TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

DIPLOMARBEIT

Mathematical Methods in Single Cell RNA Sequencing Analysis with an Emphasis on the Validation of Clustering Results

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Technische Mathematik

eingereicht von

Stephan Reichl, BSc

Matrikelnummer 01029023

ausgeführt am Institut für Stochastik und Wirtschaftsmathematik
der Fakultät für Mathematik und Geoinformation der Technischen Universität Wien

Betreuung
Betreuer: Ao. Univ. Prof. Dipl.-Ing. Dr. techn. Karl Grill

Wien, 25.03.2018

(Unterschrift Verfasser) (Unterschrift Betreuer)

Abstracts

Abstract

Next generation sequencing underwent drastic developments in the last years. One of the newest developments enables researchers to extract gene expression data from single cells. One of these techniques managed to catapult single cell RNA (transcriptome) sequencing to the top of the field, because it provides high throughput and high accuracy sequencing information for a fraction of the costs. This technology is called micro-droplet sequencing and is based on the principle of separating the cells before the sequencing process by encapsulating them into droplets with the help of microfluidics. This new technology introduces researchers to a new era of single cell sequencing and thereby understanding of the mechanisms in biology on a single cell level. Different applications are the discovery of new cell types, identification of targets for drug development or the observation of biological reactions on a cellular level to name just a few.

With this new technology a new kind of data is generated. Due to the higher sensitivity of such sequencing processes we have to deal with a lot of noise or potentially distorted measurements due to environmental factors. At the same time the amount of data increased significantly, which results in high dimensional problems on large datasets. Although more dimensions are added to the data, most of it is very sparse and therefore makes the analysis more difficult. More than ever, sophisticated algorithms and mathematical methods are needed to deal with this very sparse, high dimensional and noisy signals.

This work focuses on the mathematical methods used and needed for the analysis of single cell RNA sequencing (scRNAseq) data. For that, we introduce a mathematical framework to describe the process of scRNAseq analysis in a rigorous way. The goal is to find and evaluate different methods for each of the identified procedures in the process of scRNAseq analysis. These procedures include quality control, normalization, identification and removal of confounding factors, dimensionality reduction and clustering with appropriate visualizations. Mathematical methods are needed to perform every one of these steps and we try to find and discuss the best approaches to overcome the novel challenges. Furthermore, we put an emphasis on the validation of clustering results within scRNAseq analysis and develop two approaches to resolve this issue. All of the described and developed methods are applied on a simulated dataset, based on a real scRNAseq dataset, and presented for the purpose of better understanding and validation.

Zusammenfassung

Die Sequenzierungsmethoden der zweiten Generation (engl. next generation sequencing) durchliefen in den letzten Jahren rapide Entwicklungen. Eine der neuesten Entwicklungen ermöglicht es Wissenschaftlern Genexpressionsdaten von einzelnen Zellen zu ermitteln. Eine dieser Technologien schaffte es die Einzelzell-RNA (Transkriptom) -Sequenzierung (engl. single cell RNA sequencing) an die Spitze dieses Forschungsgebietes zu katapultieren, da sie einen hohen Durchsatz und präzise Sequenzierungsinformationen für einen Bruchteil der Kosten zur Verfügung stellt. Diese Technologie wird als "micro-droplet sequencing" bezeichnet und basiert auf dem Prinzip die Zellen vor dem Sequenzierungsprozess durch das Einschließen in Tröpfchen mit Hilfe von Mikrofluidik zu isolieren. Diese neue Technologie ermöglicht es Forschern eine neue Ära der Einzelzell-Sequenzierung einzuläuten und damit die Erforschung von biologischen Mechanismen auf zellulärer Ebene voran zu treiben. Verschiedene Anwendungen sind die Entdeckung neuer Zelltypen, die Identifikation von Angriffspunkten in der Medikamentenentwicklung und die Beobachtung biologischer Reaktionen auf zellulärer Ebene, um nur einige wenige zu nennen.

Durch die Anwendung dieser neuartigen Technologie wird eine noch nie dagewesene Art von Daten generiert. Aufgrund der höheren Empfindlichkeit solcher Sequenzierungsverfahren treten, wegen äußerer Einflüsse, viele Störungen (Rauschen) oder möglicherweise verzerrte Messungen auf. Gleichzeitig hat die Datenmenge signifikant zugenommen, was zu hochdimensionalen Problemstellungen in großen Datensätzen führt. Obwohl den Daten mehr Dimensionen hinzugefügt werden, sind die meisten davon nur sehr dünnbesetzt und erschweren daher die Analyse. Mehr denn je sind komplexe Algorithmen und mathematische Methoden erforderlich, um diese sehr dünnbesetzten, hochdimensionalen und verrauschten Datensätze professionell verarbeiten zu können.

Diese Arbeit konzentriert sich auf die mathematischen Methoden, die für die Analyse von Einzelzell-RNA-Sequenzierungs-Daten (engl. scRNAseq data) verwendet werden. Dafür konstruieren wir einen mathematischen Rahmen, der es ermöglicht den Prozess der scRNAseq-Analyse auf eine rigorose Weise zu beschreiben. Das Ziel ist es verschiedene Methoden für jede der identifizierten Vorgehensweisen im Prozess der scRNAseq-Analyse zu finden, zu beschreiben und zu vergleichen. Diese Vorgehensweisen inkludieren unter anderem Qualitätskontrolle, Normalisierung, Identifikation und Entfernung von Störfaktoren sowie Dimensionsreduktion und Clustering mit geeigneten Visualisierungen. Für jeden dieser Schritte sind mathematische Methoden erforderlich und wir versuchen die besten Ansätze zur Überwindung dieser neuen Herausforderungen zu finden und zu diskutieren. Darüber hinaus konzentrieren wir uns auf die Validierung von Clustering-Ergebnissen innerhalb der scRNAseq-Analyse und entwickeln zwei Ansätze, um diese Problemstellung zu bewältigen. Alle beschriebenen und entwickelten Methoden werden auf einen simulierten Datensatz, basierend auf einem realen scRNAseq-Datensatz, angewendet und zum besseren Verständnis und zur Validierung präsentiert.

Acknowledgments

> "We must find time to stop and thank the people who make a difference in our lives."
>
> John F. Kennedy

First of all, I would like to thank my supervisor Professor Karl Grill for his expert advice, input and guidance throughout the making of my Master's thesis. I always enjoyed our conversations on a variety of topics and occasions and I am grateful for the trust and freedom that was given to me.

Of course, I want to thank the TU Wien for the possibility to study mathematics at such a high level and for the last few years, in which I called the Freihaus my second home.

I thank as well my former colleagues in Basel for their support and friendship. Especially, I want to thank Andreas Roller and Fabian Birzele for giving me the opportunity to work in this field and supporting me with all their experience and knowledge in bioinformatics. Furthermore, I want to thank Tony Kam-Thong and Javier Gayan for their help in dealing with difficult biostatistical matters. For the private tutoring in biology, biochemistry and genomics many thanks go to Nils Grabole and Benjamin Loos.

A very warm thank you is dedicated to my parents Monika and Christoph, and my aunt Irene, who have laid the foundation for the person I am today and supported me constantly throughout my studies with a home, delicious meals, valuable advice and much more.

Special thanks go to my girlfriend Stefanie Schinnerl for being always there for me and keeping me motivated no matter what I put on my agenda.

Last but not least, I want to thank my fellow students Viktoria Freingruber, Mariella Gregorich, Thomas Tanzer and Fritz Winkelbauer. Without you guys the study of mathematics would not have been so much fun and I enjoyed having a laugh with you, even during the high intensity sessions before exercises and exams.

Without all of you this would not have been possible, Thank You!

Yours truly,

Stephan

Contents

1. Motivation **1**

2. Introduction to Single Cell RNA Sequencing Analysis **3**
2.1. Next Generation Sequencing (NGS) . . . 3
2.1.1. Applications of NGS in the Past . . . 4
2.1.2. Potential Applications in the Future . . . 6
2.1.3. The Role of Mathematics in NGS . . . 6
2.2. The Technology . . . 8
2.2.1. Next Generation Sequencing (NGS) . . . 8
2.2.2. (Bulk) RNA Sequencing (RNAseq) . . . 10
2.2.3. Single Cell RNA Sequencing (scRNAseq) . . . 11
2.2.4. The 10x Genomics Single Cell Capturing Technology . . . 11
2.2.5. Challenges in scRNAseq . . . 14
2.3. The Analysis . . . 15
2.3.1. Goals of the Analysis . . . 15
2.3.2. Data . . . 16
2.3.3. General Workflow . . . 17
2.3.4. Quality Control . . . 17
2.3.5. Normalization . . . 18
2.3.6. Confounding Factor Analysis . . . 19
2.3.7. Dimensionality Reduction . . . 20
2.3.8. Clustering . . . 20
2.3.9. Visualization . . . 22

3. Mathematical Methods in scRNAseq - Analysis **23**
3.1. Introduction to the Mathematical Framework . . . 24
3.2. Outlier Detection . . . 26
3.3. Normalization . . . 29
3.3.1. Standard Methods . . . 29
3.3.2. Size Factor (SF) . . . 31
3.3.3. Upper Quartile (UQ) . . . 32
3.3.4. Weighted Trimmed-Mean of M-Values (TMM) . . . 33
3.3.5. Downsampling (DS) - A Stochastic Approach . . . 35
3.3.6. Size Factor by Pooling Across Cells (LSF) . . . 36
3.3.7. Comparison & Conclusion . . . 40

3.4. Explanatory Variables . . . 42
3.4.1. Variable Types . . . 43
3.4.2. Linear Regression . . . 44
3.4.3. Coefficient of Determination R^2 . . . 48
3.4.4. Correlation . . . 50
3.4.5. Statistical Hypothesis Testing . . . 52
3.4.6. McFadden's Pseudo - R^2 . . . 53
3.4.7. Discussion on Adjusted R^2 Measures . . . 57
3.4.8. Comparison . . . 57
3.4.9. Application . . . 58
3.5. Dimensionality Reduction . . . 65
3.5.1. Principal Component Analysis (PCA) . . . 65
3.5.2. t-Distributed Stochastic Neighbor Embedding (t-SNE) . . . 71
3.5.3. Application & Comparison . . . 77
3.6. Clustering . . . 82
3.6.1. Partitioning-Based Clustering . . . 83
3.6.2. Further Clustering Approaches . . . 85
3.6.3. Comparison & Conclusion . . . 88

4. Cluster Validation in scRNAseq - Analysis 89
4.1. The Starting Point . . . 90
4.2. Quality Measures for Clustering Results . . . 91
4.2.1. External Indices . . . 91
4.2.2. Internal Indices . . . 94
4.2.3. Information Criteria . . . 98
4.3. Determination of the Final Clustering Result . . . 100
4.3.1. The Favorite Approach . . . 100
4.3.2. The Consensus Approach . . . 103
4.3.3. Interpretation & Application . . . 105

5. Conclusion & Outlook 113
5.1. Conclusion . . . 113
5.2. Outlook & Further Work . . . 114

A. The Simulated Dataset 115

Bibliography 121

1. Motivation

> "Leave this world a little better than you found it."
>
> Robert Stephenson Smyth Baden-Powell

Right now the medical and biological landscape is transforming at a rapid speed. One of the reasons is the increased development and use of sophisticated technologies. Next generation sequencing (NGS) is one of those technologies, which are driving the fast development and advances. As these technologies get smaller, faster, better and cheaper they leave the realm of experimental research and enter into the applied sciences, where the benefit is apparent and methods are developed around the technology to generate further insights in their respective fields. One of these fast developing fields is single cell RNA sequencing (scRNAseq), which is a collection of methods that allow the capture of individual cells, followed by NGS to quantify the gene expression level of single cells by determining their ribonucleic acid (RNA) content.

These developments are relevant for advances in medicine, biology and chemistry or the life sciences in general, but especially in understanding of biological mechanisms, differentiation of disease characteristics, discovery of new cell types or drug development.

Therefore, research in this field directly influences the academic community by driving forward this progress, the industry by supplying a basis for further development of products and society by helping understand how the biological systems of our body work or as patients, in need of new treatment options. Obviously the list of applications, which result from advances in this field, can be continued endlessly, due to omnipresent biological mechanisms occurring on a single cell level.

In the process of sequencing, a lot of data is generated, therefore it is critical to invest time and effort in the research of the analysis of this data. In the case of scRNAseq we are faced with new kind of data and therefore challenges, which are novel in this field. We want to emphasize that the best and most rigorous scientific advances in the technology or experimental setup are rendered useless, if the data is not analyzed in a proper manner. Only after thorough analysis and exploration of the data, interpretation of results is possible, reasonable and meaningful. One exemplary challenge in dealing with biological measurements is usually the question of artificial biases introduced by environmental or technical interferences. There has to be a certain confidence that these influences did not distort or change the information, which is intended to be used to generate new knowledge or inform decisions, within the data or experiment. These challenges are bringing a lot of

different scientific fields together as physics, biology, mathematics, statistics and computer science.

This work is trying to aid in the effort of making the critical part, the data analysis, mathematically sound.

To position this work and put the mathematical aspects into context we start with an introduction into the realm of next generation sequencing and continue by explaining the existing technologies, which lead to a novel approach in scRNAseq developed by 10x Genomics. After that, we present a general workflow for scRNAseq analysis, with a qualitative description of all the steps involved.

Followed by that, we construct a rigorous mathematical framework to describe the process of scRNAseq analysis. Within this framework we systematically analyze different mathematical methods used and needed in scRNAseq analysis. The result represents a clear picture of the methods recommended to use with strict argumentation.

Furthermore, new ways to analyze scRNAseq analysis clustering results are investigated and novel approaches for the decision making process are presented.

2. Introduction to Single Cell RNA Sequencing Analysis

> "We wish to suggest a structure for the salt of deoxyribose nucleic acid (D.N.A.). This structure has novel features which are of considerable biological interest."
>
> J. D. Watson and F. H. C. Crick [WC53]

In this chapter we will first give a general introduction into the world of next generation sequencing followed by a more detailed description of the relevant technologies. The chapters are building on each other and lead us finally to single cell RNA sequencing. Here, we describe a certain method in more detail. At last a general workflow for single cell RNA sequencing analysis is presented. This chapter represents the basis and at the same time the starting point of this work.

2.1. Next Generation Sequencing (NGS)

Following Kulski [Kul16] we will give a short introduction into the general topic of DNA sequencing, especially next generation sequencing.

Next generation sequencing or in short NGS, is the name of a group of deep, high-throughput, in-parallel DNA (deoxyribonucleic acid) sequencing technologies. The goal of DNA sequencing technologies is to determine the order of nucleobases (for DNA or RNA: **A**denine, **C**ytosine, **G**uanine, **T**hymine or **U**racil) within a DNA molecule of a given organism. To determine the exact order of nucleotides different techniques were developed.

Before NGS, the first technology developed for sequencing DNA was called Sanger sequencing, which was introduced by Frederick Sanger in 1977. It was the gold standard for DNA sequencing for the next three decades. With the Sanger method scientists gained the ability to sequence DNA in a reliable, reproducible manner. Nowadays it is still in use for routine sequencing applications and validation of NGS data.

With the development of machines with higher sequencing throughput, by carrying out millions of sequencing reactions in parallel, at much reduced cost around year 2005, the second generation of DNA sequencing was introduced. The second generation can also be described by the need to prepare amplified sequencing libraries before undertaking sequencing of the amplified DNA clones.

Right now the third generation of sequencing or single molecular sequencing is starting to gain momentum. Here, sequencing can be done without the need for creating the time-consuming and costly amplification libraries, and therefore reduce cost and simplify the sequencing process. We will not further elaborate on the third generation, due to the fact that these technologies are not subject of this work, as we will discuss sequencing data generated by second generation technologies.

Due to these developments over the last decades the cost for sequencing one genome decreased significantly. In Figure 2.1 on page 4 we see the decrease in cost over the years from 2001 until 2017. Furthermore, a line describing Moore's Law (initially used in the computer hardware industry to describe the increase of transistors in processors), which dictates a doubling of "performance" (in our case halving of the price) every two years, is drawn to indicate that the reduction in costs is doing extremely well.

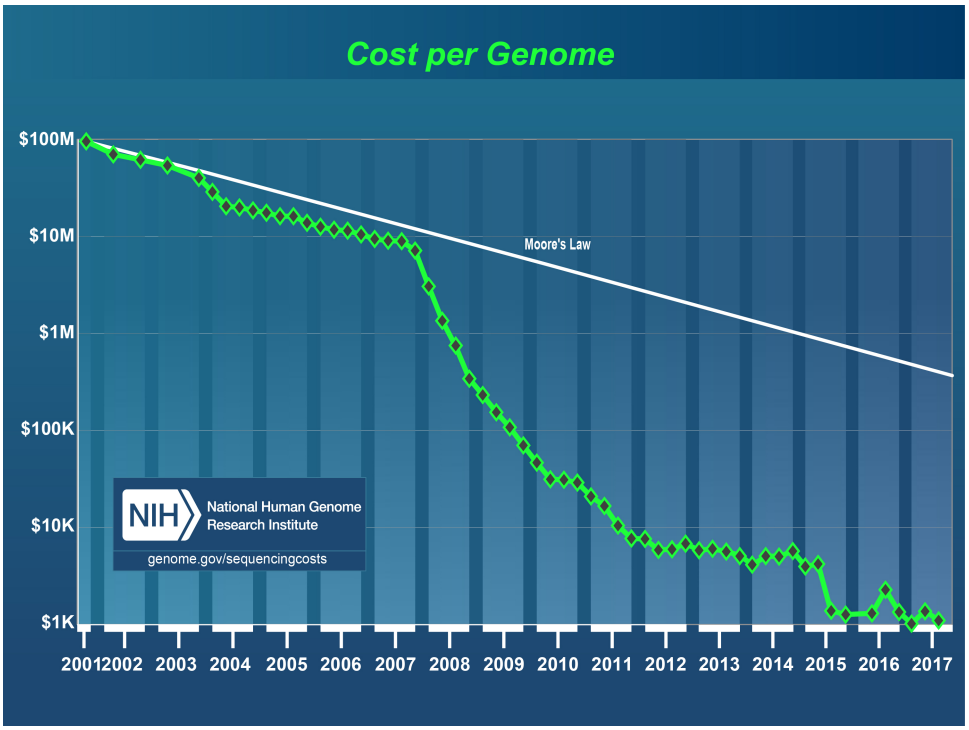

Figure 2.1.: Sequencing costs per genome over the years [Nat17]

2.1.1. Applications of NGS in the Past

One of the biggest achievements in the field of genetics was the Human Genome Project (HGP) [Nat15]. The goal was to sequence and map all of the genes (=genome) of members of the human species, Homo sapiens. Initiated in 1990 the HGP consisted of an international team of scientists. In 2003 the project was finished and a

complete genetic blueprint for building a human being was presented. The completion of this project took 13 years and cost approximately $2.7 billion. Compared to that, it would take nowadays only one day to sequence up to 45 human genomes, with a state of the art NGS system and the cost of a thousand dollar per genome [Ill17].

In the following we will describe three applications of NGS, which are relevant in the medical sciences and therefore more tangible as a motivation to drive advances in this field.

Cancer Research Medical centers specialized on cancer treatments have already incorporated protocols to screen the tumor for certain genetic mutations to identify the cause of cancerogenesis. Furthermore the choice of treatment is dependent on the mutations present within the tumor. With the help of sequencing as an analytical tool physicians are able to devise better therapeutic strategies for each patient.

Infectious Disease Outbreaks as for example Ebola virus or antibiotic resistant bacteria, are analyzed with the help of sequencing technologies with the goal of identifying the strains involved and finding a valid therapeutic agent or trace back the route of the outbreak.

Drug Discovery uses NGS technologies for conducting experiments with diseased tissue or cells to understand disease characteristics and look for a cure or potential targets to inhibit the progression of the disease. Additionally it is used in testing the effect of a potential treatment on diseased cells.

In general more evidence-based approaches are possible to be integrated in the medical practice, based on genomic research. Decision-making tools for patients and providers are improved through better screening and diagnostic options. Medical professionals have a better chance to tailor treatments to a patient's individual genomic profile.

The impact and the continuously declining costs of NGS technologies make the use feasible for small and large institutions all the same. Therefore research in a lot of different fields as genetics, biology, medicine, agriculture, forensic science, virology, microbiology, marine and plant biology can be driven to an extent never seen before.

With this drastic increase in data generation in a multitude of different scientific fields, the need for proper data analysis increased as well. Additionally, the type of data is changing with each generation of technologies and even for different systems within the same generation. Therefore, new methods have to be developed to deal in a standardized, reproducible and rigorous way with the data.

2.1.2. Potential Applications in the Future

In the future NGS might be used as a common tool by doctors or health practitioners alike. The potential applications for this technology in general are gaining a deeper understanding of the (human) genome in a short amount of time (hours or even minutes) to derive actionable knowledge concerning risks for potential diseases or genome related interventions.

Examples for such applications concerning human health would be

- sequencing the genome of a fetus as soon as possible after conception to determine potential health risk factors and intervene. Discussions concerning sequencing of newborns are already ongoing [Nat16].
- assessing potential health implications in a regular manner and interventions can be started immediately.
- discovering new cell types and understand biological mechanisms completely.
- building better models describing biological mechanisms and use simulations to extrapolate collected data to the future. One could even devise individualized models based on the personal genomic profile, which get updated by new data inputs.
- stratifying patients into groups that will benefit from a particular drug or avoid adverse reactions to a particular drug.
- characterizing diseases in a more precise way to immediately devise counter measures, which lead to meaningful interventions with a significantly higher probability of success.
- empowering individuals by giving them all the information about their genome and context on what to do with it.

Evidently, the applications of NGS technologies will not be restricted to human health interventions. It will also provide insights in all the other fields of biology and medicine as for example discovering new cell types (also non human) and trying to understand mechanisms of evolution or gene regulation in different types of animals and organisms, by looking at their genome.

2.1.3. The Role of Mathematics in NGS

NGS technologies work faster and more precise than ever before. Also a lot of sample preparation steps and the sequencing itself is mostly automated nowadays. This results in more data in less time and therefore the rigorous analysis of sequencing data becomes more important than ever before. This analysis is usually done by computational biologists, bioinformaticians and biostatisticians, depending on the question at hand.

Therefore we postulate that an appropriate and rigorous way to analyze data, derived from next generation sequencing, is essential to conduct proper research. The chosen methods for such an analysis heavily depend on the exact technologies used in the course of the sequencing process. In general, the data generated has to reflect the measured values by quantification of some kind. Then there has to be quality control steps to ensure to get rid of "bad" data and noise. Noise can be described as artificial signals added either by biological contamination or by technical errors during the sequencing or the preparation process. In both of these pre-processing steps mathematical methods are essential as for example outlier detection, normalization of the data, identification and removal of confounding factors. In the downstream analysis of such data the density of mathematical methods increases significantly to tackle problems as dimensionality reduction, proper visualization and clustering, to name just a few.

Of course, it always depends on the goal of the experiment, which therefore dictates the goal of the analysis. In this work, we focus on the mathematical methods which are or should be used to analyze NGS data derived from single cell RNA sequencing through a micro-droplet sequencing technology. The main goal of the analysis is to establish a clustering by the features of the cells, which in this case are the gene expression counts. Following the gene expression, clusters representing cell types, which generally speaking can be determined by a certain set of active genes, should emerge or other biological properties present in the cells.

In consideration of this goal the most common problems solved with the help of mathematical methods are

- quality control and outlier detection
- normalization of data to account for distortion by different sequencing depths
- identification and removal of confounding factors
- dimensionality reduction by determining most relevant dimensions
- visualization of the main features of the data in two or three dimension
- clustering by gene expression to identify cell types or other biological effects

The next chapters will give a brief introduction into the technical details of next generation sequencing, especially single cell RNA sequencing. The output of these technologies represents the basis of the data analysis.

2.2. The Technology

We will give a short technical introduction into next generation sequencing, especially into the rather new technology of single cell RNA sequencing. To understand the challenges we are facing in single cell RNA sequencing, we have to understand the difference between two kinds of RNA sequencing, namely bulk- and single cell-RNA sequencing. After that we will describe a technology, which enables high throughput single cell RNA sequencing with a never before seen level of quality in the data, called micro-droplet sequencing. To be precise, we will explain an implementation developed by 10x Genomics. The data, which is generated by this technique, will be of our main concern throughout this thesis.

The following chapters will gradually describe the way from the general process of next generation sequencing to one very recently introduced technology for single cell RNA sequencing. To ensure that the differences and similarities are clear, we will always refer to the preceding chapter(s), respectively.

2.2.1. Next Generation Sequencing (NGS)

In every NGS sequencing protocol the following four basic steps are included as described by NGS technology providers ABM [ABM] and Illumina [Ill17].

1. Sample/Library preparation
   Every NGS system requires a "library", which is a collection of fragmented DNA strands. Adapters have been added to these strands through ligation (=process of joining two nucleic acid fragments through the action of an enzyme). These Adapters are needed in the following step of the process.

2. Cluster generation by amplification
   With the help of those adapters the fragments can be oriented, positioned and targeted by other components. To ensure the detection of every nucleotide within one fragment, amplification is performed. Most of the time some kind of polymerase chain reaction (PCR) is used. The results are clusters of DNA fragments, each consisting of copies originated from one library fragment.

3. Sequencing
   Through chemical reactions the sequence of each cluster (every fragment in one cluster has the same sequence, due to the fact that they are copies) is determined by optical reads. Optical signals can either be triggered through the generation of light or a fluorescent signal. We now see that the amplification of the fragments is critical for reducing errors and increasing the strength of the optical signal. The following four main DNA sequencing methods are used in NGS systems nowadays. We characterize them by briefly describing the details of the sequencing reaction. All of the methods have different advantages compared to each other, but we will not discuss them here.

- Pyrosequencing - the optical signals are triggered by the release of the pyrophosphate during nucleotide incorporation.
- Sequencing by synthesis - by using reversibly fluorescently labelled nucleotides, that cannot support incorporation of a second nucleotide due to a missing hydroxy group at the 3' end, each base of DNA strand is sequenced in a step-by-step manner. In every step the nucleotide that is complementary to the base being sequenced is incorporated. These bring a unique (for each nucleotide type) fluorescent to the site. After detection of the fluorescent signal, that corresponds in color to the base being sequenced, the fluorophores are detached and washed away and a hydroxy group at the 3' end is recreated. This process is repeated until the sequencing reaction is complete.
- Sequencing by ligation - this method differs from the previous methods, because it does not use a DNA polymerase to incorporate nucleotides. Instead it uses short oligonucleotides that are ligated to one another and drive the sequencing reaction.
- Ion semiconductor sequencing - by locating each cluster directly above a semiconductor transistor the release of hydrogen ions during the incorporation of the correct nucleotide is detected as change in the pH of the solution and the sequence of each cluster is elucidated.

4. Data Analysis

   a) The data output in its raw form, consisting of sequences of nucleotides, gets mapped or aligned to a reference genome (for example the human genome). Most of the time error correction methods are applied.

   b) Analysis is performed on the mapped data, which represents a quantification of gene levels within the considered sample. This step in a NGS workflow represents the starting point for this work. The mathematical framework and methods, which are derived, developed and discussed in the following chapters, are entirely based on this data.

Following Kulski [Kul16] NGS technologies are not only used for whole genome sequencing (WGS), but can also be adapted for the following goals:

- whole transcriptome sequencing (WTS), also called RNA sequencing (RNAseq)
- whole-exome sequencing (WES)
- targeted (TS) or candidate gene sequencing (CGS)
- methylation sequencing (MeS)

In the following chapter we will describe RNAseq in more detail, with the final goal of describing a method for single cell RNA sequencing.

2.2.2. (Bulk) RNA Sequencing (RNAseq)

The goal in RNA (ribonucleic acid) sequencing (RNAseq) is to determine the gene expression level of a sample. This is achieved by quantification of the transcripts (coding and noncoding) present in the sample, consisting of a certain amount of cells. The basic assumption is that we can investigate the gene expression level of a sample by determining how many RNA molecules of the respective genes are present within the sample. Although, there is no information about the amount of proteins synthesized with the help of the detected RNA molecules, it has proven very useful in different scientific fields to investigate the gene expression level through RNAseq. Compared to older technologies as microarray analysis it provides a more precise and sensitive way to quantify gene expression levels. [Kul16]

To leverage the previously introduced NGS standard workflow for RNAseq, we have to take into account the fact that RNA, compared to DNA, is single-stranded and less robust. Therefore, we need a way to generate a more robust, DNA-like, structure which preserves the information of the respective RNA molecule. The solution to this problem is called complementary DNA (cDNA), which is synthesized DNA based on RNA by reverse transcription.

One important remark on the sample preparation part has to be made at this point, to better distinguish between RNAseq and scRNAseq. In RNAseq we have to extract the RNA from the cells within the sample by lysis and some quality enhancing steps (e.g. purification). Therefore, we end up with all the RNA, present in the whole sample, mixed into one sample ready for cDNA synthesis. This strategy makes it impossible to trace each RNA molecule back to its cell of origin.

With that, all the requirements are fulfilled and the standard NGS workflow can be used to quantify gene expression levels. The result of such a process are “average” gene expression levels of the cells within the sample, or in other terms, we get one gene expression value per gene, within the whole sample. Therefore we quantified the gene expression of the sample, but not the individual gene expression level of every single cell within the sample. This brings us to the idea of single cell RNA sequencing.

With the advent of single cell RNA sequencing, the term RNA sequencing for this approach was not precise enough anymore. Therefore, it was changed to ”bulk-RNAseq” due to the nature of the method, using the mixed bulk of all RNA molecules within the sample, compared to scRNAseq. From this point on we will be using bulk-RNAseq to refer to this method.

2.2.3. Single Cell RNA Sequencing (scRNAseq)

In bulk-RNAseq, as we described above, RNA is extracted from a sample, a mixture of many individual cells, to quantify the "averaged" gene expression levels of the entire cell population within the sample. In a recent article in Nature [Now17], this tactic is compared to a fruit smoothie. By the color, taste and smell we can get an idea of what is in there, but a single blueberry or a few more, can be impossible to detect or easily overlooked. Here the idea of single cell RNA sequencing (scRNAseq) comes into play. This approach would relate to having a fruit salad instead of a smoothie. In that case you can distinguish not only the blueberries from the apples, but also from different types of berries. This method promises to detect a lot of masked or hidden cellular variation, which could lead to the discovery of new cell types or biological mechanisms. One global effort, similar to the Human Genome Project in regards to genetics, is to build a Human Cell Atlas [RTL+17], with the help of scRNAseq.

As a result of this approach the generated data differs drastically from the previous generation. Now every cell within a sample, several thousand, have their own gene expression values. Therefore, instead of one value per gene we generate one value per gene and cell. This leads to a never before seen increase in volume and complexity, due to the high dimensionality and at the same time sparseness of the data. The sparseness is especially tricky, because it can originate from the fact that most cells only express a certain set of genes or that many transcripts present are not detected with the current sensitivity achieved. To ensure high quality research and comparability within efforts as the Human Cell Atlas project, we need standards and (mathematical) rigorous ways to analyze this new type of data. These challenges represent the main subject of this work.

Compared to bulk-RNAseq the sample preparation is more sophisticated, due to the fact that the cells have to be separated in some way to enable an individual analysis. To ensure this "cell-capturing", different technologies and methods have been developed. These can be sorted into three main categories: cell-sorting based, micro-well based and micro-droplet based. Every method has its merits and drawbacks, which we will not discuss any further. Instead we will focus on an implementation of the last category, the micro-droplet based approach, by the company 10x Genomics, and give a description in the following chapter.

2.2.4. The 10x Genomics Single Cell Capturing Technology

Following Zheng et al [ZTB+17] and 10x Genomics [10x16b] we will now describe the micro-droplet based cell capturing technology introduced by 10x Genomics. Similar to bulk-RNAseq we have to focus on the steps before the NGS standard workflow, which we presented in the preceding Section 2.2.1. Additionally, to the extra steps from bulk-RNAseq, extraction of the RNA and synthesis of cDNA, it is crucial to

treat the cells individually to ensure that the generated data represents single cell level gene expressions. To achieve this goal, 10x Genomics developed a step-wise approach to capture and tag the individual cells. Furthermore, they incorporated a way to get rid of amplification biases, which would occur during the standard NGS workflow. For this procedure 10x Genomics developed a machine, called Chromium. We will now describe the sample preparation steps with the help of the 10x Genomics Chromium machine.

The steps from the sample (cells) to the cDNA molecules, which are ready for the NGS standard workflow, are shown in Figure 2.2 on page 12 and are as follows:

1. Load oil, sample and **10x gel beads** on a microfluidics chip from 10x Genomics.
2. With the help of the Chromium machine cells are pumped into microfluidic channels, which are a little broader than the cells to avoid parallel transport.
3. At the first orthogonal intersection, cells and **10x gel beads** are brought together, to form pairs (ideally).
4. Each pair reaches the second orthogonal intersection and micro-droplets, called GEMs (Gel Bead in Emulsion), are formed in an oil solution.
5. Inside of these micro-droplets the cells are lysed, the RNA transcripts get captured and tagged followed by a reverse transcription (RT), which yields the desired cDNA.
6. At last, remove oil and proceed with the standard NGS workflow, presented before, by using sequencing-by-synthesis in step 3., provided for example by Illumina machines.

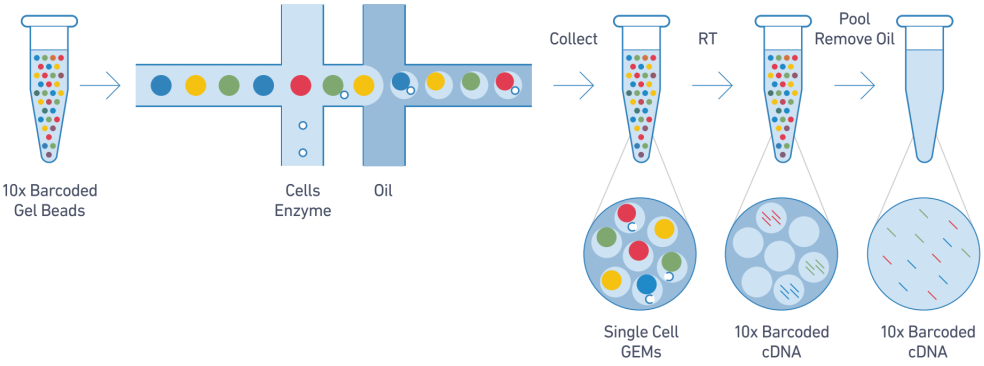

Figure 2.2.: Formation of GEMs, RT takes place inside each GEM, which is then pooled for cDNA amplification and library construction in bulk [10x16b]

Obviously, the most interesting and important reactions happen in step 5, that is why we will take a closer look at the 10x gel beads and all of its components.

