axis). Furthermore, a threshold line is introduced using the histogram at y = 10.

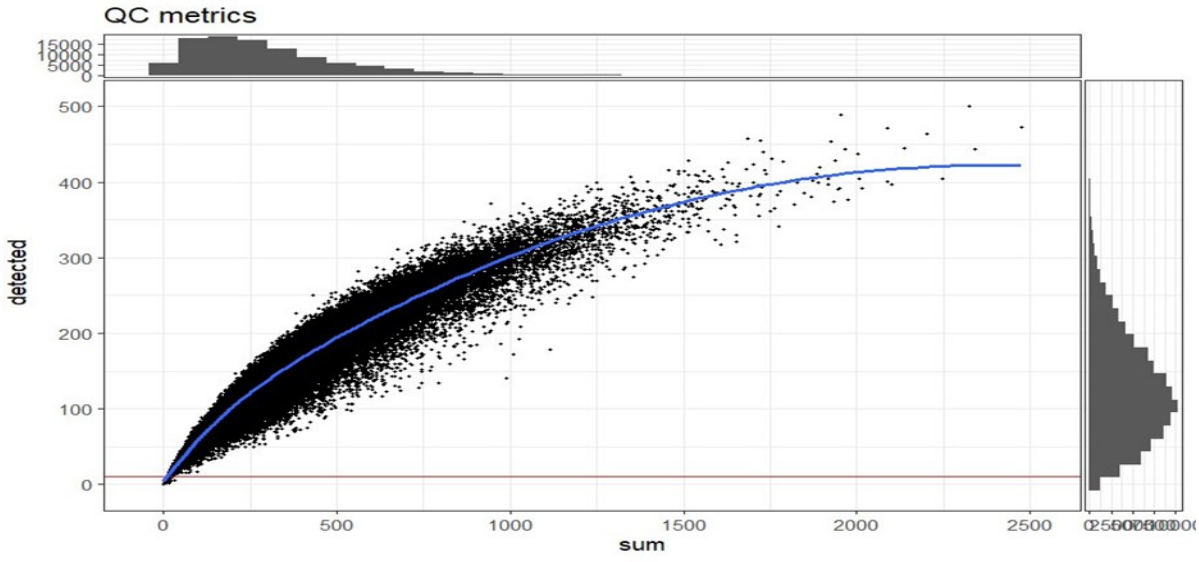

Figure 17 Thresholds for Number of expressed genes

4.2. Normalization

4.2.1. compute logcounts and store the results in an object.

```
spe = spe[, colSums(counts(spe)) > 0]

spe = computeLibraryFactors(spe)

summary(sizeFactors(spe))

hist(sizeFactors(spe), breaks = 100)
```

