An Integrated Bioinformatics Pipeline for Single Cell RNA-seq Analysis

Application in Natural Killer Cell Differentiation

Herman K. Netskar

Thesis submitted for the degree of Master in Informatics: Technical and Scientific Applications (Imaging and Biomedical Computing) 60 credits

Department of Informatics
Faculty of Mathematics and Natural Sciences

UNIVERSITY OF OSLO

Spring 2019
An Integrated Bioinformatics Pipeline for Single Cell RNA-seq Analysis

Application in Natural Killer Cell Differentiation

Herman K. Netskar
Abstract

Single cell RNA-sequencing is an increasingly popular tool for investigating the variability in gene expression between individual cells. Compared to the previously wide spread methods such as bulk RNA-sequencing, the single cell approach gives the advantage of a much higher cellular resolution, but it also provides us with much noisier data. In the recent years a large number of bioinformatics tools have been developed to analyze scRNA-seq data. There is an abundance of methods, for example more than 50 methods for trajectory inference have been developed since 2014 [1]. Many of the tools previously developed for bulk RNA-seq can also be applied to single cell data, but there are some crucial differences in the inherent characteristics of the data that differentiates scRNA-seq data from its bulk counterpart, among others in the statistical characteristics of the data [2].

In order to use the large amounts of data generated by scRNA-seq to produce new biological insights, we need to integrate the relevant tools into an integrated coherent framework. This thesis presents a pipeline that I developed, called SingleFlow, to perform large scale analysis in such an integrated framework. The pipeline’s usefulness was validated by applying it in the context of natural killer (NK) cell biology. There are a number of questions unanswered in the field of NK cell biology. By applying the pipeline to a unique scRNA-seq data set of NK cells from two different donors, we identified a temporal transcriptional map of human NK cell differentiation.

By mapping gene expression trends to pseudotime, we identified distinct transcriptional checkpoints that represent changes during NK cell differentiation. We also identified previously undescribed subsets within the CD56\textsuperscript{bright} subset of NK cells. The combination of the pipeline’s analysis and the potential of the novel data set proved useful in identifying important gene programs that are associated with NK cell differentiation. This knowledge holds potential to guide the development of new strategies for NK cell-based cancer immunotherapy.
# Contents

## 1 Introduction

1. Introduction ................................................. 3  
   1.1 Motivation ......................................... 3  
   1.2 Objectives ........................................ 3  
   1.3 About the thesis .................................... 4  

## 2 Background

2.1 Cell fate and differentiation .............................. 7  
2.2 RNA .................................................. 7  
   2.2.1 Splicing ......................................... 8  
2.3 NK cell biology ....................................... 8  
   2.3.1 NK cell differentiation and education .......... 8  
   2.3.2 Unknown factors in NK cell differentiation .... 9  
   2.3.3 Use of NK cells in cell therapies for cancer ... 10  
2.4 Single cell RNA sequencing .............................. 10  
   2.4.1 How scRNA-seq data is generated ............... 11  
   2.4.2 Challenges working with scRNA-seq data ...... 11  
   2.4.3 The need for bioinformatics .................... 12  

## 3 Data processing and statistical methods

3.1 Manifold model .......................................... 15  
3.2 Dimensionality reduction ............................... 16  
   3.2.1 PCA ........................................... 16  
   3.2.2 Diffusion maps ................................ 16  
   3.2.3 t-SNE .......................................... 16  
   3.2.4 UMAP .......................................... 17  
3.3 Artificial neural networks .............................. 17  
   3.3.1 Autoencoders .................................. 18  
3.4 Clustering ............................................ 18  
   3.4.1 Louvain modularity ............................ 18  
   3.4.2 Leiden ......................................... 19  
   3.4.3 K-means ....................................... 19  
3.5 Generalized additive models ............................ 19  

## 4 scRNA-seq bioinformatics tools

4.1 Data imputation ....................................... 21  
   4.1.1 MAGIC ...................................... 22
4.1.2 SAVER ........................................... 23
4.1.3 DCA ............................................ 23
4.2 Trajectory inference .................................. 24
4.2.1 Wanderlust ....................................... 24
4.2.2 Monocle 2 ........................................ 25
4.2.3 Slingshot ......................................... 25
4.2.4 PAGA ............................................. 25
4.2.5 Palantir ........................................... 26
4.3 RNA velocity ......................................... 26
4.4 Factor analysis ....................................... 27
4.4.1 f-scLVM .......................................... 28
4.5 Clustering ........................................... 28
4.5.1 Phenograph ....................................... 28
4.5.2 AP Clustering ..................................... 28
4.6 Differentially expressed genes ...................... 28
4.6.1 SCDE ............................................. 29
4.6.2 Bulk RNA-seq DEG methods .................... 29
4.7 Databases ........................................... 29
4.7.1 Gene Ontology ..................................... 29
4.7.2 Kyoto Encyclopedia of Genes and Genomes ... 29
4.8 Gene set enrichment analysis ....................... 30
4.8.1 GO enrichment analysis .......................... 30
4.8.2 KEGG enrichment ................................. 30
4.9 Correcting for batch effects ......................... 30
4.10 Deconvolution ...................................... 31
4.11 Scanpy ............................................. 31
4.12 Seurat ............................................. 31

II Methods 33

5 Technologies 35
5.1 Programming languages ............................ 35
5.1.1 Dependencies ..................................... 35
5.2 Nextflow ............................................ 36

6 Bioinformatics methods 37
6.1 Filtering, feature selection, normalization and transformation 37
6.2 Dimensionality reduction methods .................. 38
6.3 Gene expression imputation ........................ 38
6.4 Differentiation trajectories ........................ 38
6.5 Gene trends ......................................... 39
6.5.1 Clustering of gene trends ....................... 40
6.6 Single cell clusters .................................. 40
6.7 Gene set enrichment analysis ....................... 40
6.8 Factor analysis ..................................... 41
6.8.1 Cell cycle correction .............................. 41
6.9 Differentially expressed genes ..................... 41
6.10 Deconvolution ........................................ 41
6.11 Velocyto .............................................. 42
  6.11.1 Identify most important genes ...................... 42

7 Data sets ........................................... 43
  7.1 scRNA-seq NK cell data from Oslo University Hospital . . 43
    7.1.1 Cell Ranger ........................................ 44

8 Validating and assessing the results .................. 45
  8.1 Reproducibility ....................................... 45
  8.2 Collaboration with the Malmberg Lab ................. 46
  8.3 Validating the deconvolution approach ............... 46

III Results ........................................... 47

9 SingleFlow: an improved modularized scRNA-seq pipeline 49
  9.1 The pipeline .......................................... 49
  9.2 Graphical user interface applications .................. 52
    9.2.1 Customized clusters .............................. 52
    9.2.2 Inspecting cells contributing the most to gene trend
         phenomenon ........................................ 52
    9.2.3 Custom range gene trends ........................ 53
    9.2.4 Gene trend clusters for GO enrichment analysis ... 53
  9.3 A comparison of SingleFlow to other scRNA-seq pipeline tools 57
  9.4 User guide ............................................ 59

10 Validation .......................................... 61
  10.1 Reproduce results across donors ...................... 61
  10.2 Reproducing results from NK cell differentiation literature 65
  10.3 Recover the cell type composition in RNA-seq data using
       deconvolution ....................................... 67

11 User scenarios: applications in single cell NK cell biology 71
  11.1 NK cell differentiation defined through single cell RNA-seq 71
  11.2 Continuous and coordinated transcriptional changes in
       pseudotime ........................................ 76
    11.2.1 Gene regulatory programs and RNA velocity ........ 76
    11.2.2 Custom clusters for analysis ..................... 76
  11.3 Transcriptional checkpoints and gene-expression trends during
       NK cell differentiation ............................. 78
    11.3.1 Identifying transcriptional checkpoints ............ 78
    11.3.2 Combine pseudotime and clustering ................ 78
  11.4 Diversified gene-expression patterns during terminal NK
       cell differentiation .................................. 80
### IV Discussion and conclusions

12 What was achieved

13 Limitations of scRNA-seq and the methods integrated in Single-Flow
   13.1 Statistical assumptions and characteristics of scRNA-seq UMI data 89
   13.2 Imputation 90
   13.3 Single modality 90
   13.4 The application of SingleFlow to NK cell differentiation 91

14 Proposal for future work
   14.1 New domains 93
   14.2 Reproducible analysis 93
      14.2.1 Docker image 94
   14.3 Other databases of gene sets for enrichment 94
   14.4 Additional clustering methods 94
   14.5 Other trajectory inference tools and complementary implementation 95
   14.6 New preprocessing methods 95
   14.7 Multimodal analysis 96
   14.8 New imputation methods 96
   14.9 Deep learning models for scRNA-seq data analysis 97
      14.9.1 Challenges with deep learning models in genomics 98
   14.10 Publish paper in collaboration with the Malmberg Lab 99
      14.10.1 More sequencing data 99

15 Final remarks

---

vi
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>The subset sorting done prior to scRNA-seq</td>
<td>43</td>
</tr>
<tr>
<td>7.2</td>
<td>Pipeline to generate count data using Cell Ranger</td>
<td>44</td>
</tr>
<tr>
<td>9.1</td>
<td>Execution time of a typical SingleFlow analysis</td>
<td>50</td>
</tr>
<tr>
<td>9.2</td>
<td>Outline of the processes in SingleFlow</td>
<td>51</td>
</tr>
<tr>
<td>9.3</td>
<td>GUI for selecting custom clusters for further downstream analysis.</td>
<td>53</td>
</tr>
<tr>
<td>9.4</td>
<td>GUI for choosing the gene trend to study and map gene trend phenomenons to the embedded cellular space</td>
<td>54</td>
</tr>
<tr>
<td>9.5</td>
<td>GUI for inspecting the cells that has been assigned a given pseudotime</td>
<td>55</td>
</tr>
<tr>
<td>9.6</td>
<td>GUI for inspecting local gene trends</td>
<td>56</td>
</tr>
<tr>
<td>10.1</td>
<td>Phenotypically sorted subsets for donor 1</td>
<td>62</td>
</tr>
<tr>
<td>10.2</td>
<td>Cell clusters determined by Phenograph for donor 1</td>
<td>62</td>
</tr>
<tr>
<td>10.3</td>
<td>Phenotypically sorted subsets for donor 2</td>
<td>62</td>
</tr>
<tr>
<td>10.4</td>
<td>Cell clusters determined by Phenograph for donor 2</td>
<td>63</td>
</tr>
<tr>
<td>10.5</td>
<td>The global gene trends for donor 1</td>
<td>63</td>
</tr>
<tr>
<td>10.6</td>
<td>GO enrichment analysis of the gene sets associated with each gene trend</td>
<td>64</td>
</tr>
<tr>
<td>10.7</td>
<td>Volcano plot showing DEGs between conventional and adaptive subsets in donor 1</td>
<td>66</td>
</tr>
<tr>
<td>10.8</td>
<td>tSNE lot incorporating the factor analysis for lysosomal biogenesis</td>
<td>67</td>
</tr>
<tr>
<td>10.9</td>
<td>The constructed signature matrix</td>
<td>69</td>
</tr>
<tr>
<td>11.1</td>
<td>Volcano plots showing differentially expressed genes between adjacent clusters identified by Phenograph</td>
<td>73</td>
</tr>
<tr>
<td>11.2</td>
<td>Gene expression after MAGIC imputation for genes relevant for studying NK cell biology</td>
<td>74</td>
</tr>
<tr>
<td>11.3</td>
<td>The composition of the different clusters</td>
<td>75</td>
</tr>
<tr>
<td>11.4</td>
<td>RNA velocity embedded in the t-SNE plot</td>
<td>76</td>
</tr>
<tr>
<td>11.5</td>
<td>MAGIC imputed gene expression of transcription factors important in NK cell differentiation</td>
<td>77</td>
</tr>
<tr>
<td>11.6</td>
<td>Clusters before and after the bridge region for donor 1</td>
<td>78</td>
</tr>
<tr>
<td>11.7</td>
<td>Pseudotime plotted onto the t-SNE embedding</td>
<td>79</td>
</tr>
<tr>
<td>11.8</td>
<td>Boxplot combining the pseudotime computation with the phenograph clustering for donor 1</td>
<td>80</td>
</tr>
</tbody>
</table>
11.9 The dim gene trends for donor 1 ........................................... 82
11.10 GO enrichment analysis of the gene sets associated with the
dim gene trends ................................................................. 83
# List of Tables

7.1 Number of cells belonging to the various sorted subsets for donor 1 ................................. 44
7.2 Number of cells belonging to the various sorted subsets for donor 2 ................................. 44
9.1 Comparison between SingleFlow and Scanpy ......................................................... 58
10.1 Ground truth composition of Phenograph clusters .............................................. 68
10.2 Results of running CIBERSORT on donor 2 derived artificial bulk RNA-seq data set ................................. 68
Preface

Before I started working on this project I had no experience working in the field of single cell RNA-sequencing. My background was mostly in informatics generally, with some statistics courses and a couple of years of studying medicine as a good complement. I had also taken a course in bioinformatics, and found the idea of combining the study of biological and medical phenomenons with the quantitative nature of computer science, appealing.

I learned a lot from working on this project. Both about the field of scRNA-seq and NK cell biology specifically, but also about the scientific process generally. Especially working towards writing a paper was a very rewarding experience. The interdisciplinary nature of bioinformatics made it even more fulfilling.

First and foremost I would like to thank my main supervisor Trevor Clancy for his help and guidance throughout my work on this project. I would also like to thank my co-supervisor Eivind Hovig. Working in the interdisciplinary field that bioinformatics is, the project called for collaborations. Throughout this project I have collaborated with biologists from the Malmberg Lab at the Oslo University Hospital. I would specifically like to thank Professor Karl-Johan Malmberg and Aline Pfefferle for the collaboration and the valuable biological insight they provided.

Herman K. Netskar
Oslo, May 2019
Part I

Introduction
Chapter 1

Introduction

1.1 Motivation

Natural killer (NK) cells are central cells of our innate immune system that can lyse tumor cells. NK-cell-based immunotherapeutic strategies have recently been developed to target human cancers [3]. The successful application of this type of therapy requires an in depth understanding of NK cells, their biology and their development. The regulatory gene programs that define NK cell states and their differentiation dynamics are not fully understood and an improved description of the programs that control clinically beneficial NK cell subtypes would be of great value for future immunotherapeutic strategies.

Recent advances in scRNA-seq analysis has provided new insights in a variety of fields within molecular biology [4][5][6]. scRNA-seq can potentially revolutionize the way we characterize immune cells and their dynamics [7]. Utilizing new sequencing technologies and newly developed analysis tools for scRNA-seq data could potentially dramatically improve our understanding of NK cells, its regulatory mechanisms and differentiation in particular.

There is an abundance of tools available for scRNA-seq data analysis. Standalone tools and integrated toolkits already exist [8], but to gain advanced biological insights the field needs a comprehensive and reproducible analysis of single cell data. It would be useful to develop a framework and a pipeline to perform such an analysis. With the rapid development in this field in mind, such a pipeline must also be developed in a way so that new tools easily can be included in the framework in order to complement and extend the analyses in the future.

The goal of using NK cells in immunotherapy and the potential of scRNA-seq data analysis to retrieve the necessary biological insight to move towards this goal, was the main motivation behind this project.

1.2 Objectives

The main objective of this project was to develop an integrated and modular pipeline for studying scRNA-seq data by integrating existing tools
as well as to develop new components where I found it necessary. Specifically I wanted to achieve the following:

- An improved modularization of scRNA-seq workflows: One of the main aims of this project was to establish a bioinformatics framework that uses existing tools and integrate them so that different tools easily can be switched out for newly developed tools (modularity). I wanted the pipeline to be able to save the intermediate states, so that analysis can be run with different tools with the same input, as well as to facilitate automation and reproducibility of advanced scRNA-seq analysis.

- Apply the framework to data sets of NK cells from the Malmberg Lab, Oslo University Hospital, and verify that the analysis provided by the pipeline offered biological insights in collaboration with NK cell biologists. By applying the developed pipeline to a novel NK cell data set, we wanted to try to answer some of the unanswered questions about NK cell differentiation and NK cell subsets outlined in the background chapter on NK cell biology.

1.3 About the thesis

This thesis consists of four main parts. Part I is an introduction to the field of single cell RNA-sequencing and some of the biology underpinning this field. In this part I discuss a little about cell fate and differentiation and why we want to study this, specifically I talk about NK cells and their biology. In this part I also describe existing tools for processing scRNA-seq data and bioinformatics tools for analyzing this data. Part II discusses the methods used as part of the project. It introduces the data sets that the pipeline was applied to and describes how these data sets were generated. It describes which tools I used and how these were integrated into a coherent framework for scRNA-seq data analysis. I also present justification for choosing these specific tools.