The 10x Gel Beads are the components which drive the reaction and thereby tag each cell and RNA strand and synthesis the cDNA molecules. As shown in Figure 2.3 on page 13 they host a multitude of oligonucleotides (=short DNA or RNA strands with different applications) which consist of 4 parts. We will focus on the **10x barcode** and the **UMI** components, because the R1 sequence serves as primer-site for the amplification and the poly-T-sequence is used to capture the RNA transcripts.

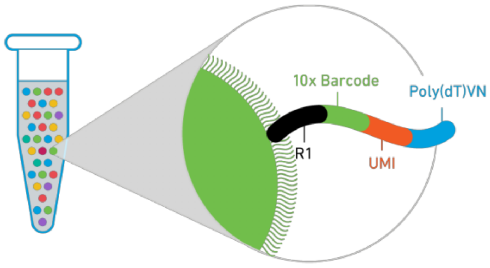

Figure 2.3.: 10x Genomics Gel Beads. Figure adopted from 10x Genomics

The 10x Barcode is a sequence, which is unique for every gel bead. Thereby it is afterwards possible to assign the RNA molecules back to their cell of origin.

Unique Molecular Identifier (UMI) these randomly generated sequences are individual for every transcript and represent in combination with the 10x barcode a truly unique way to tag each RNA molecule and trace it back to its cell of origin. Not only is it used to assign the correct RNAs to a certain cell but also helps in getting rid of a bias, which would be introduced through amplification. What we mean by that is that due to the fact that every RNA molecule has an unique tag (10x barcode + UMI) it is possible to trace back each copy made by amplification to its RNA molecule of origin. Thereby it is ensured that in the end one UMI count does really represent one RNA transcript within a certain cell. This is very important, because other methods do not account for that and take the total number of RNAs aligned to one cell and gene from the sequencer result. Here, with the help of UMIs, this problem is solved by introducing an aggregation step after the alignment of the sequencing result.

To sum up, barcodes and UMIs are crucial to the process of linking the RNA transcripts back to their original cells and accounting for amplification bias during the NGS workflow.

With this technology up to 10.000 cells can be captured per sample, with a capture efficiency rate of approximately 50 %. Therefore scRNAseq gets more interesting for industrial research applications rising above the purely curiosity-driven research

by academia. We enter a new era for (sc)RNAseq, because of the high throughput, quality and affordability this technology entails.

2.2.5. Challenges in scRNAseq

Although this technology provides a state of the art solution to quantify gene expression on a single cell level, there are some challenges due to the nature of the process. In the following we want to point out just a few:

- It can happen that micro-droplets form, which only contain a bead, a cell or remain empty.
- Doublets or multiplets, more than one cell is captured by a micro-droplet, can occur. To test for the amount of multiplets experiments with samples from two different species get mixed and analyzed together. Thereby the multiplet rate can be determined. Multiplets are very difficult to detect during the analysis of the data due to the fact that transcripts originating from cells within the same droplet are assigned to the same cell in the data.
- Free floating RNA due to already lysed cells or contaminated samples falsify the results to a certain degree.
- Dying cells can result in drastically shifted gene expression levels.
- Due to the fact that most of the genes are not expressed in a cell, we end up with a lot of gene expression levels of zero. This inflates the data immensely.
- The procedure is very sensitive to biological (contamination, storage, handling) and technical (anything else) noise.
- Only 5-15% of transcripts within a cell are captured, resulting in false zero-levels (sometimes called dropouts) of expression for certain genes.
- Cell types have different amounts of RNA content, making generalizing normalization difficult for samples consisting of more than one cell-type.

Some of these issues translate directly into challenges we are facing in the analysis of scRNAseq data. Therefore, this work is dedicated to give such an analysis a rigorous framework to operate in.

2.3. The Analysis

To understand what the goal of the analysis is, we have to get a sense of the position of the analysis in the big picture of single cell RNA sequencing (scRNAseq). Between the biological part of an experiment (the setup, the experiment itself and the sequencing) and the interpretation and discussion of the results, we can find the right spot for the analysis in the whole procedure. Simplified we could see the process before the analysis as data generation and the steps afterwards as discussion of the results presented by the analysis. The analysis itself, of course, depends on the kind of data and the question at hand. Although the analysis is the last component in the whole procedure, it extracts the relevant information out of the generated data and supplies it with meaning, context and a certain confidence concerning the accuracy of the statements, which can be derived.

The goal of this chapter is to establish an understanding of the procedure of scRNAseq analysis as it is presented in the literature, for example by Stegle et al [STM15]. We are trying to give the reader a comprehensive description of the steps which are involved in a standard scRNAseq analysis workflow. We do not want to present a manual or instruction guide on how to build a workflow for processing such data, rather a sense of the steps, which have to be taken to ensure the desired result.

Every chapter will describe one major component of the analysis. The order of the chapters is by design rather a recommendation of the way the data should be processed. We try to describe every step qualitatively. No methods will be explicitly mentioned, because the scope of this chapter, as stated above, is purely to give a qualitative feel for the process, in which the mathematical methods are implicitly integrated. The mathematical methods themselves will be explained in detail in the following chapters in relation to what is presented here as a generic workflow. To ensure that these conditions are met, we will answer the following questions in every chapter and try to follow the same structure as we move on.

- What is the aim of this step? What should be the result?
- Why is it necessary for the analysis? What is the problem at hand?
- How can it be achieved?

2.3.1. Goals of the Analysis

Before we start describing the actual steps, which have to be taken in the analysis, we have to ask ourselves: What are potential goals? This is important to keep an open mind concerning the methods, restrictions or decisions, which will be applied in the different steps on the data. Generally speaking it always depends on the scientific question presented, before conducting an experiment. Here we try to give some examples of potential goals in single cell RNA sequencing experiments, reaching from general biological to very specific medical or pharmaceutical research questions:

- characterization of cell types by differential gene expression analysis
  - e.g. biomarker identification for diseases
- identification of new cell populations or subpopulations
  - e.g. target and binding site identification for pharmaceutical agents
- cell cycle/reaction development by pseudotime analysis
  - e.g. investigate therapy effects

Most of these goals are based on the fact that there has to exist quality controlled data and a stable clustering of cells by their gene expression. Without these two prerequisites any further downstream analysis is done on a very weak basis. Therefore, we intend to present solid mathematical arguments to establish a sound basis for further investigation of scRNAseq data.

2.3.2. Data

It all starts with the data we want to analyze. There is no general rule how data from RNA sequencing looks like, but for scRNAseq a few statements can be made. First of all, we have to remember how the data were generated. After the sequencing process a few data format conversions happen (which we will not discuss in this work, because they heavily depend on the machines used) and a mapping takes place. In the mapping step the read nucleotide chains are mapped to a given genome for example human or mouse. This is necessary, because the sequencing instrument is not able to identify species-specific sequences by itself, it is only capable of detecting the order of nucleotides. After mapping a correctly sequenced string of nucleotides to the associated gene, a gene by cell matrix is generated. Every row is a gene and every column a cell. The values in this matrix are the gene expression level of the respective cell and gene. This matrix represents the basis of the analysis.

At this point it should be stated that it can happen that more sophisticated methods are at work, depending on the implementation of the single cell RNA sequencing process. For example, in micro-droplet sequencing we can distinguish between different cells, because of a separation process applied to the cells prior to sequencing. As we have discussed earlier in Chapter 2.2.4, the sample preparation machine Chromium from 10x Genomics tags not only each RNA molecule with the information of the cell it is related to (= 10x gel barcode), but also every RNA inside of each cell gets its individual tag, called UMI (=Unique Molecular Identifier). Both tags together yield a unique identifier for each RNA, which additionally relates to exactly one cell. With the help of these tags it is possible to account for duplicates introduced by the amplification process during sequencing. Therefore, we do not get just the "raw" counts (= number of RNA molecules related to the same gene per cell **after** the amplification process), but the UMI counts, which represent the actual number of individual RNA molecules expressing the same gene per cell.

Therefore, the actual analysis starts with a gene by cell matrix filled with UMI counts. We will refer to it as **expression matrix** or **count matrix**. Due to the fact that every cell expresses a different amount of genes and technical errors occur during the sequencing process, we introduce a lot of zeros into the data. The exact mechanisms behind some cell-specific dropout profiles are still unknown. Therefore we speak about sparse matrices, when we talk about scRNAseq data. There are a lot of consequences resulting from this fact, which we will discuss throughout this work. The dimensions of a given expression matrix can vary according to the used genome and the number of detected cells. For example the human reference genome 19 (hg19) used by 10x Genomics consists of 32.738 genes. Sample sizes can vary between a few hundred to tens of thousands of cells. One last aspect concerning the input data for the analysis is **annotation** or **meta data**, here we talk about information which is specific to each cell. A good example is the processing batch of the sample or the donor from which the sample was taken. This information will be relevant during the analysis, because most often these variables influence the data heavily. This is not necessarily a bad thing, because sometimes we want to see the different effects in the data (e.g. donor), but there is also the case where we want to get rid of this influence (confounding factors, e.g. batch).

2.3.3. General Workflow

According to the literature there are certain steps, which should be taken to get in the end what we desire: interpretable, corrected data which can be used to answer the questions at hand with a certain degree of confidence. These steps are summarized in the following chapters and are presented as a workflow through which the data travels in consecutive order.

Every step involves mathematical methods, which we will present and discuss in detail in Chapter 3. Here, we want to give an overview of the tasks in scRNAseq analysis in a qualitative manner and what kind of questions should be answered by the mathematical methods applied.

2.3.4. Quality Control

What is the aim of this step? What should be the result? As this is the first step after the data was generated we have to deal with the fact that there has not been done any filtering and the data, a gene by cell matrix, is in its rawest form, at least from an analytical point of view. Sometimes we will refer to this state as the original or just the count matrix. In this step we try to get rid of genes, which do not convey meaningful information, and filter out cells, which are either not relevant, seem damaged or have suspiciously high or low (UMI) counts. At the end of this step we should have removed all the cells and genes, which are not relevant to the analysis due to low quality or lack of informative value. Sometimes cells with very

high UMI counts will be filtered, because an error in the sequencing process or the prior preparation process might have occurred. For example more than one cell was captured within one micro-droplet or free-floating RNA could have contaminated the micro-droplet, which resulted in an artificially high UMI count. These cells would distort the data immensely and are therefore excluded from the downstream analysis.

Why is it necessary for the analysis? What is the problem at hand? We need to get rid of cells and genes that do not enrich our dataset in a meaningful way. If we would skip this step a lot of unnecessary data would influence each and every other step along the workflow and the results may not be conclusive in any way. If we remove the less meaningful or simply uninteresting cells from our dataset what we will obtain are less datapoints with more meaning. In the case of genes, by the removal of less meaningful, uninteresting or low expressed ones, we reduce the dimensions of the space (feature space) in which our datapoints are embedded.

How can it be achieved? Cells can be simply controlled by filters using thresholds or more sophisticated approaches as looking for outliers in the data or investigation of the distribution either of the (UMI) counts per cell or expressed genes per cell. Genes can be filtered by looking at metrics like the count values per gene over all cells.

2.3.5. Normalization

What is the aim of this step? What should be the result? We now have the quality controlled data from the step before, where we removed irrelevant data. Now we want to account for the big differences in sequencing depth, which are usually present in the data. Basically, we want to homogenize the total counts of our cells.

Why is it necessary for the analysis? What is the problem at hand? We do this to get rid of the library size (= number of UMI counts per cell/per gene) as a confounding factor (= a variable/factor which distorts or influences the data in a non biological or unwanted way) and to ensure that the influence of some cells does not dominate the results of the methods applied on the dataset and therefore the conclusions, which are drawn by the analysis.

How can it be achieved? There are a lot of normalization methods out there, because this is also done in other sequencing analyses. The difficulty in our case is the new kind of data, which we are faced with. Some of the older methods do not work properly, because of the massively increased presence of zeros in the data. Therefore, new methods based on old ones, were developed and will be discussed in Section 3.3.

2.3.6. Confounding Factor Analysis

What is the aim of this step? What should be the result? Here we try to identify confounding factors, which are variables that influence our data in a way we do not want it to be influenced. After identifying such variables their influence has to be removed from the data. A good example is a difference between two datasets, consisting of cells from exactly the same sample, which can be mostly explained by the fact that they were processed separately from each other. This variable usually is called batch effect. If the variation in the data introduced through this batch effect is above a certain threshold, we have to remove the effect of this variable on the data. Other examples of such factors are donor, treatment or storage method (frozen, fresh, etc.) of the sample. It always depends on the question, whether a variable is identified and labeled as a confounding factor or not, as we do not want to remove an investigated effect, which can be observed in the data. In the end we want to identify all of the factors, which influence the data and remove the confounding ones.

Why is it necessary for the analysis? What is the problem at hand? Obviously the outcome of the analysis differs significantly, if the confounding factors are removed or not. This process involves a lot of challenges: Identifying confounding factors, determining correlations between them or influence on each other, deciding on which to remove and finally the removal itself. If two confounding factors, which correlate heavily with each other, are removed, the effects introduced and the results obtained are not foreseeable without a more detailed analysis of the relationship between those particular factors. Nevertheless, it is critical to identify them, to be able to make an informed decision, and then to remove a certain subset of the identified ones. Otherwise these most influencing factors dominate following steps, as for example the dimensionality reduction and/or clustering processes.

How can it be achieved? There are various ways to determine if a variable is influencing the data. Most of the methods are based on a dimensionality reduced dataset on which the analysis concerning confounding factors is done. One can determine (for example with the coefficient of determination) the influence of a given variable on each dimension of the reduced dataset. Furthermore, the relation amongst the variables has to be determined for the reason described above. The removal of the chosen factors can be done with the help of regression models, which are in turn applied on the whole dataset instead on the reduced one. We want to state, that the dimensionality reduction in this procedure was just a way to identify the most influencing factors. In the next part we apply dimensionality reduction on the obtained confounding factor free data.

2.3.7. Dimensionality Reduction

What is the aim of this step? What should be the result? After this step we should have a smaller feature space in which the datapoints of our dataset, the cells, are embedded, without losing relevant information concerning the goals of the analysis. With less dimensions to worry about, and having determined the most meaningful ones, a lot of computational efforts (both quantitative and qualitative ones) can be reduced and the conclusions are more precise. At this point it should be stated that dimensionality reduction also finds its application in the confounding factor analysis, as most of the analysis is done on a reduced space to determine which factors heavily influence the most important dimensions. As we can see this step is of integral importance in various cases.

Why is it necessary for the analysis? What is the problem at hand? In the reference genome hg19 of the human genome we are confronted with more than 32.000 genes, which translates to exactly the same amount of dimensions. Most of the time, depending heavily on the dataset at hand, we can reduce the feature space itself to roughly 10.000 genes, but with the help of different dimensionality reduction methods we can aggregate some of them to be able to reduce the number of dimensions even further. It is important to keep in mind that one dimension, after a dimensionality reduction method was applied, does not have to relate directly to one gene. Of course, if a certain gene is very important (e.g. displays a high variance throughout the dataset) it still can happen, but most of the time the newly acquired dimensions are a combination of various genes. Further operations on the reduced dataset are more easily done and the results yielded convey more meaning. However, one should be aware that calculations done on a reduced dataset do not include all the aspects of the whole dataset, but only the most "relevant" ones.

How can it be achieved? There are various different methods, which we will discuss later, ranging from classical deterministic ones to stochastic ones. Some of them occurred in the literature at the same time as an increase of interest in data science by the industry was noticed. Basically it always comes down to determine the lowest number of (new) dimensions, which can explain most of the information within the data.

2.3.8. Clustering

What is the aim of this step? What should be the result? In general we are talking about a typical clustering problem. We have data points (cells) which are described by a certain set of features (genes) and our goal is to find clusters of cells, which are similar to each other in regard of their gene expression values. Most of the time the goal of the clustering process, biologically speaking, is to group them

by cell population or rather cell types. Of course, mathematically speaking, the concept cell type (which can be described through gene expression) is not conveyed by the data. So it can happen, and this is very interesting when it does, that the clustering happens due to other characteristics of the cells embedded in the gene expression information. As we have discussed above already, confounding factors can be such an information dominating the clustering process and we could for example be confronted with a classification by batch effect. Therefore we remove these factors, by regressing them out, to get to the bottom of the relevant information on the research question, which is contained in the processed data.

Why is it necessary for the analysis? What is the problem at hand? This step can represent one possible endpoint of a scRNAseq analysis and is therefore of high significance for the presented process. After a successful clustering for example by cell types the interpretation of the results can be initiated and an attempt to answer the scientific question, which stood at the beginning of the experiment, can be made. It also represents a solid basis for discussing the potential findings or to further explore the dataset in a different way to discover another solution to the current problem.

One of the biggest problems in this step, which we already touched upon briefly before, is that the mathematical methods or algorithms applied on the data do not concern themselves with the goal of the analysis. What we mean by that is, that the result of a clustering attempt does not necessarily represent a classification by cell type or cell population, although it may have been the desired outcome. If there are other effects present in the data, or to be more precise in the gene expression values of the cells, this could influence or even dominate the clustering process immensely.

Another problem is the huge amount of different methods which are present in the literature. This directly results in the question: which method should we chose or which is the best for our problem? There is not a clear answer to that, because every method or algorithm has its justification and therefore yields a “correct” result in the sense of their approach. One could argue that it makes sense to use more than one method with different parameters and then rather focus on the analysis of the results than on the methods. This is exactly what we try to achieve later in this work. Confronted with an amount of different clustering results we try to find a quality measure for comparing them and deciding on the “best one”. Although “best one” will be up for discussion, as we have already mentioned above, it strongly depends on the questions we want to answer.

How can it be achieved? Clustering of vast amounts of data generally speaking is not a new problem. Nevertheless in the immediate past the interest in this problem went hand in hand with the development of machine learning algorithms, which mainly concern themselves with classification problems. Furthermore, the novelty of the increase in scRNAseq data and its peculiar characteristics motivated the

scientific community to increase research efforts to address this problem in particular. Therefore, we have quite a lot of methods at our disposal, which makes the task of finding the qualitatively best clustering result not easy.

2.3.9. Visualization

What is the aim of this step? What should be the result? One could argue that we are confronted with the same challenge as with dimensionality reduction, but there is actually more to it. The goal is to inspect the data visually and therefore we need to reduce the dimensions even more drastically than before, because usually humans have problems visualizing and comprehending more than three dimensions in space. As a result we would like to get a two or three dimensional image of our data which conveys potential findings. With the goal of clustering by cell types for example we would like to see that the calculated clusters can also be found with the help of appropriate visualization methods. Of course there are a lot of other ways to visualize scRNAseq data than plotting the datapoints (cells) on two or three axis of a dimensionality reduced space. Heatmaps to visualize gene expressions or histograms to analyze the distribution of counts come to mind.

Why is it necessary for the analysis? What is the problem at hand? Visualization helps to understand or find structures present in the data in a qualitative manner. This seems at the first glance as something of minor relevance but sometimes data with exactly the same statistical metrics as mean or variance can look very different when plotted. So, it truly is a valid way to analyze the data from a new perspective or at least to check if the understanding of the data by quantification (metrics) resembles the one by a qualitative analysis. Apart from this reason, plots and graphs can be a lot easier communicated and presented and therefore at least the presentation of findings or evidence should be done with the help of proper visualization. One thing has to be always kept in mind: most of the visualizations are drastic simplifications or represent only one perspective. Nevertheless, it is a useful and necessary tool, especially if one is aware of the potential pitfalls.

How can it be achieved? To answer this question we, as often before, have to know the final goal of the analysis. Therefore, no general rule exists. Nevertheless, there are methods, which were created exactly for the purpose of visualizing great amounts of data in reasonable time and in a meaningful way.

3. Mathematical Methods in scRNAseq - Analysis

"The Book of Nature is written in the language of mathematics."
Galileo Galilei

In this chapter we give a qualitative analysis of the methods used in single cell RNA sequencing analysis by analyzing the mathematical aspects behind the applied best practice methods. These methods are used on different occasions within a scRNAseq analysis workflow, such as the previously presented workflow in Chapter 2.3.3. Therefore, we focus on the mathematical methods and not necessarily on the steps in the analysis, as we did in the last chapter. That is why the structure of this chapter is driven by the mathematical categorization of the methods.

The following chapters start with stating the main goal of the following methods and where they are applied within the analysis. At the end we compare, if necessary, the presented methods and put them into context concerning their use in scRNAseq analysis. Every method will be presented by a detailed mathematical description followed by an explanation on the legitimation of applying it on scRNAseq data. Some of those methods were not necessarily developed for scRNAseq data, but prior to the rise of this new field. They still yield robust and valid results and therefore should be mentioned.

We want to emphasize that there is no claim for completeness due to the fact that there are a lot of alternatives for each of the presented methods and the amount of methods is constantly increasing. The goal is to show what the mathematical background of one possible single cell RNA sequencing workflow is, which we believe yields solid results. Of course, we try to give some valid alternatives, which should be part of a stable workflow. Furthermore, we try to embed everything in the a general mathematical framework, which we setup and expand throughout this work. Therefore, every mathematical description follows the same notation, which gets introduced in the next chapter and is extended as needed. This shall aid in the understanding of the concepts and discussions on different methods.

Examples, to illustrate the described methods will be provided throughout the following chapters. Furthermore, we will use specific plots at the end of every chapter to present the results of the recommended methods. For this purpose a simulated dataset is used. Further information on its origin, a detailed description and the R-packages used to process and plot it, are provided in Appendix A on page 115.

3.1. Introduction to the Mathematical Framework

The following definitions will be used throughout this work. The starting point for every analysis is the data, in our case an expression matrix. To analyze the different methods we have to mathematically describe what an expression matrix in our case is. Before we do that, we establish what this matrix describes, therefore we have to define the meaning of its dimensions. First we look at the genes, which expression levels we want to analyze.

Definition 1. The **gene set** G consists of all the genes g_i in the reference genome of the species, where our sample is from.

$$G = \{g_i | i \in I\},$$

where $I = \{1, \ldots, m\}$ and m is the number of different genes in the used genome. This set is always the same, if the samples and thus the datasets are from the same species.

Next we define the cells, whose gene expression levels are of interest.

Definition 2. The **cell set** C consists of all the cells c_j, which make up our dataset or the sample, respectively.

$$C = \{c_j | j \in J\},$$

where $J = \{1, \ldots, n\}$ and n is the number of cells detected in the sample at hand. This set is always unique because no cell can be processed twice in single cell RNA sequencing.

Finally we can put it all together and arrive at a definition for the starting point of our analysis.

Definition 3. We define an **expression matrix** M as a gene by cell matrix. Every entry m_{ij} is the detected expression level of a certain gene g_i in a certain cell c_j.

$$(m_{ij})_{i \in I, j \in J} = M \in \mathbb{R}^{m \times n}$$

Depending on the subject we want to discuss we will use either the format c_j/g_i or j/i in sums and similar expressions, where sets of cells or genes or their respective indices are needed. This very general definition holds true for every of the following chapters, but in the beginning, before we manipulate the data in any way we are actually confronted with $m_{ij} \in \mathbb{R}_{\geq 0} = \{x \in \mathbb{R} \mid x \geq 0\}$, because there can not be a negative expression value. In the course of this chapter we manipulate this matrix heavily and sometimes introduce negative values, for example by removing the influence of an explanatory variable from the data.

In the case of data generated by the 10x Genomics micro-droplet sequencing technology with the help of the 10x Genomics Chromium machine we can even further

restrict our definition, of the data immediately after the sequencing procedure, to $m_{ij} \in \mathbb{N}^0 = \{x \in \mathbb{N} \cup \{0\}\}$. The elements m_{ij} are the UMI counts, the number of different RNA molecules originated from the same gene in a certain cell, which we introduced in Chapter 2.2.4 on page 13.

Now we want to describe the sparseness present in the data. Therefore we define the sparsity of a matrix.

Definition 4. We define the **sparsity** sp of an expression matrix $M \in \mathbb{R}^{m \times n}$ as the number of zero values divided by the total number of elements.

$$\text{sp} = \frac{|\{m_{ij} \in M \mid m_{ij} = 0\}|}{m \cdot n},$$

with $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

We can also account for the inverse information, the density.

Definition 5. We define the **density** de of an expression matrix $M \in \mathbb{R}^{m \times n}$ as the number of values greater than zero divided by the total number of elements.

$$\text{de} = \frac{|\{m_{ij} \in M \mid m_{ij} > 0\}|}{m \cdot n}$$

Or with the definition of the sparsity

$$\text{de} = 1 - \text{sp},$$

with $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

It is not uncommon to encounter datasets with sp = 0.7, implying de = 0.3, for example. This would mean that 70% of the data consists of zeros. Most of the time the sparsity is even higher (around 0.9).

The dimensions of M (m and n) can vary a lot and are dependent on the species investigated and the size of the sample respectively. Examples for human experiments could be 32000 genes and 3000 to 33000 cells.

As we can see the number of zeros in the data represents by far the majority of values. Therefore, the question of imputation is often raised by statisticians. Of course, it can be reasonable to impute most of the zeros, because they could be a result of shallow sequencing or technical errors. In our case, micro-droplet sequencing, we do not resort to this method, because of the nature of the sequencing process. Reminder: The data we intend to analyze consists of UMI counts, which convey the exact number of RNAs, and therefore expressed genes, present in the cell. Thats why a zero in this case carries more weight compared to zeros yielded by other methods. It describes the fact that there was no RNA molecule corresponding to that gene

detected in the cell of interest, either because of absence or shallow sequencing. As we are not able to distinguish between those cases, we do not resort to imputation. Naturally technical errors can occur, but we always have to make assumptions when dealing with biological matters.

Most of the time we are interested in the non-zero values of the matrix M. The range of these values depends, again, on the sequencing process. In our case, micro-droplet sequencing done with a 10x Genomics Chromium machine, we mostly encounter values between one and a few thousand, which are always elements of $\mathbb{N}$.

At the end we wanted to briefly discuss the distribution of expression values per cell, gene and expressed genes per cell. In all cases we have seen a dependency between the used cells, to be more precise the cell types present in the sample, and the distribution of the count values. What we can say is that we usually encounter a normal distribution in the first and the last case. Concerning the counts per cell and expressed genes per cell we observed always a heavy tail on the low end of the spectrum and no tail at the high end. Here, we have to mention that certain samples (for example with tumor cells in it) usually yield an additional peak in the higher end of the spectrum, due to the fact that a certain cell type expressed a lot more genes than the rest of the cells in the sample. In the case of counts per gene we see a peak at the low end, because a lot of genes are simply not expressed in the cell types present in the sample, and another peak further along the spectrum, depending on the sample.

In general there are no other special structures (banded, symmetric, diagonal,...) present in the expression matrix apart from the ones mentioned above.

For better understanding we display some properties of the simulated dataset, before any method is applied on it, in Table 3.1 on page 26.

dataset	#cells	#genes	sp	de
sim	3000	32000	0.948	0.052

Table 3.1.: Original dimensions, sparsity and density of the simulated dataset

3.2. Outlier Detection

Defining exactly which datapoints should be considered outliers is a difficult job. Unfortunately there is no consensus in the literature or community concerning a rigorous definition of the term outlier. Wether or not a datapoint (in our case a cell) should be called an outlier is ultimately a subjective decision. Nevertheless, outlier detection is sometimes complementary used in the quality control steps of scRNAseq analysis, most of the time positioned right at the beginning of a workflow. The goal is to determine cells, which are not consistent with the rest of the sample. This can be due to technical errors or biological circumstances as dying or damaged cells.

According to Filzmoser et al [FMW08] there are two approaches for outlier detection, namely distance-based methods and projection pursuit.

Distance-based methods are based on the idea to detect outliers by measuring their distance to the center of the data or a set of datapoints. Therefore, the first thing needed is a distance measure. Often a robust version of the Mahalanobis distance is used.

Definition 6. We define the **robust Mahalanobis distance** as

$$RD_i = \sqrt{(x_i - T)^T C^{-1} (x_i - T)},$$

where $x_i \in X \subset \mathbb{R}^p$ is the datapoint in question, T is a robust measure of location and C is a robust estimate for the covariance matrix

Following this strategy we need reliable estimators for T (e.g. the coordinate-wise median) and C (e.g. the median absolute deviation - MAD). Furthermore, the decision on the distance RD_i at which a point should be called an outlier represents a major challenge. All of those tasks are non trivial to solve.

Projection pursuit methods have the goal of finding a projection, which represents high dimensional data in the most meaningful or interesting way. In this case the goal is to find a projection in which the outliers are readily apparent. With this it is possible to yield robust estimators, which can be used to identify outliers. The advantage of this family of methods is that they can be applied on any kind of data, without assuming that the data originates from a particular distribution. This flexibility comes with the price of a huge computational burden, because all possible projections have to be explored.

The Problem of the currently implemented approaches in both categories is that they have issues in dealing with high dimensional data (in the range of thousands and not hundreds of dimensions), as it is the case with scRNAseq data. In an effort to overcome these obstacles a more sophisticated and commonly used way to identify and get rid of outliers in very large high dimensional datasets is presented by Filzmoser et al [FMW08]. The presented algorithm, called PCOut, basically first detects location outliers and then detects scatter outliers. The difference between those outlier types is that the former ones possess a different scatter matrix compared to the rest of the data, while the latter ones are described by a different location parameter. Finally, it combines the weights resulted from the previous steps to determine final weights for all datapoints to identify outliers. A detailed description of steps involved in this procedure would go beyond the scope of this work. Additionally we will not discuss these approaches any further, because their application in scRNAseq analysis is very limited due to the nature of the data.

With scRNAseq data we are often confronted with quite a lot of "outliers" or low quality cells, which should be marked as outliers. These cells do not enrich the dataset in a meaningful way, on the contrary they are basically noise. The presented methods have not proven to be considered reliable in discarding such damaged or low quality cells as stated by Ilicic et al [IKK+16]. They even go one step further and conclude that most of the time the low quality cells form dense clouds and thereby outlier detection algorithms treat them as high quality datapoints. According to them the best way to get rid of low quality cells is by visualizing the data with the help of principal component analysis or other dimensionality reduction methods to locate these low quality clusters.

Therefore we propose to use cut offs instead of sophisticated outlier detection methods in scRNAseq analysis.

Cell Filtering by different metrics is one efficient way to get rid of low quality cells. Some commonly used metrics for each cell are:

- total number of UMI counts (with upper and lower limits)
- total number of expressed genes
- percentage of a certain set of genes expressed, for example mitochondrial genes (can be an indicator for cell viability)

The difficulty lies in the selection of the limits and most of the time it depends on the sample itself. In a homogenous sample (all cells have the same cell type) for example the total UMI count is expected to be more or less the same across cells. By looking at the distribution of these metrics the selection of a limit gets easier. We are often confronted with heavy tails in the distributions, which sometimes should be removed.

Gene Filtering does not technically leverage outlier detection, but it is still very useful to first get rid of non informative dimensions e.g. genes. This can be done by simply removing every gene that was not expressed by any cell or more sophisticated by defining the term expressed in a deterministic way. This can be done for example by using the following condition: a gene is considered as expressed if at least two cells with at least two UMI counts each, in this particular gene, exist within the dataset. Or, mathematically speaking, we would define the set of filtered genes G^* as a subset of all genes G restricted by the described condition.

$$G^* = \{g_i \in G : |\{c_j \in C : m_{ij} \geq 2\}| \geq 2\}$$

In conclusion it can be said that outlier detection complements the quality control process, but is not necessary, because filtering with the help of thresholds yields better results. Therefore, after such a quality control step we end up with a reduced

set of genes G^* and cells C^*. Consequently, the reduction of low quality cells and not "expressed" genes leads to a significant decrease in sparseness, which benefits further analysis steps downstream. For the sake of simplicity we will use the symbols G and C for the reduced sets, on which we perform the following analysis of methods.

To illustrate the change in the data established through the described actions we display the same properties as before of the "quality controlled" simulated dataset in Table 3.2 on page 29.

dataset	#cells	#genes	sp	de
sim	2995	6428	0.745	0.255

Table 3.2.: Updated dimensions, sparsity and density of the simulated dataset after some quality control steps

3.3. Normalization

In this section we will describe a lot of different methods which are used for normalizing the data. Most of them are deterministic and therefore yield always the same result, but one is stochastic and will be labeled as such. The goal is to make data comparable within itself to eliminate cell-specific biases. So we have to account for highly expressed genes as well as cells with unusually high UMI counts compared to others.

Basically we do not want the outcomes of methods, applied on the data further down the analysis, to be influenced by only a few genes or cells respectively. It is very important to ensure that the starting point (normalized data) for most of the methods is meaningful. We will denote the normalized matrix with M_{NORM} and its items with $(m_{ij})_{NORM}$.

One could argue that the total number of counts per cell would be a good factor to normalize the data by, we will even see that this method is used in the counts per million approach. However, real data shows that a few highly and differentially expressed genes may influence this total number more and thereby causing it to be not enough of a normalization measure.

3.3.1. Standard Methods

Two very common, intuitive and easily implemented methods are applying a logarithm on the data or calculating the counts per million values of the expression matrix M. Oftentimes they are used in succession as a combination.

Logarithm

Most of the time either the logarithm with the basis $b = 2$ or $b = 10$ is used. This is easily done by first defining an offset c which is added to M to account for the zeros in the data, which would yield $-\infty$ as result, and second applying the chosen logarithm.

$$(m_{ij})_{NORM} = \log_b (m_{ij} + c)$$

Counts Per Million (CPM) - Basic Normalization by Library Size

This is the first real normalization method in this work and commonly used for the normalization of sequencing data of any kind. It also is a representative of a bigger group of methods called normalization by library size, where the main goal is to use the total number of (UMI) counts of each cell for the normalization. In this case we use it to scale the counts such that the total count per cell is the same across all cells. The idea of this method is closely related to already established ones in similar fields, namely nucleotides per million (NPM), transcripts per million (TPM) or reads per kilobase per million mapped reads (RPKM) described by Li et al [LRS+09].

Definition 7. We define the **library size** or **total number of counts** N_j of cell c_j by

$$N_{c_j} = N_j = \sum_{i=1}^{m} m_{ij} \tag{3.1}$$

Now we put the count value of one cell concerning one gene in a relative context of the total number of counts, which were found in that cell.