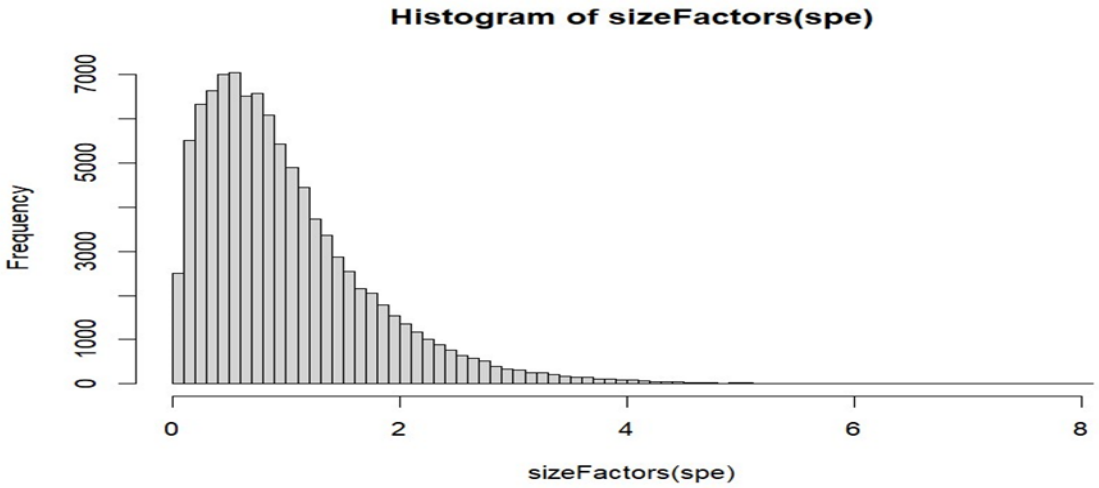

Figure 18histogram ***logcounts***

4.3. Feature selection

4.3.1Highly variable genes (HVGs)

To illustrate the mean-variance correlation:

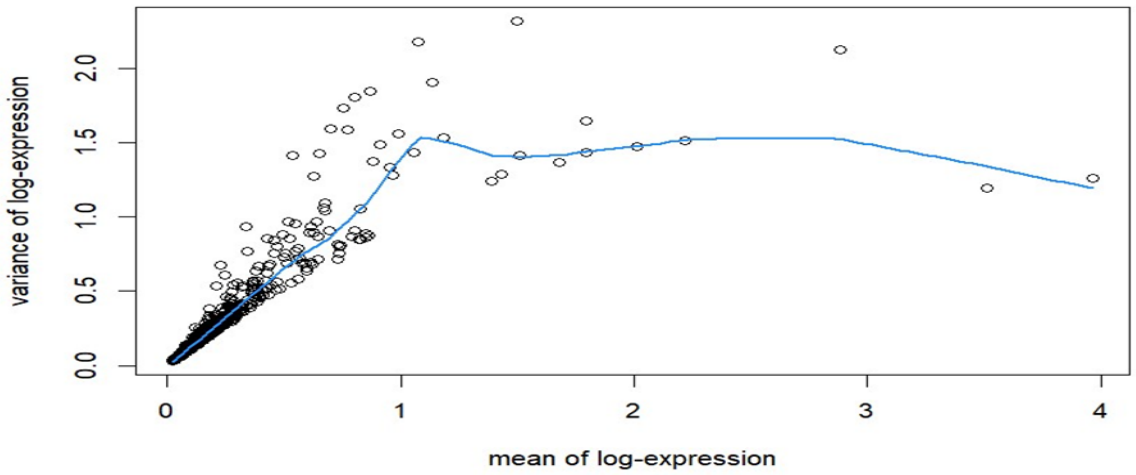

Figure 19 Highly variable genes (HVGs)

references

1. Graf, T. and M. Stadtfeld, Heterogeneity of embryonic and adult stem cells. Cell stem cell, 2008. 3(5): p. 480-483.

2. Trapnell, C., Defining cell types and states with single-cell genomics. Genome research, 2015. 25(10): p. 1491-1498.

3. Yao, Z., et al., A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain. bioRxiv, 2023.

4. Newman, A.M., et al., Determining cell type abundance and expression from bulk tissues with digital cytometry. Nature biotechnology, 2019. 37(7): p. 773-782.

5. Wang, D. and S. Bodovitz, Single cell analysis: the new frontier in 'omics. Trends in biotechnology, 2010. 28(6): p. 281-290.

6. Weaver, W.M., et al., Advances in high-throughput single-cell microtechnologies. Current opinion in biotechnology, 2014. 25: p. 114-123.

7. Navin, N., et al., Tumour evolution inferred by single-cell sequencing. Nature, 2011. 472(7341): p. 90-94.

8. Xu, X., et al., Single-cell exome sequencing reveals single-nucleotide mutation characteristics of a kidney tumor. Cell, 2012. 148(5): p. 886-895.

9. Levsky, J.M. and R.H. Singer, Gene expression and the myth of the average cell. Trends in cell biology, 2003. 13(1): p. 4-6.

10. Tang, F., et al., Tracing the derivation of embryonic stem cells from the inner cell mass by single-cell RNA-Seq analysis. Cell stem cell, 2010. 6(5): p. 468-478.