Part III consists of the results from the project. SingleFlow, the pipeline that I developed, is the main outcome of this project. However the project would not have been complete without the ability to gain some biological insights from the final framework. I therefore present a set of user scenarios in the field of NK cell differentiation and describe the corresponding biological interpretation of the results that SingleFlow produced. The final part, part IV, discusses these results and puts them into perspective. It looks at limitations of the analysis framework and looks at possible future work. I also present some ideas of future applications of NK cells in immunotherapeutic strategies and discuss why the analysis presented here is useful in a broader perspective.

This project was carried out in collaboration with the Malmberg Lab at the Oslo University Hospital. This research group studies the molecular and cellular basis of NK cell dynamics. By collaborating with biologists who work in the field of NK cell biology I was able to assess the
tools I applied and gain insight into the biological questions that would be relevant to try to answer using the scRNA-seq data sets and the subsequent analysis. I have personally carried out the development of the SingleFlow framework and performed the analysis. Throughout this thesis I will use the term ‘I’ where I refer to work on SingleFlow and I will use ‘we’ when discussing its application to NK cell biology and the corresponding analysis and interpretation. The development of SingleFlow was in large part motivated from discussions with the Malmberg Lab and the subsequent collaboration. This collaboration helped me put the bioinformatics analysis into an appropriate context. As a result of this we have written a manuscript which is already submitted for publication.
Chapter 2

Background

2.1 Cell fate and differentiation

Immune cells develop over time by interactions with antigens and other cells, facilitated by controlled modifications in gene expression in the cells as they develop [9]. Certain subsets of genes, known as gene regulatory programs, are important in regulating this dynamic process known as cell differentiation. In this process cells develop from one cell type to another, often more restricted, cell type [10]. One goal in biology is to understand how cells develop, how they differentiate and how they end up in their final state. This is known as the cell’s fate, and biologists want to understand the factors that determine it and which regulatory gene programs that are relevant.

Differentiation has in the past been understood as a series of discrete cell states, where there exist marker genes that are mutually exclusive between cells and therefore provides us with a clear classification of a given cell into one subset or another [11]. This has also been the main assumption in the mathematical and statistical models that have underpinned the study of cell fate [12]. However, recent developments [12] [13] have indicated that cell states make up a continuum and that the assumption of discrete states therefore is flawed. New sequencing technologies, and statistical and bioinformatics methods that will be discussed later, has been important in this development.

2.2 RNA

RNA molecules are transcribed from a gene’s DNA template and some of the produced RNA molecules serve as templates for protein synthesis [14]. The RNA molecules that provide genetic information for protein synthesis are known as messenger RNA (mRNA). The collection of all (protein-coding) mRNA in a cell is known as the cell’s transcriptome. Whether a gene is actively being transcribed and at what level this transcription occurs tells us something about the cell’s state and each cell can in principal be placed in a number-of-genes dimensional space where each feature corresponds to a gene and the value for that feature is the level of
transcription for that gene. The value for an individual gene in a given cell can for example be the number of RNA molecules corresponding to that specific gene in that cell. This means that a measurement of the whole transcriptome of a cell effectively gives us a high dimensional vector to represent that cell.

2.2.1 Splicing

For most human (and other eukaryotic) genes the initial RNA transcript (pre-mRNA) must be processed to become mature mRNA before protein synthesis can be carried out [14]. An important part of this processing is splicing, where certain parts of the RNA sequence is removed, or spliced out, in a multi-step process. After splicing we have the exon, which is the sequence that goes on to become the mature mRNA, and the introns, which are the removed parts, separated. This means that when we do RNA sequencing of cells, the resulting data will contain transcripts which are spliced and other transcripts which are unspliced. These transcripts correspond to the same genes even if the actual sequence of the transcripts are different. The information of which genes whose corresponding mRNA molecules are spliced and unspliced, in what proportion they are found in these two forms, and in which cells these transcripts are found, can be used in downstream analysis, for example in the computation of RNA velocity [15] which is one of the bioinformatics tools described and used later (see section 4.3).

2.3 NK cell biology

Natural killer (NK) cells are lymphocytes (white blood cells) that sit on the crossroad of innate immune response, which is the first step in the immune defense, and the adaptive immune response, the specific part of the immune defense [16] [9] [17]. NK cells recognize and kill infected and stressed cells by secreting cytokines and chemokines [18] [9]. This secretion also influence the adaptive immune response that follows. We can divide NK cells into two broad subsets based on their expression of the gene CD56, CD56\textsuperscript{bright} and CD56\textsuperscript{dim}. The bright subset is considered a set of less mature cells that can differentiate into dim NK cells. The dynamics of NK cell development is however very complex and the number of differentiation paths and subsets is very high [19].

2.3.1 NK cell differentiation and education

NK cells develop from common progenitors but diverge into distinct subsets, which differ in cytokine production, cytotoxicity and other aspects [5]. An analysis in a 2013 paper revealed a large degree of NK cell diversity [19]. The authors of this paper estimated a total of more than 100,000 NK cell phenotypes.

There is a continuous differentiation of NK cells through a set of intermediate states, from CD56\textsuperscript{bright} NK cells to terminally differentiated,
so called adaptive NK cells. As CD56$^{\text{dim}}$ cells continue to differentiate they lose expression of the gene NKG2A and they acquire inhibitory killer cell inhibitory immunoglobulin-like receptors (KIR) and CD57. They also show a decline in the cells’ proliferation, which is the cells’ ability to increase in number [18]. The adaptive NK cells are called such because they have functions generally associated with the adaptive immune response. The presence of these adaptive cells is associated with past infection by a virus called cytomegalovirus [20] [21]. Physical interactions between cells lead to development of the NK cells’ functional potential [22]. The diversity of NK cell phenotypes also stems from the process known as education.

NK cells have inhibitory receptors that suppress the cytotoxic activity of the cells. These receptors are specific for certain cell-surface proteins called HLA, which are expressed on healthy human cells. These molecules tell the immune system that the given cell is part of the “self”, i.e. that these cells should not be attacked by the immune system [9]. Without this inhibitory system, the NK cells would not only kill infected and otherwise unhealthy cells, but would also be able to kill the healthy cells. There also exist other mechanisms for preventing NK cells from killing the healthy cells [23]. The cells that express the inhibitory receptors that are capable of binding to these HLA molecules get “educated” through a set of combinations of receptors and HLA molecules. The education of an NK cell by a specific HLA molecule is defined by whether it can sense if the given HLA molecule is downregulated on a cell in order to activate its response against that cell [24].

The known marker genes and differentiation processes in NK cell biology briefly outlined above can be incorporated into our analysis and be used to verify the results of the scRNA-seq analysis to see if the analysis can reproduce some of these. This is further discussed in section 8.1.

### 2.3.2 Unknown factors in NK cell differentiation

Several regulatory programs that define the differences between the bright and the dim NK cells are already established in the biological literature, but there are some major unknown factors in NK cell differentiation. It is not clear how the bright and dim NK cell populations relate to other phenotypically defined stages of NK cell differentiation. It is not known whether there exists intermediate cell states which can be described by their transcriptional signature even if they might be considered part of the same NK cell subset when we only consider a few selected cell surface marker genes as we typically do when sorting cells before sequencing them (see section 7.1). Another unknown factor in NK cell differentiation is whether it is a linear process with distinct transcriptional checkpoints or not. These questions will be studied and discussed later as I apply SingleFlow to a novel NK cell data set (see section 11).
2.3.3 Use of NK cells in cell therapies for cancer

There is evidence for NK-cell targeting of human tumors [3] and NK cells have shown promise in so called adoptive cell therapies (ACT) [5]. ACT is a kind of cell therapy where phenotypically beneficial immune cells are transferred into the patient with the goal of ending up with an improved immune response to the cancer [25]. The possibility of off-the-shelf cell therapy based on NK cells has also been described [26]. In order to further incorporate NK cells into therapeutic strategies, we would need to obtain a deeper understanding of regulatory modules controlling clinically beneficial NK phenotypes [5] [3]. As described previously (section 1.1) this is one of the main motivations behind this project. The prospects of NK cell based therapy and how this project fits into this context is discussed in more detail in chapter 15.

2.4 Single cell RNA sequencing

Medical research increasingly deals with the cellular and molecular side of biology [27], where the modification and understanding of cellular behavior through targeted approaches are important. One way to measure a cell’s state is to look at the transcriptome of the cell as described in section 2.2. Single cell RNA sequencing (scRNA-seq) has recently become a very popular method in biological research [28]. This method measures transcriptome-wide gene expression in individual cells, in other words it counts the number of different mRNA molecules found in each of the cells in the sample being studied [29]. This provides us with a high dimensional vector representing the transcriptional state of each cell as described in section 2.2.

Prior to the development of single cell technologies, biologist had to settle for so called bulk-sequencing methods. For measuring transcriptional states, the bulk-sequencing method is known as bulk RNA-seq. This method pools together millions of cells, and therefore masks the differences between individual cells [11]. It effectively averages out some of the heterogeneity of cellular states in the samples we are studying. The expression patterns found in the data derived from bulk RNA-seq might represent the expression of very few cells in the sample or potentially of no cells at all.

scRNA-seq has played a major role in widening our understanding of the rich heterogeneous cell population that we deal with in a given sample [30] [12]. Since this technology makes it possible to study cell-to-cell differences, we are provided with a much higher cellular resolution than with the traditional bulk RNA-seq methods [31] and it facilitates analysis of cellular states in a more unbiased way because it has access to more information about the cellular content of the samples that we are studying.

The very high resolution that scRNA-seq provides, allows us to study new cellular states as well as the variation between these that are simply not possible using methods where we only have an average expression level over a set of cells [32]. This has lead to new discoveries, and novel
technological developments are constantly being applied to new data sets and new cell types. New and rare cell populations have been identified thanks to this method [27]. It has been applied in the field of immune cell biology and provided insights with implications for immune therapies [4], and it has been used to determine the molecular programs defining the identity and function of human NK cells [5]. The analysis of scRNA-seq data can be done on a large number of cells. It has for example been used to profile the transcriptomes of 2 million cells to characterize the transcriptional landscape of mammalian organogenesis [6]. In this case the single cell resolution made it possible to identify many cell types and cell differentiation trajectories that would have been impossible to discover with bulk RNA-seq methods.

2.4.1 How scRNA-seq data is generated

In order to generate scRNA-seq data we first need to isolate the individual cells and lyse them [33]. Following this we perform reverse transcription on the RNA into so called cDNA using uniquely barcoded beads where the barcode identifies the individual cell. This ensures that we know which cell’s transcripts are being sequenced. We then perform PCR, a method for amplifying the DNA signal by creating copies of the DNA sequence, on the cDNA.

One of the main differences between scRNA-seq and bulk RNA-seq data is the low quantity of mRNA isolated from each individual cell. Since bulk RNA-seq sequences many cells simultaneously, there is a much higher number of mRNA molecules available. Therefore scRNA-seq requires us to perform a large number of PCR cycles to end up with enough molecules to successfully perform the sequencing [2]. We therefore need to computationally remove duplicates after counting the molecules. These computationally computed counts are known as unique molecular identifier (UMI) counts.

Eventually we end up with a feature-barcode matrix with UMI counts as the data in the matrix. Each gene is a feature and each cell has a barcode so that we know which cell’s feature we have measured. For generating the sequencing libraries there exist a number of different protocols. In this project we used the recommended 10x Genomics protocol. The specifics of our data sets are described in section 7.1.

2.4.2 Challenges working with scRNA-seq data

The high precision and resolution that the single cell approach provides us with, comes with a cost: single-cell data is much noisier than bulk sequencing data. Two effects that are especially important are so called dropouts and batch effects. The problem of dropout does not exist for bulk-RNA-seq data, because this data is generated by an average over a set of cells. Batch effects exist also for bulk data, but the correction for this effect is different for scRNA-seq data. Here follows a description of these two effects.
Dropouts

Only 10-40% of the transcripts in a given cell are captured in current scRNA-seq methods [34]. This means that all genes in all cells are undercounted. Genes which have very low levels of expression might be measured to be 0, even if these genes are actually expressed in the given cell.

Dropout is the phenomenon of having measured a 0, when the gene is actually expressed. This is therefore known as a technical 0, as opposed to a biological 0, where a gene is actually not expressed. However, as mentioned, this undercounting is present for all cells and all genes, not just those that are lowly expressed and therefore risk resulting in a 0 value.

This means that scRNA-seq works as a kind of sampling method, and dropout is essentially undersampling of RNA molecules. This undercounting obscures many biological signals, such as gene-gene relationships [35], which makes working with raw scRNA-seq data very challenging.

Batch effects

If we perform scRNA-seq on one batch of cells, the gene expression might differ systematically from the gene expression in a different batch. This is known as batch effects and occurs because scRNA-seq data sets are generated in different laboratories at different times with potentially different techniques [36]. Since this is also a problem when studying bulk RNA-seq data, methods exist in well established bioinformatics packages such as limma [37] to tackle this. However, the problem of batch effects in scRNA-seq data is different than for bulk data. The main assumption in the bulk RNA-seq data approaches is that differences in mean gene expression between batches is due to the batch effect and therefore should be removed. For scRNA-seq data this assumption is false and new, single cell specific methods for dealing with batch effects have been developed [36] [38].

2.4.3 The need for bioinformatics

Despite the challenges described above, there have been a lot of interesting discoveries using scRNA-seq data [29]. The technology is constantly evolving and new bioinformatics tools are being developed. To tackle the inherent challenges with the scRNA-seq data as described above, specific tools for analyzing this data have been developed.

scRNA-seq data is a transcriptional snapshot of a single cell. We need to use this data to infer other information, such as the differentiation trajectories or imputed data matrices to tackle dropout. Different statistical models and bioinformatics tools have been developed for these purposes. A lot of the tools that previously were developed for use in bulk RNA-seq analysis can also be applied in the context of scRNA-seq data. But, as discussed, there are some characteristics which are specific to the single cell approach and inherent to the data we are provided with by this sequencing method.
The data that we get from performing scRNA-seq is of a very high dimension: we have thousands of cells with expression levels across thousands of genes. It’s the high dimensional nature of the data that has opened up the possibility to study cell states as a continuous gene expression space, as opposed to consider it as a set of discrete states [11]. However, the high dimensionality also poses some problems in terms of interpretation, visualization and computational complexity, in addition to the problem of noisy data. Dimensionality reduction methods are therefore central to the analysis of scRNA-seq data sets. These and other methods will be discussed in the next chapter, chapter 3. Following this, in chapter 4, I will describe more specific bioinformatics tools for performing analysis of scRNA-seq data.
Chapter 3

Data processing and statistical methods

In this chapter I present some general data processing and statistical methods that have been used as part of tools and pipelines for analyzing scRNA-seq data that have been used in previously published research. These tools provide us with standalone analysis as well as the mathematical and statistical models that underpin the scRNA-seq tools discussed in chapter 4.

3.1 Manifold model

The complexity of the scRNA-seq data, caused both by biological factors such as gene regulation and cellular behavior, as well as technical ones such as those described in the previous chapter (see section 2.4.2), has called for some simplifying assumptions. One such assumption that has proved to work well for scRNA-seq data is the so called manifold assumption [39] [35]. The assumption is that the data actually comes from a relatively low dimensional manifold. Under this assumption noise is treated as a high-dimensional phenomenon that can be alleviated by projecting the data onto the lower dimensional manifold.

The justification for this assumption comes from known biology: the cell state space consists of smooth transitions and genes are regulated in a coordinated way. Transcription factors are proteins that control the rate of transcription for specific genes. From biology we know that sets of transcription factors regulate modules of genes together. This means that the underlying structure of the cells, the gene expression vector space, can be embedded in a lower dimensional space without losing too much information. To put this in statistical terms we can say that the features (i.e. the genes) are not truly independent. This assumption is central to a set of scRNA-seq tools, some of which will be described in more detail in chapter 4.
3.2 Dimensionality reduction

As previously discussed, the data we get from performing scRNA-seq is of very high dimension (see section 2.4.3). For every cell we get a count of every gene. Combined with the single cell phenomenon of dropout that calls for some noise reduction efforts, this makes it necessary to perform dimensionality reduction.

3.2.1 PCA

Principal component analysis (PCA) is a linear dimensionality reduction method that identifies a sequence of projections of the data that are mutually uncorrelated and ordered by variance [40]. These projections are known as the principal components, and the first principal component has the largest possible variance. By considering only the top principal components, we project the data into a lower dimensional space and we still preserve a lot of the variance in the data. This helps to reduce the dimension of our high dimensional data and at the same time it removes some of the noise.

3.2.2 Diffusion maps

Diffusion maps (DM) is a nonlinear dimensionality reduction method [41] that has been used in multiple papers as a method when analyzing scRNA-seq data, both for dealing with the problem of dropout [35] and for studying differentiation and trajectory inference [42]. Both these use cases are dependent on a metric for the distance between cells which originally are placed in a high dimensional space. Diffusion maps embeds the cells into a lower dimensional space, while still preserving some key characteristics of the data it operates on. Cellular differentiation is considered a non-linear continuous process [43] and linear dimensionality reduction methods usually will not be able to preserve the continuous trajectories in the data [42]. Diffusion maps can be used to discover the underlying structure of the data by providing us with an estimate of the low dimensional phenotypic manifold of the data (see section 3.1).