Definition 8. To obtain the **counts per million (CPM) normalized values** we divide each value m_{ij} of the expression matrix M by the total number of counts found in the cell N_j and then multiply it by a million.

$$(m_{ij})_{NORM} = \left(\frac{m_{ij}}{N_j} 10^6 \right) \tag{3.2}$$

Other scRNAseq technologies sometimes yield much higher count values, because they do not posses the means to account for duplicates introduced by the amplification process as the 10x Genomics micro-droplet sequencing technology with the help of UMIs does. Therefore, this approach is much more meaningful for expression data resulting from those technologies. The results of these simple normalization methods by library size can be dominated by a few highly expressed genes, which in turn influence the downstream analysis. Nevertheless, it is still a standard on which the community has agreed on and therefore important to mention and essential for

the following comparison. Furthermore, some of the following methods build on this approach.

The Combination

The combination of those methods is also a very popular and straight forward technique to normalize the data, that's why we will define also this approach for the sake of completeness.

Definition 9. Normalization by the combination of the logarithm and CPM is performed by the following calculation.

$$(m_{ij})_{NORM} = \log_b \left(\frac{m_{ij}}{N_j} 10^6 + c \right)$$

3.3.2. Size Factor (SF)

Now we start explaining the more sophisticated normalization procedures. Before single cell RNA sequencing there was bulk RNA sequencing, as we explained in the beginning of this work. That is one reason why a lot of the following methods were developed in the first place. One popular method among them was introduced by Anders et al [AH10], who use particular size factors for the normalization. This method has two other names, which are also often used in the literature: DESeq- and Relative Log Expression-normalization.

Definition 10. We determine the **size factor** sf for each cell by calculating the median over all genes of the result of the count value of the cell, concerning the current gene, divided by the geometric mean of the counts per gene. In other words we define the size factor per cell as the median of the ratio between the count value and a pseudo-reference of the respective gene, latter results from the geometric mean of the counts per gene.

$$sf_j = \underset{i \in I}{median} \left(\frac{m_{ij}}{\left(\prod_{k=1}^{n} m_{ik} \right)^{\frac{1}{n}}} \right) \tag{3.3}$$

Together, the gene-wise pseudo-references (geometric mean in the denominator) can be seen as a reference pseudo-cell used to normalize against.

The yielded size factors should **multiply to one**, therefore we apply one additional step after the size factor calculation.

$$sf_j := \frac{sf_j}{\exp\left(\frac{1}{n} \sum_{k=1}^{n} (\log sf_k) \right)} \tag{3.4}$$

Usually this step is followed by a **rescaling** of the factors by an average measure like the mean or median.

$$sf_j := \frac{sf_j}{\frac{1}{n}\sum_{k=1}^{n} sf_k} \tag{3.5}$$

Sometimes, before this centering step is done, the size factor sf_j is multiplied by the library size N_j.

How the obtained size factors are used in the analysis depends on the goal of the normalization. To present one way of applying the size factors, we want to introduce the concept of effective library sizes. The idea is to apply the previously introduced CPM method on the dataset, but instead of using the library size N_j we want to integrate the information gained through the derived size factor. Therefore, we have to define the term effective library size.

Definition 11. The **effective library size** N_j^* is defined as

$$N_j^* = N_j \cdot sf_j. \tag{3.6}$$

With the help of the effective library size N_j^* we can get modified CPM values by replacing N_j with N_j^* in equation (3.2). These CPM normalized counts $(m_{ij})_{NORM}$ incorporate the adjustments made through the size factor. These results are more convenient for comparing different normalization methods and for the downstream analysis.

If we would want to apply this approach on the size factors we obtained in equation (3.3) we have to divide them by the library size N_j before we use equation (3.4) to transform them. After that, we multiply the library size with the transformed size factor, as noted before this is in some cases necessary, and perform the scaling with equation (3.5). Finally we follow the described steps until we have the modified CPM count values $(m_{ij})_{NORM}$.

One major drawback of this method in regards of scRNAseq data is the fact that the geometric mean only includes non-zero values, this usually is ensured by the respective implementations. As we discussed earlier we are confronted with a very sparse matrix and therefore not able to account for the amount of zeros present in the data, which can lead to a loss of information

3.3.3. Upper Quartile (UQ)

This method leverages the idea of dividing the values of each cell by the value of the upper quartile Q_3 (= 75th percentile p_{75}) of the count values, of the according cell, as proposed by Bullard et al [BPHD09]. Again we encounter a problem concerning the fact that the data is very sparse and therefore it can happen that the upper quartile Q_3 is still zero or very close to it. To avoid this case it is recommended to simply use a higher threshold, for example the 99th percentile p_{99}, or exclude the

zeros beforehand, as we did before. The latter option could lead to a loss of valuable information, hence we recommend the first one.

Definition 12. Therefore we define the **upper-quartile size factor** as follows.

$$sf_j = \underset{i \in I}{p_x} \left(\frac{m_{ij}}{N_j} \right)$$

In this step we calculated the xth-percentile p_x of the expression values for every cell. Here we could choose between the previously stated two valid options to avoid the zero problem: exclude the zeros or choose a higher percentile for example $x = 99$.

To obtain **UQ-normalized values** $(m_{ij})_{NORM}$ we refer to the presented modified CPM approach in Chapter 3.3.2 with an exception in the rescaling step. To keep the level of expression relative consistent we center the size factors. In this case we recommend using a more robust method for the centering of the size factors in equation (3.5), namely the median.

$$sf_j := \frac{sf_j}{\underset{k \in J}{median}\,(sf_k)}$$

Here, the issue with the sparsity is also present but can be bypassed by either skipping the zeros and therefore their informational value or with the help of a more restrictive threshold, concerning the chosen percentile. Due to the fact the we use a percentile instead of a quartile the method is also known as Upper Quantile, which reflects the more general approach of using a specific quantile.

3.3.4. Weighted Trimmed-Mean of M-Values (TMM)

This method was developed by Robinson et al [RO10] as a new asset for normalizing RNA sequencing data in general, coming from an era of microarray data analysis. Therefore, it is not specialized for scRNAseq data, but it still yields valid results and aids in achieving our goal of normalizing the data. As most of the other normalization methods it tries to account for different library sizes. The main assumption, as with most of the methods, is that most of the genes are not differentially expressed, this may result in some problems, depending on the goal of the analysis.

We will now describe the normalization process for one cell. The first step is to choose a reference cell, this reference is fixed for all the cells in our data set. The authors do not elaborate on the selection of such a reference cell. We would recommend a cell with average total UMI count and as few zeros as possible, because genes with zero counts in the reference cell are not used in the normalization process. Another option would be to calculate a pseudo reference cell from all cells. For each cell we want to get the weighted trimmed mean of the log expression ratios, or

as we will call it the trimmed mean of M-values (TMM). To get this normalization factor we calculate the M-values for each gene expressed in the cell (this implies only expression values $m_{ij}, m_{ir} > 0$ are used), which we intend to normalize.

Definition 13. The **M-value for sequencing data** is defined as the gene-wise log-fold-changes

$$M_j^r(g_i) = \log_2\left(\frac{\left(\frac{m_{ij}}{N_j}\right)}{\left(\frac{m_{ir}}{N_r}\right)}\right),$$

where r denotes the index of the reference cell, j is the index of the cell, which we want to normalize, and i the index of a particular gene.

We also need the absolute intensity of the expression levels.

Definition 14. The **absolute intensity** of a gene g_i in a cell c_j with respect to the reference cell c_r is defined by

$$A_j^r(g_i) = \frac{1}{2}\log_2\left(\frac{m_{ij}}{N_j} \cdot \frac{m_{ir}}{N_r}\right).$$

The M-value and absolute intensity has to be calculated for every gene expressed within the considered cell. When we have calculated $M_j^r(g_i)$ and $A_j^r(g_i)$ for all genes $g_{i\in I} \in G$ expressed in the considered cell c_j and c_r, we determine a trimmed gene set by discarding the upper and lower $x\%$ concerning the M-value and $y\%$ concerning the absolute intensity. Often $x = 30$ and $y = 5$ are taken as limits. We denote by G^{tr} the gene set, which is left after trimming. Instead of the standard mean we take a weighted mean of the M-values of the remaining genes in G^{tr}.

Definition 15. The used **weights** ω are defined as the inverse of the approximate asymptotic variances.

$$\omega_j^r(g_i) = \left(\frac{N_j - m_{ij}}{N_j m_{ij}} + \frac{N_r - m_{ir}}{N_r m_{ir}}\right)^{-1}$$

So we can now define the TMM value.

Definition 16. The **trimmed mean of M-values (TMM)**, which is the desired cell-specific size factor, for the cell c_j is defined as follows.

$$\log_2\left(TMM_j^r\right) = \frac{\sum_{g_i \in G^{tr}}\left[\omega_j^r(g_i)\, M_j^r(g_i)\right]}{\sum_{g_i \in G^{tr}}\left[\omega_j^r(g_i)\right]}$$

$$sf_j = TMM_j^r$$

The **TMM-normalized values** $(m_{ij})_{NORM}$ are obtained by applying the approach presented in Chapter 3.3.2.

Two potential problems arise while using this method. First, as already mentioned, it is assumed that most of the genes are not differently expressed and second, again, we do not account for the sparsity of the expression matrix, we simply exclude zero values.

3.3.5. Downsampling (DS) - A Stochastic Approach

This method represents the only stochastic normalization method in this work. In contrary to all the other presented methods, which are deterministic, we can not expect to get every time the same result. Nevertheless, this method yields good results and a Monte-Carlo approach would also ensure reproducibility. The basic idea is to "downsample" the counts of each cell in that way, that all the cells have approximately the same total number of counts. Parekh et al [PZV+17] use downsampling by targeting a specific number or range of total counts for all cells, chosen by the respective analyst. Thereby, we again account for the library size and at the same time still introduce or preserve the presence of zeros. There are multiple ways to achieve this and only one possible way will be described at this point with the help of the binomial distribution.

First we determine the minimum of the total number of counts per cell N_j (calculated as stated in equation (3.1)).

$$N_{min} = \min \{N_j | j \in J\}$$

This number N_{min} is used as a target library size for all the other cells. Next, we define a probability for every cell, which will be used to determine the normalized count values of the cell.

Definition 17. The **cell-specific-probability** $prob_j$ for downsampling is defined by

$$prob_j = \frac{N_{min}}{N_j}.$$

Finally we generate binomially distributed random variables with the parameters we just calculated to get the "normalized" values.

Definition 18. The **DS-normalized values** are obtained by

$$(m_{ij})_{NORM} \sim B\left(m_{ij}, prob_j\right).$$

Interestingly this method is not bothered by the sparsity of the expression matrix, in contrast to the previous methods, which were not able to deal with zeros. When

calculating the correlation of the results of this method with the results of the previously mentioned methods, as presented in Figure 3.2 on page 42, we notice that the DS method always yields slightly different normalized values. This can be easily explained by the fact that unlike the other methods, it is not leveraging the CPM approach, which we described earlier in Chapter 3.3.2.

3.3.6. Size Factor by Pooling Across Cells (LSF)

The drawback of the last methods and plain normalization by library size in general led to the development of a modified normalization method, which will loosely be called Lun Size Factor (LSF), because of the first authors name of the paper which introduced it [LBM16]. Most of the methods so far had a major drawback by assuming that most of the genes are not differentially expressed, especially TMM and SF. Therefore, large size factors are often underestimated and small size factors are overestimated. Additionally, they can not account for the sparsity of the data and therefore remove zeroes beforehand in different ways.

This method tries to account for the sparsity of the matrix by pooling across cells with similar total number of counts N_j, summing them gene-wise up to reduce the incidence of those problematic zeros and calculating size factors for these pools. Finally, by solving a system of linear equations, the size factor for each individual cell can be determined.

We can characterize the method by these five key steps:

1. Defining pools of cells with similar total number of counts N_j
2. Summing up gene-wise across all cells per pool to receive a representative pseudo-cell per pool
3. Normalization of each pool-pseudo-cell relative to a defined reference cell
4. By executing steps 1.-3. a linear system is constructed
5. Deconvolve the pool-based size factors to the individual cell-based size factors by solving the constructed linear system

The following steps are taken to derive the desired linear system, which solution yields the cell specific size factors sf_j we are looking for. In the course of this explanation we deviate from the previously introduced and used mathematical framework, because the following derivation is done with the help of random variables and their respective expected values.

Definition 19. We define a random variable Y_{ij} representing the **count value** of gene g_i in cell c_j, with its expected value

$$E[Y_{ij}] = sf_j \cdot \lambda_{i0},$$

where sf_j denotes the desired cell specific size factor or bias and λ_{i0} the **expected count** for the gene g_i.

Furthermore we define the **adjusted count value** Z_{ij} by

$$Z_{ij} = Y_{ij} \cdot t_j^{-1},$$

where t_j describes a **constant adjustment factor** for c_j.

It makes sense to set t_j as the total sum of counts of c_j, the library size N_j (the basic idea of normalization by library size is applied, see Chapter 3.3.1).

Theorem 20. *By combining the previous definitions we obtain the expectation value of Z_{ij} by*

$$E\left[Z_{ij}\right] = sf_j \cdot \lambda_{i0} \cdot t_j^{-1}.$$

Now, we need a few more definitions.

Definition 21. We define a **pool** k consisting of a set of cells

$$C_k = \left\{c_j | j \in J_k\right\},$$

where $J_k \subset J$ denotes the indices of cells, which are in the pool k. The set C_0 refers to the set of all cells, hence $J_0 = J$.

Next, we need to sum up all of the cells in pool k to get a representative cell for the set C_k, thereby we get rid of most of the unwanted zeros.

Theorem 22. *We derive V_{ik} as the random variable of the **representative cell of pool k** by summing up the values of gene g_j over all the cells in pool k or the set C_k, respectively. This yields*

$$E\left[V_{ik}\right] = \lambda_{i0} \cdot \sum_{c_j \in C_k} sf_j \cdot t_j^{-1},$$

as the expected value of V_{ik}.

The observed values of V_{ik} for $i = 1, \ldots, m$ get us an overall expression profile of the cells in pool k or as we called it before a **representative (cell)** of the pooled cells in C_k.

Now we have to define the before mentioned reference cell, which we use to normalize the newly acquired representatives of the pools.

Definition 23. Therefore we set U_i as the mean of the Z_{ij} values across all N cells as the random variable of the **reference cell**.

$$U_i = \underset{j \in J}{mean}\left(Z_{ij}\right)$$

Theorem 24. *With this we can derive the expected value of U_i as*

$$E[U_i] = \lambda_{i0} \cdot N^{-1} \cdot \sum_{j \in J} sf_j \cdot t_j^{-1}.$$

In other words, the observed values of U_i represent the average gene expression of gene g_i over all cells C_0, regardless of which pool they belong to. The last step on our way to the desired linear system is to normalize the pool representatives against the reference cell.

Definition 25. Therefore, we define R_{ik} as the **true size factor for gene** g_i **regarding the cell pool** k as the ratio of V_{ik} to U_i. As expectation we get

$$E[R_{ik}] \approx \frac{E[V_{ik}]}{E[U_i]}.$$

Now, we have everything defined and derived to simplify the last equation to

$$E[R_{ik}] \approx \frac{\sum_{c_j \in C_k} sf_j \cdot t_j^{-1}}{N^{-1} \cdot \sum_{c_j \in C_0} sf_j \cdot t_j^{-1}},$$

additionally we now see that the denominator can be seen as a constant in regards to the whole dataset, because it neither depends on a cell nor a gene (nor a pool). Therefore, we will replace the denominator with the constant U. So, this results in

$$E[R_{ik}] \approx \frac{\sum_{c_j \in C_k} sf_j \cdot t_j^{-1}}{U}. \tag{3.7}$$

In the end, we replace $E[R_{ik}]$ with its estimate, calculate a robust average (e.g. median) across all genes (under the assumption that most genes are not differentially expressed between the pool and the reference cell) and treat $sf_j \cdot t_j^{-1}$ as unknown parameters. We can set U to unity and therefore ignore it, because it is the same constant factor for every pool. For every pool k we now have an equation (based on equation (3.7)) to calculate the pool specific size factor. At the same time this means we can derive a system of linear equations with one equation per pool.

We can repeat this process with different pools of cells until we get an overdetermined system of linear equations in which every $sf_j \cdot t_j^{-1}$ corresponding to each cell is represented at least once. This system can be solved with a least-squares method. Thereby, we obtain estimates of $sf_j \cdot t_j^{-1}$ for all cells. By multiplying t_j (the library size of cell c_j) we get an estimate for the desired size factor sf_j for each cell. This value can be used to normalize the expression values of each cell, for example by applying the presented CPM method in Chapter 3.3.2.

The authors themselves admit that this process seems circuitous, because one could also directly try to estimate sf_j from the count values. However, summation reduces

the number of zeroes, which, as we know, could cause problems as we observed in other methods.

Remarks on LSF

Selecting cell pools The pooling of the cells with similar library sizes is designed to provide some robustness to estimation errors of $sf_j \cdot t_j^{-1}$. The authors recommend the following pooling strategy. The cells are ordered by their library size and split into two groups: odd and even library sizes. Then the even ones are arranged in a clockwise manner starting with the largest library sizes at 12 o'clock, and the smallest ones at 6 o'clock. The odd ones are arranged exactly the same way starting also at 12 o'clock but progressing counter-clockwise to 6 o'clock. Now the pooling process can be done by simply sliding a window of the desired pool size along this clock and thereby avoiding truncated windows as we would get in a linear ordering at the boundaries.

Solvability of the linear system With the described approach the next question would be how does the size w of the mentioned window, which slides along the clock, influence the resulting linear system? The total number of equations in the system is equal to the number of cells, but the number of equations in which the term $sf_j \cdot t_j^{-1}$ for one cell shows up is determined by w. If we would like to increase the precision of the estimates one way could be to use different values for w. Thereby the total number of equations naturally increases.

Before solving the linear system a set of equations is added in which every estimated size factor is equated to its directly derived size factor. The directly derived size factor is obtained by normalizing the count values of the cell directly against the reference cell. By adding these additional equations we ensure that the columns are linearly independent and therefore a single solution can be obtained. It is important to note that these equations are assigned a very low weight in the least-squares approach so that they just ensure independency but do not influence the least-squares outcome substantially.

Weakening the DE assumption by clustering As we still use the assumption (as other methods too) that most of the genes in one pool are not differentially expressed the authors recommend to cluster the data in the beginning and selecting the cell pools within the found clusters. Thereby, the assumption is at least reduced to a certain degree, because the cells in one cluster should have similar expression profiles, due to the fact that the clustering is based on the expression values of the respective cells.

3.3.7. Comparison & Conclusion

Comparison

To compare results of the presented normalization methods we recommend using relative log expression (RLE) plots. In general RLE plots are used to asses (unwanted) variation in high dimensional data. In our case we want to compare the RLE before and after normalization. Furthermore, it can be used to compare the normalization results of different methods to decide which normalizes the current data best. To get a RLE plot some calculations have to be done. First, for each gene g_i we calculate the median of the expression of that gene across all cells c_j.

$$g_i^* = \underset{c_j}{median}\,(m_{ij})$$

Next, with the help of the acquired median per gene g_i^*, we calculate the deviation matrix.

Definition 26. We define the **deviation matrix** by subtracting the median from the expression values of the according gene.

$$(m_{ij})_{deviation} = m_{ij} - g_i^*$$

Finally, every element $(m_{ij})_{deviation}$ is transformed, by adding an offset and applying the logarithm. For every cell c_j a boxplot for all the log-transformed deviations $(m_{ij})_{deviation}$ for $i = 1, \ldots, m$ is generated. By looking at all the boxplots at once, variations in groups of samples can be spotted. Most of the time it is helpful to apply some kind of prior knowledge (e.g. batch, donor or treatment) on the generated plot by coloring the cells according to potential or suspicious explanatory variables (this term will be defined in the next chapter). After applying an appropriate normalization method the RLE plot should display all cells in a similar manner. What we mean by that is that there should not be a cell or group of cells, which behave differently compared to all the others. If that happens it can be suspected that they have something in common, which distinguishes them heavily from all the other cells (e.g. damaged cells).

If we would like to decide for a method without testing all of them and only by looking at their mathematical description, we would recommend using the LSF normalization method. The main argument is of course the fact that this method accounts for all the new challenges in scRNAseq data and at the same time incorporates mechanisms, which seem to work on previously generated data in the sequencing domain.

One last remark on deciding for a method: Sometimes both the sophisticated methods and the straight forward ones yield very similar results. Therefore, it is recommended to choose the simpler one as they do not modify the original data as much as the other methods.

Conclusion

We have looked at different methods for normalizing scRNAseq data starting at the most intuitive one, applying the logarithm and simple normalization by library size, working through some established methods, which still yield valid results for scRNAseq data, SF, UQ and TMM, looking into stochastic ways to normalize data, DS, and finally describing in detail a method especially developed for scRNAseq data, LSF.

By normalizing the data we remove potential technical noise and establish a basis for comparability between cells. If we would describe the normalization process in terms of removing explanatory variables, we would best describe it by trying to remove the library size factor. The drawback in normalizing the data is always the loss of information (wanted and unwanted sources of variation to some degree), but the proposed methods try to ensure that the cell's inherent structure or information is not lost and only the biases introduced through the data generation process are removed.

After removing or weakening the effect of the library size N_j an intuitive next question in an analysis would be: What kind of other variables, which could describe our data, are present? Therefore the following chapter will deal with explanatory variables. We will discuss how they are defined, which types can be encountered, how to determine their influence on the data and how to get rid of this influence, if it is of an unwanted source. The methods presented in the next chapter are used throughout the course of a scRNAseq analysis.

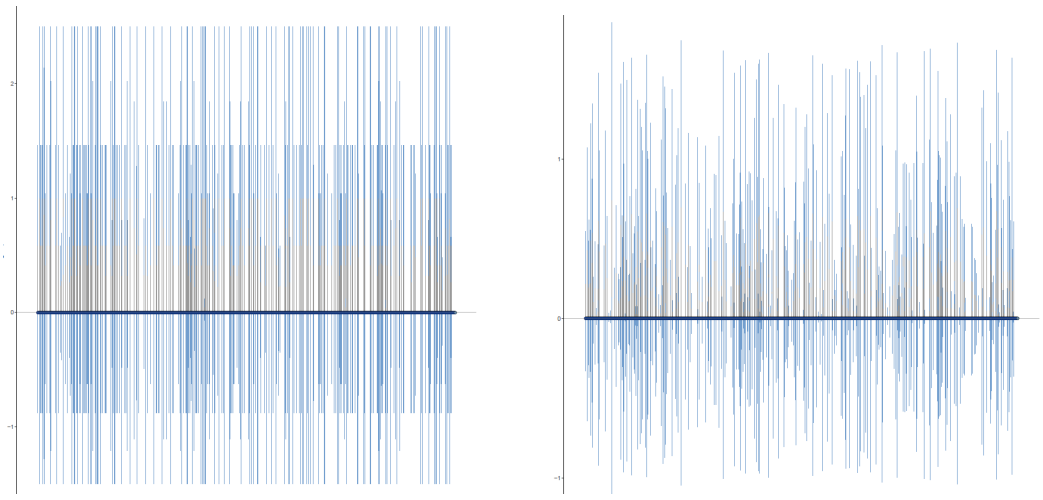

Figure 3.1.: RLE plots of the simulated dataset with the RLE values on the y-axis and the cells on the x-axis. Every vertical line is a boxplot of the deviated log-transformed counts of the respective cell. On the left the logarithm-normalization, with amplitudes larger than +2 and -1 respectively, and on the right the LSF-normalization, with amplitudes smaller +2 and -1 respectively, are shown.

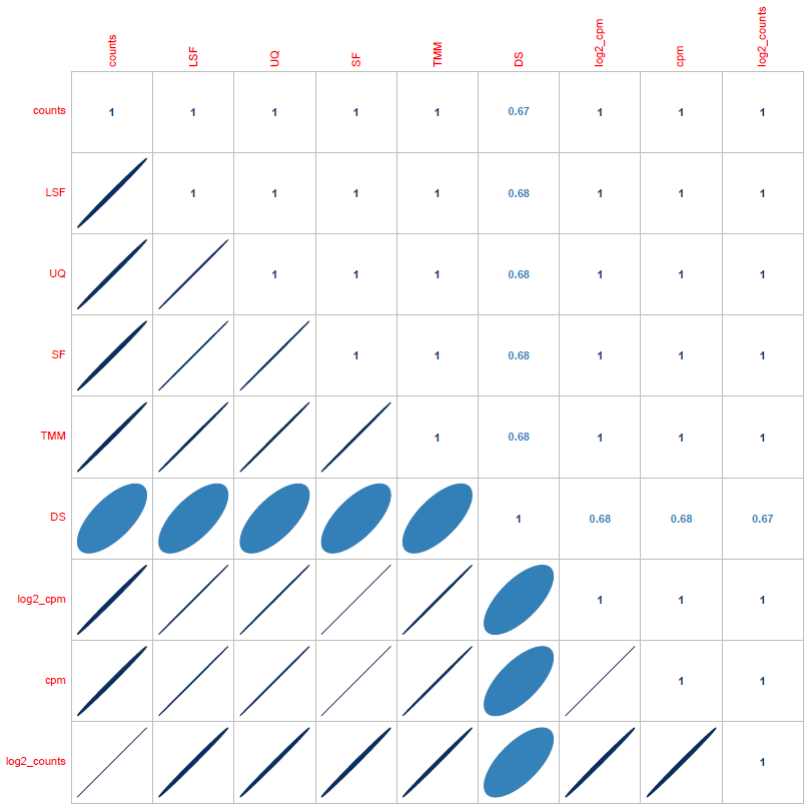

Figure 3.2.: Correlation plot based on 599 randomly selected cells out of the normalized matrices of the simulated dataset, generated by different methods.

3.4. Explanatory Variables

This chapter deals with attributes of the data, to be more precise of the cells. Mathematically speaking we add other dimensions to the already existing ones (genes), where every data point (cell) is supplied with further specific information. The questions at hand are: How big is the influence or the descriptive power of a specific variable, represented by an additional dimension, on the data? and How much do these variables describe each other?

With the help of the next section and the examples supplied within it, we try to give a feeling about the kind of variables we are dealing with. Followed by that, we will discuss different methods for determining descriptive power of variables on the data and each other. Thereby, we will establish a basis for a comparison. We will end with a discussion on the presented methods and their application in scRNAseq analysis. In the last part a method for the removal of the influence of confounding factors on the data is presented. This method is only applied in scRNAseq analysis, if a variable is identified as a confounding factor (which we will define later) and not a biological information, which helps answering the experimental question and therefore should be preserved.

In the course of this chapter we will always want to determine the influence of one variable $x \in \mathbb{R}^n$ on another variable $y \in \mathbb{R}^n$, where n is the number of cells c_j in our data set. The previously introduced mathematical framework will be used when we put the the explained methods into context of scRNAseq analysis.

3.4.1. Variable Types

We encounter three different data types in our variables, which will be described in the following and highlighted by examples. It is important to understand that different data types lead to different approaches for determining the influence of said variable on another one or the data itself, because most of the methods have a mathematical limitation concerning the desired comparison.

Before we describe the types, we have to distinguish between the two general sources of variables: internal and external ones.

Internal variables can be derived directly from the data, without additional knowledge about the data. An example would be the total UMI counts per cell N_j as previously defined in equation (3.1) or the number of expressed genes per cell $\#g_j^{expr} = |\{m_{ij} : m_{ij} > 0\}|$. These values exist intrinsically within the data, but are not provided in such an explicit way. Therefore, they have to be calculated and supplied as an artificial dimension.

External variables on the other hand, have to be supplied from an data independent source. We are talking about questions like: Who performed the experiment?, What kind of storage was used? or the most common one: From which batch (e.g. sample/experiment/run) is a certain cell? It is important to state that the effect of these variables (if there is one) would always be present in the data, whether we know about the variable or not. The problem is we would not be able to detect it and thereby also not able to remove the effect, in case of absence of the explicitly stated explanatory variable. Sometimes the term "Hidden Confounding Factor" or "Hidden Covariate" is used to address these unknown variables, but we will not further discuss this subject as we want to focus on the challenges, which come with known variables.

Independent of the source either of those variables can be represented by the following datatypes.

Categorical variables These variables describe the cell in a non numerical way and are most of the time used to convey information, which is supplied externally and concerns the preparation of the cell or other circumstances, which lead to a classification of some sort. The different values, of one categorical variable, are fixed and often called levels. Common examples are batch, donor, storage, treatment,

gender of donor, etc. Actually, it depends on the setup of the current experiment and the effects the researchers want to see. One other case of categorical values is presented in the form of boolean entries (true or false), which could either be derived from within the data (e.g. cell has less than 1000 UMI counts) or outside the data (e.g. cell was exposed to radiation).

Continuous variables Most of the time we are dealing with this kind of variables, which are an element of $\mathbb{R}$ and often describe metrics from within the data. Examples are percentages of certain genes expressed or mean values of the top 100 expressed genes within a cell. Whenever there is some kind of calculation involved other than summing up UMI counts (which are always integers), the result usually falls into this category.

Integer variables Here we are faced with values, which are either an element of $\mathbb{N}$ or $\mathbb{Z}$ and therefore can not be classified in the same way as continuous variables and sometimes are even interpreted as discrete or categorical values, although it is not the case. Dealing with expression matrices, which consist of UMI counts and therefore are elements of $\mathbb{N}$, we are often confronted with the fact that internally derived descriptive variables are also integers. To bypass possible problems arising from the nature of such variables we can transform them to another type, without losing their descriptive power. One way to transform such discrete values to continuous values is the application of the logarithm. Common examples are total UMI counts or the number of expressed genes per cell.

Complications The following measures, which describe the influence of said variables on the data or each other, have certain limitations concerning the data type they support. We will always clearly define in which case this measure makes sense or when it should not be used and why. Part of the solution is the fact that sometimes one variable can be represented by more than one type and still has, in both cases, the same descriptive power (e.g. total UMI counts or logarithm with base 10 of total UMI counts). Thereby one can simply transform a variable which is not supported by a method to a supported one.

3.4.2. Linear Regression

We will give a brief introduction into the topic of linear regression following the lecture notes from "Mathematical Statistics" held by Prof. Grill [Gri00]. Linear regression represents the basis of most of the methods, we present in this chapter. With the goal of using the same notation throughout this chapter we will define some important terms. After that we will discuss simple linear regression, multiple linear regression and finally an approach to deal with categorical variables in this context. All of these modeling approaches will be needed in the course of this chapter.

The general idea is to model the relationship between a dependent variable and one (simple linear regression) or more (multiple linear regression) explanatory (or independent) variables.

Definition 27. We will denote **the dependent variable** with $y \in \mathbb{R}^n$ and its mean with $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. Furthermore we define the variable (for simple linear regression), which explanatory power is in question, as the **independent variable** $x \in \mathbb{R}^n$ and its mean with $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

We assume that the dependent variable is related to the independent variable(s) through a linear function.

Definition 28. The vector $\hat{y} \in \mathbb{R}^n$ represents the **predicted values** of the linear regression model of the dependent y and independent variable(s) $x_{(1,\dots,k)}$.

One further definition will be very important.

Definition 29. The deviations of the dependent variable y from the predicted values $\hat{y}$ are the **residuals** $res_i \in \mathbb{R}$ of the linear regression model.

$$res_i = y_i - \hat{y}_i$$

Now the only thing missing is an explanation on how the predicted values are determined. Here we have to distinct between the two linear regression approaches.

Simple Linear Regression

We assume that the dependent variable y is related to one independent or explanatory variable x in a linear fashion, through the equation

$$y = ax + b,$$

where $a, b \in \mathbb{R}$ are constants. If the relationship would truly be linear we could determine the exact relation, the constant a for the slope and b for the y-intercept, already with the help of only two observations. Having said that, we know that most relationships in the real world, especially in the life sciences, are not linear in their nature. That's why, we have to account for the deviations of the true values from the linear function values. These deviations were previously defined as residuals and denoted by $res \in \mathbb{R}^n$. Therefore we write

$$y = ax + b + res \tag{3.8}$$

and modify the previously stated equation to

$$\hat{y} = ax + b.$$

This is then the model function for the predicted values $\hat{y}$ and equation (3.8) represents the related linear regression model. Therefore, we are looking for a linear function, characterized by a and b, which predicts the values of y as accurate as possible depending on variable x. This can be achieved for example through minimizing the the sum of squared residuals as in the ordinary least-squares method. Hence, this approach is sometimes called ordinary least squares regression.

We are using the previous definitions to frame the problem in mathematical terms: Find $\min_{a,b} R(a,b)$ for

$$R(a,b) = \sum_{i=1}^{n} res_i^2 = \sum_{i=1}^{n} (y_i - ax_i - b)^2 .$$

With the assumptions of the residuals being independent and normally distributed with expectation zero and constant variance, the solutions $\hat{a}$ and $\hat{b}$ to this minimization problem are given by the following least squares estimators.

$$\hat{a} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} = \frac{\mathrm{Cov}(x,y)}{\mathrm{Var}(x)}$$

$$\hat{b} = \bar{y} - \hat{a}\bar{x}$$

We want to mention that this is not the only way to determine the best fitting function and there are even some, which do not involve the minimization of the residuals.

Multiple Linear Regression

This approach represents a generalization to k different explanatory variables $x_1, \ldots, x_k$ and we describe them as n sets of numbers $(x_{1l}, \ldots, x_{kl})$, with $l = 1, \ldots, n$. This could practically be written as a matrix, where the rows represent the relation to a particular y-value and the columns denote the different explanatory variables $x_1, \ldots, x_k$. Sometimes this matrix is referred to as **design matrix**.

Consequently we have a model function of the form

$$\hat{y} = a_1 x_1 + \ldots + a_k x_k$$

and the related linear regression model function

$$y_l = a_1 x_{1l} + \ldots + a_k x_{kl} + res_l, \tag{3.9}$$

with res_l as the residuals. Compared to the previous model function we do not have a variable b anymore, because it can be included by letting $x_i = 1$ for some i. The corresponding a_i would represent the variable b.

Analogue to the previous derivation of the variables a and b in simple linear regression, we again apply the ordinary least squares method. This yields the following system of equations for the least squares estimators a_i

$$\sum_{i=1}^{k} a_i \left(\sum_{l=1}^{n} x_{il} x_{jl} \right) = \sum_{l=1}^{n} y_l x_{jl},$$

for $j = 1, \ldots, k$.

Linear Regression with categorical variables

We can also apply the presented methods if the explanatory variable x or one of the explanatory variables is categorical in nature. The trick is to transform the categorical variable into a design matrix, which we can use in multiple linear regression. This can be achieved by the use of coding systems. This can be done by introducing a new independent variable for every level of the respective categorical variable but one. Thereby, we reduce redundancy, while still having all the information. No additional information would be contained in the coded matrix if the final level also gets a variable, because the information is already implied.