11. Natarajan, K.N., Single-cell tagged reverse transcription (STRT-Seq). Single Cell Methods: Sequencing and Proteomics, 2019: p. 133-153.

12. Hashimshony, T., et al., CEL-Seq: single-cell RNA-Seq by multiplexed linear amplification. Cell reports, 2012. 2(3): p. 666-673.

13. Jaitin, D.A., et al., Massively parallel single-cell RNA-seq for marker-free decomposition of tissues into cell types. Science, 2014. 343(6172): p. 776-779.

14. Picelli, S., et al., Smart-seq2 for sensitive full-length transcriptome profiling in single cells. Nature methods, 2013. 10(11): p. 1096-1098.

15. Ramsköld, D., et al., Full-length mRNA-Seq from single-cell levels of RNA and individual circulating tumor cells. Nature biotechnology, 2012. 30(8): p. 777-782.

16. Aldridge, S. and S.A. Teichmann, Single cell transcriptomics comes of age. Nature Communications, 2020. 11(1): p. 4307.

17. Han, X., et al., Mapping the mouse cell atlas by microwell-seq. Cell, 2018. 172(5): p. 1091-1107. e17.

18. Schaum, N., et al., Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris: The Tabula Muris Consortium. Nature, 2018. 562(7727): p. 367.

19. Fan, Y., et al., Reliable Identification and Interpretation of Single-Cell Molecular Heterogeneity and Transcriptional Regulation using Dynamic Ensemble Pruning. Advanced Science, 2023. 10(22): p. 2205442.

20. Grün, D., et al., Single-cell messenger RNA sequencing reveals rare intestinal cell types. Nature, 2015. 525(7568): p. 251-255.

21. Mantri, M., et al., Spatiotemporal single-cell RNA sequencing of developing chicken hearts identifies interplay between cellular differentiation and morphogenesis. Nature communications, 2021. 12(1): p. 1771.

22. Grauel, A.L., et al., TGFβ-blockade uncovers stromal plasticity in tumors by revealing the existence of a subset of interferon-licensed fibroblasts. Nature communications, 2020. 11(1): p. 6315.

23. Davidson, E.H. and D.H. Erwin, Gene regulatory networks and the evolution of animal body plans. Science, 2006. 311(5762): p. 796-800.

24. Peter, I.S. and E.H. Davidson, Evolution of gene regulatory networks controlling body plan development. Cell, 2011. 144(6): p. 970-985.

25. Crosetto, N., M. Bienko, and A. Van Oudenaarden, Spatially resolved transcriptomics and beyond. Nature Reviews Genetics, 2015. 16(1): p. 57-66.

26. Ramos-Vara, J.A., Technical aspects of immunohistochemistry. Veterinary pathology, 2005. 42(4): p. 405-426.

27. Medaglia, C., et al., Spatial reconstruction of immune niches by combining photoactivatable reporters and scRNA-seq. Science, 2017. 358(6370): p. 1622-1626.

28. Chen, J., et al., Spatial transcriptomic analysis of cryosectioned tissue samples with Geo-seq. Nature protocols, 2017. 12(3): p. 566-580.

29. Nichterwitz, S., et al., Laser capture microscopy coupled with Smart-seq2 for precise spatial transcriptomic profiling. Nature communications, 2016. 7(1): p. 12139.

30. Giacomello, S., et al., Spatially resolved transcriptome profiling in model plant species. Nature plants, 2017. 3(6): p. 1-11.

31. Rodriques, S.G., et al., Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution. Science, 2019. 363(6434): p. 1463-1467.

32. Vickovic, S., et al., High-density spatial transcriptomics arrays for in situ tissue profiling. bioRXiv Preprint at https://doi. org/10.1101/563338, 2018.

33. Righelli, D., et al., SpatialExperiment: infrastructure for spatially-resolved transcriptomics data in R using Bioconductor. Bioinformatics, 2022. 38(11): p. 3128-3131.

34. Maynard, K.R., et al., Transcriptome-scale spatial gene expression in the human dorsolateral prefrontal cortex. Nature neuroscience, 2021. 24(3): p. 425-436.