3.2.3 t-SNE

t-distributed Stochastic Neighbor Embedding (t-SNE) is a dimensionality reduction method introduced in 2008 [44]. It provides us with a two or three dimensional embedding of the data and is frequently used for visualization purposes. As the name implies, there is a stochastic element to this method. The algorithm constructs a probability distribution with the objective of preserving local relationships (the neighborhood). Since the embedding that results from running the method is based on this probability distribution, we will be provided with different results if we run the algorithm on the same data set multiple times. These differences are often small and insignificant [45]. Over the last few years t-SNE has
become a well-established tool for use in biological papers for visualization of genomics and transcriptional data [46] [47] and is currently one of the most commonly used technique in scRNA-seq data analysis [48].

3.2.4 UMAP

More recently, the dimensionality reduction method Uniform Manifold Approximation and Projection (UMAP) has been proposed as an alternative to t-SNE for visualization of high dimensional scRNA-seq data [48]. This method is based on manifold theory and topological data analysis [49] [50] and has been tested on a variety of data sets in bioinformatics and other fields [50]. In a 2019 comparison [48] between t-SNE and UMAP on their ability to produce meaningful representations, UMAP was found to produce equally good representations of the cellular space, especially when it came to separating out cell populations with very subtle differences defining them. UMAP was also found to preserve more of the global structure then t-SNE, and to preserve the continuity of cell subsets better. UMAP also had shorter run time than t-SNE in general. How much faster UMAP was, depended on the specific t-SNE implementation they compared it to as there exist numerous implementations of t-SNE. Consequently UMAP has grown in popularity and has since been implemented in established scRNA-seq frameworks [8].

3.3 Artificial neural networks

Artificial neural networks (ANNs) are the main deep learning models and a major part of the field of machine learning [51]. The influence of ANNs has grown rapidly in recent years as they have proved to outperform a number of models in a variety of areas. A standard so called feed forward network aims to approximate a function by learning the parameters of the model by updating the parameters based on the data we feed the model. In this supervised case we need input-output pairs and we want the model to approximate a function that maps a given input to the corresponding output. These models are called networks because they typically compose together many different functions, which are modeled as a directed acyclic graph that describe how these are composed together. These chain structures are the most commonly used structures of neural networks, and we aim to learn the value of the parameters in this model to minimize the difference between the proposed output by the model and the ground truth output. The “deep” part comes from the use of multiple layers of functions being connected. The layers in the middle of the models, which typically don’t have any obvious interpretations, are called hidden layers. Feed forward ANNs have not really been applied in any significant way in the field of scRNA-seq, however they provide the basis for another type of ANN that recently has been applied, namely the so called autoencoders. These are described next.
### 3.3.1 Autoencoders

A more recent development in the field of neural networks are the so called autoencoders [51]. These models are not dependent on us providing input-output pairs, but work in an unsupervised way. The goal of these models are not to approximate some mapping function, but rather to learn the underlying structure of a data set. This is done by constructing both an encoder (that learns the representation) and a decoder (that uncompresses the data again). By putting these two parts together, the autoencoder outputs a reconstruction of the input. The learning process updates the parameters of the model to minimize the error (often squared error) between the original input and the reconstructed one. After training such a network we can use the decoder part of the autoencoder to perform dimensionality reduction. The decoder has then effectively learnt, in an unsupervised way, a way to represent the data in a lower dimensional space and consequently ignore the signal noise. Autoencoders have been applied for dimensionality reduction, data imputation and clustering in the field of scRNA-seq [52] [53]. The application of deep learning models for analysis of scRNA-seq is a field of growing interest [54]. Future applications of ANNs and the potential use of ANN models other than autoencoders for analyzing scRNA-seq data will be discussed in more detail in section 14.9.

### 3.4 Clustering

#### 3.4.1 Louvain modularity

In many complex networks, such as the transcriptional representation of the cells that we get from performing scRNA-seq data, the data points cluster and form relatively dense groups. We often refer to these groups as communities and if we can compute these communities, we can use this to find clusters of the scRNA-seq data. In 2008 Blondel et al. proposed a community detection method known as the Louvain method for community detection [55]. The method was first applied to a data set from the Belgian mobile phone network to identify language communities. It seeks to optimize the network modularity, which is a measure of the strength of division, in a graph. Since going through all possible iterations of nodes is computationally too expensive, the Louvain method is a heuristic method that first optimizes modularity locally and then iterates to optimize the global community detection. The Louvian algorithm has become one of the most popular and most cited algorithms for community detection [56] and is a central component in clustering tools that are often applied in the field of scRNA-seq [57]. Phenograph is the most prominent example of a clustering method based on Louvain modularity as described in section 4.5.1.
3.4.2 Leiden

More recently the Leiden algorithm for community detection has been developed as an alternative to the Louvain algorithm [56]. Just like Louvain, the Leiden method can also be applied to optimize modularity. In the paper that introduced the Leiden algorithm, the authors identified some problems with the Louvain approach. The main problem is that it under certain circumstances can result in arbitrarily badly connected communities. They therefore proposed their own method that guarantees well-connected communities based on some previous work [58] [59] [60] to improve the Louvain algorithm. The resulting method that they call the Leiden algorithm has gained some popularity and has also been applied in the field of scRNA-seq data analysis [8].

3.4.3 K-means

K-means is a clustering method that has been around for a long time [61]. The algorithm aims to partition the data points into k clusters. Each data point should belong to the cluster whose mean, known as the centroid, is closest to that given data point. This results in a partitioning of the data space into regions based on distance to points in a specific subset of the plane. The algorithm starts by choosing k random centroids, or it chooses these based on some heuristic or another domain specific process. It then assigns the cells to a cluster defined by the closest centroid. It then recalculates the centroids based on the actual data points in all the clusters. Then it reassigns the cells to clusters based on these new centroids. This process is iterated until it converges. This is a very simple and efficient clustering method, but it comes with some major drawbacks. One of these drawbacks is that k-means tends to produce equally sized clusters. These are spherically shaped due to the distance metric that is used to assign data points to clusters. The fact that we have to specify the k number of clusters in advance is also a drawback of this method. Despite this, k-means is a widely used method, often in conjunction with other more sophisticated clustering methods.

3.5 Generalized additive models

Generalized additive models (GAMs) [62] are statistical regression models where we have predictors and a dependent variable. The relationships between these follow smooth patterns that can either be linear or nonlinear depending on the data that the models are fitted on. GAMs strike a balance between the very complex and flexible black box learning algorithms (such as ANNs) and the linear, biased and rigid linear models for regression [40]. In the field of scRNA-seq, GAMs have been applied to the calculation of gene trends [63], which are trends showing how the gene expression levels develops as the cellular development proceeds. GAMs are used for this because they are useful in deriving robust estimates of non-linear trends.
How GAMs can be applied to calculate gene trends discussed in more detail in section 6.5.
Chapter 4

scRNA-seq bioinformatics tools

In this chapter I present some of the main bioinformatics tools that already exist for conducting analysis of scRNA-seq data.

4.1 Data imputation

As discussed in section 2.4.2, one of the problems with single cell genomics is that the measured counts only capture a small random sample of the transcripts that are actually present in a given cell. Imputation is an approach for dealing with sparse genomics data that is common in a variety of fields in bioinformatics [64]. Imputation methods essentially replaces missing values with substituted values that can come from varying sources and models depending on the specific method that is applied [39] [64]. The sparseness of the scRNA-seq data comes in part from the undersampling and dropout phenomenon that is inherent to scRNA-seq data. However not all zeroes in the data matrix are equal. Some zeroes come from the fact that the given gene is actually not expressed in the given cell. This makes some traditional imputation methods, methods that have been applied in statistics generally and in other bioinformatics fields, unsuitable in this case as a lot of these methods assume that all zeroes should be imputed and/or that the non-zero values should not be changed.

A number of approaches for dealing with dropout and undersampling in scRNA-seq data have been proposed based on a number of mathematical and statistical models [35] [65] [52]. Broadly speaking they fall into two categories: either they apply a model of the expected gene expression distribution to distinguish true zeros from dropouts in the data matrix, or they apply a data smoothing method [64]. The most recently developed method discussed here, DCA, uses a deep learning autoencoder (see section 3.3) and is a combination of these two categories.
4.1.1 MAGIC

Even if we only observe a small sample of the mRNA in a cell, we can still make useful changes to the data matrix if we incorporate some basic biological insights and some statistical and mathematical methods in our approach. Many of the genes we measure are redundant from a biological perspective because they are regulated together in a coordinated way. This is the realization that is central to the use of the manifold assumption as described in section 3.1. This assumption was central to the development of the Markov affinity-based graph imputation of cells (MAGIC) method that was published in 2018 [35]. It exploits this underlying structure, the manifold, of the transcriptional data to impute missing and undercounted values. The main idea behind MAGIC is to learn the manifold of the scRNA-seq data and use it to recover the gene expression values. MAGIC performs data smoothing for scRNA-seq data based on each cell’s k nearest neighbors and thereby falls into the first of the two categories described above.

MAGIC is based on the use of diffusion maps to estimate the low dimensional phenotypic manifold and looks at the neighborhoods in this space. Euclidean distance gives the incorrect neighbors because cell development in the space twists and turns, as marker genes rise and fall in expression. Therefore cells are embedded into a graph structure and the neighbors are considered based on how many steps away a cell is and weighted accordingly.

Imputing and denoising of the gene counts are done by filtering them as signals on this manifold. MAGIC denoises the data by sharing information across similar cells, and consequently it will also impute missing values (dropout), but it is not restricted to imputing only these values. MAGIC essentially imputes values for each cell based on cells that are most similar to it by using the covariate relationships between genes as justified by the manifold assumption. This incorporates the biological insight discussed above, that the gene set is not independent. This results in an imputed data matrix with modified expression levels for the genes in the data matrix and can be used for downstream analysis.

Validation

In the paper that describes MAGIC [35], the authors showed that the imputed data matrices outputted by MAGIC gave meaningful results for a lot of different applications. One of the main focuses in the paper was the method’s ability to recover gene-gene relations. Because of the high degree of dropout, it is very unlikely to measure two individual genes in the same cell. Gene-gene relations that are already known are therefore often impossible to see in the scRNA-seq data. By applying MAGIC to different data sets, these relations were restored.
4.1.2 SAVER

More recently SAVER was developed as an alternative imputation method to MAGIC [65]. It’s development partly came from the observation that MAGIC’s approach to imputation can lead to oversmoothing and remove some natural cell-to-cell stochasticity in the gene expression that actually captures some meaningful biological signals. SAVER belongs to the first category of imputation methods outlined above, and hence it applies a model of the expected gene expression distribution. SAVER assumes that the count of each gene in each cell follows a negative binomial model and takes a UMI count matrix as input. It then estimates the prior parameters and outputs an estimation uncertainty (unlike MAGIC) and a matrix of imputed gene expression values. SAVER was tested on a number of data sets and performed well in recovering gene expression values and showed improvements also compared to MAGIC on downstream analysis.

4.1.3 DCA

Deep count autoencoder network (DCA) was proposed in a 2019 paper [52] as a new method for denoising of scRNA-seq data. The main component of this method is a deep learning autoencoder (see section 3.3.1) that compresses the scRNA-seq data using specialized loss functions targeted towards scRNA-seq data. Since the compression forces the autoencoder to learn only the essential latent features, the reconstruction ignores non-essential sources of variation such as random noise. The neural network model underpinning DCA is built so that it learns the gene-specific distribution parameters by minimizing the error. The compression of the representation performed by the decoder causes it to learn gene-gene dependencies because some genes can be considered as dependent features. By default DCA uses three hidden layers which allows for non-linear mappings.

One major advantage of DCA is that it allows the user to decide the noise model. As the field of scRNA-seq analysis keeps developing the underlying statistical assumptions researchers build their analysis on are under constant discussion and it has been suggested that the apparent zero-inflation in scRNA-seq data, that a lot of methods assume, is not present when using UMI counts and that it also depends on the normalization method used [2] [66]. It would therefore be useful to let the users themselves decide on a noise model based on the assumptions they make. This also helps keep the method relevant if new insights are encountered as these easily can be incorporated into DCA. These aspects will be discussed in more detail in section 13.1 and section 14.9. DCA is based on the state-of-the-art deep learning Python library TensorFlow and its higher level API Keras [67] which provides it with very good performance.
4.2 Trajectory inference

Trajectory inference, also known as pseudotemporal ordering, is a technique used to determine the fate and the dynamics of cellular differentiation. One important concept in this field is pseudotime. The concept of pseudotime was introduced in one of the early trajectory inference algorithms, Monocle, which since then has developed into Monocle 2 [68]. Pseudotime measures a cell’s biological progression: later in pseudotime means that the cell is considered more mature and later in development towards its terminal state. This same concept has since been used in a number of newly developed tools for analyzing trajectories and differentiation by studying scRNA-seq data [1] [63]. Trajectories inferred from scRNA-seq data can unveil how gene regulation governs cell fate decisions and a number of methods have been developed to this end.

According to a 2019 comparison of trajectory inference methods [69], 50 different methods have been developed since 2014. In this paper they compared the methods both by using a synthetic dataset, which provides the most exact measure for comparing to a reference result, and by using real datasets, which tells us about the biological relevance of the analysis. This comparison concluded that a method called Slingshot predicted the most accurate trajectories. PAGA was another method that seemed to perform well in this comparison. Generally, it found that Slingshot worked best for inferring simpler trajectory structures, while PAGA tended to do better if the underlying trajectory was more complex. The analysis in the paper indicates that some of the methods are complementary and that one preferably should choose a method based on the underlying data if one has additional insight into its structure.

The trajectory inference methods considered in this comparison tend to model differentiation as a series of discrete states and deterministic bifurcations [63]. As discussed in section 2.1, this view of differentiation does not fit with more recent developments in biology and conveys a limiting view of how differentiation progresses. The most recent trend in trajectory inference methods is to model the distribution of a cell population across a continuous cell state coordinate [12]. PAGA incorporate some of these aspects by generating a graph-like map of cells that preserve continuous structures in the data. Methods, which are not included in the comparison mentioned above, have been developed since then to incorporate this biological insight more explicitly. Palantir [63] is one of these methods.

4.2.1 Wanderlust

Wanderlust [43] was introduced in 2014 as one of the earlier developments of trajectory inference methods in the field of scRNA-seq. It is a linear method and only provides a trajectory inference if all the cells can be considered part of the same branch, i.e. it only provides us with an ordering of the cells along a fixed topology that is predefined. This is typical of the early methods that were developed. Other early methods
suffered from the requirement of the user to specify the number of branches and cell fates as a parameter. Since 2014, a number of new methods have been develop that have proven better at identifying known trajectories in well-studied systems and at identifying trajectories in synthetic data sets [69].

4.2.2 Monocle 2

Monocle 2 [70] [68] [71] first learns the overall trajectory topology through a machine learning based dimensionality reduction method called reversed graph embedding (RGE) [72]. The RGE method learns a function that maps data points in a high-dimensional space to points in a lower dimensional space. Monocle uses this to construct the graph that constitutes the trajectory topology, and it then places each of the cells in the data set at its proper place in the trajectory. This results in an ordering of the cells and a basis for calculating pseudotime along the different trajectories. Monocle 2 requires explicit specification of the terminal states, which limits its applications if this information is unknown or if this is the exact thing that we want to calculate. In another comparison where known trajectories and gene expression trends in human hematopoisis was studied, Monocle 2 was also shown to perform worse in recovering the differentiation lineages compared to Slingshot, PAGA and Palantir [63]. Monocle 2 was also found to have worse performance on data sets as the number of cells increased. This indicates some fundamental limitations in its application especially as the field moves toward methods that are able to take advantage of the rapidly increasing amount of scRNA-seq data that is available. This is the opposite of most other methods [73], which tend to perform better given more data.

4.2.3 Slingshot

Slingshot [73] is a more recent method for inferring cell developmental trajectories in scRNA-seq data. It overcomes some of the limitations of both Wanderlust and Monocle 2. Among other things it does not require explicit specification of the terminal states. Slingshot first constructs a minimum spanning tree (MST) on cell clusters to identify the topology of the trajectory structure, i. e. to identify the lineages. It then calculates the pseudotime of each cell.