We give an example to better illustrate this approach. We take the categorical variable of the Batch, in which the cells were processed, with 6 levels (Batch1,..., Batch6). Following the previous argumentation we need 5 variables for the coded system. Table Table 3.3 on page 47 illustrates the coding system of the variable Batch.

Batch	x_1	x_2	x_3	x_4	x_5
Batch1	0	0	0	0	0
Batch2	1	0	0	0	0
Batch3	0	1	0	0	0
Batch4	0	0	1	0	0
Batch5	0	0	0	1	0
Batch6	0	0	0	0	1

Table 3.3.: Coding system of the categorical variable Batch

With this coding system the design matrix can be derived for that variable or even additional explanatory variables. Following the constructed example, the first line of the design matrix, representing the explanatory variables for the first cell, would look like

$$\begin{pmatrix} x_{11} & x_{21} & x_{31} & x_{41} & x_{51} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

in the case of the first cell being originally from Batch3.

We face a problem if a categorical variable is used as the dependent variable y. This is not possible, because we can not determine the sum, mean, median or any

other statistic needed. In the case of only one independent variable x and it being continuous in nature we can simply swap their respective role. In any other case, for example if the influence of more than one explanatory variable on the categorical variable is of interest or if we want to determine the influence of two categorical variables on each other, we can not use linear regression models.

Remarks

In scRNAseq analysis linear regression is commonly used due to its transparency concerning the understanding of the derivation of the result and its wide field of application, as we will see in the following. Having said that, it is important to keep in mind that some assumptions are not correct for biological matters and the results can be misleading. As for example demonstrated with the help of Anscombe's quartet [Ans73], which is a group of four datasets that have very similar descriptive statistics (e.g. mean of the variables x and y, linear regression function, coefficient of determination R^2) but appear very different when plotted. This remark holds also true for the following sections of this chapter. That's why, there are also implementations and practices in scRNAseq analysis, which experiment with other modeling approaches (e.g. non-linear). The problem with those more complex models is their restricted field of application (e.g. special data structures required) and the insufficient control over the effects, which result from applying it on the data.

When we will use the term model in the following, we usually refer to linear regression models as presented in this section, unless stated otherwise. Now, we are finally moving on to methods for the measurement of explanatory power, which sometimes leverage linear regression as modeling approach.

3.4.3. Coefficient of Determination R^2

The coefficient of determination R^2 is an important measure for the formal assessment of the quality of a fit by a regression model. The R^2 describes the proportion of the variance in a dataset, which can be predicted by another (explanatory) variable. In other words: How much of the variance in the data can be explained by a specific variable? We chose to present the most general version following Gujarati [Guj04], which uses linear regression. We will use the mathematical definitions from the previous Section 3.4.2.

To determine the explanatory power of the independent variable x on the dependent variable y we need the predicted values $\hat{y}$. Therefore, we will apply linear regression, as described before. In the case of using simple linear regression the coefficient of determination is often denoted by r^2 instead of R^2, but we will see that both modeling approaches will be needed. Before we can start to formulate the definition of the R^2 we need three measurements for the variability of the data and the model,

which are the total sum of squares, the explained sum of squares and the residual sum of squares.

Definition 30. The **total sum of squares** SS_{tot} yields the difference or error of the data to its own mean, thereby describing the total variance within the data and is given by

$$SS_{tot} = \sum_{i=1}^{n} (y_i - \bar{y})^2 .$$

To get the proportion of the variance explained through the model, we need a measure for the variance explained by the respective model.

Definition 31. The **explained sum of squares** SS_{expl} describes the difference of the predicted values to its mean and is obtained by

$$SS_{expl} = \sum_{i=1}^{n} \left(\hat{y}_i - \bar{\hat{y}}\right)^2 .$$

Furthermore, we define the inverse measure, the variance which is not explained by the model, with the help of the residuals of the model.

Definition 32. The **residual sum of squares** SS_{res} describes the difference of the data to its modeled value and is calculated by

$$SS_{res} = \sum_{i=1}^{n} res_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 .$$

At this point we want to show a convenient relation connecting those measures.

Theorem 33. *In the case of using linear regression with the least squares approach (including a constant in the model function, the y-intercept) to obtain the model values $\hat{y}$, the following partition of the sum of squares holds.*

$$SS_{expl} + SS_{res} = SS_{tot}$$

Furthermore the mean of the dependent variable and the mean of the modeled values are the same.

$$\bar{y} = \bar{\hat{y}}$$

Now we can formulate the definition of the R^2.

Definition 34. The **coefficient of determination** R^2 of a linear regression model, describing the proportion of the variance in the data that is predictable from the explanatory variable(s), is defined by the ratio of the explained variance by the model to the total variance within the data

$$R^2 = \frac{SS_{expl}}{SS_{tot}} = 1 - \frac{SS_{res}}{SS_{tot}} \in [0, 1] .$$

If $x = y$ the value of the coefficient of determination R^2 would be 1. Therefore, a higher R^2 value indicates that the two variables are very similar to each other or in other words: one variable can predict the other to a high degree. In the case of scRNAseq analysis this would relate to a high influence of one explanatory variable on the data.

If we have continuous variables y and x this works out fine, but we encounter problems if either one of them is of another type. In the case of integer values we already presented an easy solution of transforming them to continuous variables by applying for example the logarithm. In the case of categorical variables we sadly inherit the limitations, concerning the data types of the variables, we already faced with linear regression, due to the fact that we base the calculation of the predicted value $\hat{y}$ on a linear regression model.

As we already described in Section 3.4.2 in the case of x being categorical and y being continuous, we can apply a coding system to the categorical variable and use multiple linear regression. If, on the other hand, x is continuous and y is categorical, we can not apply the same strategy. Due to the fact that the R^2 for simple linear regression is symmetric we solve that issue by simply swapping the roles of x and y to get the mirrored case, where we use y as the independent and x as the dependent variable, which we can solve as before.

In the remaining option we are faced with a different situation. Here both, x and y, are categorical. In this setup it is not possible to determine the mean of our dependent variable or the residuals of the model, because linear regression, as we described it, can not be applied. Therefore, we have to conclude that the case of comparing the influence of categorical variables on each other can not be described with the help of the R^2 of linear regression models and we need another, hopefully comparable, measure.

3.4.4. Correlation

When thinking about the comparison of data in general or two vectors in particular, correlation immediately comes to mind. The standard measure of the linear correlation between two variables x and y in statistics is called the Pearson correlation coefficient PCC (also known as Pearson's product-moment correlation coefficient PPMCC). It yields values in the interval of -1 to 1. With the interpretation of 1 describing total positive linear correlation, 0 no linear correlation, and -1 total negative linear correlation. Following the lecture notes from "Applied Mathematical Statistics" held by Prof. Gurker [Gur13] we will define the PCC, in addition we will present a relationship to the previously described coefficient of determination and discuss the application of correlation in scRNAseq analysis.

Definition 35. We define the **Pearson correlation coefficient** PCC also known as r_{xy} as the ratio between the (empirical) covariance of x and y, and the square root

of the product of the (empirical) variances, or simply the product of the standard deviations, of x and y.

$$r_{xy} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

We have already discussed before that linear regression is a powerful and flexible method for the analysis of linear dependencies. Following this argumentation we want to show a relationship between the coefficient of determination R^2 and the PCC.

In the case of applying simple linear regression using the least squares approach the R^2 and the r_{xy} are linked by the following relationship.

Theorem 36. *Suppose the conditions of Theorem 33 are met with only one explanatory variable x, then it can be shown that*

$$R^2 = r_{xy}^2,$$

where R^2 describes the coefficient of determination of the simple linear regression model of y and x, and r_{xy} denotes the PCC of x and y.

In the more general case of wanting to measure the influence of multiple explanatory variables, we can formulate the following relationship.

Theorem 37. *Assuming again that the conditions of Theorem 33 are met, it can be shown that*

$$R^2 = r_{y\hat{y}}^2,$$

where R^2 describes the coefficient of determination of the multiple linear regression model of y and $x_1, \ldots, x_k$, and $r_{y\hat{y}}$ denotes the PCC of y and the predicted linear regression model value $\hat{y}$.

Of course these relationships are not valid for every correlation method, but they illustrate nicely that these measures capture the same kind of information.

We immediately realize one major shortcoming of the PCC or correlation methods in general: we can only measure the correlation of two variables on each other, but not the correlation of a set of variables with a single one, as we do with the coefficient of determination and multiple linear regression. Furthermore, in the case of PCC we are restricted to continuous variables x and y, which is not the case for all correlation coefficients.

One other popular correlation coefficient, in the analysis of life science derived, data is the Spearman's rank correlation coefficient, due to its more robust nature. In contrast to the PCC it does not try to capture linear relationships, but measures monotonic relationships or in other words: the best correlation is achieved, when

each of the variables is a perfect monotone function of the other. Here, we still face the restricting condition that the variables must be ordinal in nature, which is not the case with categorical variables.

Due to the shown nature of the relationship between the PCC and the R^2, we again face the same limitations concerning categorical variables. Therefore, we are still confronted with the unsolved case of determining the influence of categorical variables on each other. That's why other approaches, which can deal with this setting in particular, have to be investigated in the following.

3.4.5. Statistical Hypothesis Testing

One attempt to solve the issue of determining the influence of categorical variables on each other is to use statistical hypothesis testing. The null hypothesis in our case proposes that no relationship between the two variables exist or in other words, that the two variables are independent. To investigate this idea we chose the χ^2 test of independence as presented by Prof. Grill [Gri00]. We will define it and then discuss arising challenges.

As a starting point we are faced with two categorical variables x and y with their respective number of levels k and m and length n. Now we define some helpful terms for $1 \leq j \leq k$ and $1 \leq l \leq m$. The probabilities $p_j = \mathbb{P}(x = j)$ and $q_l = \mathbb{P}(y = l)$ together with the assumption that the variables are independent get us the following equation.

$$\mathbb{P}(x = j, y = l) = \mathbb{P}(x = j)\,\mathbb{P}(y = l) = p_j q_l$$

Moreover, we need the expected and observed values for the different cases. Therefore, we define

$$\begin{aligned} z_{jl} &= \#\{i : x_i = j, y_i = l\}, \\ z_{\cdot l} &= \#\{i : y_i = l\} = \sum_{j=1}^{k} z_{jl}, \\ z_{j\cdot} &= \#\{i : x_i = j\} = \sum_{l=1}^{m} z_{jl} \end{aligned}$$

and get the estimator $\frac{Z_{j\cdot}}{n}$ for p_j and $\frac{Z_{\cdot l}}{n}$ for q_l. With that we are ready to apply the χ^2 test.

Definition 38. The **χ^2 test of independence** is defined as

$$\chi^2 = \sum_{j=1}^{k} \sum_{l=1}^{m} \frac{\left(z_{jl} - \frac{Z_{j\cdot} Z_{\cdot l}}{n}\right)^2}{\frac{Z_{j\cdot} Z_{\cdot l}}{n}}.$$

The last thing we need to get the desired result, a p-value for the null hypothesis, are the degrees of freedom of this statistic.

Theorem 39. *The number of degrees of freedom (dof) of the χ^2 statistic from Definition 38 is*

$$dof = (k-1)(m-1).$$

With this we can determine a p-value, which is compared to a given significance level (conventionally 0.05). If the p-value is smaller than the given significance level, then the assumption (null hypothesis) of x and y being independent is rejected and we can conclude that the variables x and y influence each other in a significant way.

Having reached our goal of measuring the influence of two categorical variables, we do have a problem of comparability concerning the previously introduced methods, namely the coefficient of determination and correlation. Another issue is the predetermination of a level of significance to which the yielded p-value shall be compared. Additionally sometimes the p-value itself has to be adjusted to be meaningful. Due to all of these complications a hopefully more suitable approach was investigated.

3.4.6. McFadden's Pseudo - R^2

Another attempt to solve the problem of having two categorical variables with possibly different numbers of categories (levels) and trying to describe the influence on each other, is the application of pseudo-R^2 measures on models, which capture these aspects. The idea is the construction of an analogues to the classical R^2 in linear regression for more general regression models. There is not one but several measures, which try to be the appropriate analogues to the R^2, each with different approaches and advantages. We decided on McFadden's pseudo-R^2, which is motivated by constructing an analogue to the classical R^2 measure for multinomial logistic regression models. Here, we assume that the dependent variable y is categorical in nature with potentially more than two levels. This condition describes nicely the limitation, which we were not able to overcome with the previous methods and especially with linear regression. As the name suggests, this measure was first proposed by McFadden, therefore we will define it with the help of the according publication [McF73].

Without going to deep into multinomial logistic regression, which is needed to describe this measure, we want to give a brief introduction into the terminology first, before defining the desired measure.

We start with the simplest case, which tries to build a regression model for the categorical dependent variable y, with the two levels 0 and 1 (binary), and only one continuous explanatory variable x as discussed by Gujarati [Guj04]. The basis of this approach, as the name suggests, is the logistic function.

Definition 40. The **logistic function** is defined as

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}},$$

with $t \in \mathbb{R}$.

This function is very useful, because the input t can be any real number, but the output value is between zero and one, $\sigma(t) \in [0, 1]$. Thereby it can can be interpreted as a probability. Next, we want to look at the inverse of that function. Before we do that, we choose for t a linear regression model with one explanatory variable $t(x) = ax + b$, as we have done before in Section 3.4.2. This results in

$$\sigma(t(x)) = \sigma(x) = \frac{1}{1 + e^{-(ax+b)}}. \tag{3.10}$$

With that we can define the so called logit.

Definition 41. We define the inverse of the logistic function, the **logit function**, as follows

$$L(\sigma(x)) = \ln\left(\frac{\sigma(x)}{1 - \sigma(x)}\right) = ax + b,$$

which immediately gives us the following equation

$$\frac{\sigma(x)}{1 - \sigma(x)} = e^{ax+b}$$

through exponentiation.

When investigating equation (3.10) again, we can see that it can be interpreted as the probability of the dependent variable y equaling 1 rather than 0, $\mathrm{P}(y = 1|x)$, given some linear combination of predictors. This is also known as the **(cumulative) logistic distribution function**. Looking at the previous definition in this context, we can see that the logit is simply the logarithm of the odds ratio in favor of y equaling 1 or in other words the ratio of the probability that y equals 1 to the probability that it will equal 0. That is why these models are also called logit models. This shows us, that the logit enables us to link a linear regression model to a probability.

So, we now have derived a probability for the case of having a binary categorical dependent variable y and one continuous independent variable x. By extending this approach to more than one explanatory variable, we will also include the possibility to use categorical variables with any number of levels as independent variables. We can simply achieve that by replacing the simple linear regression model by a multiple

linear regression model in equation (3.10). Thereby we get

$$\mathrm{P}\left(y = 1 | x_1, \ldots, x_K\right) = \frac{1}{1 + e^{-\sum_{k=1}^{K} a_k x_k}},$$

with K explanatory variables. How to deal with categorical variables in this case was already explained in Chapter 3.4.2 with the help of a design matrix X and coding systems.

The last thing missing now is the possibility of the dependent variable being categorical with more than two levels, namely the multinomial part. To get there we keep building on what we have so far, because luckily we can directly extend the binary approach to the multinomial one, as described by Czepiel [Cze12].

Definition 42. The family of probabilities, which describes the **multinomial logistic regression model**, is obtained by

$$\mathrm{P}\left(y = c | x_1, \ldots, x_k\right) = \frac{e^{\sum_{k=1}^{K} \alpha_{kc} \cdot x_k}}{\sum_{l=1}^{L} e^{\sum_{k=1}^{K} \alpha_{kl} \cdot x_k}},$$

where c denotes the respective category we want to determine the probability for, the variable α_{kc} is the coefficient of the explanatory variable x_k associated with category c and L is the number of categories, which the independent variable y could choose from. For the sake of completeness the respective linear regression function for category c looks like

$$\hat{y}_c = \alpha_{1c} x_1 + \ldots + \alpha_{Kc} x_K,$$

where K denotes the number of explanatory variables.

Having now the models properly derived we still need two more things to be able to define McFadden's pseudo-R^2. The first thing is a special variant of the just described model, the associated null model, which we need for a comparison later on.

Definition 43. The **null model** of a binary logistic regression model contains just the intercept b,

$$\ln\left(\frac{\mathrm{P}\left(y = 1\right)}{1 - \mathrm{P}\left(y = 1\right)}\right) = b.$$

Analogously, in the case of a multinomial logistic regression model, the null model is defined by a vector b_c, with the proportion of the respective category c present in y. Thereby, we determine the probabilities without including the explanatory variables $x_1, \ldots, x_K$ or in other words we build a model with zero values for the explanatory variables.

So far, we have derived everything for only one observation y. To get a useful model we have to derive it by considering every y_i with $i = 1, \ldots, n$. Then this model has to be fitted, which means that the regression coefficients α_{kc} have to be determined for every category and explanatory variable. This is usually done by maximum likelihood estimation. For that we need the likelihood function of such models.

Therefore, the last thing left is to answer the question of how the likelihood functions of those models are determined.

Definition 44. We obtain the kernel (= only non constant components) of the **likelihood function $\mathbf{L_M}$ of a binary logistic regression model** M by

$$L_M \simeq \prod_{i=1}^{n} P_i^{y_i} \left(1 - P_i\right)^{1-y_i},$$

where n denotes the number of observations y_i and P_i describes the probability $\mathrm{P}\left(y_i = 1 | x_1, \ldots, x_k\right)$.

Similarly, we obtain the kernel of the **likelihood function $\mathbf{L_M}$ of a multinomial logistic regression model** by

$$L_M \simeq \prod_{i=1}^{n} \prod_{l=1}^{L} P_{il}^{y_{il}},$$

where L is the number of categories, y_{il} is 1 when $y_i = l$ and otherwise 0, and P_{il} describes the probability of y_i being category l

$$\mathrm{P}\left(y_i = l | x_1, \ldots, x_k\right).$$

Finally, we have everything in place to define the desired measure.

Definition 45. We define **McFaddens pseudo-$\mathbf{R^2}$** or ρ^2 as one minus the ratio of the log likelihood values of the fitted model with predictors and the null model

$$R^2_{McFadden} = \rho^2 = 1 - \frac{\ln\left(L_M\right)}{\ln\left(L_0\right)},$$

where L_M is the likelihood for the fitted model and L_0 is the likelihood for the null model. The measure takes values between zero and one, $R^2_{McFadden} \in [0, 1)$.

Now we have a powerful solution at our disposal to the problem of comparing categorical variables with each other, but this was already ensured by statistical hypothesis testing. The motivation was the question whether we can find a measure, which is more compatible to the previously introduced methods, namely the coefficient of determination and correlation. This is the case, but only to some degree. We still have to keep in mind that this measure can not be exactly interpreted in the same way as the standard R^2 (proportion of variance explained) and therefore the

two measures are not directly comparable. Fortunately, McFadden himself pointed out the following, in a discussion on model evaluation in the context of multinomial logistic models:

> "While the R^2 index is a more familiar concept to planners who are experienced in ordinary regression analysis, it is not as well-behaved a statistic as the ρ^2 measure for maximum likelihood estimates. Those unfamiliar with the ρ^2 index should be forewarned that its values tend to be considerably lower than those of the R^2 index and should not be judged by the standards for a "good fit" in ordinary regression analysis. For example, values of .2 to .4 for ρ^2 represent an excellent fit." [McF77]

Therefore, we conclude that we have found a reasonable measure for the determination of the influence of categorical variables on each other, which is in some sense comparable to the previously established methods.

3.4.7. Discussion on Adjusted R^2 Measures

There are also adjusted versions of the corresponding R^2 measures. The goal of these adjusted versions is to counteract the effect of getting a better model fit by simply increasing the number of variables or categories of an explanatory variable. This is a common issue in statistics and therefore often a penalizing term is introduced into the calculation. This holds also true for the two presented R^2 measures. We now want to argue why we think that an adjustment, in our case, would not make any sense.

Generally speaking our main goal is to determine the influence of given variables of different types on each other. We do not want to find the best model to describe our data. Therefore, we argue that penalizing by deliberately decreasing the corresponding R^2 measure would decrease the value of the information we want to get. It would be another story if we would try to build or find the best variable to describe our data, in that case it would totally make sense to penalize. However, in our case the variables are predetermined by external sources and we have to assess how much they influence each other or the data, respectively.

3.4.8. Comparison

We have investigated two established methods to determine the influence of one variable on another namely the coefficient of determination R^2 and correlation coefficients. Both methods were lacking the capability of determining the influence of categorical variables on each other and therefore we had to look for methods to solve this problem. This challenge resulted in two additional approaches, statistical hypothesis testing and McFadden's pseudo-R^2 measure, both with their own limitations.

Representing the statistical hypothesis testing approach, we tried using the χ^2-test of independence to determine the relation between explanatory variables. Due to the fact that we could not find a general rule for the comparison p-value, the lack of comparability to the other measures and the question of appropriate p-value adjustment we had to consider the alternative.

The McFadden pseudo-R^2 on the other hand, yielded stable and comparable results, within the categorical vs. categorical domain. Having already established that the R^2 measure could be used for any constellation but the categorical vs. categorical one, we conclude that the McFadden pseudo R^2 would complement nicely. Although, we want to repeat that these two different R^2 measures can not directly be compared, this solution is more consistent than any other combination of methods, as we have demonstrated by direct comparison. Therefore, using the McFadden pseudo-R^2 for categorical vs. categorical variables and the standard R^2 for the remaining constellations is the best way to go.

3.4.9. Application

In single cell RNA sequencing analysis the above described methods are applied in the identification of influencing variables, comparing their influence on each other, removal of confounding factors and even in clustering analysis.

Identification of influencing factors

We loosely define confounding factors as variables, amongst all explanatory variables, which have a big influence on the data but actually should not describe the data at all or at least not to a big extent. Therefore, the influence of given variables (internal or external) on the data is of interest. For this analysis most of the time the data is projected in a space of reduced and more meaningful dimensions (we will come to that in Section 3.5 on Dimensionality Reduction). This means that the dependent variable y, in the above described methods, would represent the values of every datapoint (cell) concerning one of these dimensions. The explanatory variable in question would be denoted by the variable x, as in the presented methods. After the influence of x on y is determined by the respective R^2 measure, a threshold is needed which establishes a rule on how much a variable is allowed to influence the data. If the regarded R^2 value surpasses this threshold, we call the respective variable a potential confounding factor. This process has to be repeated for every dimension of the reduced space or at least for the most informative ones. Having done this, we have a list of potential confounding factors, but still need to establish which of them to remove.

Analysis of potential confounding factors

Before we can remove the influence of the obtained potential confounding factors we have to analyze their influence on each other to detect hidden dynamics or dependencies. The motivation of this procedure requires some knowledge about the following step in the analysis, which is the removal of the influence of confounding variables on the data with the help of a linear regression model. With that in mind, we have to be careful to not remove two variables, which influence each other heavily, because the effects introduced by that and the results obtained may not be conclusive or aid in the further analysis. To reiterate: If the potential confounding factors influence each other to a certain degree and the influence of both is removed, the result might not be useful for the downstream analysis. Of course, a closer investigation of the relationship of such variables and the respective dataset, would aid in predicting the effects or consequences of a removal, but in general we want to avoid the removal of effects of related variables. Therefore, we state that it is sufficient and safer to remove only the effect of the confounding factor with the greater descriptive power concerning the data. What we mean by greater descriptive power naturally relates to greater influence on the data, which we determined with the respective R^2 measure. As we mentioned before the data is embedded in a space with reduced dimensions, which are often ranked according to their explanatory power within the data. Therefore, the selection also depends on the respective influenced dimension of the reduced space. This should also be taken into account.

Removal of confounding factors

Ultimately we are facing the following two questions: Which of the identified variables are confounding factors? and How to remove their effect from the data?

Before we try to answer the first question, we want to stress that not every highly influential variable should automatically be removed, because it depends on the question which is tried to be answered by the analysis. Therefore, we deliberately called the influential factors with the greatest descriptive powers so far only "potential" confounding factors. To illustrate what we mean by that, let us look at the following brief example. We assume the goal of an analysis is to establish whether cells, which were exposed to radiation, can be distinguished by their gene expression from untreated cells. Here, we would have the prior knowledge which cells were exposed and which were not. This translates to a categorical variable with two levels. Furthermore, we assume that the radiation heavily changed the gene expression profiles of the cells and therefore this variable gets labeled as a potential confounding factor. Now, it would not make any sense to remove the effect of radiation from the cells, because exactly this effect is the subject of the analysis. Therefore, the answer to the first question whether the effect of a variable, which was identified as potential confounding factor, should be removed from the data or not, is up to the question driving the analysis.

After we have established a certain set of potential confounding factors, which do not depend on each other in a significant way, and decided on the truly unwanted variables, thereby labeling them as confounding factors, we now have to remove their effect from the data. To remove the effect of a set of confounding factors from the data, linear regression can be used. Suppose we identified k confounding factors. The first step is to declare the confounding factors as explanatory variables $x_1, \ldots, x_k$ and construct a multiple linear regression model. This is done with the help of a design matrix, representing the confounding factors in the model, and/or coding systems in the case of categorical variables, as described in Section 3.4.2. Thereby, we obtain residuals res for every observation (cell). The last step is to use these residuals instead of the original values y for the downstream analysis, because they are per definition independent of the identified confounding factors.

Cluster Analysis

We want to briefly mention that the described methods can also be applied in the analysis of clustering results in two ways. Firstly, the clustering results, categorical variables which describe the cluster membership of every cell, can be analyzed in the same way as the explanatory variables. Due to the nature of the question in this case adjusted R^2 measures would be necessary, because we want to determine which clustering result best describes the data. Therefore, we have to penalize the number of found clusters, because more clusters would always induce higher R^2 measures. Secondly, we can determine whether there are explanatory variables (e.g. metadata), which explain the clustering results to a high degree. This helps in understanding driving factors within the clustering process and whether particular explanatory variables, which were not detected before, influence the clustering process to an unwanted extent.

Conclusion

Coming back to the topic of confounding factors after a detour into cluster analysis we are confronted with the following scenario: We have identified potential confounding factors, established their relation among each other and determined the ones with the greatest descriptive power. Finally, we showed how to decide on the real confounding factors and a way to remove their effect from the data. The presented methods were mostly linear in nature and based on the variance within the data or variables. During the identification of the influencing variables, we needed vectors y representing our data, therefore we applied dimensionality reduction. One of the most popular methods for dimensionality reduction is Principal Component Analysis (PCA), which is also linear in nature and based on the variance within the data. This fits well to the presented methods, for example the standard R^2 which determines the proportion of the variance explained by an explanatory variable, and therefore will be a main subject of the next chapter on dimensionality reduction.

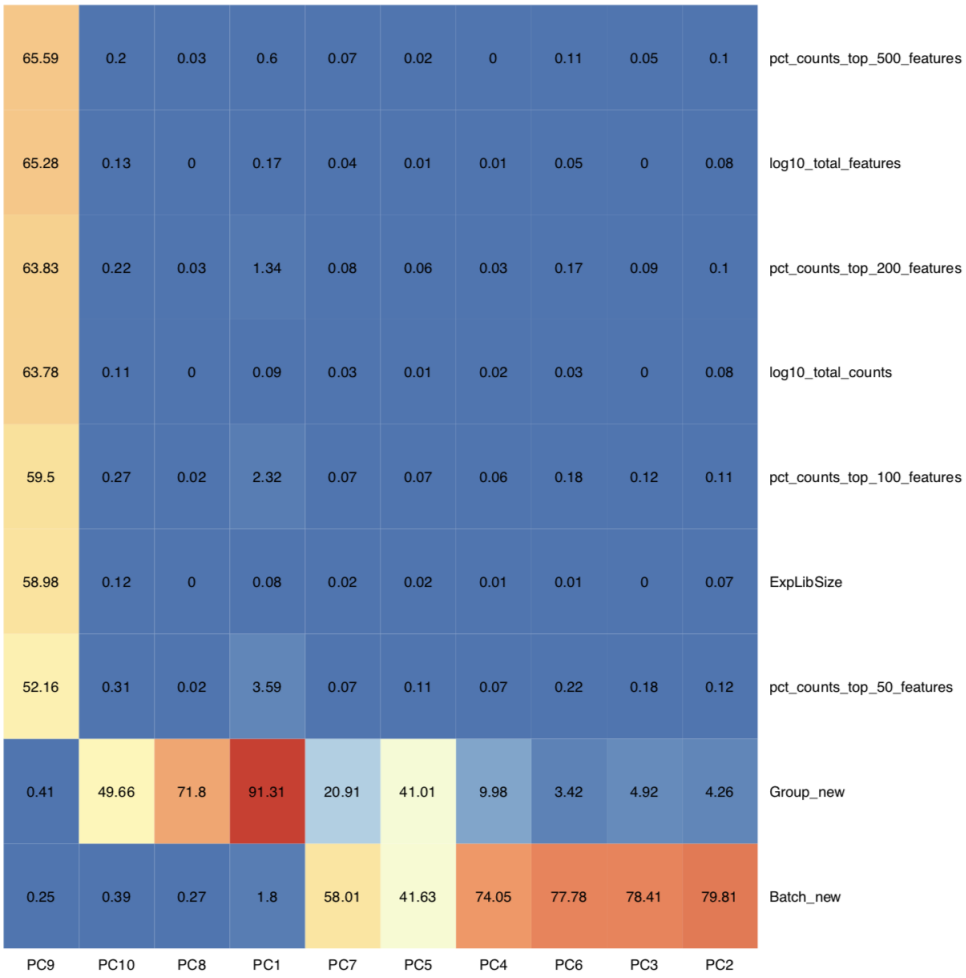

Figure 3.3.: The Heatmap is displaying the influence in percentage, determined by the standard R^2, of metadata variables on the 10 most informative principal components of the simulated dataset. The heatmap is clearly indicating that the variable Group_new (true clustering) and Batch_new are explaining a great proportion of the variance within the data.

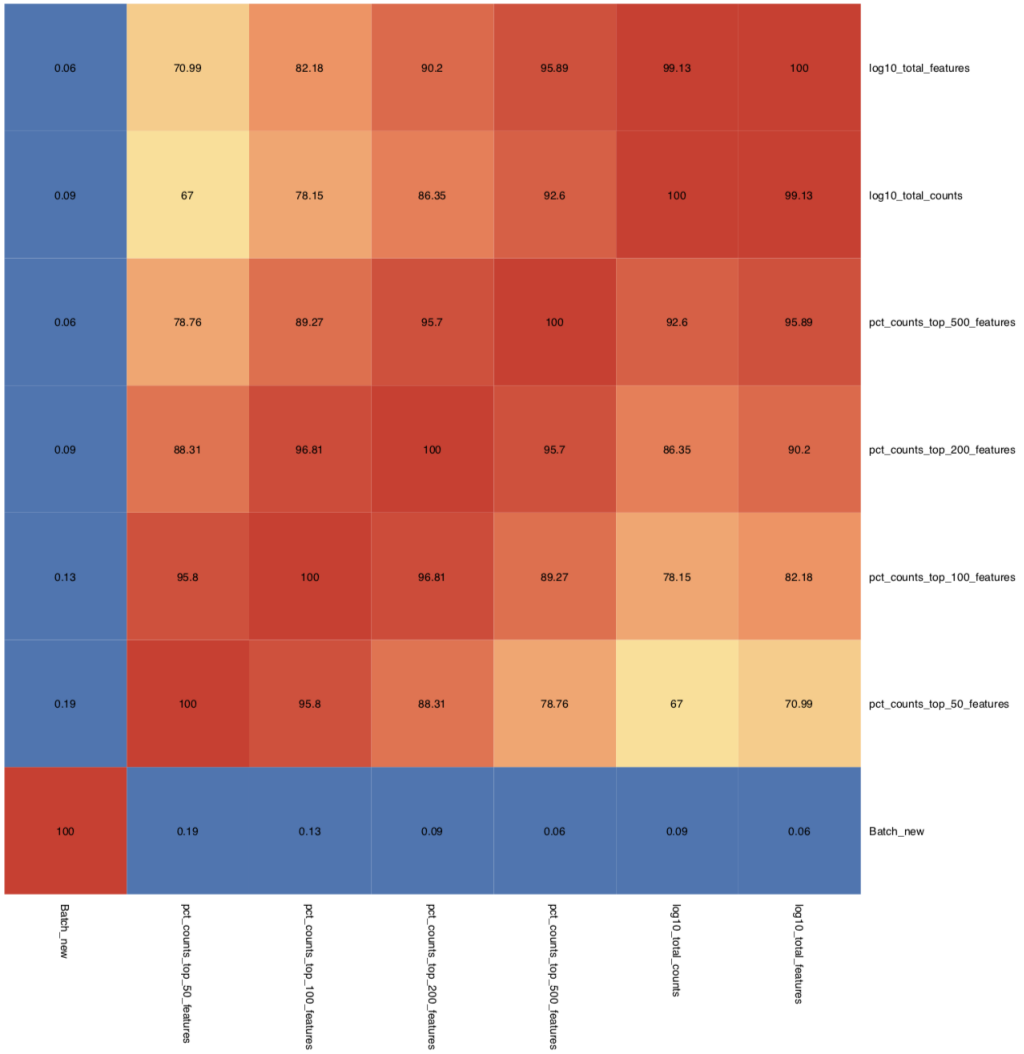

Figure 3.4.: The Heatmap is displaying the influence in percentage, determined by the appropriate R^2, of metadata variables on each other. We can see that the remaining variables, after applying analysis specific rules on which variables' effects are not allowed to be removed (e.g. Group_New), split up into two groups. Therefore, the decision of removing the effect of the variables Batch_new and pct_counts_top_500_features can be made.