35. Ortiz, C., et al., Molecular atlas of the adult mouse brain. Science advances, 2020. 6(26): p. eabb3446.

36. Berglund, E., et al., Spatial maps of prostate cancer transcriptomes reveal an unexplored landscape of heterogeneity. Nature communications, 2018. 9(1): p. 2419.

37. Ji, A.L., et al., Multimodal analysis of composition and spatial architecture in human squamous cell carcinoma. Cell, 2020. 182(2): p. 497-514. e22.

38. Lohoff, T., et al., Integration of spatial and single-cell transcriptomic data elucidates mouse organogenesis. Nature biotechnology, 2022. 40(1): p. 74-85.

39. Srivatsan, S.R., et al., Embryo-scale, single-cell spatial transcriptomics. Science, 2021. 373(6550): p. 111-117.

40. Brennecke, P., et al., Accounting for technical noise in single-cell RNA-seq experiments. Nature methods, 2013. 10(11): p. 1093-1095.

41. Marinov, G.K., et al., From single-cell to cell-pool transcriptomes: stochasticity in gene expression and RNA splicing. Genome research, 2014. 24(3): p. 496-510.

42. Efremova, M. and S.A. Teichmann, Computational methods for single-cell omics across modalities. Nature methods, 2020. 17(1): p. 14-17.

43. Moses, L. and L. Pachter, Museum of spatial transcriptomics. Nature Methods, 2022. 19(5): p. 534-546.

44. Bergenstråhle, J., L. Larsson, and J. Lundeberg, Seamless integration of image and molecular analysis for spatial transcriptomics workflows. BMC genomics, 2020. 21(1): p. 1-7.

45. Palla, G., et al., Squidpy: a scalable framework for spatial omics analysis. Nature methods, 2022. 19(2): p. 171-178.

46. Stuart, T., et al., Comprehensive integration of single-cell data. Cell, 2019. 177(7): p. 1888-1902. e21.

47. Hao, Y., et al., Integrated analysis of multimodal single-cell data. Cell, 2021. 184(13): p. 3573-3587. e29.

48. Dries, R., et al., Giotto: a toolbox for integrative analysis and visualization of spatial expression data. Genome biology, 2021. 22: p. 1-31.

49. Virshup, I., et al., anndata: Annotated data. BioRxiv, 2021: p. 2021.12. 16.473007.

50. Siegel, R.L., et al., Cancer statistics, 2023. Ca Cancer J Clin, 2023. 73(1): p. 17-48.

51. Leader, A.M., et al., Single-cell analysis of human non-small cell lung cancer lesions refines tumor classification and patient stratification. Cancer cell, 2021. 39(12): p. 1594-1609. e12.

52. Liu, B., et al., Temporal single-cell tracing reveals clonal revival and expansion of precursor exhausted T cells during anti-PD-1 therapy in lung cancer. Nature Cancer, 2022. 3(1): p. 108-121.

53. Marjanovic, N.D., et al., Emergence of a high-plasticity cell state during lung cancer evolution. Cancer cell, 2020. 38(2): p. 229-246. e13.

54. Zheng, L., et al., Pan-cancer single-cell landscape of tumor-infiltrating T cells. Science, 2021. 374(6574): p. abe6474.

55. Ali, H.R., et al., Imaging mass cytometry and multiplatform genomics define the phenogenomic landscape of breast cancer. Nature Cancer, 2020. 1(2): p. 163-175.

56. Jackson, H.W., et al., The single-cell pathology landscape of breast cancer. Nature, 2020. 578(7796): p. 615-620.

57. Schürch, C.M., et al., Coordinated cellular neighborhoods orchestrate antitumoral immunity at the colorectal cancer invasive front. Cell, 2020. 182(5): p. 1341-1359. e19.

58. Sorin, M., et al., Single-cell spatial landscapes of the lung tumour immune microenvironment. Nature, 2023. 614(7948): p. 548-554.