4.2.4 PAGA

Partition-based graph abstraction (PAGA) [74] is one of the more recently developed methods for trajectory inference. As mentioned above it has been shown to give good results on a variety of data sets. PAGA provides an interpretable graph-like map of the data manifold. This graph is based on the connectivity in this partition. While Palantir and Slingshot automatically can determine the terminal states, PAGA requires specification of the PAGA clusters that belong to a particular lineage.
4.2.5 Palantir

Palantir is one of the most recent developments when it comes to trajectory inference method [63]. It was developed by the same lab as MAGIC (see section 4.1.1) and in some ways it is based on the same underlying assumptions of the existence of a lower dimensional phenotypic manifold. Similar to MAGIC, Palantir uses diffusion maps to estimate this manifold. Palantir was designed to investigate cell plasticity and fate decisions, based upon a continuous, probabilistic model for a cell’s potential to reach different cell fates. Palantir treats cell-fate as a probabilistic process. A cell is not assumed to commit to a given path in a bifurcation of trajectories, but each cell is assigned a probability of ending up in each of the terminal states that the algorithm identifies. These probabilities are known as branch probabilities.

The aim of the model is to build in the assumption of the continuous nature of cell fate and differentiation as discussed above. The actual differentiation process is modeled as a Markov chain, which is turned into an absorbing Markov chain where the terminally differentiated cells are the absorbing states. Based on the graph structure and the Markov chain the cells are ordered and the pseudotime of each cell is calculated. Pseudotime is a measure of the distance between the starting cell and any given cell. Based on the branch probabilities, Palantir calculates the entropy (the negative log of the probability mass function). Higher entropy means that the given cell has a higher potential to reach different terminal states. The entropy is therefore a measure of differentiation potential (DP).

DP captures an aspect of the continuity in cell fate determination. This provides us with a better view of differentiation processes compared to well-defined bifurcations. Cell fate is modeled as a stochastic process and Palantir requires the least amount of a priori biological information among the methods discussed here. We only need to provide the starting cell as well as the data matrix as input and we are provided with pseudotime, branching probabilities and differentiation potential as output.

In the paper where Palantir was presented [63], the authors compared it to the most commonly used competing methods, such as Slingshot and PAGA, and found it to provide better results when inferring trajectories in human hematopoiesis, which is a very well-studied system where the inferred trajectories easily can be tested against known biology.

4.3 RNA velocity

So far we have looked at methods based on studying the RNA abundance of the cells. All of these methods analyze a data matrix where we have the genes and we have a count for each of these genes for each of the cells in the sample. As discussed above (see section 2.4.3), this is just a snapshot of that cell and it does not in itself tell us anything about the dynamics of the cell in terms of differentiation. The trajectory inference methods infer this information from looking at the landscape of cells that all the cells make up.
A different approach for studying the dynamics of cellular development, called RNA velocity, was proposed in 2018 [15].

As alluded to in the background chapter about RNA splicing, the difference between unspliced and spliced mRNAs in a given cell can be used to predict the cell’s cellular state progression. This adds a new layer of information to the analysis. The RNA velocity calculation is based on looking at not only the gene-cell matrix, but by looking at the transcript level counts. It looks at both unspliced and spliced RNA and calculates the first time derivative of the difference in abundance between these as well as at the degradation of mRNA. The resulting metric is called RNA velocity. This can then be used to identify the dynamics and direction of differentiation. More details on how to combine this velocity vector with other analysis tools and how to visualize the results will be discussed in section 6.11.

4.4 Factor analysis

Because of the very high dimensional nature of the scRNA-seq data it would be useful to be able to get a metric for the expression values of a set of genes instead of only considering individual genes. To capture these type of aggregated values, known as factors or metagenes, we can use so called factor analysis.

Factor analysis is a statistical analysis that aims to describe the variability among many observed factors in terms of a preferably lower number of unobserved variables. In our case of scRNA-seq data the many observed factors are the gene expression levels that we have measured, and the lower number of unobserved factors can be a functional factor that consists of a list of genes which together represent a given functional role. The factor is then essentially a weighted list of the genes that go into that gene list. The unobserved factors are metagenes that vary smoothly and are less skewed compared to the expression of single genes. They should be able to identify some broader trends and are not that dependent on the value of one single measurement.

The problem of factor analysis is essentially a factorization problem. We want to factorize our scRNA-seq data matrix to enable this analysis [75]. A lot of different methods have been proposed for achieving this factorization. Typically these are based on singular-value decomposition (SVD), regression or principal component analysis (PCA)[76]. However these methods do not model error in the way the gene sets that we use as factor are defined and they do not take into account unannotated factors. f-scLVM is perhaps the most prominent factor analysis method developed for scRNA-seq and its development was in part motivated by these limitations of the already existing methods [76].
4.4.1 f-scLVM

Factorial single-cell latent variable model (f-scLVM) is a factor analysis method that not only computes estimates of the relevance of the factors it infers, but it also lets us predefine gene set annotations which results in refined factors. We can provide a set of gene lists (these can come from various databases, see section 4.7) which constitute the annotated factors and f-scLVM infers additional unannounced factors based on the variability in the data.

In the paper where it was presented [76], f-scLVM was shown to successfully decomposes scRNA-seq datasets into interpretable components. Since the method provide us with a metric for different factors and their contribution to the variance in the expression levels, it can also be used to regress out the effect of given factors. One example of this is the use of f-scLVM to correct the expression matrix for the effect of the cell cycle as done in various published papers [13].

4.5 Clustering

4.5.1 Phenograph

Phenograph is a clustering method that algorithmically defines phenotypes in the high-dimensional scRNA-seq data [57]. It infers transcriptionally defined clusters in an unbiased way. Phenograph is based on the Louvain modularity (see section 3.4.1). After creating a weighted graph where the weight is dependent on the neighborhood of the two connected nodes (a set of cells), the Phenograph algorithm uses this community detection method to divide the graph into parts which then constitutes the final clusters. Phenograph is currently one of the most established methods for scRNA-seq cluster analysis and is implemented in the most established toolkits [8] [77] and have successfully been applied in a number of scRNA-seq data analysis papers [63] [48].

4.5.2 AP Clustering

Affinity propagation (AP) was introduced as a clustering method in 2007 [78]. It is based on the idea of passing messages between data points. These messages are real-valued and are exchanged until a set of exemplars and their clusters emerges. AP clustering has showed useful for clustering in some fields of computational biology. In the paper where AP clustering was introduced, they applied it to identify genes in expression data of transcripts of possible exons, and to identify regulated transcripts. AP clustering was first implemented in R [79] for use in bioinformatics.

4.6 Differentially expressed genes

Differentially expressed genes (DEG) are genes which are significantly higher or lower expressed in one sample compared to another sample.
This can for example be used to compare the gene expression of two phenotypic clusters or arbitrarily defined sets of cells. The need to calculate differentially expressed genes is also present in the context of bulk RNA-seq, and there are many well-established methods for performing this type of analysis. However, it is unclear whether the methods developed for bulk RNA-seq can be applied reliably to scRNA-seq data [80]. Therefore there has been recent developments to build single cell specific DEG analysis, such as SCDE [81] and MAST [82]. Both these methods were developed with the objective of dealing with the single cell specific challenge of dropout.

### 4.6.1 SCDE

Single-cell differential expression (SCDE) [83] is a single cell specific R package developed by Kharchenko et al. for performing analysis of differentially expressed genes (DEG). The Bayesian approach to single cell DEG analysis that this packages implements was described in a 2014 paper [81] and have proved useful for this purpose [80].

### 4.6.2 Bulk RNA-seq DEG methods

Multiple bulk RNA-seq DEG methods have previously been developed. The most prominent of these have also been implemented for scRNA-seq data through toolkits such as Seurat (see section 4.12) and Scanpy (see section 4.11).

### 4.7 Databases

#### 4.7.1 Gene Ontolgy

Gene Ontology (GO) is a resource of annotated genes and gene products that provides us with a unified definition of terms that represent gene product properties [84]. Traditionally, different areas of biology, such as genetics and biochemistry, used different terminology even if they agreed on the underlying concepts. GO was developed to deal with this and the consequential lack of interoperability of different genomic databases. There are three domains that the GO terms in the ontology can belong to: cellular component, molecular function and biological process. 85 % of human protein-coding genes have GO annotations.

#### 4.7.2 Kyoto Encyclopedia of Genes and Genomes

Like GO, Kyoto Encyclopedia of Genes and Genomes (KEGG) is a data base which contains information about gene functions in the context of molecular pathways in the cell [85].
4.8 Gene set enrichment analysis

If we have a gene list and want to understand which functions or properties this gene list encompasses, we can use gene set enrichment analysis (GSEA). This is a method where we perform a statistical test to see how similar the input gene list is to a predefined database of gene lists. If a list in the predefined database is statistically significantly over-represented in the input gene list, we say that this list is enriched for that input [86]. The predefined set of gene lists can come from any source, but typically we use GO, KEGG or another established functional database. The input gene list is typically constructed based on some shared property among the given genes in an experiment, for example genes that show the same expression pattern or genes which are differentially expressed between two sets of cells.

4.8.1 GO enrichment analysis

Gene ontology enrichment analysis (GOEA) is when we perform enrichment using the GO data sets. There exists a number of tools for performing this type of analysis. GOATOOLS [87] was developed in 2018 and is a Python library. There also exists R packages to perform this analysis such as clusterProfiler [88].

4.8.2 KEGG enrichment

KEGG enrichment analysis is used to extract relevant functional features of gene lists using the KEGG database. There exist a number of packages, mostly in R, to perform this type of analysis. clusterProfiler [88] is one of such package.

4.9 Correcting for batch effects

Batch effects are a problem when studying scRNA-seq data sets that have been produced in different laboratories and at different times. As mentioned in section 2.4.2 the methods developed previously to the development of single cell specific solutions to this effect, was mostly based on assuming that the cell populations were similar across batches so that the mean expression values could be used to remove the batch effects. This assumption does not hold true and a single cell specific method based on the detection of mutual nearest neighbors (MNNs) has been proposed [36]. These mutual nearest neighbors are cells that have similar expression profiles across different batches. We can then use the matching of mutual neighbors to correct for the batch effects. This approach was originally implemented in R as mnnCorrect in the scran package [89] [36], but has since then also been implemented in Python [38].
4.10 Deconvolution

Despite the rapid advancement in scRNA-seq technology, performing scRNA-seq heterogeneous tissues still requires labor-intensive protocols. This has hindered their establishment in a clinical setting. Computational approaches have therefore been developed to infer the abundance of different cell types in samples on which bulk sequencing has been performed. In addition to making the analysis of tissue samples faster and cheaper, the computational deconvolution approaches also lets us gain insight into the composition of pre-existing data sets.

Deconvolution of the cell composition of a sample can be considered as a factorization problem [90]. Some of the most recently developed deconvolution tools [91] [92] rely on a signature matrix that captures the gene signatures of the different cells whose abundance we want to compute. CIBERSORT is perhaps the most well established deconvolution tool and has among other things been used to deconvolute the immune cell content in various cancer types [91] [93].

In addition to deciding on the actual factorization method, the main problem in the field of deconvolution is to create an accurate signature matrix and the construction of this has been one of the main challenges. Signature matrices have previously mostly been constructed by considering existing data bases of marker genes and sequenced cells [94]. More recently however, it has been proposed that we can use single cell data to create this matrix [95]. This may allow us to combine the new insight provided by scRNA-seq data with the advantages of studying bulk sequencing samples and analyzing them using computational deconvolution.

4.11 Scanpy

Scanpy is a toolkit for analyzing scRNA-seq data that was developed to integrate different scRNA-seq data tool [8]. The motivation was to develop a scalable toolkit to deal with the increasingly large data sets that are generated by the rapidly increasing use of scRNA-seq. Where most frameworks and toolkits previously had been developed in R, the researchers behind Scanpy opted for a Python-based implementation. Scanpy continues to be developed and has gotten a lot of tools added to it since it was first published. Some of the bioinformatics methods that I have discussed so far has been implemented as a part of Scanpy and some methods that previously only was found as R packages has also been implemented in Python to make it compatible with Scanpy [96].

4.12 Seurat

Seurat [77] is an R toolkit developed to enable analysis of scRNA-seq data. It is in many ways the R equivalent of Scanpy and enables the integration of various scRNA-seq tools. It was initially develop previous to Scanpy
and offers many of the same features in terms of preprocessing, clustering and visualization.
Part II

Methods
Chapter 5

Technologies

5.1 Programming languages

For this project I have primarily used Python and R for development and for incorporating existing tools into my own scripts. I used Conda as the package manager and took advantage of Conda’s feature of environments. Most of the packages and libraries I used are available through Conda using various repositories. Some of the most important Python libraries I have used include Pandas for data frames, Numpy for matrices and matrix operations, and PyQt for the graphical user interface components of SingleFlow. PyQt is the Python binding of the cross-platform GUI toolkit Qt. R provided me with some of the statistical libraries and some of the bioinformatics methods discussed previously. In order to access packages written in R, I used rpy2 in Python, which is an interface to R from Python, or simply ran R scripts independently as separate Nextflow processes. To perform the initial exploration of the data and to test out the different tools that I have used, I used Jupyter notebooks.

For the version control and to facilitate development, I used GitHub. The final version of the SingleFlow code is available at https://github.com/hernet/SingleFlow. For some of the more computationally intensive calculations, I used the Abel server which the University of Oslo gave me access to. Nextflow is the fundamental framework that SingleFlow was built on, in order to build reproducible, automated and modular workflows. This framework ties together the use of different tools and programming languages. Nextflow is described in more detail in section 5.2.

5.1.1 Dependencies

SingleFlow has a set of software dependencies. Fundamentally it requires the Java Virtual Machine (JVM) and Java 8 or later to run Nextflow. It also requires the installation of the workflow manager Nextflow. The required Python and R packages, and the specific versions that I have used, are listed on the GitHub page (https://github.com/hernet/SingleFlow).
5.2 Nextflow

A bioinformatics pipeline consists of a number of different tasks that can be used in various sequences and combinations. The many permutations a pipeline can follow leads to a certain complexity. There exist a number of bioinformatics pipeline frameworks to deal with the problem of handling the execution of a large number of different software packages that might not be easily bundled together [97]. These frameworks generally work as workflow management systems.

Nextflow is perhaps the most prominent example of such a framework in the bioinformatics discipline [98] [97] and is the one I decided to use for this project. In the field of bioinformatics and biostatistics there exist a number of specialized software packages in different languages to perform specific analyses. The methods that any given pipeline uses might be very specialized and might be most easily implemented by accessing libraries available in a specific scripting language. Tools have therefore been developed in specific languages to easily incorporate already existing packages. One task might require the use of R, while others might require the use of Python, because of the libraries or APIs available in the respective languages. Through Nextflow’s management system we can easily manage these different processes and integrate them. Nextflow also provides efficient parallel execution and traceability [98].

Nextflow implements the dataflow programming paradigm [99]. This paradigm models the data flow as a directed graph and ensures that tasks are automatically started once they receive data through the defined input channels. This allows for very effective parallel execution in a pipeline. The computational dataflow is defined by implementing separate processes, as they’re called in Nextflow, for a given module, and then define channels and connections between these. One process can for example perform dimensionality reduction and then output this to a channel that is then used as input to a downstream process that requires a lower dimensional representation of the data set. The downstream process won’t start until the channel whose content it takes as input has received the data from the upstream process. If multiple processes both depend on receiving this data (but don’t depend on each other) they can be started simultaneously once the dimensionality reduction is performed and their execution will be parallelized. Nextflow also provides us with statistics and figures describing the dataflow and the execution of the various processes. We can for example have Nextflow generate a flowchart of the processes that goes into the analysis or make it report run time, CPU usage and other metrics, if we provide the appropriate parameters when running the pipeline.
Chapter 6

Bioinformatics methods

To build the SingleFlow pipeline I worked closely with the Malmberg Lab to determine what kind of analysis we would want to perform on the scRNA-seq NK cell data set. The biological insight gained from this collaboration, formed the basis for deciding which tools I should include and which biological questions we would try to answer using that analysis.

6.1 Filtering, feature selection, normalization and transformation

Data cleaning was implemented by allowing the user to decide the minimum number of molecules a cell must have in order to be considered part of the analysis and by deciding the minimum number of cells that must exhibit the gene for the gene to be one of the features that will be considered. This type of filtering has been proposed in a number of previous studies [63] [35] [96]. It also provides the user with some flexibility in choosing the parameters for the filtering.

As in other contexts where we are dealing with high dimensional data, feature selection is an important step in the analysis of scRNA-seq data. As mentioned above, the features in this context are the genes. The gene expression vector of a cell puts the cell in a very high dimensional space. Some genes are filtered out during data cleaning, which therefore constitutes the initial feature selection. Other feature selection methods were also implemented based on looking at variable genes and differentially expressed genes. When performing the gene set enrichment analysis we did feature selection by only looking at those genes which had log2-fold change greater than a given number (we used 1 for the results in chapter 11) and that where significantly differentially expressed.

I also implemented normalization methods based on what can be considered the standard pipeline for scRNA-seq preprocessing [96] [2]. I implemented the normalization methods that come with the Palantir library, which performs normalization of the gene expression based on the total expression of the gene in the sample. I also implemented log transformation, which is widely used as a transformation step in scRNA-seq analysis. Since it recently has been suggested that this standard
pipeline of preprocessing might suffer from some flaws [2] [66], I also included the option to not normalize the data at all. Other future changes to this will be discussed in section 13.1.