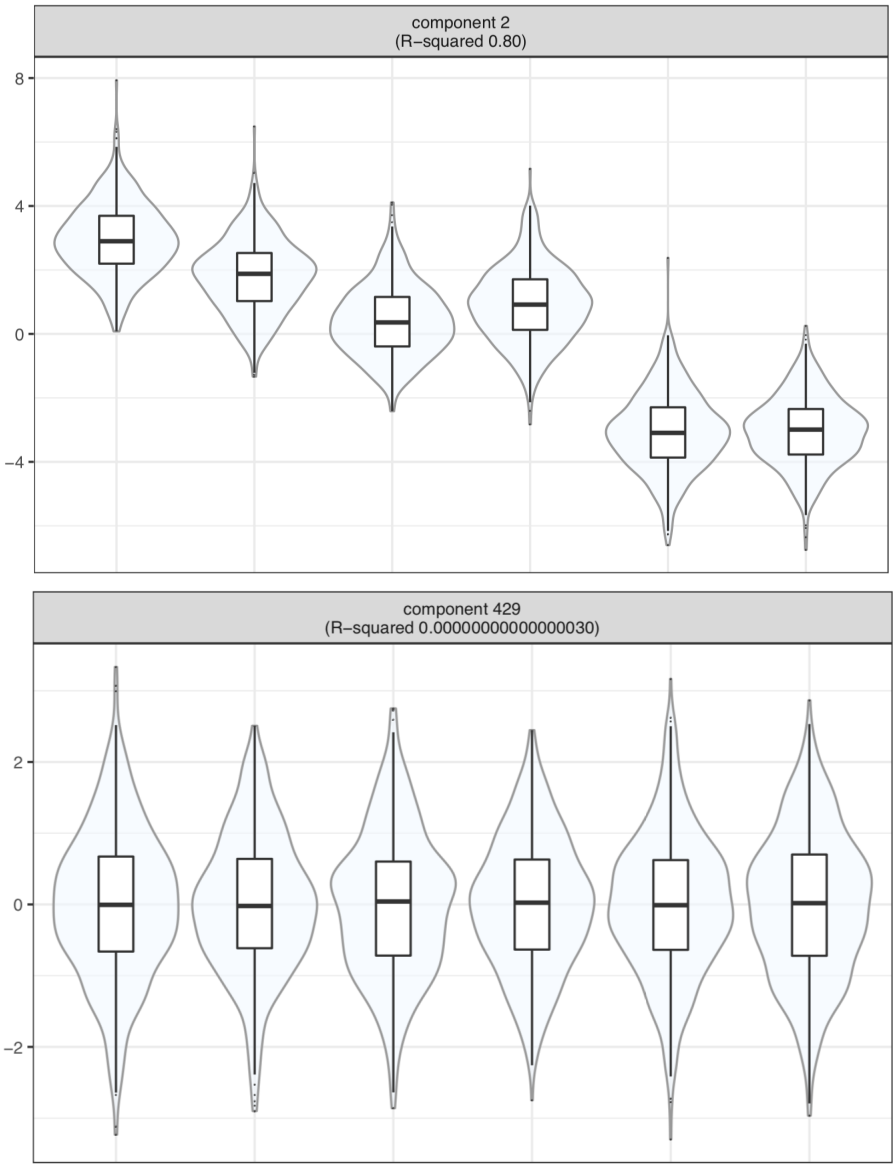

Figure 3.5.: Violin plots of the respective most influenced principal component by the categorical variable Batch_new before (top) and after (bottom) confounding factor removal .

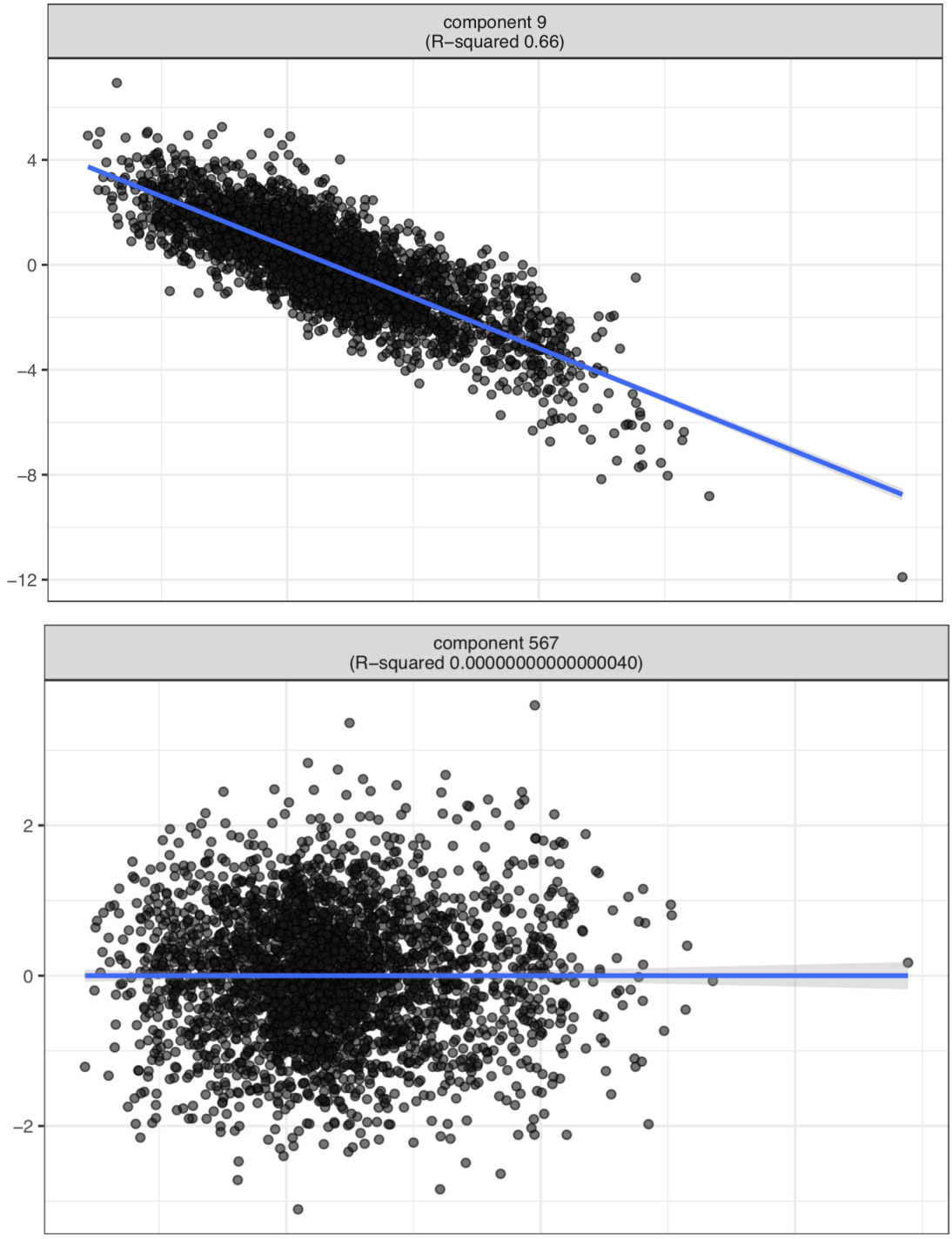

Figure 3.6.: Scatter plots of the respective most influenced principal component by the continuous variable pct_counts_top_500_features before (top) and after (bottom) confounding factor removal.

3.5. Dimensionality Reduction

One of the biggest challenges in dealing with data from scRNAseq experiments is the high number of dimensions (e.g. more than 32.000) present in a dataset. Therefore, one of the most critical procedures in scRNAseq analysis is the reduction of the dimensions. It is absolutely essential to the analysis, because the computational effort decreases with every dimension we can get rid of. Furthermore, the mechanisms and effects, we seek to understand, are easier to reveal by focusing on the most "relevant" dimensions regarding the experimental setup. By reducing dimensions we hope to increase the individual characteristics of particular effects and thereby make it easier to distinguish them from each other more distinctively. In the end the tough questions are: How do we find or define the most relevant dimensions? and What happens to the information lost by removing dimensions?

Often the most relevant dimensions can be found by combining the most descriptive dimensions or the ones with the biggest explanatory power of the variance and removing dimensions below a certain threshold of explanatory power. Thereby, we define on which aspect we want to focus (e.g. the variance and therefore on the differently expressed genes). The reduced dataset will be used in most of the computational expensive downstream analysis procedures and therefore the decision on the relevance is rather impactful. As stated above it is difficult to define what may be of importance and at the same time we always lose information by reducing dimensions. We have to always keep in mind that the reduced dataset only represents those aspects, which we deemed as important or relevant before. Consequently, only mechanisms within the remaining dimensions can be revealed or analyzed, respectively.

We will present a standard method to deterministically reduce dimensions (Principal Component Analysis) and a rather novel one (t-Distributed Stochastic Neighbor Embedding), which is stochastic in nature, but yields faster and often more insightful results. Both methods are frequently used, when dealing with high dimensional data.

At the end we compare the presented methods and point out explicitly where in a scRNAseq analysis workflow dimensionality reduction will be used.

3.5.1. Principal Component Analysis (PCA)

The goal of Principal Component Analysis (PCA) lies in determining a lower-dimensional picture, a projection of the data when looked at from its most descriptive angle, of the data, which actually lies in a much higher dimensional space. Often times it can be described as reduction to the drivers of the variance within the internal structure of the data.

The method has a lot of different names or variants depending on the mathematical field for example the eigenvalue decomposition (EVD) or its generalization singular

value decomposition (SVD) in linear algebra, factor analysis (FA) in statistics or the spectral decomposition in functional analysis. The main idea is to determine the principal components of a set of observations in our case the expression matrix. This transformation results in a set of independent vectors, which are an orthogonal basis set. The vectors are ranked by their associated eigenvalues. This implies that the first vector or principal component explains the largest proportion of the variance within the data and therefore turns out to be the most informative. Thereby, we already get a ranking from the most informative components to the least informative ones.

The method needs normalized data, either centered (=subtract the mean of each variable from the dataset), Z-score (=through standardization) or by another method. We have already discussed how important normalization for the downstream analysis is before in Section 2.3.5 and Section 3.3. In this chapter we assume that one of the previously presented normalization methods was applied on the data. PCA is sensitive to the relative scaling of the data and there is no measure to best transform the data to get optimal results.

We will now explain in detail how a PCA is done, based on Shlens tutorial on PCA [Shl03].

Mathematically speaking our goal comes down to a change of basis, which constructs the reduced space. To change the basis of the space, in which our data is embedded, we have to assume linearity. At this point we have to mention that this assumption was already implicitly expressed when we stated that the data itself can be described by its measured dimensions (e.g. genes). In such a linear system we want to get rid of two things: noise and redundancy. Noise can be defined as the irrelevant information or technical artifacts within the data. Therefore, we presume that the variance in the noise is significantly smaller than the variance of our signal or otherwise it is not possible to get any information out of the data. This relates to a very high signal-to-noise ratio (SNR), a common measure for noise.

Definition 46. We define the **signal-to-noise ratio** SNR, also known as ratio of variances as

$$\text{SNR} = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}$$

, where σ represents the standard deviation of the signal or noise, respectively.

Redundancy on the other hand can be described as the issue of two dimensions containing the same or highly correlating information. We capture this by using the covariance.

The new basis has to be found according to a goal. In our case we focus on the variance within the data to get rid of both above mentioned issues. This is achieved by looking at the components with the biggest variance within the data (which should not be noise) and the covariance of the given dimensions (to spot redundancy).

These considerations lead us to the fact that the covariance matrix of our data is an integral part of our further discussions.

To put this in a more rigorous mathematical context we use our previously defined expression matrix $M \in \mathbb{R}^{m \times n}$. Every row represents a gene and therefore a different measurement type or dimension. Columns are the denoted as cells and could be seen as the observations.

We are now looking for a matrix $P \in \mathbb{R}^{m \times m}$ to transform M to $N \in \mathbb{R}^{m \times n}$ by the following linear transformation

$$N = PM.$$

We could also interpret this transformation geometrically as a rotation and a stretch or even better that the rows of P are a set of new basis vectors for expressing the columns of M.

Next we think about the already mentioned covariance matrix S_M. We get this matrix by

$$S_M = \frac{1}{n-1} MM^T$$

with the following attributes. The elements of $S_M \in \mathbb{R}^{m \times m}$ describe the covariance of the i^{th} to the j^{th} gene. The diagonal terms of S_M are the variances of the respective genes. Therefore, S_M is a square symmetric matrix. Now we have a matrix (S_M) which exactly describes to us the relationship between the genes and at the same time gives us the variance of each gene within the data. What we would like to have is S_M to be diagonalized and ranked, because this would mean that all the genes do not have anything in common and we would get the most informative ones.

At the end we want to bring it all together. So lets rephrase our goal in a mathematical way: We are looking for a matrix P which transforms M, through $N = PM$, in a way that $S_N = \frac{1}{n-1} NN^T$ is diagonalized. The rows of such a P would then represent the principal components of M. We start out with the equation of our final term and then substitute N.

$$\begin{aligned} S_N &= \frac{1}{n-1} NN^T \\ &= \frac{1}{n-1} (PM) (PM)^T \\ &= \frac{1}{n-1} P \left(MM^T \right) P^T \end{aligned}$$

We define a new matrix $A = MM^T$, where A is symmetric. From linear algebra we know that symmetric matrices can be diagonalized with the help of a matrix E consisting of the eigenvectors of A as columns and a diagonal matrix D.

$$A = EDE^T$$

If the matrix A has a rank $r < m$, which would mean A is degenerate or all of the data occupies a subspace of dimension r, we fill up the matrix E with $(m - r)$ additional orthonormal vectors to maintain the constraint of orthogonality. This does not influence the result of our calculation, because the variances of those filled up dimensions are zero. The final and most critical step is now to select the matrix P as a matrix where each row is an eigenvector of $A = MM^T$. In other words we select

$$P = E^T$$

and substitute it in our equation

$$\begin{aligned} S_N &= \frac{1}{n-1} P\left(A\right) P^T \\ &= \frac{1}{n-1} P\left(EDE^T\right) P^T \\ &= \frac{1}{n-1} P\left(P^T D P\right) P^T, \end{aligned}$$

at last we know that the inverse of an orthogonal matrix is its transpose. So we use $P^{-1} = P^T$ and get

$$\begin{aligned} S_N &= \frac{1}{n-1} P P^T D P P^T \\ &= \frac{1}{n-1} \left(P P^{-1}\right) D \left(P P^{-1}\right) \\ S_N &= \frac{1}{n-1} D. \end{aligned}$$

With this choice for P we successfully diagonalized S_N and thereby reached our previously formulated goal. In the end the matrix M is transformed to N by $N = PM$. To keep everything consistent we denote $M_{redcued} = N$.

To summarize, we derived a way to get a diagonalized version of the covariance matrix of our data, which implies the independence of the dimensions. Furthermore, the values of this matrix represent the variance of the respective dimension. For the computation we need to do two things: first normalize the data and then compute the eigenvectors of the normalized dataset, to get the principal components.

Assumptions & Limitations

In the described process we have made some **assumptions**, which we want to summarize in the following

- **Linearity** is the main assumption for our goal of changing the basis of the space in which the data initially is embedded. Without linearity a basis change would not be a valid approach.

- **Mean and variance** are the only statistics used to capture all the information within the data and decide on what are the most informative components. The only distribution, which is fully described by those two measures, is the Gaussian distribution, therefore we implicitly assume that our data is usually normal distributed. Luckily, a lot of the observed data in nature is normally distributed and additionally we can achieve normalization by applying previously introduced methods.
- **Large variance means important information**. And lower ones represent noise. Those two statements are the basis for the ranking done by the eigenvalues.
- **Orthogonality** of the principal components. This assumption provides a good way to tackle the problem with the help of linear algebra.

The second and the third point additionally assume or imply the fact that the data should have a high SNR.

Furthermore this method has its **limitations**

- **No parametrization**, which is actually also a strength, because independent of the user the answer is always unique. It gets problematic in the case of having a-priori knowledge about the system and not being able to incorporate it in the process.
- **Linearity** is a necessary condition for using methods of linear algebra, but in certain cases a non-linear transformation prior or within the dimensionality reduction would lead to far better results. Sometimes those transformations are called kernel transformations and therefore the modified PCA approach is named kernel PCA. Such a kernel could be the transformation of cartesian coordinates to polar coordinates, Fourier- or Gaussian transformations.
- The above described **assumptions** can be to strict, as for example in a situation where the orthogonality of the components or the distribution to be Gaussian is not needed. With less constraints we face a set of problems which are not easily solved. One rather recent approach is called Independent Component Analysis (ICA) where the goal, to find a matrix P with $N = PM$ and S_N is diagonalized, is the same without the above stated assumptions except linearity, which is still required.
- **Preserving large pairwise distances.** Dissimilar points in the high dimensional space are kept far apart, but for high dimensional data that lies on a low dimensional non-linear(!) manifold it is often more important to keep similar datapoints close together. This can not be achieved consistently with linear transformations.

We mentioned that there are more variants of PCA and further developed methods. One of them was ICA, which relies on the concept of finding a basis such that the joint probability distribution can be factorized, which in turn results in statistical independence of the components. In contrary to PCA the result is not unique.

Singular Value Decomposition (SVD)

In the beginning of this chapter we stated that PCA is sometimes, depending on the mathematical discipline, also called Singular Value Decomposition (SVD). Actually, SVD is a more general method of factorizing any matrix M into three distinct components. The exact theorem is formulated as follows.

Theorem 47. *For every $n \times m$ matrix N, with either real or complex numbers, there exists a factorization, called **singular value decomposition (SVD) of** N,*

$$N = U\Sigma V^*,$$

*with U being a $n \times n$ unitary matrix (=columns and rows are orthonormal to each other regarding the scalar product), Σ is a diagonal matrix consisting of a ranked set of **singular values**, which are uniquely defined by N, and V^* is the conjugate of a $m \times m$ unitary matrix.*

Let us now link this theorem to the concept of PCA. Suppose we define a new $n \times m$ matrix Y with the help of the expression matrix M,

$$Y = \frac{1}{\sqrt{n-1}} M^T$$

and investigate the term $Y^T Y$. Thereby we get

$$Y^T Y = \frac{1}{n-1} M M^T = S_M,$$

which is the covariance matrix of M. We already know that the principal components of M are the eigenvectors of S_M. By applying Theorem 47 on Y we can show the following relationship.

$$\begin{aligned} S_M &= Y^T Y \\ &= (U\Sigma V^*)^T (U\Sigma V^*) \\ &= V\Sigma U^T U\Sigma V^* \\ &= V\Sigma^2 V^* \\ \Rightarrow S_M V &= V\Sigma^2 \end{aligned}$$

At last we see that the columns of V will give us exactly the eigenvectors of $S_M = Y^T Y$. Therefore, we can calculate the principal components of M, by determining the SVD of Y.

Another method, which is also widely used in scRNAseq analysis and leverages probability theory, is t-Distributed Stochastic Neighbor Embedding (t-SNE). The next chapter will give an introduction to this rather novel method.

3.5.2. t-Distributed Stochastic Neighbor Embedding (t-SNE)

Van der Maaten and Hinton presented in [VH08] a new method for dimensionality reduction with the goal of visualizing high dimensional data. They recognized the absence of visualization methods, which capture more than a few variables at a time and yield results in reasonable amounts of time and computational effort. Visualization, as stated before, is basically a very rigorous way of reducing dimensions to only two or three.

The idea of t-SNE lies in building a low dimensional map in which distances between datapoints reflect similarities in the dataset. The main focus lies in preserving smaller pairwise distances, in contrast to linear methods as PCA. It then tries to iteratively minimize the discrepancies between similarities in the high dimensional space and similarities in the low dimensional representation. To measure the discrepancy we need a function to describe it. This function should then be minimized.

We now take a step away from our mathematical framework for scRNAseq analysis and introduce a more generic setting for the explanation of this method. At the end we will put the derived method back into our context. Our goal is to convert a high dimensional dataset $X = \{x_1, \dots, x_m\}$ into a two or three dimensional dataset $Y = \{y_1, \dots, y_m\}$. Due to the fact that we want to reduce dimensions down to two or three, the dataset Y can be referred to as a map.

To measure the pairwise similarities between the high dimensional data points we convert euclidean distances to conditional probabilities. In other words we want to convert the distance between x_i and x_j to a conditional probability $p_{j|i}$. This value should represent the probability that the point x_i chooses the point x_j as its neighbor, assuming neighbors are chosen according to their probability density under a Gaussian with mean at x_i. This results for nearby points in a high value and for far reaching points in a value close to zero. In other words, we are mainly looking at local distances and thereby similarities, in contrast to global ones. Additionally, to the conditional probability we need to put the value into a context. This can be achieved by normalizing only over pairs of points that involve the point of interest x_i. This is done to focus even more on the local mechanisms.

Definition 48. We define the **conditional probabilities $p_{j|i}$ of neighborhoods in the high dimensional space** as

$$p_{j|i} = \frac{\exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(\frac{-\|x_i - x_k\|^2}{2\sigma_i^2}\right)},$$

where x_i is the point of origin, x_j is the point in question and σ_i^2 is the variance of the Gaussian distribution with mean x_i.

We are only concerned with pairwise similarities and therefore set $p_{i|i} = 0$.

To explain the origin of the parameter σ_i^2, the variance of the used Gaussian distribution, we have to first look at some challenges concerning the parameter selection. Due to the fact that the density of datapoints in the dataset varies drastically it is unlikely that there is an optimal value σ_i^2 for all datapoints. Dense regions would benefit of smaller σ_i^2 values and sparser areas need bigger ones. The parameter σ_i^2 induces a probability distribution P_i with a certain entropy. Before we elaborate further, we have to put the term entropy into context.

Definition 49. We define the **Shannon entropy H** of P_i, measured in bits by

$$H(P_i) = -\sum_j p_{j|i} \log_2 \left(p_{j|i}\right),$$

where $p_{j|i}$ is the previously defined conditional probability of x_i choosing x_j as its neighbor.

With increasing σ_i^2 the entropy of the corresponding probability distribution P_i increases as well. Having the term entropy in place we will use it to define the term perplexity.

Definition 50. The **perplexity Perp** of a certain probability distribution P_i is defined as follows

$$\text{Perp}(P_i) = 2^{H(P_i)},$$

where $H(P_i)$ denotes the just defined Shannon entropy of probability distribution P_i.

Having already stated that it is unlikely to find an optimal σ_i^2 value for all datapoints, we rather choose a perplexity Perp* beforehand and then look for an appropriate σ_i^2 (e.g. by binary search) that generates a probability distribution P_i with $\text{Perp}(P_i) = \text{Perp}^*$. As we can see the perplexity increases monotonically with the variance.

One last remark on the interpretation of the perplexity term. The authors state that it can be seen as a smooth measure for the effective number of neighbors. Furthermore, they say that the performance is rather robust concerning variations in the perplexity and that values ranging from 5 to 50 are typical.

Next, we symmetrize those conditional probabilities, to define symmetric joint probabilities in the high dimensional space, by basically averaging the conditional probabilities.

Definition 51. We define the **joint probabilities** p_{ij} in the high dimensional space as the symmetrized conditional probabilities

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2m},$$

where m is the total number of points in the high dimensional data set X.

Thereby, we can simplify the function (= cost function, which will be defined later), which we have to minimize, and reduce computational effort. Additionally, by symmetrizing it in this way we ensure that every datapoint x_i makes a significant contribution to the cost function, which would not have been the case if we would have achieved symmetrization by

$$p_{ij} = \frac{\exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)}{\sum_k \sum_{l \neq k} \exp\left(\frac{-\|x_k - x_l\|^2}{2\sigma^2}\right)}.$$

Here we would have encountered a problem with outliers, because their p_{ij} values would have been very small and therefore not impacted the cost function.

To measure the pairwise similarities in the low dimensional space we use the same approach, but execute it slightly different. In contrast to the conditional probabilities $p_{j|i}$, which use a Gaussian distribution, we take advantage of the properties of the Student t-distribution with one degree of freedom (= standard Cauchy distribution).

Definition 52. Using the Student t-distribution with one degree of freedom we define the **joint probabilities q_{ij} of neighborhood in the low dimensional space** as

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_k \sum_{l \neq k} \left(1 + \|y_k - y_l\|^2\right)^{-1}},$$

where y_i is the point of origin and y_j the point in question.

As before we set $q_{ii} = 0$.

Why did the authors use another distribution and not again a Gaussian? The answer to this question is manifold and we will go through it step by step.

- The **heavy tails** of this distribution, compared to a Gaussian, give us a natural way to alleviate a problem, which is called the **crowding problem**. This problem describes the difficulty of perfectly preserving distances from the high dimensional space in the map. This can not be done in every case, especially when the data is intrinsically high dimensional. This means that the data itself exhibits a high dimensionality. It is for example not possible to map three points, with the same distance between each other, in two dimensions (a triangle) to one dimension (on a line) with preserved distances. Therefore, the heavy tails of the Student's t-distribution allow dissimilar points to be modeled as "too" far apart in the map.
- For large pairwise distances $\|y_i - y_j\|$ the term $\left(1 + \|y_i - y_j\|^2\right)^{-1}$ gets close to an inverse square law in the low dimensional map. Therefore, we introduce an **invariance to changes in the scale of the map** for points, which are far apart.

- The Student's t-distribution is **closely related to the Gaussian distribution**, because it is equivalent to an infinite mixture of Gaussians with different variances.
- It is a lot **faster to evaluate** the density, because no exponential calculations are involved.

The main goal now, is to achieve a mapping, which yields $p_{ij} = q_{ij}$ as a result for all points. In other words we want to minimize the mismatch between q_{ij} and p_{ij}.

The last thing missing is the before mentioned (cost) function, which should describe the discrepancy between the high dimensional similarities of the data points and the corresponding similarities on the low dimensional map. For this purpose we introduce the Kullback-Leibler divergence as a natural measure of how good q_{ij} models p_{ij}.

Definition 53. The Kullback-Leibler divergence between a joint probability distribution P, in the high dimensional space and a joint probability distribution Q in the low dimensional space is defined by

$$KL\left(P||Q\right) = \sum_{i} \sum_{j \neq i} p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right).$$

Since we want to minimize this Kullback-Leibler divergence, we set it as the cost function $C\left(P, Q\right)$.

$$C\left(P, Q\right) = KL\left(P||Q\right)$$

At last we need the gradient of this cost function $C\left(P, Q\right)$, to minimize it.

Theorem 54. *The gradient of the defined cost function $C\left(P, Q\right)$ is given by*

$$\frac{\delta C}{\delta y_i} = 4 \sum_{j} \frac{\left(p_{ij} - q_{ij}\right)\left(y_i - y_j\right)}{\left(1 + \left\|y_i - y_j\right\|^2\right)}.$$

The low dimensional map Y is constructed by using an algorithm, which starts with a **random** sampled initial solution $Y_{(0)}$ and iteratively improves this solution until $Y_{(T)}$ is generated, where T describes the number of iterations. The improvement or learning process is achieved through recomputing the low dimensional similarities q_{ij} of $Y_{(t-1)}$ after every iteration and using the updated gradient of the cost function $\frac{\delta C}{\delta Y}$ to calculate a new low dimensional data set Y_t.

Theorem 55. *The improvement can be achieved by iteratively determining $Y_{(t)}$ based on the previously calculated $Y_{(t-1)}$, with an initial sample solution $Y_{(0)} \sim \mathcal{N}\left(0, 10^{-4} I\right)$, where I is the $m \times n$ Identity Matrix, and $Y_{(-1)} = 0_{m,n}$ a zero matrix, where m is the number of datapoints and n is the number of dimensions of the low dimensional map Y. This is achieved by*

$$Y_{(t)} = Y_{(t-1)} + \eta \frac{\delta C}{\delta Y} + \alpha\left(t\right)\left(Y_{t-1} - Y_{t-2}\right),$$

where η is the learning rate, $\frac{\delta C}{\delta Y}$ is the gradient of the cost function, which has to be updated after every iteration, and $\alpha(t)$ is the momentum.

A few remarks on the parameter η and the function $\alpha(t)$. The learning rate η describes how fast new learnings, which are determined by the gradient of the cost function, are adopted by the algorithm. If η is to big, the changes within the cost function would have a greater impact on the adoption of the equation and this would lead to an overcorrection and thereby to an increase in error or even divergence. If it is to small, the adoption to the changes would be slow and it would take a lot longer to converge or in this case at time T the result $Y_{(T)}$ would not be as conclusive. In the publication the authors chose initially $\eta = 100$, but then used an adaptive learning rate scheme, which adapts η in every iteration, to reduce computational effort.

The function $\alpha(t)$ describes the momentum at time t and it actually represents a time dependent factor for the whole momentum term $\alpha(t)(Y_{t-1} - Y_{t-2})$. This term is added to speed up the optimization process and helps to avoid poor local minima. In this case the authors chose $\alpha(t) = 0.5$ for $t < 250$ and $\alpha(t) = 0.8$ for $t \geq 250$, with the total number of iterations $T = 1000$.

Limitations & Extensions

At the end we want to point out some limitations and extensions of t-SNE.

Limitations

- This method should **only** be used for the purpose of **visualizing** high dimensional data. The authors specifically stated that it is not foreseeable how it will behave when used like a standard dimensionality reduction method. Furthermore, the primary goal of the method was to give the analyst a feeling for the data and what kind of behavior to expect. It is also used to check if certain attributes (dimensions) capture the nature of the data.
- The **computational effort** can be too high, because we have to consider all parallel interactions between points. This means if we have m points we have to look at m^2 interactions between points and those have to be summed whenever the gradient is calculated, which happens in every iteration of the process. Therefore t-SNE is computationally very expensive and limiting, when visualizing 5.000 to 10.000 datapoints. In scRNAseq it is not uncommon to analyze more than 10.000 cells.
- There are situations in which it is simply **not possible to get a correct map**, based on similarities. One example would be the attempt to visualize words by their semantic similarities or associations in a single map. This can never be done right, because of different meanings of one word. The same problem occurs if authors and co-authors are mapped as relationships. It can happen

that author A and author B wrote a lot of papers together and therefore will be mapped close to each other. If author B also wrote a lot together with author C, but A did not write anything with author C, they will still be mapped closely together, because of their relationship with author B. So we can not model that correctly. In this case it is a matter of perspective. Mathematically we face the problem of the similarity structure not being metric, because the triangle inequality is not satisfied.

- **Too high intrinsic dimensionality** and an underlying highly varying manifold violates the implicitly made assumption of local linearity. As a result of these circumstances t-SNE might not be successful in preserving local similarities and yielding a map that represents the relations of datapoints to each other.

- The **non-convexity** of cost function leads to the usage of more optimization parameters and consequentially the result depends on the choice of those parameters.

Possible **extensions** to overcome the stated limitations of the standard t-SNE method are as follows.

- The **Barnes-Hut-SNE** described by van der Maaten [**Van14**] tries to reduce the computational effort through approximation. Thereby, decreasing the cost from m^2 to $m \log(m)$. The idea is based on the assumption that many of the pairwise interactions between points are very similar. The approach is to find the center of mass of close together groups and calculate the similarity of that center to far away points. This similarity is multiplied by the number of points in the group. The method is called Barnes-Hut approximation and it is coming from astronomy. There they try to model the interaction between stars. This variant of t-SNE can be implemented with Quadtrees to determine the groups.

- To address the issue of non-metric similarity structures, an alternative approach using **multiple maps** instead of a single map was proposed by van der Maaten and Hinton [VH12]. The authors state, that due to the probabilistic nature of t-SNE it can be easily extended to leverage multiple maps. In multiple-maps t-SNE each object gets a point in each map. Then an importance weight is assigned to each point in each map. In the end the low-dimensional similarity is defined between two points under the multiple maps model as a weighted sum over the similarities in the individual maps. Mathematically the similarities in the low dimensional maps q_{ij} are differently defined by incorporating the different maps and the weights of the objects in the respective maps. Thereby, the "perspective" on the map can be shifted. For the sake of a clearer notation we denote with N the number of datapoints and M the collection of maps m. As already mentioned we have to redefine the

similarities q_{ij} of the low-dimensional map

$$q_{ij} = \frac{\sum_m \pi_i^{(m)} \pi_j^{(m)} \left(1 + \left\| y_i^{(m)} - y_j^{(m)} \right\|^2\right)^{-1}}{\sum_k \sum_{l \neq k} \sum_{m'} \pi_k^{(m')} \pi_l^{(m')} \left(1 + \left\| y_k^{(m')} - y_l^{(m')} \right\|^2\right)^{-1}},$$

where $y_i^{(m)}$ denotes the location of word i in map m and $\pi_i^{(m)}$ the importance weight that measures the importance of word i in map m. The weights have to fulfill $\forall i \forall m : \pi_i^{(m)} \geq 0$ and $\forall i : \sum_m \pi_i^{(m)} = 1$. In the nominator $\left(1 + \left\| y_i^{(m)} - y_j^{(m)} \right\|^2\right)^{-1}$ describes the similarity of words i and j in map m and we multiply that by the according weights and finally sum over all maps. Thereby, all the maps and the importance weights are learned simultaneously. The goal remains the same, namely to minimize the Kullback-Leibler divergence, as we have shown before.

In conclusion, the visualization method t-SNE tries to lead to insights in high dimensional data by reducing dimensions to two or three, while trying to preserve local similarities within the data. It is widely used in a range of domains and modified variants are in development. Most of the time already reduced datasets, for example by PCA, are used as input for t-SNE to decrease computational effort.

3.5.3. Application & Comparison

Two very different methods were presented and now we will discuss their application in scRNAseq analysis and how the before introduced terms translate into our mathematical framework.

The first approach was Principal Component Analysis (PCA), which is a deterministic method to reduce the existing dimensions to its principal components based on the variance within the data. The second method, t-Distributed Stochastic Neighbor Embedding (t-SNE), on the other hand is stochastic in nature and tries to preserve local similarities in a two or three dimensional map of the data. Both methods are applied in scRNAseq analysis, but on rather different occasions.

Principal Component Analysis

In the case of PCA the observations are the cells $c_j \in C$ and the measurement types are the genes $g_i \in G$. After performing the PCA on the (normalized) expression matrix M_{NORM} downstream analyses are computationally less expensive and more importantly will yield more meaningful results, because noise was reduced and the most informative dimensions (by variance) within the data were highlighted. Most of the time further analyses are done only on the most informative principal components derived by PCA. The number of components chosen depends on the data set.

Usually bigger datasets convey more information and therefore more components are used. This reduced dataset will be used in

- **Confounding Factor Analysis** for identification of potential explanatory variables by analyzing their influence on the principal components and thereby on the data,
- **Clustering** as an input, because of the more pronounced attributes and computational simplifications,
- **Cluster Analysis**, when trying to determine which clustering result best describes the data, we can use PCA to represent the data,
- **Differential Gene Expression Analysis** for determining differences in found clusters by looking at the difference on the reduced instead of the entire dataset and
- **Visualization**, either by plotting two or three principal components or as input for methods such as t-SNE.

Actually nearly every downstream analysis step will be performed on some kind of reduced dataset.

t-Distributed Stochastic Neighbor Embedding

In t-SNE the high dimensional datapoints x_i are the cells $c_j \in C$ and the dimensions of x_i are the genes $g_i \in G$. This means that the columns of our expression matrix M represent the high dimensional datapoints x_i. In scRNAseq analysis t-SNE is commonly used for visualization (its main purpose) to achieve two things.