59. Azizi, E., et al., Single-cell map of diverse immune phenotypes in the breast tumor microenvironment. Cell, 2018. 174(5): p. 1293-1308. e36.

60. Giladi, A. and I. Amit, Single-cell genomics: a stepping stone for future immunology discoveries. Cell, 2018. 172(1): p. 14-21.

61. Lavin, Y., et al., Innate immune landscape in early lung adenocarcinoma by paired single-cell analyses. Cell, 2017. 169(4): p. 750-765. e17.

62. Chen, J., et al., Single-cell transcriptome and antigen-immunoglobin analysis reveals the diversity of B cells in non-small cell lung cancer. Genome biology, 2020. 21(1): p. 1-21.

63. Zheng, C., et al., Landscape of infiltrating T cells in liver cancer revealed by single-cell sequencing. Cell, 2017. 169(7): p. 1342-1356. e16.

64. Zhang, L., et al., Lineage tracking reveals dynamic relationships of T cells in colorectal cancer. Nature, 2018. 564(7735): p. 268-272.

65. Ho, Y.-J., et al., Single-cell RNA-seq analysis identifies markers of resistance to targeted BRAF inhibitors in melanoma cell populations. Genome research, 2018. 28(9): p. 1353-1363.

66. Qi, Z., et al., Single-cell sequencing and its applications in head and neck cancer. Oral oncology, 2019. 99: p. 104441.

67. Karacosta, L.G., et al., Mapping lung cancer epithelial-mesenchymal transition states and trajectories with single-cell resolution. Nature communications, 2019. 10(1): p. 5587.

68. He, D., et al., Single-cell RNA sequencing reveals heterogeneous tumor and immune cell populations in early-stage lung adenocarcinomas harboring EGFR mutations. Oncogene, 2021. 40(2): p. 355-368.

69. Sun, X., et al., 2012P Spatially resolved transcriptomics deciphers inter-and intra-tumor heterogeneity of small cell lung cancer. Annals of Oncology, 2023. 34: p. S1071.

70. Zhu, J., et al., Delineating the dynamic evolution from preneoplasia to invasive lung adenocarcinoma by integrating single-cell RNA sequencing and spatial transcriptomics. Experimental & Molecular Medicine, 2022. 54(11): p. 2060-2076.

71. Larroquette, M., et al., Spatial transcriptomics of macrophage infiltration in non-small cell lung cancer reveals determinants of sensitivity and resistance to anti-PD1/PD-L1 antibodies. Journal for ImmunoTherapy of Cancer, 2022. 10(5).

72. Moses, L., et al., Voyager: exploratory single-cell genomics data analysis with geospatial statistics. bioRxiv, 2023: p. 2023.07. 20.549945.

73. Amezquita RA, Lun ATL, Becht E, Carey VJ, Carpp LN, Geistlinger L, et al. Orchestrating single-cell analysis with Bioconductor. Nat Methods. 2020. 17:137–45.

74. Moses L, Pachter L. Museum of spatial transcriptomics. Nat Methods. 2022.19:534–46.

75. McCarthy DJ, Campbell KR, Lun ATL, Wills QF. Scater: pre-processing, quality control, normalization, and visualization of single-cell RNA-seq data in R. Hofacker I, editor. Bioinformatics. 2017. 33:1179–86.

76. Zhou R, Yang G, Zhang Y, Wang Y. Spatial transcriptomics in development and disease. Mol Biomed. 2023. 4:32.

77. Wang Y, Liu B, Zhao G, Lee Y, Buzdin A, Mu X, et al. Spatial transcriptomics: Technologies, applications, and experimental considerations. Genomics. 2023. 115:110671.

78. Williams CG, Lee HJ, Asatsuma T, Vento-Tormo R, Haque A. An introduction to spatial transcriptomics for biomedical research. Genome Med. 2022. 14:68.

79. Danishuddin, Khan S, Kim JJ. Spatial transcriptomics data and analytical methods: An updated perspective. Drug Discov Today. 2024. 29:103889.