6.2 Dimensionality reduction methods

I implemented a set of dimensionality reduction methods, both for visualization purposes and for preprocessing the data for further downstream analysis. All the methods discussed in section 3.2 were implemented because they can serve different purposes and they can complement each other. In the suggested analysis in the MAGIC paper for example [35] they perform both PCA (linear dimensionality reduction) and DM (non-linear). The PCA was used for initial noise reduction and the non-linear diffusion maps was used to estimate a lower dimensional manifold that could then be used for further downstream tasks. This order of applying dimensionality reduction method is also the default in SingleFlow. However SingleFlow’s flexibility allows for alternative execution paths and to integrate other methods, such as the recently proposed GLM-PCA [2]. This will be discussed in more detail in section 14.6. t-SNE and UMAP are mostly used for visualization purposes and consequently fits together with the other tools by providing embeddings for us to incorporate other metrics into.

6.3 Gene expression imputation

As discussed in detail in section 4.1 the problems that the undercounted data that scRNA-seq provides us with has shown to be alleviated by various imputation methods. As part of the development of SingleFlow, MAGIC was the primary imputation method that I implemented. The more recent methods of SAVER and DCA were also implemented as alternative imputation methods. The data outputted from any of these methods can be used for computing downstream analysis. It has however recently been suggested that imputation methods are prone to generate false positive or irreproducible results when computing differentially expressed genes using the imputed data [64]. The default workflow of SingleFlow therefore does not use the imputed data matrix for this computation. As with the other workflow specifics this is also something the user can change by using the parameters when running SingleFlow.

6.4 Differentiation trajectories

In order to gain insights into NK cell differentiation SingleFlow needed to implement differentiation trajectory inference methods. As discussed in section 4.2 there exist a number of tools for this, all with their own draw backs. Palantir is the primary method I used when developing SingleFlow. It is one of the most recently developed methods and incorporates important biological insights in its statistical models. It has
shown to work better at certain data sets than methods such as Slingshot [73] and PAGA [74] [63]. Palantir also provides us with a probability of each cell to end up in each of the inferred terminal states, which gives us a useful metric for weighing the cell’s contribution when computing gene trends as discussed in section 6.5. Another advantage of Palantir compared to some of the other methods is that the only input that we need to give is the starting cell, i.e. the cell that is assumed to be the least developed cell. For the specific NK cell user scenarios in this project we identified starting cells by looking at the lowest expression of genes that we know from biology to be associated with cellular development in NK cell biology. We used the imputed data matrix to identify the cell with the lowest (or highest, depending on the gene) expression of a given gene so that the specific cell would not be some outlier in the middle of the manifold, but a cell that resides in a region where the expression of the given gene is low.

Since SingleFlow is designed to be a modular pipeline, long-term we can easily implement potentially all of the other trajectory inference methods discussed. We can also easily compare the results from these, both the actual resulting analysis and trajectories, and their performance in terms of running time and resources required. I have implemented the option to run PAGA and Slingshot using SingleFlow to allow the user to compare the results of the most prominent trajectory inference methods as discussed in section 4.2.

6.5 Gene trends

The computation of differentiation trajectories and the subsequent pseudo-time calculation provided by the previously described methods in section 4.2 and section 6.4 provides us with an ordering of the cells. In order to study how gene expression levels develop as the cellular development progresses, I combined this ordering with the gene expression levels of the cells to produce gene trends. These trends showed us how the expression levels of the different genes develop with pseudotime. These trends were calculated by fitting a GAM, and weighing the contribution of the cells to the expression at the given pseudotime based on their branch probabilities (as calculated by Palantir) so that cells with a higher probability of committing to a given trajectory contributes more to the gene trend for that trajectory. SingleFlow lets us fit a GAM for each of the lineages identified by Palantir so that we end up with a trend for each gene for each of the lineages. GAMs have been found to produce robust trends [63].

Initially the gene trends were only computed with MAGIC imputed data and the trajectory inference as calculated by Palantir as described in the Palantir paper [63]. Imputed data is used here to avoid the effect of dropout to interfere with the computation of the gene trends. SingleFlow allows however for the use of data that has been imputed using other methods, or the use of data that has not been imputed at all. We can also use other pseudotime or trajectory inference methods, but to fit the GAM we need a metric for weighing the cell’s contribution to the model fitting. With
it’s probability metric Palantir provides us with this. When referring to the computed gene trends we call the trends from pseudotime 0 to pseudotime 1, i.e. the whole trend that we compute, the global gene trends. As I discuss in more detail in section 9.2.3, I also implemented a method to zoom in on a specific range of pseudotime providing us with the local gene trends.

6.5.1 Clustering of gene trends

After computing the gene trends of potentially hundreds or even thousands of genes it will be interesting to see if there are certain genes which inhibit a similar pattern with respect to their trends of expression level as pseudotime progresses. Based on known biology we expect sets of genes to be expressed in a similar pattern. We can identify the different ways in which genes and sets of genes can develop with cellular development by clustering the gene trends. I performed this clustering by using Phenograph (see section 4.5.1) as suggested in previous work [63]. After clustering the gene trends SingleFlow performs normalization before it computes the standard deviation and the mean of the gene trends in each cluster. Each of the clusters constitutes a list of genes that can be used for further analysis, such as gene set enrichment analysis to identify which biological programs these trends correspond to.

6.6 Single cell clusters

There exist a variety of methods for clustering. Phenograph is perhaps the most established one in scRNA-seq data analysis and the one I used as the primary clustering method for this project. I also included other clustering methods, such as AP clustering and k-means clustering. For the NK cell biology application we only used k-means in order to confirm the clusters identified by Phenograph, by using the number of clusters Phenograph identified as the k input parameter to k-means.

6.7 Gene set enrichment analysis

The primary source of functional information in the gene set enrichment analysis SingleFlow provided in this particular study was from the Gene Ontology (GO) database (see section 4.7.1). I implemented the Python library GOATOOLS [87] to perform the enrichment analysis. I also added KEGG enrichment (see section 4.7.2) as an alternative. In SingleFlow gene set enrichment can be performed down stream of primarily two processes: gene trend clustering and differential gene expression analysis. Having identified the clusters of gene trends SingleFlow allows for the enrichment of the different clusters to identify the molecular programs that follow the trend corresponding to that cluster. By performing gene set enrichment on the differentially expressed genes between two clusters, we can get a higher level description of what separates the clusters.
6.8 Factor analysis

In order to get a metric for a set of genes I implemented a factor analysis method (see section 4.4) as part of the pipeline. We can obtain gene lists from for example GO for specific biological functions that we are interested in studying. I used the f-scLVM method implemented in the python packages slalom for the factor analysis [76]. This package has been used in a number of other papers, for example in the Palantir paper [13].

6.8.1 Cell cycle correction

One example of the use of factor analysis is its use in correcting the gene expression matrix for the influence of cell cycle effects. The following GO annotations were used to quantify the cell cycle’s effect on gene expression in the Palantir paper [13] and I used the same terms when implementing a cell cycle correction process in SingleFlow: M phase (GO:0000279), DNA replication (GO:0006260), chromosome segregation (GO:0007059), M phase of mitotic cell cycle (GO:0000087) and organelle fission (GO:0048285).

6.9 Differentially expressed genes

SCDE was used to compute differentially expressed genes as this is a single cell specific method that has shown good performance as discussed in section 4.6. The results of the DEG computations were plotted on so called volcano plots. Volcano plots are used to identify changes in large data sets and has the fold change, which is a quantify of how much a measurement has changed, of the expression of the given gene on the x-axis and the significance of the result on the y-axis. The fold change is shown as log2 of the fold change and the significance is -log10 of the computed p-value. On these plots SingleFlow highlights the genes that show a fold change larger than a defined value as well as have a p-value lower than a given value. These cutoff values can be defined by the user or the default values of log2 fold change greater than 1 and p-value lower than 0.05 can be used.

6.10 Deconvolution

I used the scRNA-seq data to construct the signature matrix that subsequently was used as input to CIBERSORT to perform the deconvolution of inputted bulk RNA-seq samples. To create this signature matrix we need to identify the cell types that we want to deconvolute from the bulk RNA-seq sample. It is up to the user of SingleFlow to specify the cell identities that need to be deconvoluted. It could be from performing one of the clustering methods and consider each of the clusters as its own cell type, for example. Since we typically have multiple sorted subsets and their scRNA-seq data as input we could also consider these sorted subsets as the cell types in the deconvolution analysis.
Following the identification of the cell types we have to identify the marker genes, i.e. the genes whose expression value to the largest degree separate these cell types. This was achieved by looking at the differentially expressed genes between the different cell-type identities as well as the highly expressed genes in the scRNA-seq data for these cells. Based on this we then had expression values for a set of marker genes for cells of different types, this is then used to construct the final signature matrix that consists of values for each marker gene for each of the cell types to facilitate the deconvolution of these cell types in the bulk sample.

As mentioned in section 4.10 CIBERSORT is one of the most commonly used deconvolution methods and as suggested in [95] it can be combined with scRNA-seq data by constructing a signature matrix in a similar way to the preceding description. CIBERSORT is therefore the method used for this project, but any other deconvolution method could potentially be paired with the signature matrix we construct.

6.11 Velocyto

In order to calculate the RNA velocity (see section 4.3) of the cells in our data set, I used the command line tool velocyto [15]. Velocyto was ran directly on the output of Cell Ranger which is the software packaged that was used to generate the scRNA-seq data as discussed in 7.1.1. The output of velocyto includes, in addition to the count matrix, all the information about which specific transcripts that was sequenced and therefore also the abundance of the spliced versus the unspliced version of the gene transcript. This was all outputted to a .loom file. I imported this into Python to analyze it further using the velocyto python library. I implemented a calculation of the velocity vectors and calculated locally averaged vector fields because our data sets typically (as in the NK cell user scenarios) consists of many thousands cells. These vectors were the projected onto same t-SNE or UMAP embedding that is used for visualizing other analysis as well. This allows us to visualize the RNA velocity in the same tSNE plot that have visualized the cells in. All of this is implemented as a process in SingleFlow and was applied in the NK cell user scenarios to determine future cell states.

6.11.1 Identify most important genes

To expand on the analysis provided by velocyto, I implemented a method to identify the individual genes that contribute the most to the vectors with the largest magnitude. Since the vectors we are projecting onto the embedding is based on the contribution from a neighborhood of cells, I first identified the cells in the neighborhoods that gave rise to the highest magnitude vectors. I then extracted the number-of-genes dimensional vectors for each of these cells and identified the genes that contributed the most to these vectors. The resulting genes were reported as the most important genes.
Chapter 7

Data sets

7.1 scRNA-seq NK cell data from Oslo University Hospital

The data set that the pipeline has been applied to for the analysis in this project was provided by the Oslo University Hospital. The data set consists of sequenced cells which was collected with informed consent from two healthy donors at the Oslo University Hospital. The cells collected were peripheral blood mononuclear cells (PBMCs), which are a set of different blood cells including NK cells. Using AutoMACS, an instrument for high-speed cell sorting, the NK cells were separated out. After this, the cells where sorted by different marker genes and subsequently scRNA-seq was performed. This provided us with sequencing data for each of the subsets outlined in figure 7.1.

![Figure 7.1: The subset sorting done prior to scRNA-seq. The figure was provided by the Malmberg Lab.](image)

The further processing took place at the Genomics Core Facility at Oslo University Hospital. The recommended 10x Genomics protocol was
used to generate the sequencing libraries. Cell Ranger was then used to process the data, resulting in the count matrix that I used for most of this analysis as well as the transcript data used for the unspliced and spliced considerations. These steps are visualized in figure 7.2.

One of the two donors, referred to as donor 1 or the adaptive donor, had adaptive NK cells as one of the sorted subsets that we sorted for. For the other donor, donor 2 or the conventional donor, this cell population was missing. For each of the donors we had a NK bright subset and we had two so called conventional dim subsets (educated and uneducated). For donor 1 we had the additional adaptive dim subset, and donor 2 had an additional mature dim subset. The number of cells in the various sorted subsets for the two donors are shown in table 7.1 and 7.2.

<table>
<thead>
<tr>
<th></th>
<th>Bulk</th>
<th>Brights</th>
<th>NKG2A</th>
<th>Educated</th>
<th>Uneducated</th>
<th>Adaptive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Donor 1</td>
<td>1683</td>
<td>1774</td>
<td>1356</td>
<td>2242</td>
<td>1413</td>
<td>1331</td>
</tr>
</tbody>
</table>

Table 7.1: Number of cells belonging to the various sorted subsets for donor 1. Donor 1 is our adaptive donor so one of the sorted subsets provided by this donor is a set of adaptive NK cells.

<table>
<thead>
<tr>
<th></th>
<th>Bulk</th>
<th>Brights</th>
<th>NKG2A</th>
<th>Educated</th>
<th>Uneducated</th>
<th>CD57</th>
</tr>
</thead>
<tbody>
<tr>
<td>Donor 2</td>
<td>2135</td>
<td>3440</td>
<td>2544</td>
<td>2865</td>
<td>2578</td>
<td>3145</td>
</tr>
</tbody>
</table>

Table 7.2: Number of cells belonging to the various sorted subsets for donor 2. Unlike donor 1, donor 2 does not have an adaptive NK cell population, but we have a set of more mature NK cells. These cells have high expression of CD57 and are referred to by this in the table.

7.1.1 Cell Ranger

Cell Ranger is a software package developed by 10x Genomics that provide a analysis pipelines that process Chromium scRNA-seq output to align reads, generate feature-barcode matrices (cell-gene matrices) and perform clustering and gene expression analysis [100]. Cell Ranger was used to generate the data that was used for the analysis in this project. The preprocessing Cell Ranger provides gives us a count matrix for scRNA-seq data that we can use for analysis using the tools described previously.
Chapter 8

Validating and assessing the results

The main result of this project is the developed pipeline SingleFlow. Here I present the ways I validated the analysis the pipeline provided us and how I assessed its usefulness as a part of the scientific process of developing biological insights into NK cell differentiation. This will be revisited in chapter 10.

8.1 Reproducibility

As mentioned in section 7.1 we have a data set with scRNA-seq data generated from the blood of two separate donors. An important way of verifying the usefulness of the analysis pipeline I have developed would be to see whether the results from one donor could be reproduced using the data from the other donor. I ran the analysis with the same parameters for both the two donors and compared the resulting analysis. As mentioned in section 7.1 we have one adaptive donor and one conventional donor. This means that we will expect this difference to have manifested itself in the results, but if SingleFlow provided us with a robust analysis we should still be able to see similarities when it comes to the other subsets and other general trends. The results that can be reproduced will also be considered stronger and more robust when we consider SingleFlow’s application in generating novel biological insight.

Another way of assessing the results, was to see if SingleFlow could reproduce results that have been found in other studies and capture biological insights that are established in the previously published NK cell literature. NK cells are a well studied cell type and a lot of its biology, differentiation dynamics and specific genes associated with various processes and subsets are well known. If SingleFlow can reproduce some of this known biology in an unbiased way, this will speak to SingleFlow’s ability to model NK cell biology and hence its ability to be applied in novel ways.
8.2 Collaboration with the Malmberg Lab

Throughout this project I have collaborated with NK cell biologists at the Malmberg Lab. By assessing whether SingleFlow is able to provide these biologist with an analysis that they previously did not have and whether it is useful for their work will also be an important part of assessing SingleFlow’s usefulness.

8.3 Validating the deconvolution approach

In order to validate the results of the deconvolution approach I used the scRNA-seq data sets to create matrices resembling bulk RNA-seq. By averaging the expression values over all the cells in the scRNA-seq data matrix I ended up with two samples with characteristics similar to bulk RNA-seq samples, but with known ground truth with respect to both the number of cells in the sample that came from the different sorted subsets and the ground truth with respect to any other clusters that we want to define or calculate using the scRAN-seq data. If it ended up working for this artificially constructed bulk RNA-seq data set, it would show that deconvolution using the signature matrix that I generate using scRNA-seq data is possible. The resulting signature matrix could potentially be applied to actual bulk RNA-seq data sets.
Part III

Results
Chapter 9

SingleFlow: an improved modularized scRNA-seq pipeline

9.1 The pipeline

The main outcome of this project is the modular pipeline SingleFlow that I developed using the Nextflow framework. Figure 9.2 is a top level overview showing how the different tools described in the preceding chapters fit together to create an integrated analysis where the different tools can complement each other. The arrows show the data flow in the pipeline, from loading the data and preprocessing it, to the generation of plots to visualize the results of the different analysis tools. The results from the analysis outlined in red are all visualized on a plot where the cells are embedded using the chosen method for this. This means that we, when performing one of these analyses, will end up with a t-SNE or UMAP plot where each individual cell is embedded in a two-dimensional space. Examples of such figures can be seen in figures 10.1 to 10.4. Two cells that are close in the embedded space are most likely close also in the higher dimensional space. By studying these plots we can identify the structure of the cellular population and identify potentially interesting regions. The embeddings provided in SingleFlow are tSNE and UMAP, but PCA and Diffusion Maps can also be used for visualization purposes although these methods primarily are used for other downstream analysis. The data imputation methods provided are MAGIC, SAVER and DCA. Trajectory inference can be done by Palantir, PAGA or Slingshot. Gene trends were calculated by fitting GAMs. The inputted gene lists can be custom lists, GO lists or KEGG lists. For performing deconvolution SingleFlow lets the user input the cell type identities that should be deconvoluted in the inputted bulk RNA-seq data set.