- **Visual investigation of the data** after quality control, normalization, confounding factor analysis and dimensionality reduction. Often to compare the visualization of the data before and after those steps. Due to the fact that t-SNE tries to preserve and visualize the local similarities within the data, clusters should form on the map. Those clusters give an idea of the different cell types, which can be expected, or other effects, present within the dataset. By coloring according to metadata, unwanted effects could be revealed (e.g. batch effect). Also the number of different clusters, which should be targeted by certain clustering approaches, can be anticipated.
- As a **control mechanism** during the cluster analysis process. After performing the clustering step and deciding on one solution we can visualize the data with t-SNE and color the cells by their cluster affiliation. If the visual clusters of t-SNE map and the colored clusters coincide, it is a strong indicator that on the one hand t-SNE already captures the found structure within the data and on the other hand that two different approaches yield the same or a similar result. This usually can be considered as a confirmation on the chosen clustering solution. To leverage these circumstances one further clustering attempt

is to use a centroid-based clustering approach on the two or three dimensional map, generated by t-SNE. This option will be discussed in the next chapter.

This approach should be taken with a grain of salt, because t-SNE is only able to visualize two or three dimensions. Therefore it could happen that one visual cluster found by t-SNE can consist of a homogenous mixture of cells from two or more cell types. This would result in a mixed color pattern and thereby contradict the previously stated strategy for decision making. However, this could still be a valid result of both processes, visualization by t-SNE and clustering respectively, because t-SNE could have captured the general cell-type and the clustering found subpopulations of that cell-type. Therefore, it is important to use t-SNE and different clustering methods together to ensure optimal results.

In scRNAseq analysis the input X for t-SNE should always be the already dimensional reduced dataset $M_{reduced}$, this can be achieved by performing for example PCA beforehand.

Comparison

As we just demonstrated, the two methods can not really be compared, because of their different fields of application. Moreover t-SNE uses PCA in most implementations in scRNAseq analysis, because of the high number of dimensions and datapoints. They complement each other nicely and supply the potential of a better analysis.

We already touched upon the topic of clustering, which is one possible endpoint of a scRNAseq analysis. In the next chapter we give an overview of the clustering approaches used in scRNAseq analysis.

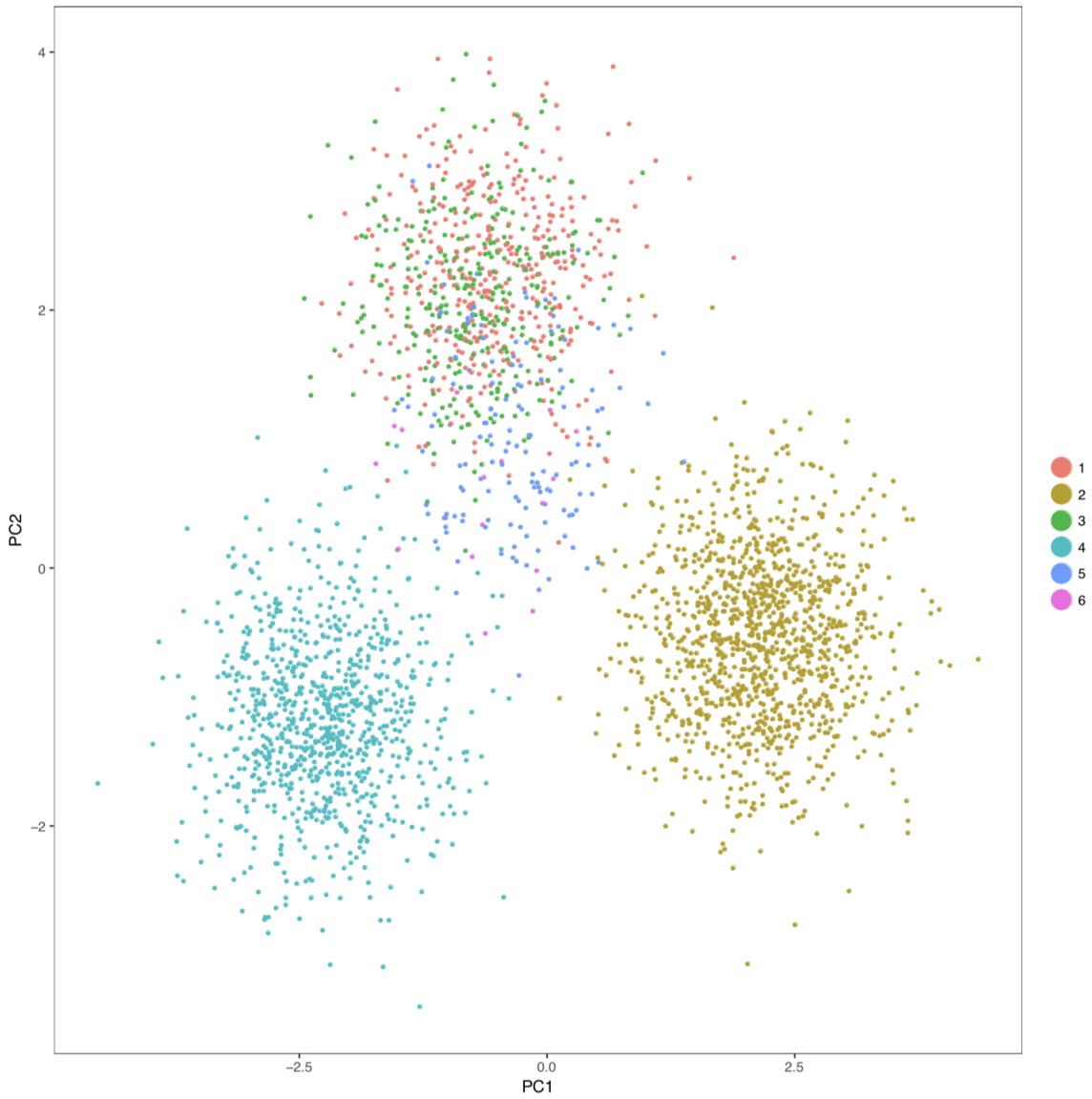

Figure 3.7.: PCA plot of the first and second principal component of the simulated dataset colored by the clustering result obtained by applying the consensus approach presented in Chapter 4.

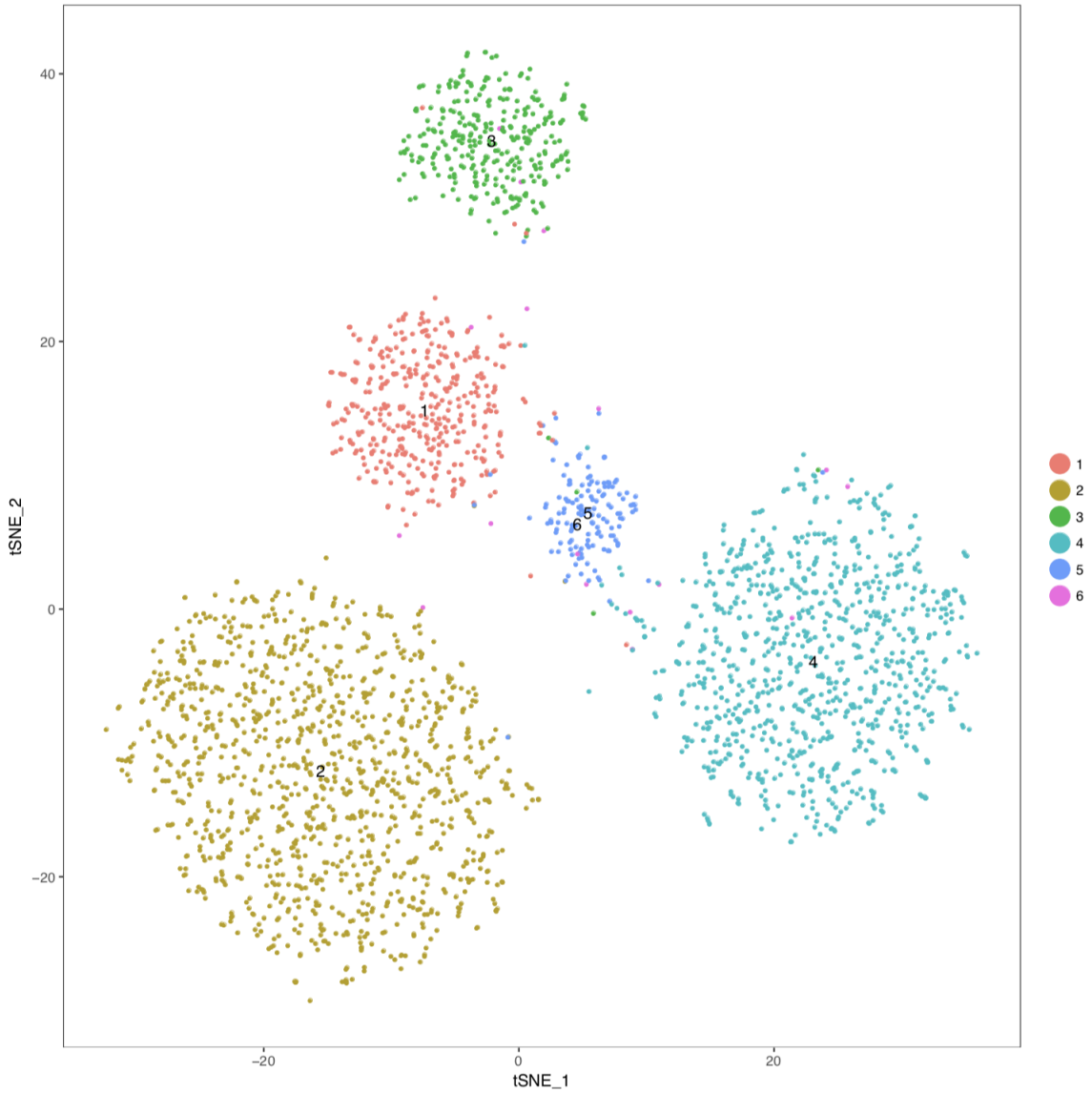

Figure 3.8.: Plot of a two dimensional t-SNE map of the simulated dataset colored by the clustering result obtained by applying the consensus approach presented in Chapter 4.

3.6. Clustering

This chapter is dedicated to a topic, which filled books for the last decades, namely clustering. We have already elaborated on the importance of clustering to identify cell types or even unknown subpopulations within a cell type in Section 2.3. Furthermore, clustering can be used to identify the time course of changes in the expression profile of a cell population, which can be used for detecting effects of a certain exposure. A few examples would be treatment with a medical compound, exposure to radiation or progression of a disease. Therefore, clustering by cell type or other effect is one of the primary objectives in scRNAseq analysis in general. As we do not have any given labels, categories or a ground truth to compare to or use as a training set we will only talk about unsupervised approaches.

We will not even try to cover every aspect of the topic clustering, because it is simply not possible in one chapter and out of scope of this work. Instead we will focus on constructing a mathematical framework in which different approaches can be described and apply it to one very popular kind of clustering algorithms. Then we will describe further approaches, which are most frequently used in scRNAseq analysis, in a qualitative manner. The general clustering approach descriptions follow the chapter "A Survey of Clustering Data Mining Techniques" by Berkhin from the book "Grouping Multidimensional Data: Recent Advances in Clustering" [KNT06]. Every approach will be highlighted by at least one exemplary algorithm, which is used in scRNAseq anaylsis. In the end we will try to compare the presented approaches and touch upon a huge challenge, which will be the subject of Chapter 4.

Let us start by defining a few key terms to setup the mathematical framework. The result of a clustering algorithm is the classification of data in different categories or groups, which we call clusters. The clustered cells should have more in common, concerning the attributes, which are captured by the algorithm, than to those in other groups. All clusters together are called a clustering and represent one possible classification solution.

Definition 56. We will call a set of cells $C_k = \{c_j | j \in J_k\}$ a **cluster** k, where $J_k \subset J$ are the indices of the cells in the cluster.

Building on that, we define the term clustering in our context.

Definition 57. A **clustering** CL is a group of clusters, which are defined on the same dataset and the union of all clusters within one clustering represents the whole data set.

$$CL = \{C_k | J_k \subset J\}$$

$$\bigcup \{C_k | C_k \in CL\} = C_0 = \{c_j | j \in J\}$$

A clustering can capture either the main aspect we are looking for (e.g. cell type) or it classifies the cells according to other attributes within the data (e.g. batch, disease or treatment effects). One important distinction has to be made.

Definition 58. If the clusters of one clustering are pairwise disjoint sets we call it a **hard clustering**.

$$\begin{aligned} C_i \cap C_j &= \emptyset \\ i &\neq j \\ \forall C_i, C_j &\in CL \end{aligned}$$

On the other hand, if a cell can belong to more than one cluster to a certain degree, we will call it a **soft clustering**.

In the following we will only discuss hard clusterings, because for most of the biological questions a distinct classification for the cells is necessary. Now, we have all the building blocks to start exploring the realm of clustering.

3.6.1. Partitioning-Based Clustering

This approach and the respective example (k-means) is one of the most commonly used, established and fastest ones. The idea is to divide the data into subsets without testing every possible combination. Therefore, different relocations schemes, which reassign points in an iterative manner to k different clusters until a condition is met, are applied. These schemes are referred to as iterative optimization. The relocation thereby improves the cluster assignments in a step-wise manner.

We will now discuss in detail a very popular partitioning-based algorithm, namely k-means. The basic principle of this clustering approach is to find K centroid-points within the dataset such that the distances from the datapoints, which are assigned to the K clusters, to their cluster-centroids are minimized. This centroid point is presented by a vector, which does not necessarily have to be part of the data (in contrast to the k-medoid approach, where the representative point is one of the datapoints). Therefore, we are faced with an optimization problem which is NP-hard, but very fast heuristic implementations exist.

Two problems immediately come to mind:

1. How to find the optimal number K of clusters within the data without prior knowledge?
2. How to find the best centroids of these K clusters?

Problem one is a very common question in the domain of cluster analysis with no (simple) answer. The next section will partially deal with the topic of finding or estimating the optimal number of clusters within the data. This task is especially hard, if there is no further information about the data, what we have to assume most of the time.

The second problem on the other hand has a solution. We find the "best" centroids by an iterative refinement technique after initially random centroids were set. This is

achieved by assigning the datapoints to the nearest (depending on the used metric) centroid and thereby to a cluster. Next, we calculate a new centroid m_k within each cluster C_k, by taking the mean over all the data points within this cluster $m_k = \frac{1}{|J_k|} \sum_{j \in J_k} c_j$. We continue by repeating those two steps until no further change is observed or a certain number of iterations is reached.

The result of this approach is a Voronoi diagram as presented by Joswig and Theobald [JT08], which is basically a partitioning of space into subsets following specific rules concerning distances between points. Each subset is called a Voronoi cell and in our case they represent the clusters C_k. Putting the previously explained k-means approach and this short description of the Voronoi diagram together, we end up with a definition of how the clusters are put together.

Definition 59. In mathematical terms a **cluster** in k-means is defined by

$$C_k = \{c_j \in C | d(c_j, m_k) < d(c_j, m_i) \, \forall i \neq k\},$$

where $d(\cdot, \cdot)$ denotes the chosen metric and m_k, m_i describe the respective centroids of the clusters C_k, C_i.

We stated before that this is actually an optimization problem. Therefore we also want to give the mathematical description of the measure, which we minimize.

Definition 60. We define the **within-cluster sum of squares (WCSS)** as follows

$$WCSS(CL) = \sum_{i=1}^{K} \sum_{c_j \in C_k} \|c_j - m_k\|^2.$$

To reiterate, the final goal is to find the clustering CL which minimizes the $WCSS$ which translates to finding

$$\arg\min_{CL} WCSS(CL).$$

The attentive readers have already grasped that there are a few **drawbacks** to this approach, but we will discuss them anyway in a concise way:

- The method needs a certain number of clusters to look for. This is not a trivial matter and will be discussed throughly later in Chapter 4.
- The approach yields always similar cluster sizes, because of its nature. Therefore, it might not be advisable to use it on datasets where a lot of differently sized clusters are suspected.
- Another problem occurs, when we are faced with a dataset, which has more dense clusters than we are looking for (K). Then we can not capture those dense clusters with this approach, because the centroids would be placed, in at least one case, exactly in the middle of the distance between two very dense clusters.

Despite a few drawbacks there are plenty of variations and implementations out there. We will not go into the different implementations, but we want to note that when the name k-means algorithm is used it usually refers to the standard implementation, which is called Lloyd's algorithm (here we look for local minima and therefore multiple iterations are recommended). This algorithm was roughly described before, when we discussed the best method for determining the centroids.

Concerning scRNAseq analysis the approach is often used with slight **variations or modifications**, two interesting ones are presented in the following.

- One idea is applying t-SNE on a dataset and then using k-means to cluster just the points of the low dimensional map. This comes of course with a tradeoff, because on the one hand we increase the speed immensely and k-means will yield good results, because it benefits from the structure of typical t-SNE maps. On the other hand, we loose a lot of information prior to clustering, assuming that we first use a PCA, then put the most informative components in t-SNE and then apply k-means clustering.
- Another modification is an approach called single cell consensus clustering (SC3) [KKS+17]. Here the authors calculate a lot of different results by high iteration numbers (e.g. 1.000) and varying parameters such as: switching between three metrics, namely Euclidean, Pearson and Spearman. Then they try to find a consensus of all of the results, which seems to be very robust concerning variations of parameters.
- Distribution-Based Clustering is another approach in the data partitioning realm. Here, we tackle the problem from a conceptual point of view and assume that the clustering can be done due to an underlying probabilistic mixture model. The clusters are then characterized by the parameters of the respective probabilistic model. By constructing the clusters this way it is easier to interpret the meaning of the individual clusters. A popular method is using gaussian mixture models, which assume that the datapoints of the individual clusters follow a normal distribution. Gaussian mixture models are an integral part of the MCLUST algorithm [FR99], which is applied in scRNAseq analysis, although it is hierarchical in nature (will be explained in Section 3.6.2).

Therefore partitioning-based clustering methods as k-means are a valid approach and a valuable asset in finding clusters within scRNAseq datasets.

3.6.2. Further Clustering Approaches

As mentioned before, this work is not about clustering methods but their application in scRNAseq analysis. Therefore, the analysis of clustering results is more interesting than the methods themselves. That's why we will give an overview of other clustering approaches used in scRNAseq analysis to give the reader a feeling for the vast amount of valid approaches and to set the scene for Chapter 4, where we try to tackle a major challenge in scRNAseq analysis.

Density-Based Clustering This approach can be seen as a close relative of the partition-based clustering approach, because the goal of partitioning the topological space, in which the data is embedded, is the same. Here, the partitioning is achieved according to the densities within the data, instead of distances between certain points. Therefore, the main difference lies in the strategy of how the partitioning is implemented. By using concepts of density, connectivity, and boundary, arbitrary shaped clusters can be discovered in contrast to partition-based clustering. Additionally, this approach is very robust in respect to outliers, which is not the case for partitioning-based approaches we presented previously. A often used representative of this approach in scRNAseq analysis is DBSCAN [EHPSX96].

Hierarchical-Based Clustering The goal of this approach is to derive an hierarchical tree, called dendrogram, whose leaves are the datapoints. Moving from one leaf up to the parent node we have a cluster of two datapoints. This process can be repeated and yields larger clusters in every step until we reach the root of the tree, which represents the whole dataset. There are two common approaches to generate such a dendrogram following either an agglomerative (bottom-up) or a divisive (top-down) strategy. In the agglomerative case we proceed as just described, by starting with singleton clusters (=clusters consisting of only one datapoint) and merge each with the most similar other singleton cluster. This process is repeated with the newly generated clusters until we have built a complete dendrogram. To reiterate, this was achieved by recursively merging the most similar clusters together until only one cluster is left. The divisive method starts on the opposite end of the dendrogram. Therefore, it starts with all the datapoints and divides them always in appropriate subclusters until a level is reached where all the leaves of the tree are singletons. If the computational burden is too high, a stopping criterion has to be applied, which specifies at which level of granularity to stop the process. This directly relates to the number K of clusters we are looking for.

In biological applications these approaches are very popular due to the fact that biology itself, or to be more precise cells, are hierarchical in nature (e.g. cell differentiation). The approach automatically generates different levels of granularity illustrated by the levels of the dendrogram. Additionally, hierarchical clusterings may indicate relationships of clusters to each other and it is easier to find subpopulations or explain effects. Another advantage of this approach is that it builds models based on any distance or similarity measure. Deciding on a proper stopping point, on the other hand, in the process of generating a dendrogram is a major challenge.

Especially for the analysis of scRNAseq data a hierarchical based algorithm, called pcaReduce [žY16], was developed. It is hierarchical in nature and, as the name suggests, applied on datasets, which dimensions were reduced by PCA.

Graph-Based Clustering Here, we see the data as a graph, where the datapoints represent vertices and their relation to each other (e.g. similarity) the weighted edges. The main idea is to gradually delete (cut) the edges with the smallest weights. Thereby, we automatically generate a partitioning. Of course the challenge is to decide which edges to cut and when to stop, because the amount of cuts should be kept to a minimum. The motivation to apply it in scRNAseq analysis is, that we hope to reveal connections and close neighborhoods, which can be interpreted as subpopulations. A rather novel approach developed for scRNAseq data, which shows good results and clusters most of the time by cell type, is SNN-Cliq [XS15]. This algorithm also captures some kind of granularity and thereby makes the search for subpopulation or inner dynamics possible and more transparent.

Combinations of the presented approaches or with other methods are becoming more popular, because disadvantages of one method can be compensated by advantages of another one. Three examples in scRNAseq analysis are as follows.

- The SC3 approach of using the k-means algorithm with a variety of different parameters and the combination of the dimensionality reduction method t-SNE with the clustering algorithm k-means, were already mentioned in Section 3.6.1.
- The authors of the SC3 [KKS+17] approach, tackled the problem of the computational burden, which comes from applying k-means multiple times (up to a 1000 times) on large datasets, by combining their approach with Support Vector Machines (SVM). When confronted with more than 5.000 datapoints (e.g. cells) they switch to a "hybrid" approach. Here, they randomly choose a small subset (e.g. 100 cells) of the dataset and perform SC3 on that. Afterwards they use the clustering result of that subset as a training set for a SVM. With that strategy they seem to achieve very similar results.
- Another approach is to presented by Tseng et al [TW05], called "Tight Clustering", where k-means clustering is applied as an intermediate clustering engine combined with early truncation of hierarchical clustering tree to overcome the local minimum problem of k-means. Additionally, the approach tries to identify the tightest and most stable clusters in a sequential manner by analyzing the tendencies of certain dimensions being grouped together under repeated resampling.
- Finally we wanted to mention CIDR (Clustering through Imputation and Dimensionality Reduction), a clustering package designed for scRNAseq data by Lin et al [LTH17]. This approach uses "implicit" imputation to account for the huge amount of dropouts present in scRNAseq data. Additionally, it performs Principal Coordinates Analysis (PCoA) for the dimensionality reduction, based on a dissimilarity matrix, and applies hierarchical clustering on the first few principal coordinates.

3.6.3. Comparison & Conclusion

We have presented a mathematical framework to describe any clustering approach, which could be applied on scRNAseq data. Then we have introduced a very popular strategy in clustering, namely partitioning-based algorithms. Here, we explained in detail one common algorithm, called k-means, with the help of the mathematical framework. Finally we presented in a qualitative manner other approaches and pointed out that new strategies often involve combinations of different methods.

A comparison between the presented approaches is neither sensible nor possible, because all of them are used in scRNAseq analysis on a regular basis and yield valid results. Depending on the dataset some algorithms work better than others, but until now there is no consensus within the community concerning the best clustering approach for scRNAseq data and it might be that there is not a best one. Additionally, it is not uncommon that different approaches yield very diverse results, which could mean that they captured different aspects within the data or simply did not work properly. A recent publication by Freytag et al [FLNB17] tried to solve this issue, even with data derived with the 10x technology, and called this situation actually a "Cluster Headache", without any more concrete results except what we already stated above. Others have already accepted this situation and tackled the problem by putting different clustering approaches together in programs, as for example the R-package clusterExperiment by Purdom et al [PRJ17]. Here, the goal is to generate a lot of clustering results by applying a variety of algorithms, each with different, algorithm specific, parameter variations.

This means we are facing the following situation. We have a lot of valid approaches, which yield biological sound results. Depending on the datasets at hand, some of them work very well and some do not even yield reasonable results. Every algorithm has their own set of parameters, which variations also have to be investigated, because the results can significantly differentiate from each other. Additionally, the amount of algorithms and combinations, which are applied in scRNAseq analysis, is increasing.

In conclusion we can see that the decision making process in cluster analysis is a big unsolved challenge in scRNAseq analysis with an immanent need to be addressed. The open question is: Which is the "best" or "correct" clustering result? This question implies the following issues and aspects:

- Which clustering result describes the current dataset best?
- Which clustering result should be chosen for further analysis or answers the initial scientific question?
- Which clustering result is the most robust, concerning parameter variations?
- How many clusters K can be expected by looking at the dataset? (this is needed as input for some clustering approaches)

The objective of the next chapter is to address this major challenge.

4. Cluster Validation in scRNAseq - Analysis

> "The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage."
>
> Algorithms for Clustering Data, Jain and Dubes [JD88]

The anecdotal quote in the beginning of this chapter nicely outlines the importance of this topic. It is true that cluster validation or the analysis of clustering results, is one of the most important steps in cluster analysis. Naturally, this translates to the analysis of clustering results in scRNAseq analysis. Without this part of the cluster analysis, it can not be determined, which result should be considered as the best and therefore investigated or used in further analyses. Differences in clustering results often indicate that the respective algorithms, or even parameter variations using the same algorithm, detect different dynamics within the data. Sometimes the approach is simply not right for the structure of the dataset at hand and therefore useless results are generated. Here, in scRNAseq analysis, it is not different and the inhomogeneity within the results of the clustering process could also simply indicate that one approach is more fit to the structure of the dataset than others. On the other hand they could reflect something more interesting, as for example the cell types within the data in different levels of granularity or they even pick up on different effects as batch or exposure to some compound. Therefore, we need to find a way to determine a result or a group of results, which we can use to answer the initial scientific question from the onset of the analysis or to perform further analyses on.

We need to mention, that the variety of results can, for the above stated reasons, be seen as an advantage. This may sound confusing, but we can use it to capture a number of different aspects within the data at the same time.

One major challenge in cluster analysis has always been the determination of the number of clusters K we are looking for. Or in other words, how many different groups should we expect to find within the data. This is not only interesting, but also often a prerequisite for a lot of clustering approaches to operate properly and highly influences the quality of the results. One popular example is the k-means algorithm, which was previously presented in Section 3.6.1. This conundrum motivated the development of so called "Clustering Indices" to evaluate, with or

without additional knowledge, which clustering result, and therefore which input parameters, yields the highest quality. Exactly such clustering indices are the main drivers of this chapter and how they can be used to solve the problem of finding the best clustering result among a huge amount of potentially valid ones.

4.1. The Starting Point

We basically start off where Chapter 3 ended, namely with the open question of finding the "best" or "correct" clustering result among an arbitrary number of potentially useful ones. The large amount of results origins from a lot of different clustering approaches and therefore a multitude of different algorithms, each with their own sets of different parameter variations, which can be successfully applied on scRNAseq data, as we have elaborated before. Therefore, we are faced with the challenge of determining which of the given results is the best. Maybe it would be even better, if we were able to derive a ranking according to quality and additionally find some consensus between the best results. We have already indicated that we propose to use already established quality measures in cluster analysis.

Before we can start with the definition and the subsequent application of these quality measures we have to agree on the terminology. We will extend the previously introduced mathematical framework, which was used to describe clusterings in Chapter 3.6, by the following terms.

Definition 61. We define the **number of clusters K** within a clustering CL implicitly by

$$CL = \{C_k | k = 1, \ldots, K\}\,.$$

The clustering CL can also be described by the **partition P** $\in \{1, \ldots, K\}^n$ of the data, where P_j describes the cluster affiliation of cell c_j.

Furthermore we want to describe the individual clusters in a more detailed way.

Definition 62. We can see a cluster C_k also as a **cluster submatrix** $M^{\{k\}}$ of the whole expression matrix M, which is reduced to only consisting of the cells (columns) c_j, which are affiliated with cluster C_k. With the help of the partition P we define it as

$$M^{\{k\}} = (m_{ij})_{i \in I, j \in \{j \in J | P_j = k\}}\,.$$

An alternative way to define the cluster submatrix is to use the previous definition of the **cluster indices** $J_k = \{j \in J | c_j \in C_k\}$ of the cells affiliated with cluster C_k and define the submatrix $M_{\{J_k\}}$ by the index set J_k

$$M_{\{J_k\}} = (m_{ij})_{i \in I, j \in J_k}\,.$$

Thereby, we get the relationship $M^{\{k\}} = M_{\{J_k\}}$, because $J_k = \{j \in J | c_j \in C_k\} = \{j \in J | P_j = k\}$.

The **number of cells within a cluster** C_k is denoted by

$$n_k = |\{c_j : P_j = k\}|$$

and therefore the dimensions of the cluster submatrix $M_{\{J_k\}}$ are $m \times n_k$ and we get $\sum_{k=1}^{K} n_k = n$.

Having now set the scene, we will introduce different kinds of quality measures for clustering results. After that, we will propose two ways of deciding on or determining a final clustering result. Finally, we apply this approach on the simulated dataset and discuss the results.

4.2. Quality Measures for Clustering Results

In general we distinguish between two different kinds of clustering indices: external and internal ones.

The external indices rely on some kind of externally provided knowledge, hence the name. In most cases this is a ground truth, a reference result or another kind of labeling, against which we want to compare. In scRNAseq analysis ground truths are very scarce. Nevertheless, we need external indices to compare clustering results with each other and thereby detect similarities between them and in the case of the simulated dataset to validate the proposed approach by comparing it to the configured true clustering. Therefore, we will present the Adjusted Rand Index (ARI) and the Normalized Mutual Information (NMI) in Section 4.2.1.

Internal clustering indices on the other hand only use metrics, which lie within the data. Therefore, we will only use internal indices for the decision making process, which we will present in Section 4.3, as we do not have any external knowledge. To do that we will briefly discuss four different indices and two information criteria in Section 4.2.2 and Section 4.2.3, respectively.

4.2.1. External Indices

We present two external indices, mainly to determine similarities of clustering results between each other, compare clustering results to published ones or to verify results in the very rare case of having a ground truth. External indices are also very useful for the investigation of relationships between clustering results derived from the same algorithm, but with different parameter combinations. Here, some approaches simply yield different levels of granularity in their results, where others do not resemble each other at all. The input for such indices are merely two clustering results or in other words the partitions P of the respective clusterings CL.

Adjusted Rand Index (ARI)

The first external index is the adjusted version of the Rand index (ARI) as described by Jain and Dubes in [JD88]. The regular Rand index describes the overlap of two clustering results with a value from 0 to 1, where 0 means no overlap at all and 1 indicates that the clusterings are the same. The adjusted version accounts for the fact that two datapoints could be clustered together due to chance. Therefore, this statistic is normalized with the expected values $ExpInd$. Thereby, the value 0 is yielded when the clusterings are selected by chance and 1 in the case of a perfect match. The adjusted Rand index, in contrast to the regular one, can return negative values in the case of the two clusterings being more different from each other than it would be expected, if they were determined by chance.

To describe the ARI we have to first define the term contingency table in this context.

Definition 63. Given two partitions P_A and P_B, with their respective index sets J_A and J_B and number of clusters K and L of a dataset, we define the contingency table and its respective row- and column-sums as

	J_{A1}	J_{A2}	$\dots$	J_{AK}	$sums$
J_{B1}	n_{11}	n_{12}	$\dots$	n_{1K}	a_1
J_{B2}	n_{21}	n_{22}	$\dots$	n_{2K}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
J_{BL}	n_{L1}	n_{L2}	$\dots$	n_{LK}	a_L
$sums$	b_1	b_2	$\dots$	b_K	

,

where $n_{ij} = |J_{Ai} \cap J_{Bj}|$.

Now, we can calculate the previously mentioned terms, namely the regular rand index, the maximum index and the expected index.

Definition 64. The **rand index** RI is defined as

$$RI = \sum_{ij} \binom{n_{ij}}{2}.$$

The **maximum index** MI, being the maximum value of the statistic, is calculated by

$$MI = \frac{1}{2}\left[\sum_{i} \binom{a_i}{2} + \sum_{j} \binom{b_j}{2}\right].$$

The **expected index** EI is determined by

$$EI = \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n_{ij}}{2}}.$$

Finally, we have everything in place to describe the adjusted version of the Rand index.

Definition 65. The **Adjusted Rand Index (ARI)** is obtained by

$$ARI = \frac{RI - EI}{MI - EI}.$$

Normalized Mutual Information (NMI)

The second external index we want to present, following Wagner et al [WW07], is the normalized mutual information (NMI), a measure from information theory. We have already used a term originated from information theory, namely the Shannon entropy, in a previous chapter about dimensionality reduction in Definition 49. We will use it here to describe the entropy associated with a clustering CL, measuring the uncertainty about the cluster affiliation of a randomly chosen cell. This is achieved by assuming that all cells have the same probability of being put in a particular cluster. Choosing one cell c_j at random, the probability that $c_j \in C_k\,(\in CL)$ is $P(k) = \frac{n_k}{n}$.

Definition 66. Therefore, we define **the associated entropy H with clustering** CL by

$$H(CL) = -\sum_{k=1}^{K} \mathrm{P}(k) \log_2 \mathrm{P}(k).$$

We now want to extend this concept to that of mutual information. Therefore, we want to determine how much this uncertainty can be reduced on average, when knowing about the cluster affiliations of randomly chosen cells in another clustering.

Definition 67. We define the **mutual information between two clusterings CL_A and CL_B**, with their respective number of clusters K and L, as

$$I(CL_A, CL_B) = \sum_{i=1}^{K} \sum_{j=1}^{L} \mathrm{P}(i,j) \log_2 \frac{\mathrm{P}(i,j)}{\mathrm{P}(i)\,\mathrm{P}(j)},$$

where $\mathrm{P}(i,j)$ is obtained by

$$\mathrm{P}(i,j) = \frac{|C_{Ai} \cap C_{Bj}|}{n}$$

and describes the probability of an element belonging to cluster C_{Ai} in CL_A and to cluster C_{Bj} in CL_B.

Due to the fact that this measure is not bound to a constant value it is rather difficult to interpret or compare. Therefore, normalization of the entropy values seem reasonable. One approach is to apply the arithmetic mean for the normalization.

Definition 68. Finally, we define the **normalized mutual information** (NMI) **between two clusterings** CL_A **and** CL_B by

$$\mathrm{NMI}\left(CL_A, CL_B\right) = \frac{2I\left(CL_A, CL_B\right)}{H\left(CL_A\right) + H\left(CL_B\right)},$$

with $0 \leq \mathrm{NMI}\left(CL_A, CL_B\right) \leq 1$ and $\mathrm{NMI}\left(CL_A, CL_B\right) = 1$ for $CL_A = CL_B$ and $\mathrm{NMI}\left(CL_A, CL_B\right) = 0$ for $\mathrm{P}\left(i,j\right) = 0$ or $\mathrm{P}\left(i,j\right) = \mathrm{P}\left(i\right) \cdot \mathrm{P}\left(j\right)$.

4.2.2. Internal Indices

This chapter concerns itself with the description, definition and occasional interpretation of four different internal clustering indices. As already stated before, internal clustering indices deliver a measure of quality concerning a clustering purely based on the data's internal information, hence the name. Therefore, the input of such indices is always the data and a certain clustering. Often the data will be used to derive further objects to analyze, for example a distance matrix. We chose the following indices to capture different aspects (e.g. density or partitioning) within the data and the clusterings, because results of certain clustering approaches fit to certain cluster indices. By using always more than one cluster index we tried to ensure that the clustering result is analyzed in a balanced way. Every index represents a function depending on clusterings, which can be seen either as a cost function or a score function and therefore has to be minimized or maximized, when looking for a better result.

The following definitions are adapted from [Des13] by Desgraupes, the author of the R-package clusterCrit, which can be used to perform cluster validation with the help of cluster indices.

The Silhouette Index

This measure is derived from the individual silhouette widths of every point, which try to capture how good a point lies within its affiliated cluster. To get a global measurement for a whole clustering, the silhouette index, all the silhouette widths have to be combined. That's why, we will start with deriving the silhouette width for one cell c_j. We start off by defining measures of the position of cell $c_j \in C_k$ with respect to the clustering CL.

Definition 69. The **within-cluster mean distance** of cell c_j to its own cluster is given by

$$a\left(j\right) = \frac{1}{n_k - 1} \sum_{i \in J_k \setminus j} d\left(c_j, c_i\right),$$

with $d\left(\cdot,\cdot\right)$being the chosen metric.

Similarly, we define the **mean distance to the other clusters** $C_l \in CL \setminus C_k$ as

$$\delta(c_j, C_l) = \frac{1}{n_l} \sum_{i \in J_l} d(c_j, c_i).$$

Lets determine the **minimum of the mean distances to the other clusters** by

$$b(j) = \min_{C_l \in CL \setminus C_k} \delta(c_j, C_l).$$

The cluster C_l, which yields the minimum $b(j)$, would be the best choice for relocating the cell c_j, if necessary.

With that, we can define the silhouette width.

Definition 70. The **silhouette width** of one cell c_j is calculated by

$$s(j) = \frac{b(j) - a(j)}{\max(a(j), b(j))}.$$

Its values range from -1 to 1, where -1 means that the cell c_j should be relocated to another cluster and 1 indicates a perfect fit.

Next we determine the mean silhouette of a cluster.

Definition 71. The **silhouette index for cluster** C_k is obtained by applying the arithmetic mean on the silhouette widths of the respective cells

$$s_k = \frac{1}{n_k} \sum_{j \in J_k} s(j).$$

Finally, we define the global silhouette index.

Definition 72. The global **Silhouette Index** SI of one clustering is defined by the arithmetic mean of the silhouette indices of its clusters

$$SI = \frac{1}{K} \sum_{k=1}^{K} s_k,$$

with values ranging again from -1 to 1, where -1 indicates a overall bad clustering and 1 a very good one. Therefore, we see ourselves confronted with a score function, which we want to maximize.

The Calinski-Harabasz Index

The idea of this index is to define a "variance ratio criterion" to get some insight into the structure, determined by the clustering, of the cells. This is achieved by

calculating the relationship between a measure of between-cluster and within-cluster variance.

Definition 73. The **Calinski-Harabasz Index** CHI is defined by

$$CHI = \frac{(n-K)}{(K-1)} \frac{BCSS}{WCSS},$$

where $WCSS$ is the previously in Definition 60 introduced within-cluster sum of squares of clustering CL and $BGSS$ denotes the **between-cluster sum of squares** defined by

$$BCSS\,(CL) = \sum_{k=1}^{K} n_k \left\| m_k - m \right\|^2,$$

with m_k being the centroid of cluster C_k and m denoting the centroid of all points given by $m = \frac{1}{n} \sum_{j=1}^{n} c_j$. Hereby, we capture the positions of the clusters in respect to all points by a weighted sum of squared distances between the centers of the clusters and the center of all cells.

As we can see this index represents a score function and therefore should be maximized while searching for better clusterings. There is no normalized range for the CHI values and therefore direct comparisons between different clusterings are necessary.

The Tau Index

Here, we formulate an analogue to the Kendall rank correlation coefficient (also known as Kendall's tau coefficient), which is a statistic used to determine the ordinal association between two vectors.

We need a few new metrics, before we can define the tau index for clusterings. We start off with counting the number of specific pairs within the data.

Definition 74. The **total number of pairs** of distinct cells is

$$N_T = \frac{n\,(n-1)}{2},$$

the **total number of distinct pairs within the clusters of a clustering** is

$$N_W = \sum_{k=1}^{K} \frac{n_k\,(n_k-1)}{2}$$

and **the number of pairs constituted of cells, which do not belong to the same cluster** is

$$N_B = \sum_{k<l} n_k n_l.$$

These measures relate to each other in the following sense

$$N_T = N_W + N_B.$$

Furthermore, we denote J_W as the index set of the N_W pairs within clusters and J_B as the index set of the N_B pairs between-clusters.

The last thing we need are two measures for the distances between those pairs.

Definition 75. We formulate the definition of the two numbers s^+ and s^- as

$$s^+ = \sum_{(r,s)\in J_B} \sum_{(u,v)\in J_W} \mathbf{1}_{d(c_u,c_v)<d(c_r,c_s)}$$
$$s^- = \sum_{(r,s)\in J_B} \sum_{(u,v)\in J_W} \mathbf{1}_{d(c_u,c_v)>d(c_r,c_s)},$$

where s^+ counts the number of times a distance between two cells belonging to the same cluster is strictly smaller than the distance between two points not belonging to the same cluster. The value of s^- on the other hand is the number of times the opposite situation occurs.

Now, we are putting it all together to get the desired measure.

Definition 76. The **Tau Index** is defined by

$$TI = \frac{s^+ - s^-}{\sqrt{N_B N_W \left(\frac{N_T(N_T-1)}{2}\right)}}.$$

This index can be classified as a score function and therefore the maximum would represent the best clustering result.

The C Index

In this case we focus on comparing distances between pairs of cells within the clusters and distances between all the pairs of cells within the data.

We can reuse a lot of the definitions presented in the previous section to define the C Index in the following.

Definition 77. The **C Index** is defined by

$$CI = \frac{S_W - S_{min}}{S_{max} - S_{min}},$$

where S_W is the sum of the N_W distances between all the pairs within each cluster, S_{min} and S_{max} are the sums of the N_W smallest and largest distances between all the pairs of cells in the whole dataset, respectively.

This index can be seen as a cost function and therefore the goal is to minimize its value in the search for better clusterings. The range of its values is also not fixed and therefore indices of different clusterings have to be compared directly.

During the investigation of cluster indices we noticed that the just described C Index and the previously presented Tau Index correlate highly (negative). We argue that this is due to the fact that they use and capture very similar aspects of a clustering. Nevertheless, we have to keep in mind that this can be the case with other indices as well. Therefore, it is important to account for it in the decision making process in which the presented indices play a key role. Otherwise, correlating indices will always influence the decision in a way that promotes clustering approaches, which fit the respective indices.

4.2.3. Information Criteria

Information criteria are a way to select the best statistical model, describing a dataset. In general they need a statistical model and the number of parameters used. With that, they try to balance the two aspects of perfectly fitting a model (and thereby increasing complexity) and the number of parameters, used in the model to describe the data. Due to the fact that it is possible to increase the likelihood of a model by increasing the number of parameters the presented criteria always penalize for that by adding a term dependent on the number of parameters. The information criteria are relative measures, therefore only meaningful when used to compare different models of the same data. A value of an information criteria for one model does not convey any information on the model's quality.

We will present two very popular information criteria, namely the Akaike (AIC) and the Bayesian Information Criterion (BIC). Both information criteria can be seen as cost functions and therefore the model with the lowest value is preferred. After describing these measures we will propose a way to apply them on our data and the large number of clustering results in a meaningful way.

Akaike Information Criterion (AIC)

Following the original publication of Akaike [Aka74], we will briefly describe the AIC.

Definition 78. The **Akaike Information Criterion** AIC of a statistical model M is determined by

$$\mathrm{AIC} = 2k - 2\ln\left(L_M\right),$$

where k denotes the number of estimated parameters in the model and L_M the maximum value of the likelihood function of the model M.

Bayesian Information Criterion (BIC)

Adapting the publication of Schwarz [Sch78], which originally proposed this criterion, we will define the BIC in the following.

Definition 79. The **Bayesian Information Criterion** BIC of a statistical model M is obtained by

$$\mathrm{BIC} = \ln(n)\, k - 2\ln(L_M),$$

where n denotes the number of datapoints, k the number of estimated parameters in the model and L_M the maximum value of the likelihood function of the model M.

Weighted Information Criteria Approach

To apply either of those criteria we try to use already derived and described techniques and results from the previous chapters. Before, we chose the most informative principal components to represent our data during the clustering process. That's why it makes sense to use them again in the analysis of the clustering results in the same way. The required models are determined with the help of linear regression, as described in Section 3.4.2, with the respective clustering result as categorical explanatory variable.

We propose the following approach to determine a comparable value for every clustering result CL with partition P.

1. Perform a PCA on the data, which was used in the clustering process, or use the already dimensionality reduced dataset $M_{redcued}$.
2. Take the L most important principal components $PC_1, \ldots, PC_L$, which were also used before in the process of confounding factor analysis, in Section 3.4.9, and clustering.
3. Build a linear regression model for every principal component as the dependent variable $y_l = PC_l$, for $l = 1, \ldots, L$, and the clustering result as categorical independent variable $x = P$.
4. Determine the respective information criterion for each model, yielding L information criterion values $IC_1, \ldots, IC_L$.
5. In the end we calculate a weighted sum IC_{sum} of those information criterion values, with the proportions of variance explained VE_l by the respective principal component PC_l as weights.

$$IC_{sum} = \sum_{l=1}^{L} VE_l \cdot IC_l$$

The result of this approach is one value per clustering result, which can be used just like a clustering index in the following decision making process. The next chapter

deals with that decision making process by using all of the above described clustering indices to determine the best clustering result.

4.3. Determination of the Final Clustering Result

We are now faced with having six different values, e.g. the internal quality measures we just described in Section 4.2, for every clustering result. The question is how to determine the "best" clustering result by using all six of these measures simultaneously and in an objective way? This situation can be framed as a multicriteria decision making (MCDM) problem, where the six different results from our quality measures per clustering result are seen as criteria.

In the following we propose a way to solve this MCDM problem and determine the "best" clustering result. Since we do not want to loose the potential knowledge within the other clustering results, we describe an approach to leverage the large amount of alternative clustering results and present an additional solution. In the end, we discuss how the two final clustering results can be interpreted and apply the proposed approaches on the simulated dataset.

4.3.1. The Favorite Approach

The goal of this section is to determine one final clustering result, which is qualitatively the best, among a large number of potentially valid ones. The quality is determined by six different measures we described previously in Section 4.2. Those measures are derived only from the data and the respective clustering result. Only one of those measures can be used in an independent way to determine the quality of clustering results, in contrast to the other five measures, which are only applicable by comparing different clustering results to each other. They are separated in two groups: cost- and score-functions, where higher quality is associated with a smaller or bigger value, respectively. In summary, we are faced with a multicriteria decision making (MCDM) problem, where the criteria are the quality measures and the alternatives between which we have to decide are the different clustering results.

As we have stated before, the issue of finding the best number of clusters K in a dataset is not new. This was one of the motivations for developing the above described clustering measures. Therefore, the problem we are facing now is also not totally new. In the past, the idea was to determine the best clustering from a lot of different results generated by the same algorithm, where different values for the number of clusters K were used as the main input parameter. We rephrase this problem to be more general: We have a large amount of clustering results generated by a lot of different algorithms with different parameter configurations and we want to use different quality measures to determine the best one. An approach to solve the initial problem of finding the correct number of clusters within a dataset was

proposed by Peng et al [PZKS12]. They used three different MCDM approaches with ten cluster validity measures on fifteen public-domain datasets. One of the conclusions was, that the most accurate results of the individual validity measures were equal to the worst performances of the three MCDM approaches. This indicates that the MCDM approach leverages the cumulative qualitative measurement of all of the indices better than either one of them individually. We decided to apply one of the three described MCDM approaches to our problem and therefore present in the following the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), adapted to our setting.

We start out by calculating a normalized decision matrix, which aggregates all the quality measures for every clustering result in a normalized and structured way.

Definition 80. We define the **normalized decision matrix** R by its entries r as

$$r_{pq} = \frac{x_{pq}}{\sqrt{\sum_{q=1}^{CR} x_{pq}^2}},$$

where $p = 1, \ldots, CM$, $q = 1, \ldots, CR$, CM and CR denote the number of clustering measures and the number of clustering results, respectively. The value of x_{pq} describes the result of the pth quality measure applied on the qth clustering result.

In other words, R is a matrix, which rows denote the different clustering measures and columns the clustering results. In our case we know that $CM = 6$, but we will keep the description abstract and continue with the general variables. The TOPSIS approach supports the option to declare different priorities for the clustering measures by weights. As we were not able to find any indication that one clustering measure is superior to others in determining the quality of clusterings on scRNAseq data, we decided to assign equal weights to all six clustering measures. Nevertheless, we give the definition of the weighted normalized decision matrix in the following.

Definition 81. We obtain the **weighted normalized decision matrix** V with its entries v by

$$v_{pq} = w_p r_{pq},$$

where the indices are defined as before and w_p denotes the weight of the pth quality measure. The weights have to fulfill the condition of summing up to 1, $\sum_{p=1}^{CM} w_p = 1$.

As just discussed, we set $w_p = \frac{1}{CM} = \frac{1}{6}$ for $p = 1, \ldots, CM$. Before we can continue, we have to distinguish between cost and score functions, with the help of two disjoint index sets. The C-index and the two information criteria (Akaike and Bayesian) were cost functions, with the minimal value indicating the highest clustering quality. The Silhouette-, Calinski-Harabasz- and Tau-index, on the other hand, were score functions, where the biggest value indicates the best clustering result. We will denote the index set of the score functions by CM' and the index set of the cost functions by CM''. This yields $CM = CM' \cup CM''$ and $CM' \cap CM'' = \{\emptyset\}$. We continue by determining the ideal solution and its counterpart the negative-ideal solution.

Definition 82. We can find the **ideal solution** S^+ by

$$S^+ = \left\{v_1^+, \ldots, v_{CM}^+\right\} = \left\{\left(\max_q v_{pq} | p \in CM'\right), \left(\min_q v_{pq} | p \in CM''\right)\right\}$$

and its counterpart the **negative-ideal solution** S^- by

$$S^- = \left\{v_1^-, \ldots, v_{CM}^-\right\} = \left\{\left(\min_q v_{pq} | p \in CM'\right), \left(\max_q v_{pq} | p \in CM''\right)\right\}.$$

With this we have the best and the worst possible solution, which are not necessarily present within the decision matrix, to calculate the relative distances of every clustering result to them.

Definition 83. With the help of the CM-dimensional Euclidean distance we determine the **separation measures**. The separation of each clustering result $q = 1, \ldots, CR$ to the ideal solution is denoted by D_q^+ and determined by

$$D_q^+ = \sqrt{\sum_{p=1}^{CM} \left(v_{pq} - v_p^+\right)^2}.$$

Similarly, the separation of each clustering result to the negative-ideal solution is denoted by D_q^- and determined by

$$D_q^- = \sqrt{\sum_{p=1}^{CM} \left(v_{pq} - v_p^-\right)^2}.$$

The last step is to determine the relative closeness of every clustering result to the ideal solution.

Definition 84. Therefore, we define the ratio R_q^+, which measures the **closeness to the ideal solution** in a relative way by

$$R_q^+ = \frac{D_q^-}{D_q^+ + D_q^-}.$$

To obtain the best or final clustering result we have to rank them in a descending order of the ratio value R_q^+. The first entry denotes the best and therefore final clustering according to our clustering indices. Due to the fact that we will propose an additional way to reach a final clustering we will call the clustering result, which is the result of this approach, the **favorite clustering**.

4.3.2. The Consensus Approach

We have already briefly mentioned that potential discoveries within the remaining clustering results, which were not chosen as the favorite through the procedure we described in the preceding chapter, are lost. Furthermore, Peng et al [PZKS12] mentioned in the conclusion of their publication on MCDM approaches in cluster analysis, that the second and third ranked clustering results should always be considered by the analyst, especially in the case of very close distances between their respective R^+ values.

Therefore, we propose an additional way of finding a final clustering by using a set of clustering results consisting of the top t_{CL} entries of the previously obtained ranked list. Inspiration on how to do that was given by Kiselev et al [KKS+17] authors of the SC3 approach, who try to find a consensus among a large amount of k-means clustering results with different parameter configurations, and Purdom et al [PRJ17], who developed the R-package clusterExperiment, where a function for combining given clustering results according to specific rules is described. The goal of this chapter is to combine the top t_{CL} clusterings, from now on called **topclusterings**, to obtain a robust and potentially more informative result. Thereby, we leverage the cumulative knowledge or potential discoveries of a lot of clustering results, which were derived from different sources (e.g. algorithms and parameter variations).

The following statements concern only the topclusterings. We start with defining a matrix for every clustering result, which describes the relation of individual cells to each other in respect to their cluster affiliation within the respective clustering.

Definition 85. The entries of the $n \times n$ **connectivity matrix** C_{CL} for a clustering CL is obtained by

$$m_{ij} = \begin{cases} 1 & P_i = P_j \\ 0 & P_i \neq P_j \end{cases},$$

where $i, j \in J$ denote the indices and P_i, P_j the cluster affiliation of the cells c_i, c_j, respectively.

In other words, the connectivity matrix value m_{ij} is 1 when cell c_i is in the same cluster as cell c_j and 0 when they are in different clusters. To aggregate these t_{CL} connectivity matrices (one for every topclustering) in a meaningful way, we apply the arithmetic mean.

Definition 86. We obtain the entries of the $n \times n$ the **consensus matrix** $Cons$ of the t_{CL} connectivity matrices by

$$(m_{ij})_{Cons} = \frac{1}{t_{CL}} \sum_{k=1}^{t_{CL}} (m_{ij})_k \,,$$

where $(m_{ij})_k$ denotes the entry $m_{i,j}$ of the connectivity matrix of the kth clustering. Naturally, $(m_{ij})_{Cons} \in [0, 1]$.

The entries m_{ij} of the matrix $Cons$ can be seen as the proportion of times that cell c_i was in the same cluster as cell c_j in respect of all topclusterings. In other words, this matrix can be described as a similarity matrix, determining the similarity of two cells c_i and c_j by the value m_{ij} following the rational of defining clusters as groups of points with similar features.

Having one matrix representing the relationship between all the individual cells concerning their cluster affiliations in all topclusterings, enables the use of hierarchical clustering on this matrix. The matrix $Cons$ is symmetric in nature, therefore it does not make any difference if the rows or columns are taken for the clustering process. Furthermore, a threshold proportion $0.5 < prop \leq 1$ has to be defined to establish a rule for putting two cells into the same cluster in the desired final clustering. The proportion threshold can be interpreted in the following way: if two cells c_i and c_j clustered into the same cluster more than $prop \cdot t_{CL}$ times, they belong together in the same cluster. Otherwise they should not be in the same cluster.

Following Purdom et al [PRJ17] to realize this strategy we apply divisive (top-down) hierarchical clustering with the matrix $Cons$ as input similarity matrix and the cut parameter (e.g. stopping criterion) $prop$. The clustering algorithm transverses down the tree until encountering a group (=potential cluster) of cells whose minimum similarity value is greater than or equal to $prop$. So, the stopping criterion for a potential cluster C_k with cells $c_j, j \in J_k$ can mathematically be described as $\min_{i,j \in J_k} \{m_{ij}\} > prop$. Attentive readers probably noticed that this procedure inheres the possibility (with higher $prop$ the probability increases) that some cells are not clustered in the course of this approach. These not-clustered cells can either be excluded from any further downstream analysis or put into an additional cluster C_{K+1}.

This process yields a final clustering, which we will call **combined clustering**.

The combined clustering leverages the additional cumulative knowledge of the next $t_{CL} - 1$ clustering results in the ranked list, which was derived in the previous section, after the favorite clustering. Furthermore, potential discoveries indicated by the additional results can be unveiled through this approach. Therefore, we were able to utilize not only the best, but the top t_{CL} clustering results concerning the presented quality measures, which was the goal of this section.

We want to make one last remark concerning this approach, relating it to graph theory. The consensus matrix $Cons$ could also be seen as an adjacency matrix. With that we could derive a weighted graph. The vertices would be the cells and the weights for the edges would be the entries of $Cons$. Applying the proportion parameter $prop$ as a condition for erasing every edge with a weight less than $prop$ and using a graph-based clustering approach would probably yield similar results, but this was not investigated in the course of this work.

4.3.3. Interpretation & Application

Having now two clustering solutions at hand we will point out some characteristics of them and how the above described approach was applied on the simulated dataset.

Interpretation

With two final results in the end, the problem of deciding which one to use for answering the scientific question from the beginning or the downstream analysis is apparent. We want to propose that both final clusterings should be used as a base for further analysis in the following way.

Lets start with the favorite clustering, here we simply chose the result with the best combination of quality measure values, determined by a MCDM approach called TOPSIS. As already mentioned the second and third best should also be investigated by the analyst, especially in the case of close R^+ values. Sometimes the favorite clustering yields the best objective quality measurements, but does not fit to the biological question at hand. Therefore, we recommend to keep an open mind and use the whole list and not only the number one.

Having in mind that the favorite clustering is a part of the combined clustering, it begs the question where their potential differences come from. First of all it depends heavily on the number t_{CL} of topclusterings used to determine the combined clustering. When more topclusterings are used to generate the combined clustering, then the influence of one single clustering result is naturally decreased.

In general the combined clustering is intended to give a result based on more than one clustering approach and the respective parameters. Therefore, it informs on the level of agreement among the topclusterings concerning the cluster affiliations of cells. When the majority of clusterings within the topclusterings agree on certain cluster affiliations it is reflected in the combined clustering result. In the course of generating the combined clustering and by looking how it was created a lot of information about potential subclusters within the data can be obtained. This could inform further downstream analysis steps or lead to modifications within the initial analysis. Biologically this is very interesting as subcluster could indicate subpopulations within cell types or effects within certain populations. Furthermore, the clustered matrix $Cons$ can be used to visualize very nicely how "good " the combined clustering harmonizes with the input clusterings, the topclusterings. This can be achieved by displaying it as a heatmap. This would be one way to detect hidden dynamics or potential subclusters. We can even derive a "level of satisfaction" for each cell with the help of the consensus matrix $Cons$ in the following way.

Definition 87. We define the **level of satisfaction** sat_j of cell c_j in the cluster C_k in the combined clustering $CL_{combined}$ as

$$sat_j = \frac{1}{n_k} \sum_{i \in J_k} m_{ij},$$

where m_{ij} denotes the values of the consensus matrix $Cons$. Naturally, $sat_j \in [0, 1]$, where a value close to 1 indicates a good fit of cell c_j within cluster C_k.

In the end we have some remarks concerning very fractured combined clustering results. If there are huge discrepancies between the topclusterings, then the combined clustering will not look very homogenous. The combined clustering will be very fractured and yield a lot of very small clusters. This is not a good result and indicates that the best clustering results picked up on very different aspects within the data. This can also be the case when the number of clusters within the topclusterings varies significantly or the proportion parameter *prop* is chosen close to 1.

Application & Results

The preceding chapters outlined nicely how this approach is applied. We want to mention some further details, which should be considered. Furthermore, we present the results of this approach applied on the simulated dataset.

Before any quality measures of clusterings are calculated the clustering results should be filtered by very basic means. We recommend to remove clusterings which are empty or have less than two clusters, have more than 20% not clustered cells, largest cluster consists of more than 90% of the total number of cells or found more clusters than twice the number of anticipated clusters K. The thresholds of the presented filter criteria should be adapted according to the data at hand.

Furthermore, we think it is reasonable to set a minimum size of clusters before the consensus approach is applied, therefore the problem of a very fractured combined clustering can be avoided or at least weakened. Again, it depends on the data, how the size of the smallest clusters is chosen.

In the following we will present the result of applying the described approach on the simulated dataset. The dataset was processed with the help of the mathematical methods we have discussed throughout this work. The favorite and combined clustering will be benchmarked against a "ground truth" clustering (=true clustering), which was used in the process of simulating the data to characterize the real clusters in the raw data. This clustering consists of six clusters of different sizes. We want to point out that in contrast to Peng et al [PZKS12] we did not only compare the number of clusters found, but the clusterings themselves. This is of course a much more rigorous and precise way to assess the accuracy of the results of the proposed approaches. The comparison is performed by the external clustering indices presented in Section 4.2.1, namely the ARI and NMI.

We have already presented PCA and t-SNE plots of the combined clustering result obtained by the consensus approach in Figure 3.7 on page 80 and Figure 3.8 on page 81. The respective plots of the favorite clustering are displayed in Figure 4.1 on page 108 and Figure 4.2 on page 109.

ground truth vs.	ARI	NMI
favorite	0.959	0.937
combined	0.998	0.99

Table 4.1.: The results of the favorite and the consensus (combined) approach are benchmarked against the true clustering of the simulated data by the external clustering indices ARI and NMI. These results describe nearly perfect agreement between the ground truth an the found clusterings (1 would mean that they are the same).

We can immediately see that the solutions do not differ significantly from each other, but that the combined clustering yielded more clusters than the favorite approach. Indeed, the combined clustering performed better than the favorite clustering. This can already be seen in Table 4.1 on page 107. That allows us to conclude that the majority of the topclusterings already indicated the correct clustering. To understand how the combined clustering was generated we can investigate Figure 4.3 on page 110, where we see the topclusterings in a more transparent way. The columns of this figure consist of cells and the rows are either clusterings (the first one, the combined clustering, dictates the order of the cells) or categorical metadata such as the true clustering (=Group) or the batch affiliation, as it is the case here. Every row is colored according to the categories (e.g. clusters) of the respective variable. We can immediately see that the topclusterings do not differ significantly from each other and therefore the combined clustering is very robust.

To investigate the result of the combined clustering further, we see in Figure 4.4 on page 111 the according consensus matrix. Again, as in Figure 4.3 on page 110, we notice that the consensus between the topclusterings is very high concerning the cluster affiliations of the individual cells. Therefore, we can see clearly distinguished clusters.

We have shown that the proposed approaches for cluster validation in scRNAseq analysis perform very well on a simulated dataset. Furthermore, we have displayed ways that aid in the analysis of clustering results and make automated decision making processes more transparent. Therefore, this represents a mathematical solid basis for further research in this direction with promising potential due to the very good results.

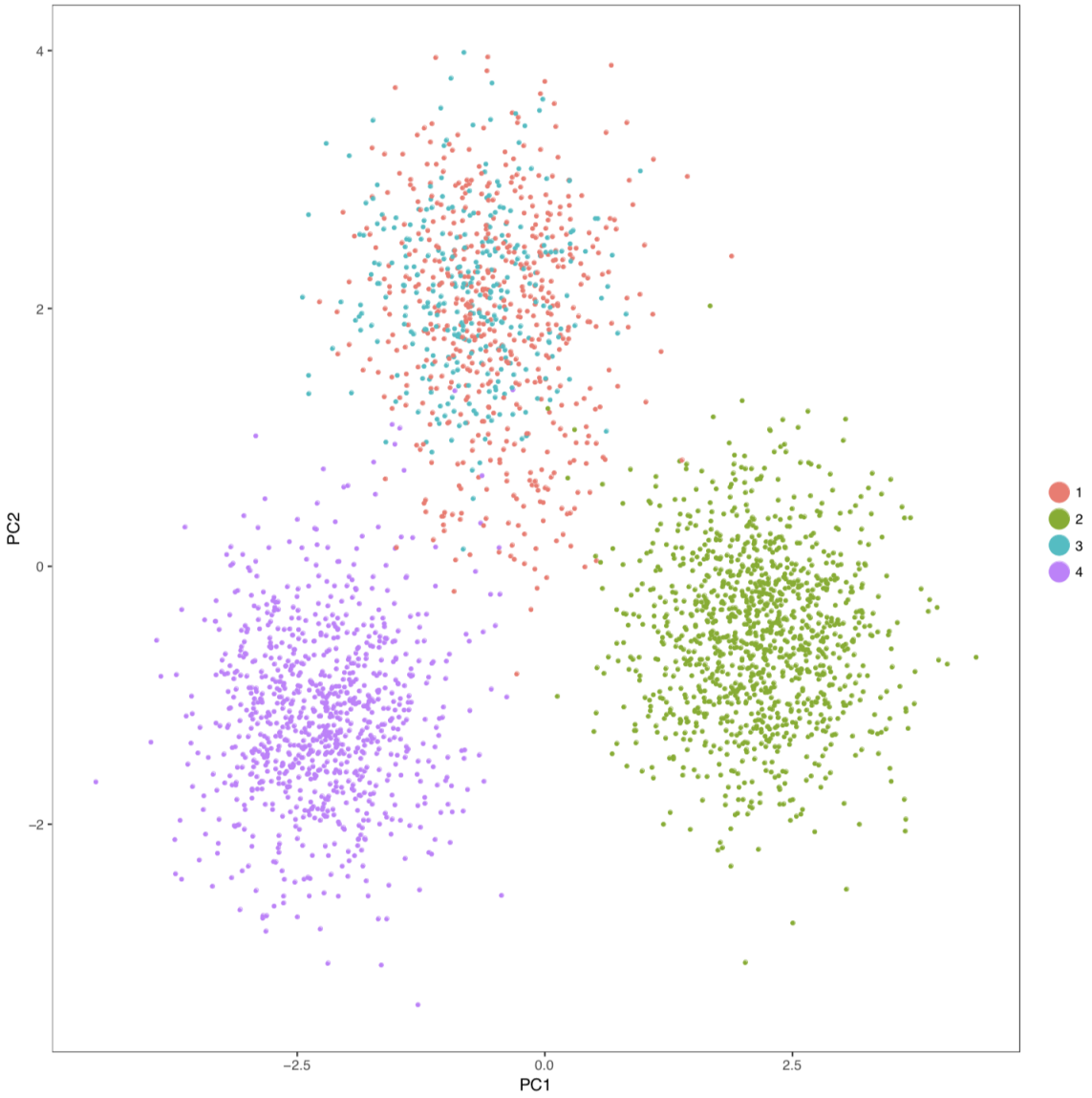

Figure 4.1.: PCA plot of the first and second principal component of the simulated dataset colored by the clustering result obtained by applying the favorite approach presented in Chapter 4.

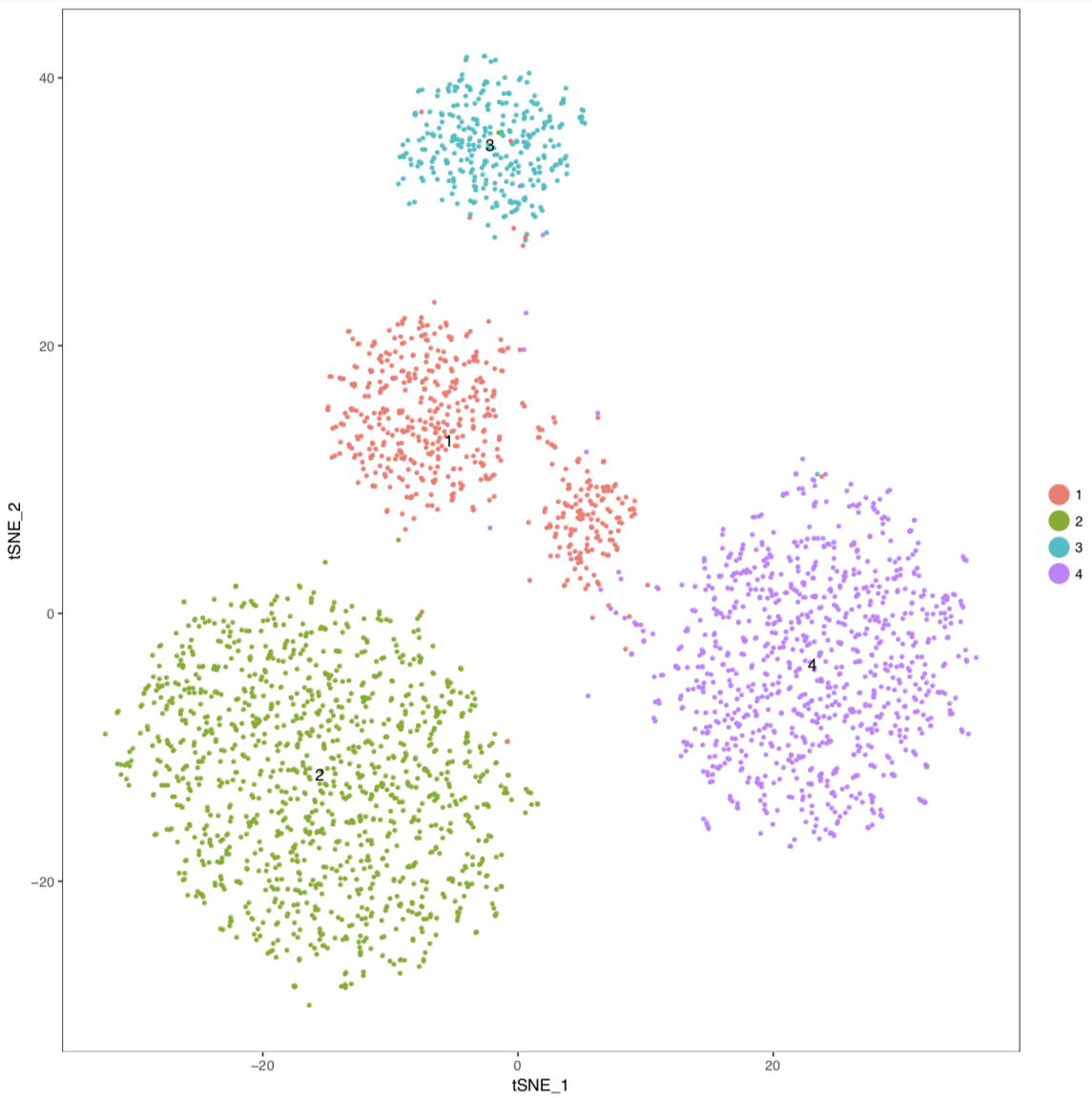

Figure 4.2.: Plot of a two dimensional t-SNE map of the simulated dataset colored by the clustering result obtained by applying the favorite approach presented in Chapter 4.