Each of the components outlined in figure 9.2 was essentially implemented as a process in Nextflow as described in section 5.2. The use of Nextflow allowed for the implementation of efficient parallelization through the use of its built-in queuing system where processes are ex-
executed based on the connections that they are part of and the dependencies that the data flow definitions imply. Nextflow provides mechanisms for monitoring the processes being executed and for reporting execution time, resource use and other metrics for the different process. An example of this can be seen in figure 9.1. Nextflow provides caching of the intermediate results to skip unnecessary computations in the future when the pipeline is being re-executed. This mechanism facilitates reproducibility and it was extensively utilized for the development of SingleFlow. In addition to the explicitly stated modules in 9.2, SingleFlow provides output of various metrics of the data such as the number of molecules for each cell plotted onto the chosen embedding and the number of cells from each sorted subset that makes up each of the computed clusters. I also provided plotting of the inputted sorted subsets onto the chosen embedding. The full code for SingleFlow has been made available through GitHub at https://github.com/hernet/SingleFlow. More extensive descriptions of how to use SingleFlow is also available through that GitHub page.

![Task execution real-time](image)

Figure 9.1: Execution time of a typical SingleFlow analysis. In this case a set of tools have been executed, including trajectory inference (Palantir) and data imputation (MAGIC).
Figure 9.2: Outline of the processes in SingleFlow. The main processes that I implemented as part of SingleFlow are presented here with arrows showing the data flow and the connections between these processes. The exact execution path is determined by user input to SingleFlow and will depend on the specific user scenario and the analysis that we want to produce.
9.2 Graphical user interface applications

Some of the analysis that was carried out in collaboration with the Malmberg Lab required me to develop some graphical tools to interact with the data and to visualize some aspects of the results from the analysis. All of the features described here were implemented in PyQt, as discussed in section 5.1 of the methods chapter, and integrated into SingleFlow.

9.2.1 Customized clusters

One of the graphical user interface applications that I developed and integrated into SingleFlow allows the user of the pipeline to specify specific custom clusters by clicking the mouse on regions of interest in the t-SNE plot. For example, regions which contains a specific pattern or otherwise is of special interest. A screenshot of this utility is provided in figure 9.3. The different colors represent the different clusters and the cells are colored according to which cluster they belong to. By clicking the mouse on specific cells or regions this feature allows us to assign the relevant cells to a new cluster which is then given a new color in order to visualize the different clusters. The analysis that can be performed downstream of selecting these custom clusters includes among others, the computation of differentially expressed genes. A specific user scenario example where this feature is utilized is provided in section 11.2.2 with corresponding analysis and biological interpretation.

9.2.2 Inspecting cells contributing the most to gene trend phenomenon

By inspecting the gene trends that the pipeline computes, we might encounter certain patterns or otherwise interesting time points that we want to study in more detail. I implemented a GUI feature that lets the user combine the gene trends with the information of each cell’s assigned pseudotime to study the cells most involved in the gene trend at that pseudotime. This allows the user to define clusters of cells consisting of cells that correspond to a given pseudotime. The resulting GUI for doing this is shown in figure 9.4 and figure 9.5.

In this window we can specify pseudotime as a number between 0 and 1 and ask to define a cluster of cells consisting of cells with an assigned pseudotime within a given range around this value. SingleFlow can then visualize a set of different clusters defined by pseudotime as shown in figure 9.4 and figure 9.5. By changing the pseudotime we are interested in, in one of the windows, it also changes in the other. We can then inspect the cells at the given pseudotime both from the perspective of their locations in the embedding and from the perspective of the gene trends. This feature also allows us to visualize the cells through pseudotime. By dragging the scale from 0 to 1 we can see how the cells progress from the starting cell to the terminal states.
9.2.3 Custom range gene trends

Another graphical tool that I developed to integrate into SingleFlow gives the ability to define the pseudotime range within which we want to study the gene trends. We can provide the pseudotime range we are interested in and thereby inspecting and cluster the local gene trends for that range. This allows us to study the gene trend of a specific maturation level in cellular development. An example of its application in studying NK cell biology is provided in the user scenario in section 11.4 where we studied the gene trends in the pseudotime range corresponding to the dim NK cell population.

9.2.4 Gene trend clusters for GO enrichment analysis

As we can see from figure 9.3 we can perform gene set enrichment on the selected gene cluster from the GUI. In this specific case we perform GO enrichment, but KEGG can also be performed given the right parameter when running the pipeline. I also developed a separate GUI feature that lets us inspect the results of GO enrichment analysis. An example of this analysis is provided in figure 10.6.
Figure 9.4: GUI for choosing the gene trend to study and map gene trend phenomena to the embedded cellular space. The clusters are annotated by numbers from 0 to \((n-1)\) where \(n\) is the number of clusters. The x-axis of the plot is the normalized gene expression of the trends in the selected cluster and the y-axis is the pseudotime.
Figure 9.5: GUI for inspecting the cells that has been assigned a given pseudotime and the gene trends at that pseudotime. This shows the cells embedded using t-SNE. This is connected to figure 9.4. By changing the pseudotime in one of these windows it updates in the other, allowing us to study both the cellular space and the gene trends at the same time.
Figure 9.6: GUI for inspecting local gene trends.
9.3 A comparison of SingleFlow to other scRNA-seq pipeline tools

Seurat (see section 4.12) and Scanpy (see section 4.11) are previously published toolkits for large scale analysis of scRNA-seq data. As their approaches for integrating tools for scRNA-seq data are quite similar I will here focus on only one of these, namely Scanpy. Table 9.1 shows an overview of the different tools that are integrated into the Scanpy toolkit and compares it to the list of tools that I have implemented as processes in the generic SingleFlow pipeline. In addition to the differences in the set of tools that they employ, one fundamental difference between SingleFlow and Scanpy is that Scanpy is a Python toolkit, while SingleFlow is an integrated pipeline incorporating several diverse environments with a defined and modularized data and tool workflow structure. There are a number of parameters that can be set when running SingleFlow, allowing a diverse set of analyses to be performed.

New tools can also easily be integrated seamlessly into SingleFlow by simply implementing the required method as a process and adding the necessary connections between it and the preceding and succeeding processes. This allows for tools written in any major programming language to be used as part of a SingleFlow analysis. I have already implemented tools in both Python and R, which are the main languages used in bioinformatics and data science generally. Scanpy is a Python toolkit and is generally used to integrate the analysis specifically with other Python libraries. There exist however a vast amount of tools available in other languages, especially in R. These features make SingleFlow a modular and very flexible pipeline. Unlike in Scanpy, the Nextflow framework in SingleFlow also allows for a seamless parallelization based on which processes can be executed in order or in parallel based on the dependencies of the different processes as defined by the connections between them.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Scanpy</th>
<th>SingleFlow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filtering/cleaning data</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PCA</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>t-SNE</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Diff map</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>UMAP</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Leiden clustering</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Louvain clustering</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(phenograph)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AP clustering</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Volcano plots</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Violin plot</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Nextflow</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>DCA imputation</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>MAGIC imputation</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GUI for studying custom cell clusters</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>GUI for studying cells at specific pseudotime</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Plotting pseudotime with phenograph clusters</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Gene trends analysis</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Palantir</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>DEGs with volcano plots and GO</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>DEG (seurat-like implementation)</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Factor analysis</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>RNA velocity</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Deconvolution of bulk seq-data</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 9.1: Comparison between SingleFlow and Scanpy. The table lists the most prominent analysis tools that SingleFlow and Scanpy incorporate and shows which tools the two have implemented.
9.4 User guide

SingleFlow is, for the most part, a command line tool that can be run with scRNA-seq data sets as input. In the case of deconvolution we can also input a bulk RNA-seq data set. Different samples can be pooled together into one matrix so that they can be analyzed together in order to study the relationship between, for example, different sorted subsets. SingleFlow also takes a number of command line parameters to specify which analysis tools to incorporate in that specific analysis as well as to adjust the default parameters for things such as filtering, normalization and clustering. A detailed user guide for SingleFlow is available from the GitHub repository.
Chapter 10

Validation

10.1 Reproduce results across donors

For each of the two donors we pooled together all of the samples and merged them into one donor-specific matrix. We then performed analysis using SingleFlow on both of these two donors separately. For both the donors the tSNE analysis revealed two transcriptionally unique islands which where connected through a narrow region that we termed the “bridge”. This can be seen from figures 10.1, 10.2, 10.3 and 10.4. The clustering performed by Phenograph revealed five clusters for donor 1 and four for donor 2. The fifth cluster for donor 1 mostly consisted of the adaptive subset of NK cells. This cell population was missing from donor 2. Except for this, both donors reproducibly gave us two bright clusters and two dim clusters.

The Palantir algorithm (see figure 11.7) identified one terminal state, located at the tip of cluster 5 furthest from the bridge, belonging to the NK adaptive population of cells. In the conventional donor, the terminal cell was identified within cluster 3, belonging to the mature NK cell population. The overall progression of cellular development was similar across the two donors. For both the donors the dim population dominated pseudotime as can be seen from figure 11.8. We also identified a set of similar gene trends computed for both the donors. In addition to being similar across the two donors, the direction of differentiation was similar across the Palantir calculation of psuedotime and the RNA velocity calculations (see figure 11.4).
Figure 10.1: Phenotypically sorted subsets for donor 1. This shows the output SingleFlow gave when specified to show the sorted subsets that were in the inputted data sets. The bright subset is highlighted on the tSNE plot on the the left. The conventional dim subsets are the educated and uneducated. Donor 1 also has an adaptive dim subset.

Figure 10.2: Cell clusters determined by Phenograph for donor 1. This shows the output of Phenograph for donor 2. We can see that SingleFlow identified five distinct clusters which are visualized with different colors in this plot.

Figure 10.3: Phenotypically sorted subsets for donor 2. Donor 2 lacks the adaptive NK cell subset that donor 1 has, other than that we see a very similar output as we observed in figure 10.1. Donor 2 has the additional mature NK cell subset characterized by high CD57 expression.
Figure 10.4: Cell clusters determined by Phenograph for donor 2. This shows the output from SingleFlow for donor 2 when performing Phenograph clustering. Compared to donor 1 we can see that we lack the fifth cluster which mostly corresponded to the adaptive subset.

Figure 10.5: The global gene trends for donor 1. Gene trends with similar genes were obtained for donor 2. The blue, green and red lines indicate different checkpoints: progenitor, bright and adaptive respectively. The bottom plot shows the cells corresponding to the pseudotime defined in the gene trend plots.
Figure 10.6: GO enrichment analysis of the gene sets associated with each gene trend. The y-axis show the different GO terms whose genes were identified to be represented in the different gene trends. The x-axis shows the p-value for the different GO terms. The number in the plot and the size of the circle represent the number of genes identified within each GO term.
10.2 Reproducing results from NK cell differentiation literature

From a top level perspective we can see that the tSNE plots in figures 10.1 to 10.4 seem to indicate a separation between two main clusters, the bright and the dim NK cells. As discussed previously, the bright and dim NK cells represent the two major subsets in NK cell biology. We used established marker genes such as MYC, TCF7, BACH2, LEF1, PRDM1, ZEB2 and MAF to establish which cell should be the starting cell when we calculated pseudotime using Palantir. We considered the MAGIC imputed data when we made the decision on which cell to use. This resulted in a cell on the bright side of the bridge region as the starting cell as we can see from the Palantir results in figure 11.7. This is in line with the biological literature [9] [17], where brights are considered a more immature subset of NK cells.

Looking at differentiation from the bright to the dim subset we can compare the different clusters as we move through them guided by pseudotime. If we compare cluster 3 and cluster 5 in figure 10.2, we are studying the transition from conventional to adaptive NK cells. This was characterized by a general loss of gene expression as indicated in the volcano plot in figure 10.7 of the differentially expressed genes between these two clusters. This is in line with what researchers in the field have reported by studying the epigenetic reprogramming during terminal NK cell differentiation [5] [20].

The conventional dim cell population (the subsets termed educated an uneducated) exhibits a high degree of heterogeneity. From figure 10.2 and figure 10.4 we can see that the analysis gave us two transcriptionally defined clusters for this cell population: clusters 3 and 4. These two clusters had a similar distribution in terms of the cells that went into them as we can see from figure 11.3. We can see that these clusters are made up of a large portion of educated and uneducated cells. This is in line with previous work based on bulk RNA-seq data [101] that showed no unique transcriptional signature between educated and uneducated NK cells. Using our analysis we also observed this in our pseudotime calculation: educated and uneducated cells occupied the same clusters and a similar space in pseudotime as we can see from figure 11.8.

Both cluster 3 and 4 were found to consist of cells with a higher expression of IKZF3 and TBX21 compared to the cells in the bright clusters. These genes are described as important transcription factors for maturation of NK cells in previous studies [102] [103] [104], so this result is as expected based on known NK cell biology.

It has been shown that remodelling of the lysosomal compartment inside NK cells is an important part of NK cell education and that it plays a role in the increased functionality of educated NK cells [101]. We observed an increase in lysosomal biogenesis in the later stages of pseudotime as shown in figure 10.8. This is in line with increased functionality within the CD56dim NK cells and in line with the NK cell biology literature.
Compared to conventional CD56$^{dim}$ NK cells, the overall transcriptome of adaptive NK cells was highly reduced. This is in line with epigenetic silencing that has been described for this population of terminally mature NK cells [20] [105] [106].

Figure 10.7: Volcano plot of differentially expressed genes between cluster 3 (blue, conventional dim) and cluster 5 (green, adaptive cluster). The transition between these two clusters represents the transition from the conventional to the adaptive NK cells. We see that there is a general down regulation of genes for cells in the adaptive cluster, in line with published NK cell literature. The cutoff for the colored genes was set so that the genes were colored if they had a p-value less than 0.05 and a log2 fold change greater than 1 or less than -1. The genes were colored according to the cluster they were higher expressed in, so the colors correspond to the ones used for the Phenograph clusters in figure 10.2.
Figure 10.8: tSNE plot incorporating the factor analysis metric for the GO term lysosomal biogenesis. Each cell is assigned a color based on its value for the lysosomal biogenesis. The green and yellow colored cells have a lower value and the more red the cell is colored the higher the value. We can observe that there is a general increase from the bright cell population to the more mature cells.

10.3 Recover the cell type composition in RNA-seq data using deconvolution

To validate the deconvolution approach that I implemented, I created an artificial bulk RNA-seq sample from the single cell data set (see section 8.3). The example discussed here is based on the data set derived from donor 2 (the conventional donor). The averaged expression values was put into a data object that I called the bulk data set. Based on the scRNA-seq data set I created a signature matrix to define the different clusters that we discovered using Phenograph as visualized in figure 10.4. Figure 10.9 shows the signature basis matrix that SingleFlow outputted for this data set. As we can see there is a cell type-specific expression pattern: the cells in a given cluster have higher expression levels of genes used as marker genes for that cluster.

Only looking at the artificial bulk data set that I generated and the signature matrix I constructed (figure 10.9), I was able to recover the cluster composition of the data set. Table 10.1 shows the ground truth values for
the Phenograph cluster composition of the scRNA-seq data set. Table 10.2 shows the predicted composition as computed using CIBERSORT and our custom signature matrix.

<table>
<thead>
<tr>
<th></th>
<th>Blue</th>
<th>Red</th>
<th>Cyan</th>
<th>Magenta</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.446</td>
<td>0.268</td>
<td>0.169</td>
<td>0.117</td>
</tr>
</tbody>
</table>

Table 10.1: Ground truth composition of Phenograph clusters. This table shows the composition of cells belonging to the different Phenograph clusters that we identified. These clusters are visualized on the t-SNE embedding in figure 10.4.