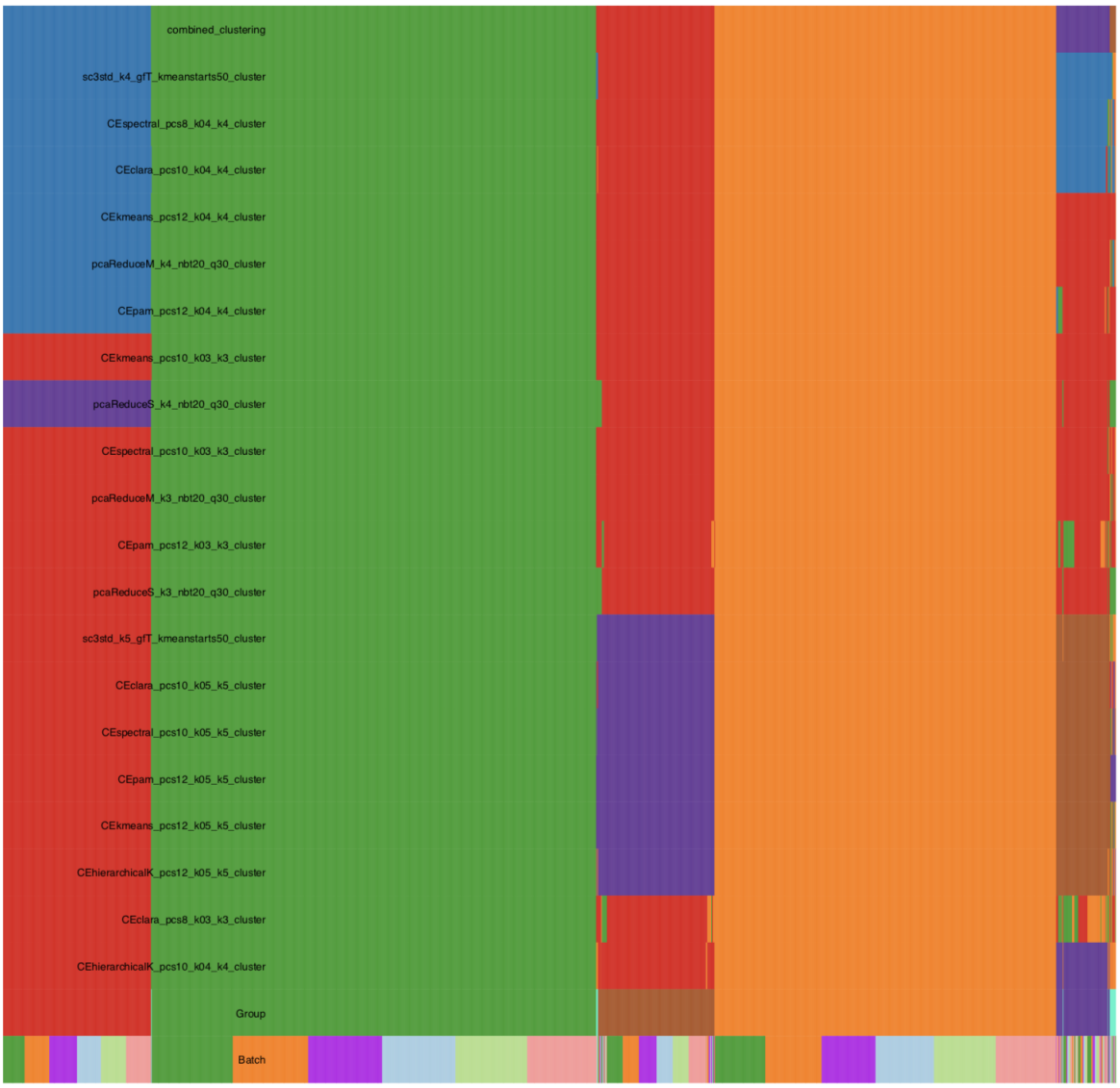

Figure 4.3.: In this plot the columns describe cells and the rows denote different categorical variables. The first row describes the combined clustering, followed by 20 topclusterings, the true clustering denoted by Group and the simulated batch affiliation at the bottom. Every row is colored by the category affiliation of the cells concerning the respective variable.

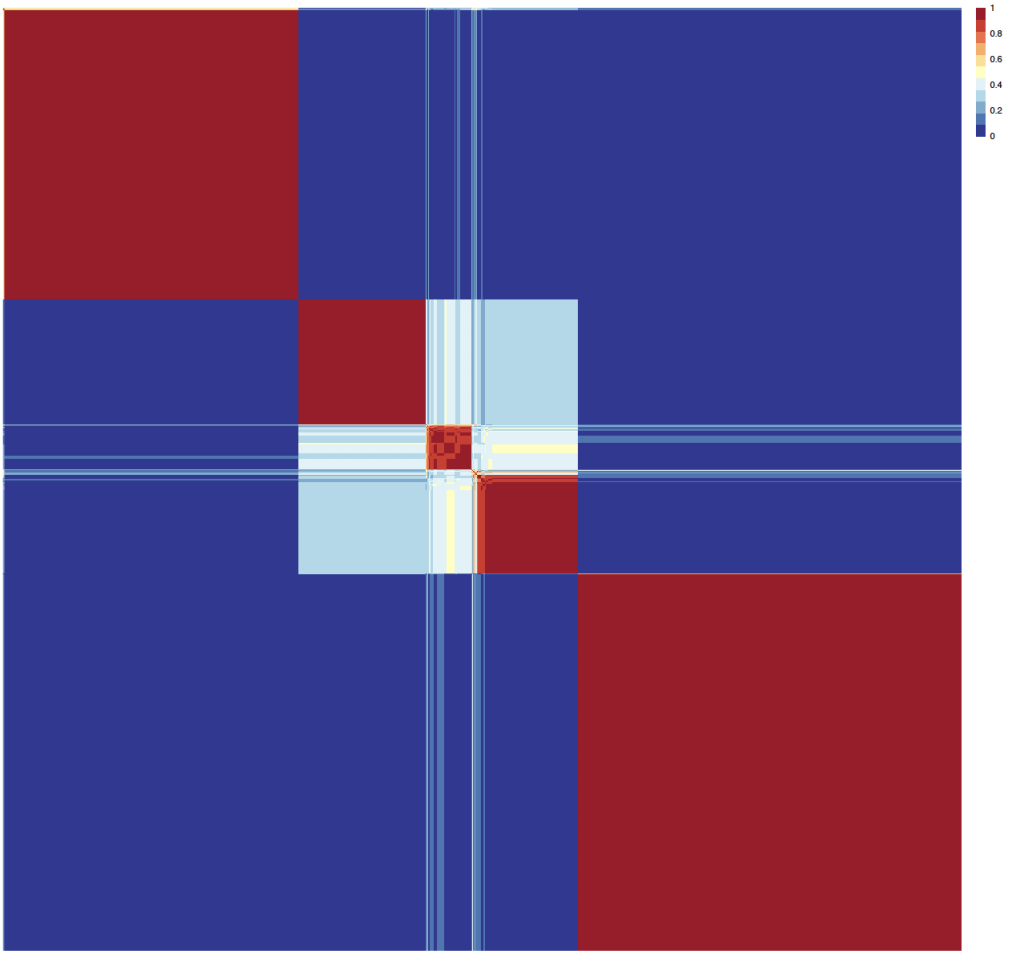

Figure 4.4.: Consensus matrix *Cons* of the combined clustering, where the rows and the columns denote the cells and the entries describe the proportion of times two cells are in the same cluster over all topclusterings.

5. Conclusion & Outlook

> "'There is only one way to learn,' the alchemist answered 'it is through action. Everything you need to know you have learned through your journey.'"
>
> Paulo Coelho, The Alchemist

5.1. Conclusion

We started out by an introduction into the general topic of next generation sequencing (NGS) and ended up with the role of mathematics within the realm of this technology. Then we took a closer look and elaborated on the technology of NGS and how it is applied. We dealt with the differences between classic DNA sequencing, bulk RNA sequencing and finally single cell RNA sequencing (scRNAseq). Here, we described the 10x Genomics technique for the capturing of single cells in detail and discussed challenges. Having understood how the data was generated in scRNAseq, we continued with describing the process of analyzing that data. This was done with the description of a general workflow, which heavily relies on mathematical methods.

We introduced a mathematical framework to discuss matters in scRNAseq analysis in a rigorous way. With that we investigated the problem of outliers, analyzed different normalization methods, described how to deal with explanatory variables, presented two methods for dimensionality reduction and discussed clustering, all in the context of scRNAseq analysis. Every single method was described in detail, using the proposed mathematical framework. The application of those methods and approaches was discussed and compared. At the end of every chapter we applied the recommended methods on a simulated dataset, which was derived from a real scRNAseq dataset, and discussed the meaning of the results.

Finally, we rose to the challenge of developing an approach on how to choose from many potentially valid clustering results, generated in the course of scRNAseq analysis, the best one. We proposed the usage of established quality measures and framed the problem as a multicriteria decision making (MCDM) problem, where the quality measures are seen as the criteria. Applying an already existing MCDM approach called Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) we were able to develop two approaches (favorite and consensus) for the determination of a final clustering result.

The favorite approach yielded the clustering result with the overall best quality measures, which was determined by TOPSIS. The consensus approach on the other hand tried to capture the additional information of the next best clustering results by combining them, according to specific rules, yielding an additional solution. We elaborated on the application of these approaches and the interpretation of its results. Ultimately, we applied the two approaches on the simulated dataset and obtained nearly perfect results by comparing them to the ground truth clustering.

5.2. Outlook & Further Work

The technology of single cell RNA sequencing is entering a new era with cheaper and faster (high throughput) sequencing machines than ever before. Therefore, the developments in this field will keep increasing and generate more data in less time.

Following the direction of this work, directly building on the determined final clustering results, we think it would be of interest to analyze the mathematical methods of downstream analysis methods such as differential gene expression analysis or pseudotime analysis in scRNAseq analysis. Apart from that, we noticed a need for proper visualization tools in scRNAseq analysis. All of these methods could be integrated in the proposed mathematical framework.

With the methods we have established in this work, the most biological knowledge would be used in the downstream analysis on the final clustering results. Therefore, we see a big potential in the life sciences to apply the knowledge of different disciplines directly in the analysis. This work was subject to the mathematical perspective and thereby kept a certain objectivity concerning the analysis of scRNAseq data. Methods, which could leverage biological knowledge in a meaningful way, could be developed in combination with rigorous mathematical approaches.

A more practical way for further development would be to build a best practice scRNAseq analysis workflow, based on the methods described in this work. This workflow could be implemented in a common programming language (e.g. R) to conduct analyses and further tests. For example more simulated datasets to test the cluster analysis approach and real publicly available datasets could be analyzed.

Furthermore, the cluster validation approach could be tested on other datasets.

In addition, standards in the realm of scRNAseq analysis could be developed not only concerning the methods and workflows but also the formats. This should aid in the comparison and understanding of results and the communication within the community. The human (single) cell atlas project [RTL+17] is one instance, which is at the forefront of such developments.

We think that we will see a lot of revolutionary developments in this field in the following years.

A. The Simulated Dataset

In this section we will discuss the origins of the simulated dataset, which we used throughout this work for the purpose of illustrating how the different presented methods are applied and in the end to test the proposed cluster validation approaches. It was important to us, that this work shows the close relationship between mathematical theories and real world applications. Therefore, we tried to point out that the theoretical argumentations of the mathematical chapters yield directly applicable methods, by presenting real results.

We chose a simulated dataset, because thereby we were able to configure a true clustering and batch affiliations (as a confounding factor). However, we wanted to keep it as close to a real dataset as possible and therefore used a public dataset from 10x Genomics to estimate the simulation parameters. This procedure enabled us to have full control over the intrinsic structure of the data, which we needed to validate our results and at the same time we kept a close relationship to a real dataset, which was generated by applying the described single cell capturing technology from 10x Genomics. This additionally underlined our argumentation for using the described recommendations in real world scenarios. Therefore, this work yields valid arguments for real world applications and not purely theoretical conclusions based on arbitrarily simulated data.

In the following we will present the dataset from 10x Genomics, which was used to model the simulated dataset, the origin and configurations of the simulated dataset and finally the R packages, which were used to process the dataset and generate most of the figures in this work.

3k PBMCs from a Healthy Donor

As stated in the beginning of this chapter we used a real dataset as basis for the estimation of simulation parameters to generate the simulated dataset. We chose the 3k PBMCs dataset from a healthy donor provided publicly by 10x Genomics[10x16a]. We will call it pbmc3k for the sake of convenience till the end of this section.

In the following we directly state some metrics from 10x Genomics concerning this dataset.

- Single Cell Gene Expression Dataset by Cell Ranger 1.1.0
- Peripheral blood mononuclear cells (PBMCs) from a healthy donor.

- PBMCs are primary cells with relatively small amounts of RNA (~1pg RNA/-cell).
- 2,700 cells detected, sequenced on Illumina NextSeq 500 with ~69,000 reads per cell 98bp read1 (transcript), 8bp I5 sample barcode, 14bp I7 GemCode barcode and 10bp read2 (UMI)
- Analysis run with --cells=3000
- Published on May 26, 2016

Before we move on to simulating the dataset based on the pbmc3k data, we wanted to elaborate on the decision making process, which resulted in choosing exactly this dataset. First of all it is provided by the inventors of the technique we described in the beginning of this work for capturing single cells. Furthermore, the number of cells (approximately 2700) is high enough to encounter the novel challenges, which come with this new kind of data. At the same time the number is low enough to work on the dataset in an iterative manner, without long computing times. This enabled us to test, refine and check the results more often. The biological origin of the sample was human PBMCs, which host a wide range of cell types, compared to samples from specific organs or tissues (e.g. liver), with differently sized cell populations. Coming from a human donor, the sample represents an easily relatable example for the use of scRNAseq analysis.

Simulating the dataset

The simulation of the dataset was done by the R package Splatter from Zappia et al [ZPO17]. It presents a standardized format for simulating scRNAseq datasets. The program itself is very sophisticated and highly customizable with the help of different parameters. Furthermore, it provides a function to estimate parameters, for the purpose of simulating similar datasets, based on a given dataset. We will not elaborate on the simulation process itself, only on how we applied this method to generate the simulated dataset.

The estimated parameters based on the pbmc3k dataset by splatter were as follows.

```
A Params object of class SplatParams
Parameters can be (estimable) or [not estimable], 'Default' or 'NOT DEFAULT'.
Global:
(GENES)  (CELLS)  [Seed]
32738     2967    362015

27 additional parameters

Batches:
[BATCHES]       [BATCH CELLS]       [Location]      [Scale]
    4       741, 742, 741, 743       0.1             0.1

Mean:
(RATE)              (SHAPE)
13.1862567561821    0.513625055001981
```

```
Library size:
(LOCATION)              (SCALE)
7.46950097836897   0.795782986717212

Exprs outliers:
(PROBABILITY)            (LOCATION)              (SCALE)
0.0192464082407156     5.13929002084815    0.994144591961186

Groups:
[GROUPS]          [GROUP PROBS]
    7    0.01, 0.25, 0.1, 0.05,...

Diff expr:
[Probability]     [Down Prob]   [Location]    [Scale]
     0.1              0.5          0.1          0.4

BCV:
(COMMON DISP)             (DOF)
0.291449770028786    28.9998030313448

Dropout:
[Present]             (MIDPOINT)                (SHAPE)
  FALSE      -0.0200665832600224      -1.02201072763585

Paths:
[From]     [Length]     [Skew]     [Non-linear]   [Sigma Factor]
  0          100          0.5          0.1              0.8
```

These parameters could have been used directly to simulate a dataset, instead we simplified and modified them slightly to fit our purpose.

We will point out the adapted and most important parameters, which play a critical role in the subsequent analysis of the dataset.

- Batches: We decided for the simulated dataset to consist of six different batches of equal size, because real datasets often consist of more than one batch. Thereby, we were able to demonstrate the usage of the methods from Section 3.4 on explanatory variables and illustrate the term "batch effect" described by a categorical variable. This information is known to the analyst in advance and therefore can be used in the course of an analysis.
- Groups: This variable describes the true clustering within the data. We decided to use very different cluster sizes, described by the group probabilities, to make it more realistic. Especially in the case of PBMCs it is possible to encounter very small cell populations. Furthermore, we decided on six clusters, because this number is close to the estimated one from the pbmc3k dataset, seven. The group affiliation is never known beforehand and therefore was only used in the end of this work for the assessment of the cluster validation results.
- BCV: The Biological Coefficient of Variation for each gene in each cell describes the underlying common dispersion across all genes. The dispersion was estimated at 0.291 for the pbmc3k dataset and we increased it to 0.5 to make the data even more disperse and thereby more difficult to distinguish between different clusters.
- Dropouts: We added dropouts, which are a major challenge in scRNAseq.

- We kept most of the other parameters as they were estimated or simplified them through rounding.

Finally, we used the following parameters to simulate the dataset.

```
A Params object of class SplatParams
Parameters can be (estimable) or [not estimable], 'Default' or 'NOT DEFAULT'.
Global:
(GENES)  (CELLS)  [Seed]
32000     3000    123456

27 additional parameters

Batches:
[BATCHES]        [BATCH CELLS]              [Location]      [Scale]
    6     500, 500, 500, 500, 500, 500        0.1             0.1

Mean:
(RATE)    (SHAPE)
10              0.1

Library size:
(LOCATION)   (SCALE)
   11          0.2

Exprs outliers:
(PROBABILITY)   (LOCATION)    (SCALE)
      0.02            5            1

Groups:
[GROUPS]          [GROUP PROBS]
    6    0.01, 0.4, 0.1, 0.05, 0.3, 0.14

Diff expr:
[Probability]     [Down Prob]   [Location]    [Scale]
      0.1             0.5           0.1          0.4

BCV:
(COMMON DISP)   (DOF)
      0.5         30

Dropout:
[Present] (MIDPOINT) (SHAPE)
  TRUE       0         -1

Paths:
[From]      [Length]      [Skew]     [Non-linear]   [Sigma Factor]
  0           100          0.5            0.1             0.8
```

In the end we manipulated the count values by dividing them by 10 to make the quality control metrics look more like the ones from the pbmc3k dataset.

R packages

Every figure in this work displaying some kind of scRNAseq data is an illustration of the simulated dataset or a modification of it. The computations and plots were done and generated with the help of the following R packages. In the following we quickly describe and categorize them according to their role in scRNAseq analysis. We will not include or mention their respective dependencies.

- scRNAseq (low-level) analysis and quality control packages
  - scater 1.6.2 [MCLW16] - "A collection of tools for doing various analyses of single-cell RNA-seq gene expression data, with a focus on quality control."
  - scran 1.6.6 [LMM16] - "Implements a variety of low-level analyses of single-cell RNA-seq data."
  - Seurat 2.2.0 [SBH+18]- "A toolkit for quality control, analysis, and exploration of single cell RNA sequencing data. "
- Regression model packages
  - nnet 7.3-12 [VR02] - "Software for feed-forward neural networks with a single hidden layer, and for multinomial log-linear models."
  - limma 3.34.9 [RPW+15] - "Data analysis, linear models and differential expression for microarray data."
- Clustering packages
  - clusterExperiment 1.4 [PRJ17] - "Provides functionality for running and comparing many different clusterings of single-cell sequencing data or other large mRNA Expression data sets."
  - SC3 1.7.7 [KKS+17] - "Consensus clustering of single-cell RNA-seq data."
  - pcaReduce 1.0 [žY16] - "Hierarchical clustering of single cell transcriptional profiles."
  - CIDR 0.1.5 [LTH17] - "Ultrafast and accurate clustering through imputation for single-cell RNA-seq data."
- Cluster validation packages
  - mclust 5.4 [SFMR16] - "Gaussian finite mixture models fitted via EM algorithm for model-based clustering, classification, and density estimation, including Bayesian regularization, dimension reduction for visualisation, and resampling-based inference." Here, used to calculate ARI measures.
  - NMI 2.0 [Wu16] - "Calculates the normalized mutual information (NMI) of two community structures in network analysis."
  - MCDM 1.2 [Ceb16] - "Implementation of several MCDM methos for crisp data for decision making problems."
- Plot packages
  - corrplot 0.84 [WS17] - "A graphical display of a correlation matrix or general matrix."
  - NMF 0.23.6 [GS10] - "Provides a framework to perform Non-negative Matrix Factorization (NMF)." Here, used for the consensus heatmap plot.

Bibliography

[10x16a] 10x Genomics. pbmc3k - Datasets - Single Cell Gene Expression - Official 10x Genomics Support, 2016.

[10x16b] 10x Genomics. Single Cell 3 ' Solution. 2016, https://www.10xgenomics.com/single-cell/.

[ABM] ABM Inc. NGS - Introduction | ABM Inc.

[AH10] S Anders and W Huber. Differential expression analysis for sequence count data. *Genome biology*, 11(R106): 1–12, 2010, http://genomebiology.com/2010/11/10/R106.

[Aka74] Hirotugu Akaike. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6): 716–723, 1974.

[Ans73] F. J. Anscombe. Graphs in Statistical Analysis. *The American Statistician*, 27(1): 17, feb 1973, http://www.jstor.org/stable/2682899?origin=crossref.

[BPHD09] James H Bullard, Elizabeth Purdom, Kasper D Hansen, and Sandrine Dudoit. Evaluation of Statistical Methods for Normalization and Differential Expression in mRNA-Seq Experiments. *U.C. Berkeley Div. Biostat. Pap. Ser.*, 11(1): 94, 2009, http://www.biomedcentral.com/1471-2105/11/94.

[Ceb16] Blanca A. Ceballos Martin. Package "MCDM", 2016.

[Cze12] Scott A Czepiel. Maximum Likelihood Estimation of Logistic Regression Models: Theory and Implementation. *Class Notes*, pages 1–23, 2012, papers3://publication/uuid/4E1E1B7E-9CAC-4570-8949-E96B51D9C91D.

[Des13] Bernard Desgraupes. Clustering Indices. *CRAN Package*, (April), 2013, https://cran.opencpu.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf cran.r-project.org/web/packages/clusterCrit.

[EHPSX96] Martin Ester, Kriegel Hans-Peter, Jörg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proceedings of the 2nd ACM SIGKDD*, pages 226–231, 1996.

[FLNB17] Saskia Freytag, Ingrid Lonnstedt, Milica Ng, and Melanie Bahlo. Cluster Headache: Comparing Clustering Tools for

10X Single Cell Sequencing Data. page 203752, oct 2017, https://www.biorxiv.org/content/early/2017/10/19/203752.

[FMW08] Peter Filzmoser, Ricardo Maronna, and Mark Werner. Outlier identification in high dimensions. *Computational Statistics and Data Analysis*, 52(3): 1694–1711, 2008, www.elsevier.com/locate/csda.

[FR99] C Fraley and A E Raftery. MCLUST : Software for Model-Based Cluster and Discriminant Analysis. (342): 1–15, 1999.

[Gri00] Karl Grill. Skriptum zur Vorlesung "Mathematical Statistics". 2000, https://institute.tuwien.ac.at/fileadmin/t/mathstoch/upload/ms0.pdf.

[GS10] Renaud Gaujoux and Cathal Seoighe. A flexible R package for nonnegative matrix factorization. *BMC Bioinformatics*, 11, 2010.

[Guj04] Damodar N. Gujarati. *Basic Econometrics*. 4 edition, 2004.

[Gur13] Werner Gurker. Angewandte Mathematische Statistik. 2013.

[IKK+16] Tomislav Ilicic, Jong Kyoung Kim, Aleksandra A Kolodziejczyk, Frederik Otzen Bagger, Davis James McCarthy, John C Marioni, and Sarah A Teichmann. Classification of low quality cells from single-cell RNA-seq data. *Genome Biology*, 17(1), 2016, https://genomebiology.biomedcentral.com/track/pdf/10.1186/s13059-016-0888-1?site=genomebiology.biomedcentral.com.

[Ill17] Illumina. An Introduction to Next-Generation Sequencing Technology. Technical Report 2, 2017.

[JD88] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., 1988.

[JT08] Michael Joswig and Thorsten Theobald. *Voronoi-Diagramme*, pages 83–102. Vieweg+Teubner, Wiesbaden, 2008.

[KKS+17] Vladimir Yu Kiselev, Kristina Kirschner, Michael T Schaub, Tallulah Andrews, Andrew Yiu, Tamir Chandra, Kedar N Natarajan, Wolf Reik, Mauricio Barahona, Anthony R Green, and Martin Hemberg. SC3: Consensus clustering of single-cell RNA-seq data. *Nature Methods*, 14(5): 483–486, 2017, https://www.nature.com/articles/nmeth.4236.pdf.

[KNT06] Jacob Kogan, Charles Nicholas, and Marc Teboulle. *Grouping multidimensional data: Recent advances in clustering*. 2006.

[Kul16] Jerzy K. Kulski. Next-Generation Sequencing - An Overview of the History, Tools, and "Omic" Applications. 2016, http://dx.doi.org/10.5772/61964 http://www.intechopen.com/books/colitis.

[LBM16] Aaron T. L. Lun, Karsten Bach, and John C. Marioni. Pooling across cells to normalize single-cell RNA sequencing data

with many zero counts. *Genome Biology*, 17(1): 75, 2016, http://genomebiology.biomedcentral.com/articles/10.1186/s13059-016-0947-7.

[LMM16] Aaron T.L. Lun, Davis J. McCarthy, and John C. Marioni. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. *F1000Research*, 5: 2122, 2016, https://f1000research.com/articles/5-2122/v2.

[LRS+09] Bo Li, Victor Ruotti, Ron M Stewart, James A Thomson, and Colin N Dewey. RNA-Seq gene expression estimation with read mapping uncertainty. *Bioinformatics*, 26(4): 493–500, feb 2009, http://www.ncbi.nlm.nih.gov/pubmed/20022975 http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2820677.

[LTH17] Peijie Lin, Michael Troup, and Joshua W.K. Ho. CIDR: Ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome Biology*, 18(1), 2017, https://genomebiology.biomedcentral.com/track/pdf/10.1186/s13059-017-1188-0?site=genomebiology.biomedcentral.com.

[McF73] Daniel McFadden. Conditional logit analysis of quzlaitative choice behavior. 1973.

[McF77] Daniel McFadden. Quantitative Methods for Analyzing Travel Behaviour of Individuals: Some Recent Developments. *Cowles Foundation Discussion Paper*, (474), 1977.

[MCLW16] Davis J McCarthy, Kieran R Campbell, Aaron T L Lun, and Quin F Wills. scater: pre-processing, quality control, normalisation and visualisation of single-cell RNA-seq data in R. *bioRxiv*, 3: 069633, 2016, http://biorxiv.org/lookup/doi/10.1101/069633.

[Nat15] National Human Genome Research Institute (NHGRI). All About The Human Genome Project (HGP) - National Human Genome Research Institute (NHGRI), 2015.

[Nat16] National Human Genome Research Institute (NHGRI). Newborn Screening Fact Sheet - National Human Genome Research Institute (NHGRI), 2016.

[Nat17] National Human Genome Research Institute (NHGRI). DNA Sequencing Costs: Data - National Human Genome Research Institute (NHGRI), 2017.

[Now17] Anna Nowogrodzki. How to build a human cell atlas. *Nature*, 547(7661): 24–26, jul 2017, http://www.nature.com/doifinder/10.1038/547024a.

[PRJ17] Elizabeth Purdom, Davide Risso, and Marla Johnson. clusterExperiment: Compare Clusterings for Single-Cell Sequencing, 2017.

[PZKS12] Yi Peng, Yong Zhang, Gang Kou, and Yong Shi. A Multicriteria Decision Making Approach for Estimating the Number of Clusters in a Data Set. *PLoS ONE*, 7(7), 2012.

[PZV+17] Swati Parekh, Christoph Ziegenhain, Beate Vieth, Wolfgang Enard, and Ines Hellmann. zUMIs: A fast and flexible pipeline to process RNA sequencing data with UMIs. *bioRxiv*, pages 2–7, 2017, https://www.biorxiv.org/content/biorxiv/early/2017/10/18/153940.full.pdf http://www.biorxiv.org/content/early/2017/06/22/153940?%3Fcollection=.

[RO10] Mark D Robinson and Alicia Oshlack. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biology*, 11(3): R25, 2010, http://genomebiology.biomedcentral.com/articles/10.1186/gb-2010-11-3-r25.

[RPW+15] Matthew E. Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W. Law, Wei Shi, and Gordon K. Smyth. limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic acids research*, 43(7): e47, 2015.

[RTL+17] Aviv Regev, Sarah A Teichmann, Eric S Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, Peter Campbell, Piero Carninci, Menna Clatworthy, Hans Clevers, Bart Deplancke, Ian Dunham, James Eberwine, Roland Eils, Wolfgang Enard, Andrew Farmer, Lars Fugger, Berthold Göttgens, Nir Hacohen, Muzlifah Haniffa, Martin Hemberg, Seung Kim, Paul Klenerman, Arnold Kriegstein, Ed Lein, Sten Linnarsson, Emma Lundberg, Joakim Lundeberg, Partha Majumder, John C Marioni, Miriam Merad, Musa Mhlanga, Martijn Nawijn, Mihai Netea, Garry Nolan, Dana Pe'er, Anthony Phillipakis, Chris P Ponting, Stephen Quake, Wolf Reik, Orit Rozenblatt-Rosen, Joshua Sanes, Rahul Satija, Ton N Schumacher, Alex Shalek, Ehud Shapiro, Padmanee Sharma, Jay W Shin, Oliver Stegle, Michael Stratton, Michael J T Stubbington, Fabian J Theis, Matthias Uhlen, Alexander van Oudenaarden, Allon Wagner, Fiona Watt, Jonathan Weissman, Barbara Wold, Ramnik Xavier, Nir Yosef, and Human Cell Atlas Meeting Participants. The Human Cell Atlas. *eLife*, 6: e27041, 2017, https://elifesciences.org/articles/27041.

[SBH+18] Rahul Satija, Andrew Butler, Paul Hoffman, Jeff Farrell, Shiwei Zheng, Christoph Hafemeister, and Patrick Roelli. Package "Seurat", 2018.

[Sch78] Gideon Schwarz. Estimating the Dimension of a Model. *Ann. Statist.*, 6(2): 461–464, 1978, https://projecteuclid.org:443/euclid.aos/1176344136.

[SFMR16] L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5: clustering, classification and density estimation using Gaussian finite

mixture models. *The R Journal*, 8(1): 205–233, 2016, https://cran.r-project.org/package=mclust.

[Shl03] Jon Shlens. A tutorial on principal component analysis: derivation, discussion and singular value decomposition. *Online Note httpwww snl salk edushlenspubnotespca pdf*, 2: 1–16, 2003, https://www.cs.princeton.edu/picasso/mats/PCA-Tutorial-Intuition_jp.pdf www.snl.salk.edu/ shlens/pca.pdf.

[STM15] Oliver Stegle, Sarah A. Teichmann, and John C. Marioni. Computational and analytical challenges in single-cell transcriptomics. *Nature Reviews Genetics*, 16(3): 133–145, 2015, http://dx.doi.org/10.1038/nrg3833.

[TW05] George C. Tseng and Wing H. Wong. Tight clustering: A resampling-based approach for identifying stable and tight patterns in data. *Biometrics*, 61(1), 2005.

[Van14] Laurens Van Der Maaten. Accelerating t-SNE using Tree-Based Algorithms. *Journal of Machine Learning Research*, 15: 1–21, 2014.

[VH08] Laurens Van Der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9: 2579–2605, 2008, http://jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf.

[VH12] Laurens Van Der Maaten and Geoffrey Hinton. Visualizing non-metric similarities in multiple maps. *Machine Learning*, 87(1): 33–55, 2012.

[VR02] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, 4 edition, 2002.

[WC53] J. D. Watson and F. H. C. Crick. Molecular structure of nucleic aids: A structure for deoxyribose nucleic acid. *Nature*, 1953.

[WS17] Taiyun Wei and Viliam Simko. R package "corrplot": Visualization of a Correlation Matrix, 2017.

[Wu16] Tianhao Wu. Package "NMI", 2016.

[WW07] Silke Wagner and Dorothea Wagner. Comparing Clusterings - An Overview. *Analysis*, 4769(001907): 1–19, 2007, https://publikationen.bibliothek.kit.edu/1000011477.

[XS15] Chen Xu and Zhengchang Su. Identification of cell types from single-cell transcriptomes using a novel clustering method. *Bioinformatics*, 31(12): 1974–1980, jun 2015, https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv088.

[ZPO17] Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: Simulation of single-cell RNA sequencing data. *Genome Biology*, 18(1), 2017, https://genomebiology.biomedcentral.com/track/pdf/10.1186/s13059-017-1305-0?site=genomebiology.biomedcentral.com.

[ZTB+17] Grace X.Y. Zheng, Jessica M. Terry, Phillip Belgrader, Paul Ryvkin, Zachary W. Bent, Ryan Wilson, Solongo B. Ziraldo, Tobias D. Wheeler, Geoff P. McDermott, Junjie Zhu, Mark T. Gregory, Joe Shuga, Luz Montesclaros, Jason G. Underwood, Donald A. Masquelier, Stefanie Y. Nishimura, Michael Schnall-Levin, Paul W. Wyatt, Christopher M. Hindson, Rajiv Bharadwaj, Alexander Wong, Kevin D. Ness, Lan W. Beppu, H. Joachim Deeg, Christopher McFarland, Keith R. Loeb, William J. Valente, Nolan G. Ericson, Emily A. Stevens, Jerald P. Radich, Tarjei S. Mikkelsen, Benjamin J. Hindson, and Jason H. Bielas. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8: 14049, jan 2017, http://www.nature.com/doifinder/10.1038/ncomms14049.

[žY16] Justina žurauskiene and Christopher Yau. pcaReduce: Hierarchical clustering of single cell transcriptional profiles. *BMC Bioinformatics*, 17(1), 2016.