<table>
<thead>
<tr>
<th></th>
<th>Blue</th>
<th>Green</th>
<th>Red</th>
<th>Cyan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.434</td>
<td>0.304</td>
<td>0.193</td>
<td>0.0686</td>
</tr>
</tbody>
</table>

Table 10.2: Results of running CIBERSORT on donor 2 derived artificial bulk RNA-seq data set. The table shows the composition of the Phenograph clusters (identified by their color that has been used throughout) as computed by CIBERSROT when using the signature matrix (see figure 10.9) that I constructed based on the scRNA-seq data.
Figure 10.9: The constructed signature matrix. We can see that the markers are the Phenograph clusters identified by their assigned color. y-axis we can see the genes (identified by there gene id) that is used as marker genes for the clusters. These genes are color coded (on the left side) based on which cluster they are marker genes for. On the x-axis we can see the names of the cell types (the Phenograph clusters in this case). The heatmap is colored based on the expression of the given gene in the given cell type, the darker blue the higher expression. We can observe that, as expected, genes which are used as marker genes for a given cell type tend to have much higher expression levels in the cells belonging to the given cluster.
Chapter 11

User scenarios: applications in single cell NK cell biology

11.1 NK cell differentiation defined through single cell RNA-seq

One of the major goals we set out to reach by applying SingleFlow to the scRNA-seq NK cell data set was to define the differentiation dynamics of NK cells. This was done by constructing two donor-specific tSNE plots to examine the relationship between phenotypically defined NK cell subsets across distinct stages of differentiation.

For donor 1 our data set consisted of an NK population ranging from $CD56^{bright}$ NK cells to distinct subsets of $CD56^{dim}$ NK cells, including adaptive (memory) NK cells. We wanted to use the scRNA-seq data to delineate different molecular programs involved in human NK cell differentiation. To do this we used SingleFlow to perform clustering. In this case we opted for Phenograph clustering which as discussed earlier is one of the most established clustering methods used for scRNA-seq. The result from running Phenograph on donor 1 can be seen in figure 10.2. We also used k-means, which essentially gave the same results when provided with k equal to the number of clusters outputted by Phenograph. We considered this as a way of verifying the clustering. The resulting clusters were then analyzed by calculating differentially expressed genes between adjacent clusters and we produced volcano plots to visualized these results. The order of the clusters for the volcano plots were determined based on the calculation of pseudotime. The volcano plots from comparing the Phenograph clusters are shown in figure 11.1.

In addition to the cluster analysis we also looked at individual genes that are of special interest for NK cell biology, such as genes for NK cell receptors, IL-15 signaling genes and specific transcription factors. We also looked at the genes that were used as subset markers and a set of canonical markers. We plotted the expression of all of these genes using the expression values after MAGIC imputation onto the tSNE embedding. The results are showed in figure 11.2. The expression of the canonical markers across the t-SNE embedding gives support to the idea of a gradual
transition from immature to mature NK cells with progression from cluster 1 to 5.

We can see from figure 11.3 that despite the use of a limited set of markers to define the five sorted subsets, they provided a complete representation of the total bulk NK cell signature as no bulk-specific cell cluster was identified in the composite cell population [17] [16]. We can also see that the NKG2A sorted subset exhibit a very high transcriptional variation as we find it in all of the clusters we identified.

Both donor 1 (figure 10.2) and donor 2 (figure 10.4) revealed two clusters in the CD56 bright NK cell population and two clusters in the CD56 dim population. The adaptive donor, donor 1, has an additional cluster mostly consisting of adaptive NK cells. The analysis identified unique transcriptional clusters, which only partially overlapped with phenotypic subsets as we can see from figures 10.1 - 10.4. The most distal bright cluster, cluster 1, were more transcriptionally diverse compared to cluster 2. We identified, through analyzing differentially expressed genes, specific genes that correlate with the transitions between the clusters. We found differences in important regulatory genes between these two clusters. Cluster 4 appeared to represent a slightly more mature or activated cell state within the dim NK cell population compared to cluster 3.

The clusters we identified overlapped only partially with the phenotypically defined sorted subsets and the two distinct bright clusters are previously undescribed.
Figure 11.1: Volcano plots showing differentially expressed genes between adjacent clusters identified by Phenograph. The cutoff for the colored genes was defined as a p-value less than 0.05 and a log2 fold change greater than 1 or less than -1. The genes were colored according to which cluster they were higher expressed in. The colors of the genes therefore correspond to the colors of the Phenograph clusters in figure 10.2.
Figure 11.2: Gene expression after MAGIC imputation for genes relevant for studying NK cell biology. Based on known NK cell biology we identified genes that were of special interest when studying NK cell differentiation. The MAGIC imputed expression levels generally show smooth transitions in the cellular space and we can see clear trends in the gene expression levels.
Figure 11.3: The top figure shows all the sorted subsets as well as the bulk scRNA-seq subset. The sorted subsets provided complete representation of the total bulk NK cell signature as we can see no separate cell populations in the bulk data set. The bottom plot shows the composition of the different Phenograph clusters by the sorted subsets that the cells in each cluster originate from. The colors of the subsets corresponds to the colors in figure 11.4.
11.2 Continuous and coordinated transcriptional changes in pseudotime

11.2.1 Gene regulatory programs and RNA velocity

Bulk RNA-seq of both bulk and dim NK cell populations has revealed regulatory programs driven by TCF7-MYC in bright cell populations and PRDM1-ZEB2-MAF in dim cell populations [5]. We plotted these genes after MAGIC imputation as seen in figure 11.5. This figure suggested that the TCF-MYC axis is gradually replaced by a PRDM1-ZEB2-MAF-driven program. This lead to the hypothesis that these genes may be used to investigate the direction and relationship of differentiation. We therefore used SingleFlow to apply the analysis of spliced and unspliced transcripts in our data set to calculate RNA velocity to further study this. The use of RNA velocity added another level to the analysis of the snapshot scRNA-seq data. The result of RNA velocity is shown in figure 11.4. Vector length increased with proximity to the dim cell population and cells close to the bridge region exhibited the highest RNA velocity. The direction of the vectors indicated a transition from bright NK cells to dim NK cells.

Figure 11.4: RNA velocity embedded in the t-SNE plot. We can see a region of longer arrows (higher magnitude vectors) in a region before the narrow bridge region. This indicates the transcriptional changes occurring as NK cells transition from bright to dim.

11.2.2 Custom clusters for analysis

Looking at figures of the tSNE embedding of the cells we can see that, moving from the bright cell population to the dim cell population, there exist “bridges” that the developmental path from the bright cells to the dim cells crosses over. Since we know that the transition from bright to dim, in addition to showing a very specific pattern in the t-SNE plots, also constitutes a major developmental process for NK cells, this was a region
that we wanted to study in more detail.

After performing the standard pipeline of preprocessing, dimensionality reduction and embedding, we visualized the data set as a two dimensional tSNE plot. By studying this visualization we identified the bridge regions and used the GUI feature described previously to define the relevant clusters. The selected clusters are shown in figure 11.6.

The clusters consisted of 100 of the closest cells to the bridge on both sides of the bridge. SingleFlow also provided us with information about which of the sorted subsets the cells in each of the clusters belong to. 40% of the pre-cluster consisted of CD56bright NK cells, and around 50% consisted of sorted CD56dim NK cells. This suggests that changes in phenotypic markers may be partly dissociated from underlying global transcriptional changes. After defining the custom cell clusters using the GUI, we used these in the downstream analysis by performing differential gene expression to identify genes which are significantly differentially expressed before and after the bridge.
Figure 11.6: Clusters before and after the bridge region for donor 1. We defined a cluster of cells before the bridge region and one cluster after to compare these clusters. This allowed us to study changes occurring in this specific region in more detail.

11.3 Transcriptional checkpoints and gene-expression trends during NK cell differentiation

11.3.1 Identifying transcriptional checkpoints

Gene expression trends mapped to pseudotime, defined by increasing entropy, identified three distinct transcriptional checkpoints, reflecting important changes in regulatory gene-circuits. The results of analyzing the gene trends can be seen in figure 10.5 with the corresponding GO enrichment analysis in figure 10.6.

We defined custom cell clusters by inspecting the gene trends using the feature described in section 9.2.2. If we could identify a specific pattern of interest in the gene trends, we could then go back and identify which cells that have contributed the most to the gene trend at that time point. We could then specify that this set of cells, defined by the cells within a small margin of the given pseudotime, is a cluster.

Transitioning into cluster 5 was accompanied by the third and final checkpoint, highlighting the important transcriptional changes occurring at this stage of differentiation.

11.3.2 Combine pseudotime and clustering

In order to understand how differentiation progresses and which subsets of cells that exist, we used SingleFlow to combine the pseudotime
computation and the clustering method Phenograph to produces a boxplot like the one in figure 11.8.

![Figure 11.7: Pseudotime plotted onto the t-SNE embedding. We performed trajectory inference and pseudotime computations using Palantir. Here we can see the starting cell, based on high expression of MYC, that we used as input to Palantir. We can also see the one terminal state that Palantir computed. The bottom plot shows the pseudotime of each individual cell colored on the tSNE embedding. Each cell has been assigned a value between 0 and 1 and it is colored accordingly.](image)

The CD56$^{bright}$ NK cell population dominated pseudotime with two distinct checkpoints separating precursors from intermediate states that gradually took on transcriptional signatures similar to CD56$^{dim}$ NK cells. The final checkpoint corresponded with the conventional to adaptive NK cell transition, where further transcriptional changes within the adaptive population were associated with unique divergent gene-expression trends compared to the early phases of differentiation as will be further studied in section 11.4.
As discussed previously the global gene trends are dominated by the bright NK cell population as we can see that these cells take up the majority of pseudotime (figure 11.8). We therefore wanted to zoom in on the later part of the gene trends, specifically we considered the local gene trends from pseudotime 0.8 to 1. The higher standard deviation that we can see from the figure 10.5 for the gene trends also indicates a poorer fit with the overall global gene trend clustering for the trend in this pseudotime range. We therefore applied the feature described in section 9.2.3 to cluster only this late part of the gene trends. We called these trends the dim trends and the resulting analysis is shown in figure 11.9.

This change in gene trend clusters indicates an uncoupling of transcriptional programs from the bright cell population to the dim cell population.
We ended up with three new dim trends as figure 11.9 shows. They contain between 417 and 2575 genes, two of them are down-trending and consequently the majority of the genes are down-trending in this late stage of differentiation which are in line with the previously discussed NK cell literature. The largest trend was the trend termed trend 1 in the figure and the genes in this trend were found to decrease in this pseudotime range. As we can see from figure 11.10 the genes in this trend are associated with mitochondrial translation elongation, apoptotic process, positive regulation of telomere maintenance via telomerase, mitotic cell cycle process and regulation of hematopoietic stem cell differentiation. The gene trend termed trend 2 also shows a generally decreasing trend and the genes in this trends are associated with cellular response to cytokine stimulus, positive regulation of leukocyte cell-cell adhesion, cell chemotaxis, positive regulation of metabolic process and regulation of leukocyte differentiation. The last trend, termed trend 3, showed an increase in gene expression in the final stages of pseudotime. This trend constitute a small minority of the genes. Gene ontology terms associated with these genes include leukocyte activation, negative regulation of cellular process and regulation of intracellular signal transduction.
Figure 11.9: The dim gene trends for donor 1. The plots show the three gene trends that we identified for pseudotime 0.8 to 1, which corresponded to the pseudotime dominated by dim NK cells.
Figure 11.10: GO enrichment analysis of the gene sets associated with the dim gene trends. The y-axis show the different GO terms whose genes were identified to be represented in the different gene trends. The x-axis shows the p-value for the different GO terms. The number in the plot and the size of the circle represent the number of genes identified within each GO term.
Part IV

Discussion and conclusions
Chapter 12

What was achieved

At the beginning of this project I set out to develop a scRNA-seq analysis pipeline to integrate a number of the available analysis tools. I wanted to create a pipeline that was user friendly and that easily could provide some useful analysis of scRNA-seq data, and at the same time provide some of the flexibility, automation and modularity that a complex and rapidly developing field such as scRNA-seq demands. The development of the framework was motivated by unknown questions in NK cell biology and it was developed mainly with biological NK cell differentiation questions in mind. At the same time the resulting pipeline, SingleFlow, works completely independent of its ability to be applied for studying NK cell biology. It could potential be used for analyzing any scRNA-seq data set. Other fields might require other tools, other parameters or otherwise other assumptions built into the analysis. This flexibility to scale is exactly what SingleFlow provides.

SingleFlow proved, in collaboration with NK cell biologists, to provide useful both in confirming already known aspects of NK cell differentiation (see section 10.2) as well as to provide some novel insights. The results were validated and proved to be robust across two independent donor derived data sets (see section 10.1). The analysis identified important gene programs driving functional diversification and specialization during NK cell differentiation. The collaboration resulted in a compact description of the transcriptional diversification at the single cell level during human NK cell differentiation. We identified regulatory program and transcriptional checkpoints. Our analysis also supported the hypothesis of development from brights to dim NK cells as a continuous process as the diversity among the cells in the transition between the two main subtypes contained a diverse set of cells. Specifically we studied a region of cells in the embedded space that we coined the “bridge” which consisted of cells on the crossroad between bright and dim NK cells and therefore was of special interest. The GUI tools I developed proved useful in studying this.

Our analysis found great transcriptional diversity within the bright population, it identified two distinct subsets within this population and it found brights to occupy the majority of pseudotime. These two subsets were previously undescribed and the discovery of these constitutes one
of the novel discoveries using SingleFlow. As discussed in more detail in chapter 15, the differentiation dynamics and gene programs identified using this type of analysis hold potential to guide new strategies for NK cell-based cancer immunotherapy. Despite these promising results derived from the SingleFlow analysis, there are some important limitations associated with the current version of SingleFlow, both in the methods it employs and in the underlying statistical assumptions it is built on. This will be discussed in the next chapter, chapter 13. This sets the stage for chapter 14 where I discuss how to expand the capabilities of SingleFlow as well as other future possible developments. In the final chapter I bring the perspective back to one of the main ambitions that this project may have impact on, the potential use of NK cells in cancer immunotherapy, and I make some final remarks about SingleFlow and how its analysis fits into a wider perspective in the field.
Chapter 13

Limitations of scRNA-seq and the methods integrated in SingleFlow

scRNA-seq is still a young technology, something the rapid development and publishing of new tools to analyze it speaks to [29]. The technology has recently started to show signs of maturity but there are still some major aspects, such as the viability of the statistical assumptions the methods are built on [66] and the validity of the various imputation methods [64], were the science seems to have yet to settle fully. These issues need to be studied further and taken fully into account when analyzing such data sets [2]. These and other aspects that limit the applicability of SingleFlow, as well as scRNA-seq data generally, will be discussed in this chapter.

13.1 Statistical assumptions and characteristics of scRNA-seq UMI data

SingleFlow relies to a large degree on the standard pipeline for preprocessing scRNA-seq that has been used in numerous bioinformatics papers [8] [63] [77]. Some recent papers currently in preprint [2] [66], have however suggested that some of the statistical assumptions underpinning this standard pipeline some degree are flawed. These papers state that many widely used methods for scRNA-seq analysis lack the necessary statistical rigor for being reliably applied to scRNA-seq UMI counts. These methods have, to some extent, been borrowed from bulk RNA-seq without considering the unique characteristics of scRNA-seq UMI count data.

The standard preprocessing typically consists of normalization of gene expression based on total gene expression of the genes followed log transformation. It has been showed that these methods causes false variability in dimensionality reduction downstream [2]. SingleFlow primarily uses, as discussed in section 6.1, this standard preprocessing pipeline and might therefore suffer from this. The problems with this standard pipeline stems from the characteristics of scRNA-seq UMI count
data. The large number of zeros that are apparent in scRNA-seq UMI count data mean that we have to use a so called pseudocount when performing log transform. A pseudocount is a small value we add before taking the log, because we cannot take log of 0. The use of such a pseudocount has been shown to introduce biases in the data [107]. Statistical models built on log-normal distributions can not model exact zeros, which have led a lot of methods incorporating a zero-inflated model. One such method is the SCDE method that SingleFlow employs for computing differentially expressed genes (see sections 4.6.1 and 6.9). It has recently been argued that droplet scRNA-seq is not zero-inflated [66]. It has been argued that the normalization and log transformation is what is causing this apparent zero-inflation and that the data instead can be modeled using multinomial models with no normalization [2]. Methods for incorporating this insight, that are currently being developed, will be discussed in section 14.6.

13.2 Imputation

The imputation methods (see section 4.1) are based on multiple biological and statistical assumptions that may not be true for the biological sample we are studying [108]. In a recent paper the inclination towards generating false signals for different imputation methods was emphasized [64]. Specifically they applied each of the major methods, including the ones implemented in SingleFlow (MAGIC, SAVER and DCA), to a variety of data sets and found that all of the imputation methods to varying degree generated false positive gene-gene correlations and differentially expressed genes. In the paper each of the methods was applied to both simulated datasets and permuted real scRNA-seq data sets. Benchmarking of these imputation methods has previously been done mostly to test if it can recover true signals in noisy data (true positive signals), not whether any of the results that follows the imputation leads to false positive signals. The different methods were found to vary in their inclination to generate false positives, something which is also dependent on the data set being studied. Possible future developments to tackle this problem will be considered in section 14.8.

13.3 Single modality

SingleFlow, as presented in this thesis, only considers one way of measuring a cell’s state, namely the transcriptional state of the cell. A cell is a complex system of transcription, determined by both epigenetic factors as well as the underlying genome (DNA), translation and other processes. If we have data of only one type, we call this single modality. SingleFlow is therefore considered an analysis pipeline based on a single modality, namely the transcriptome. There are a lot of other, potentially complementary, modalities we can include when studying a single cell’s state, including DNA, protein and spatial measurements. By only analyzing the transcriptional state we potentially limit our ability to
derive a deep understanding of a cell’s complexity [109]. The importance of a multimodal analysis and the possibility of extending SingleFlow to incorporate such an analysis will be discussed in more detail in section 14.7.

13.4 The application of SingleFlow to NK cell differentiation

SingleFlow has currently only been tested and validated on NK cell data sets. Although the application of SingleFlow to this domain may be considered successful as discussed in chapter 12, it must be considered a limiting factor that it has only been applied in this one specific domain. The goal of this thesis was to develop a generic scRNA-seq data analysis pipeline, but it still remains to be tested whether this was actually achieved. It would be truly demonstrable that the pipeline is useful if it was tested on other datasets in addition and found to validate what is known about their differentiation biology. The possibility of applying SingleFlow to other domains is discussed in section 14.1.
Chapter 14

Proposal for future work

14.1 New domains

As mentioned in chapter 12 the application of SingleFlow to study NK cell differentiation is arbitrary. SingleFlow could be applied to any scRNA-seq data set to study differentiation and other biologically relevant questions in any type of cell. Obvious examples of other cell types to apply SingleFlow to would be T cells and other other immune cells. There exist an increasing amount of publicly available scRNA-seq data sets for a number of different immune cells [110]. T cells is of special interest because of their application in cancer immunotherapy as discussed in chapter 15. In addition to the advantage of bringing about new biological insights, the application of SingleFlow to new domains would also serve to further validate my pipeline. If SingleFlow could be reliably applied to data sets derived from different cell types to generate novel biological insights in various fields, its robustness and flexibility would be reaffirmed.

14.2 Reproducible analysis

A major topic in biological sciences as well as in science generally, is the ability of scientist to reproduce and build upon the results that other researchers have produced. Recently the existence of a so called reproducibility crisis in basic and preclinical research has been discussed [111] [112] [113]. One study in the field of oncology found that only 11 % of research findings could be reproduced [114]. The lack of reproducibility stems from various sources, most of which are outside the realm of bioinformatics. This does however speak to the importance of producing code and software tools that support reproducible research. This is something Nature Methods, one of the highest impact scientific journals in the field of biomedical research, has emphasized the importance of. In a 2014 editorial [115] they discussed how it is important to not only publish the code that has been used as part of the scientific research, but also to facilitate the actual implementation of the computational methods for others to use. This includes things such as the dependencies and other specifics that need to be in place for the relevant code and method
to actually work. By implementing the analysis that we performed on the NK cell data for this project into an integrated Nextflow pipeline that others can easily access through GitHub and use for their own purpose, I have facilitated reproducibility of our results. To handle computational research reproducibility problems in an even better way, I could implement Docker’s container technology [116].

14.2.1 Docker image

Docker [116] provides lightweight Linux containers. It is perhaps the most prominent container technology for consistent development and deployment. Docker essentially provides us with lightweight virtual machine where we can set up a computing environment, including all dependencies, data and code, in an image. This Docker image can be shared and run seamlessly on any Linux system. Multiple containers can run in parallel to make complex computational tasks more efficient. NextFlow also provides easy integration with Docker by letting each process run in its own container. By providing a Docker image I would, in other words, be able to package the binary dependencies of SingleFlow into a standard and portable format that easily could be executed on any platform that supports the Docker engine. The combination of NextFlow and GitHub, which I have already used for SingleFlow, with Docker would make the pipeline truly replicable [98].

14.3 Other databases of gene sets for enrichment

In addition to the databases, GO and KEGG, that I have described in this thesis for extracting functional information from gene sets, there exists a number of other databases. The main ones would be the BioCarta pathway database [117] and the Reactome pathway database [118]. In addition to using these there is also the possibility of using databases of curated gene signatures for cells and cell types of interest.

14.4 Additional clustering methods

There exist a number of clustering methods that I did not implement in SingleFlow. Since identifying transcriptionally meaningful subsets, which can be achieved by clustering, is such an important task when characterizing the differentiation dynamics of NK cells and other cell, it would be interesting to apply a much bigger array of clustering methods to the data. In the application of SingleFlow to NK cell differentiation we did show the clustering to be robust across Phenograph, which is perhaps the most established method for clustering scRNA-seq data, and k-means, which is one of the simplest and most intuitive clustering methods. Since these two clustering methods are so different in their assumptions and algorithm, and they’re still able to identify more or less the same clusters, we can see this as a validation of the clusters we identified. We
should maybe also have applied the AP clustering method (see section 4.5.2) in this case. As mentioned there are also numerous other clustering methods, maybe most importantly the hierarchical clustering methods which I haven’t discussed in this project previously. Future clustering of scRNA-seq data will, in the case of integration into a multimodal analysis, have to be able to evolve to incorporate this additional data [119].

14.5 Other trajectory inference tools and complementary implementation

Given the extensive list of tools for studying differentiation and trajectory inference that, as discussed in section 4.2, has been developed over just the last five years, it is of course impossible and impractical to include all of these tools. Some of them suffer from such extensive limitations both caused by inaccuracy of the models they deploy and by the limited amount of information they are able to infer. There exist however a tools that I have yet to implement into SingleFlow that potentially can provide us with useful insights [69]. SingleFlow currently functions by choosing one of what might be considered the most promising tools (PAGA, Slingshot and Palantir) and then uses the result as the pseudotime and trajectory computations for downstream tasks. As concluded in the comparison discussed in section 4.2 [69], some of the existing tools are complementary. One future extension of SingleFlow could therefore include the option to combine the results from multiple methods as well as to implement methods that I have yet to explore.

14.6 New preprocessing methods

As discussed in section 13.1 some problems have been identified with applying the standard preprocessing pipeline for scRNA-seq data. A few new methods have therefore been proposed. GLM-PCA [120], which is a generalization of PCA (see section 3.2.1) that can incorporate different likelihood models, has been proposed as one solution [2]. A multinomial statistical model of the data can then be applied and there is no need for normalization. Townes et al. also proposed methods to approximate this type of model to make the computational task more manageable. In the same paper the use of a multinomial deviance statistic to rank genes for feature selection was also described. Another paper, also currently in preprint, suggested the use of regularized negative binomial regression for normalization which according to the paper removes technical variation while preserving biological heterogeneity [121].

Any or all of these proposed methods could be implemented in the SingleFlow framework as alternatives to the preprocessing methods that are currently being employed, while still employing the part of the analysis pipeline that is executed downstream of this. As the statistical characteristics of the scRNA-seq data is being studied and subsequently
the statistical models that should underpin the study of this data becomes apparent, these and other methods will become mainstream. Another interesting development in genomics analysis in general is increasing use of deep learning models. These models also hold great potential for preprocessing and are discussed in more detail in section 14.9.

14.7 Multimodal analysis

As mentioned in 13.3 SingleFlow suffers from the limitations of a single modal analysis when it comes to grasping the complexity of cellular development. To increase the depth of our understanding of the single cell’s differentiation and molecular dynamics we could extend SingleFlow to combine the study of scRNA-seq (the transcriptome) with other modalities such as DNA, protein, chromatin modifications and spacial genomics [29]. We could also potentially integrate the study of scRNA-seq and bulk RNA-seq data in a more comprehensive way than what the deconvolution approach in this thesis offers.

The protein modality of single cells have successfully been studied by employing single cell mass cytometry (CyTOF) [122]. It has for example been used to identify cell populations in brain myeloid cells [123]. The DNA modality can be studied by chromatin accessibility [124], assessing epigenetic heterogeneity [125] and variability in chromosome structure [126]. Spatial transcriptomics, the spacial location of gene expression within tissue, is one of the main frontier in the study of transcriptomes [127]. New and promising methods have been developed and numerous projects pursuing this are under way. There is still little data available, however methods to combine this modality with for example scRNA-seq will most certainly be of value in the future and as discussed in section 14.9 deep learning models might play an important role in this field.

Ideally we want to observe all aspects of a cell simultaneously to understand fully how it develops and interacts. This is currently technologically not possible. However, new methods to measure multimodality in single cells are being developed. Recently a method, called ECCITE-seq, was published that allows for simultaneous detection of transcriptome, proteins, clonotypes and CRISPR perturbations [128]. Integrating various modalities is critical in future applications of an analysis pipeline like SingleFlow [109]. Integrative single-cell analysis has proved useful revealing the impact of graphene on human immune cells [129] and in other fields.

14.8 New imputation methods

Other imputation methods exist in other domains of bioinformatics that might give us some ideas of possible ways of improving the imputation of scRNA-seq data. For so called genome-wide association studies (GWAS) which studies the entire genome (DNA) there exist imputation methods for imputing missing so called polymorphism, areas of the genome where nucleotides have been substituted [130]. Imputation methods for this type
of data relies on external reference datasets [131] [64]. Such reference
databases do not yet exist for scRNA-seq data and a future possible
development in the field would be to create such a database to have a
reliable reference when computing the imputed data matrix.

One of the main takeaways from the study discussed in section
13.2 on the imputation methods’ tendency to produce false positives,
is the fundamental trade-off between sensitivity and specificity that the
imputation methods have to strike. This is therefore something the user
of a given imputation tool has to be vary of. Since the results also varied
between different data sets, this means that one might want to use different
methods and strike a different balance between sensitivity and specificity
depending on the specific of the data set one is studying and of the specific
user scenario.

It would also be an idea to employ other statistical models of the data in
line with the statistical assumptions that is discussed in section 13.1. DCA
(see section 4.1.3) lets the user specify the noise model and any statistical
model can in principle be incorporated into it. Other developments in
deep learning (see section 14.9) and in the development of new multimodal
analyses (see section 14.7) could potentially also be integrated to provide
increased understanding of the noise and technical variability factors in a
scRNA-seq data set.

14.9 Deep learning models for scRNA-seq data ana-
lysis

Deep learning methods that over the last few years have led to major
breakthroughs in a variety of fields in computer science, have increasingly
been applied in the field of genomics, including for transcriptome profiling
[54] [132]. The development of new and powerful Python frameworks such
as TensorFlow, Keras and Pytorch have provided us with a fairly easy way
of implementing high performance, complex and flexible neural network
models. The combination of these frameworks and the large increase in
the amount of data both in genomics generally as well as in scRNA-seq
specifically, has driven this development.

It has been argued that machine learning is well suited to genomics
generally, including the study of transcriptomes, both because of the
complexity of the data we are dealing with and because of the amount
of data that is typical in studying genomics [133]. The complexity of the
scRNA-seq data has been a recurring theme throughout this thesis and
questions about how to best model noise and perform feature selection
are, despite a number of advances recently, still an area of research. Deep
learning models are good at incorporating complex dependencies and
could potentially help improve preprocessing of scRNA-seq data.

As discussed in section 14.6 there is a development of alternative
preprocessing pipelines for use in scRNA-seq data analysis. Machine
learning methods could for example be used to combine multiple such
pipelines and then let the data, based on an appropriate error model, decide
what type of preprocessing would be ideal for the given data set [54].

Deep neural network, which learns feature representations, have already successfully been applied to scRNA-seq data for dimensionality reduction [134] and for data imputation using DCA as described and discussed previously (see section 4.1.3). Based on the unsupervised autoencoder type of machine learning models, new clustering methods have also been proposed [53]. DCA, that has been implemented as part of SingleFlow, is still the most prominent example of the application of deep learning models in the field of scRNA-seq [54], but there is a lot of development in this field, especially with the amount of data we now have available for analysis. This hold promise for further improvements over the next few years.

One of the most prominent types of neural network in the field of machine learning generally, are so called convolutional neural networks. These ANN models have been especially popular in the field of image analysis where they have driven the advances in image recognition, segmentation and object detection [135]. Convolutional networks typically works for data sets that has a known grid-like topology [51]. If this type of model is to be applied in the field of scRNA-seq in the future, it will most likely be in a type of multimodal analysis, for example where spatial transcriptomics is analyzed [54].

Traditional statistical models and approaches, which most of the methods described throughout this thesis are based on, are still dominant in biological research [132]. Partly this is because it is only recently that the relevant machine learning methods have been able to scale and it’s only recently that the amount of data that we see today have become available. However there are also some challenges with applying deep learning models in the field of genomics that also hinders their implementation [132]. These will be discussed next.

### 14.9.1 Challenges with deep learning models in genomics

Deep learning models are inherently hard or impossible to interpret. In general they work as a black box where we only know the input and the corresponding output and they don’t provide any explanations of the features involved in the modeling. This can be a limiting factor as it can make it more challenging to extract a deep biological understanding from these models, even in cases where they model the phenomenon we’re studying well in terms of predictive accuracy [136] [132]. There have been made efforts to develop techniques to be able to make meaningful interpretations of deep learning models and this is a field of continued research [136], so this limitation might be resolved in the future.
14.10 Publish paper in collaboration with the Malmberg Lab

This project was carried out as a collaborative effort with the Malmberg Lab and culminated in the writing of a paper on the topic of NK cell differentiation. As of writing this paper has yet to be published. To expand on the analysis that we are including in the paper, all the points for future possible work raised above would be worth considering. In addition there has already been started work on sequencing new samples.

14.10.1 More sequencing data

In general, results that can be reproduced across multiple independent samples from different donors suffer from less bias and therefore can be considered more robust. In this project we applied SingleFlow to two independent NK cell data sets from two different donors. As we discussed previously (see 10.1) we were able to reproduce results across these donors. By including even more donors and testing if our results can be reproduced also across these new donors, we could potentially further strengthen our conclusions.
Chapter 15

Final remarks

One of the main ambitions of this project was to impact the value chain for the characterization and selection of optimal immune cells with the prospect of applying NK cells in immunotherapy. Cell therapy is rapidly developing as a cancer treatment strategy. So called adoptive cell immunotherapies, where immune cells are transferred into the patient after being directed into a phenotypical state that is beneficial for the cancer treatment [25], are some of the most promising immunotherapies. The cells used for this purpose may originate from the patients themselves or from other individuals. The most prominent application of adoptive immunotherapy for cancer is its application in combination with modified T cells for B cell malignancies [137]. These therapeutical techniques have been shown to have potentially curative effect in certain cancer types for some patients. We need to be able to understand the cellular and molecular mechanism that underlies the immune systems lack of response to a specific cancer in a specific patient, and then convert the immune cells from non-responders to responders.

In order to further improve the cell therapy approach to cancer treatment, to be able to increase the number of patients and cancer types that it can treat, looking at other immune cells than T cells that are involved in the anti-cancer immune response is of interest [138]. NK cells is, as described previously (see section 2.3.3), one such immune cell and it also exhibit many features normally associated with adaptive immunity [24].

NK cells are of increasing interest for use in immunotherapeutic strategies for cancer treatment. In order to use NK cells for this therapeutic strategy, we need to know which phenotypical states that exist and we need to know what drives differentiation between these states. One possibility with NK cells is to engineer the cell’s metabolism to be resistant to the metabolically restrictive tumor microenvironment and to the molecules generated by tumors that can suppress NK cell metabolism and function [139].

Recently the possibility of off-the-shelf therapy based on NK cells has been described [26]. In order to use NK cells in a immunotherapeutic strategy, we would need to develop strategies to guide NK cell differen-
tiation towards a desired functional phenotype. The goal is to be able to take NK cells from a patient and direct them towards a favorable phenotypical state that then would help target the cancer for treatment. Current approaches are unspecific and unreliable. Bioinformatics and machine learning technologies are necessary to intelligently select the relevant cells that has the desired functional phenotype and to identify which states we would want to direct them towards before transfusing them back into the patient.

I have presented SingleFlow, an integrated bioinformatics pipeline for single cell RNA-seq analysis. SingleFlow has, as I have discussed in detail in chapter 13, a number of limitations. Despite this, and by having these limitations in mind, we were able to successfully apply SingleFlow to a novel NK cell scRNA-seq data set to extract some novel biological insight to improve the understanding of NK cell differentiation and the subsets of NK cells that exist. By providing an integration of the most cutting-edge scRNA-seq analysis tools, in combination with my own developments, SingleFlow provided robust and biologically meaningful results. By this I have given a small contribution towards the overall grand objective of future development of NK cell immunotherapy.
Bibliography


[36] Laleh Haghverdi et al. ‘Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbours’. In: Nature Biotechnology 36.5 (May 2018), pp. 421–427. ISSN: 1546-1696. DOI: 10.1038/nbt.4091. URL: https://www.nature.com/articles/nbt.4091 (visited on 19/02/2019).


114


