

---

# PREPROCESSING AND ANALYSIS OF SINGLE-CELL RNA- SEQUENCING DATA

---

**Master's Thesis**

Degree Programme in Biotechnology

Faculty of Medicine and Life Sciences

University of Tampere, Finland

Meeri Pekkarinen

May 2018

## MASTER'S THESIS

Place: University of Tampere, Faculty of Medicine and Life Sciences  
Author: Meeri Pekkarinen  
Title: Preprocessing and analysis of single-cell RNA-sequencing data  
Pages: 55 pages + an appendix  
Supervisors: Dr.Tech Juha Kesseli, M.Sc Antti Ylipää  
Reviewers: Professor Matti Nykter, Dr.Tech Juha Kesseli  
Date: May 2018

---

### ABSTRACT

In recent years, single-cell measurement technologies have greatly advanced and offer a new approach to study biological problems. While the traditional RNA sequencing results in computational average transcriptome that represents the whole biopsy, scRNA-sequencing records the transcriptional differences between the cell types. However, the method is also more sensitive to various biological and technical noise, and the field still calls for further research and establishment. Especially data normalization and quality control require novel methods, since many of the tools originally developed for bulk RNA-seq data are based on assumptions that are not valid for scRNA-seq data.

The goal of this thesis was to gain insight in RNA-seq analysis and especially find the optimal ways to preprocess the data and asses its quality. The relevant tools were chosen and further tested in the computational part of the thesis. The final and the most important deliverable was an analysis pipeline constructed by combining the best approaches and necessary quality metrics.

A three-step pipeline utilizing various command line tools and Bioconductor R-packages was implemented. This pipeline performs preprocessing, transcript quantification, filtering, normalization and simple downstream steps. Most importantly, it produces both visual and statistical information for estimating various general features and quality properties of the data. The pipeline was successfully used to process two public scRNA-seq datasets.

The work was done for Genevia Technologies Oy in Tampere between October 2017 and May 2018. An ultimate goal was to develop a generalized pipeline that would be useful to the company. However, diverse analytical and technical issues make this task very challenging, and a couple of pitfalls still remain unsolved. One major reason is that no best practices are yet established. Regardless, the information provided by the pipeline should be helpful for picking suitable tools and thresholds for more sophisticated methods. Future development in the field will most certainly discover biological information that cannot be discerned by current tools.

## PRO GRADU -TUTKIELMA

Paikka:	Tampereen yliopisto, Lääketieteen ja biotieteiden tiedekunta
Tekijä:	Meeri Pekkarinen
Otsikko:	Preprocessing and analysis of single-cell RNA-sequencing data
Sivumäärä:	55 sivua, 1 liitesivu
Ohjaajat:	TkT Juha Kesseli, DI Antti Ylipää
Tarkastajat:	Prof. Matti Nykter, TkT Juha Kesseli
Päiväys:	Toukokuu 2018

---

### TIIVISTELMÄ

Solun molekulaaristen ominaispiirteiden määrittämiseen on hiljattain kehitetty RNA-sekvensointimenetelmä, jolla voidaan tutkia yksittäisten solujen mRNA-tuotantoa suurella tarkkuudella. Menetelmästä käytetään nimeä scRNA-seq (single-cell RNA-sequencing). Aikaisemmin RNA-sekvensoinnissa (bulk RNA-seq) on voitu mitata isojen solupopulaatioiden keskimääräistä käyttäytymistä, mutta ei ole pystytty erittelemään solutyypin tarkkoja piirteitä tai seuraamaan kehittyvissä solupopulaatioissa tapahtuvaa monimutkaista molekulaarista evoluutiota.

Uusi sekvensointimenetelmä avaa kiinnostavia mahdollisuuksia biologisten ilmiöiden syvällisempään ymmärtämiseen ja tiedon soveltamiseen esimerkiksi taistelussa syöpää vastaan. Toisaalta tarkkuus tekee menetelmästä herkemman erilaisille teknisille ja biologisille vääristymille, mikä asettaa haasteita erityisesti datan esikäsittelylle, laatuanalyysille ja normalisoinnille. Päivitettyjä laskennallisia ja analyttisiä metodeja tarvitaan, sillä monet aikanaan RNA-sekvensointidatalle kehitetyt työkalut tekevät datasta tietynlaisia oletuksia, jotka eivät välttämättä päde scRNA-datalle.

Tämän Pro gradu -tutkielman tarkoitus oli perehtyä scRNA-sekvensointiin ja työkaluihin, jotka on räätälöity ottamaan huomioon datan aiheuttamat haasteet. Lisäksi työn tärkein tavoite oli suunnitella ja toteuttaa analyysivuo, joka suorittaa datalle automatisoidun esikäsittelyn sisältäen mm. laatuanalyysin, suodatuksen ja normalisoinnin. Tutkielma toteutettiin yhteistyössä GenevaTechnologies Oy:n kanssa Tampereella, ja sitä työstettiin lokakuusta 2017 toukokuuhun 2018.

Työn tuloksena syntyi kolmeosainen analyysivuo, joka yhdistelee useita uusimpia työkaluja shell- ja R-pohjaisten skriptien avulla. Analyysivuon keskeisimmät vaiheet ovat sekvensointitiedostojen laatuanalyysi ja suodatus, geeniekspression kvantifiointi referenssitranskriptomia vasten sekä tuloksena syntyvän matriisin suodatus, visualisointi ja normalisaatio. Lisäksi sovellettiin muutamaa korkeamman tason työkalua, joilla voitiin tehdä datasta yksinkertaisia biologisia johtopäätöksiä ja arvioida osien toimintaa. Analyysivuon avulla prosessoitiin kahta julkista scRNA-datasettiä, ja sen räätälöityvyyttä pyrittiin parantamaan konfiguraatitiedoston avulla.

Päämäärä oli tuottaa analyysivuo, jota voidaan tulevaisuudessa soveltaa myös yrityksen tarpeisiin. Mahdollisia analyysiin vaikuttavia muuttujia, sekä datalähtöisiä että teknisiä, on kuitenkin varsin paljon, eikä kaikkiin ongelmakohtiin ole vielä kirjallisuudessakaan kehitetty yleiskäyttöistä ratkaisua. Näinollen tämäkään analyysivuo ei sellaista tarjota, ja tuloksia on osattava aina tarkastella kriittisesti ja mahdollisesti täydennettävä analyysiä räätälöidyillä menetelmillä. Ala kuitenkin kehittyy nopeasti, ja tulevaisuudessa on odotettavissa uusia, tarkempia menetelmiä ja mielenkiintoisia löytöjä.

# Acknowledgements

This thesis was an adventure that I could not have been able to complete without the help of several people. First, I am very grateful to my supervisors, Juha Kesseli and Antti Ylipää for feedback, trust and good discussion. My thanks also for Professor Matti Nykter for arranging the computational capacity from CSC and reviewing the written thesis. Moreover, Matti Annala kindly helped me to get started with the computational environment. I also want to thank the whole Genevia team for the support, trust and wonderful working environment! It has been such a pleasure to get to know you all, and I think there was not a single day without at least one good laugh and excellent coffee, even though sometimes certain challenges felt bigger than I could handle.

I probably would not be alive without my beloved husband Esko who has supported me in such many ways during my thesis and studies. In addition to his endless patience and love, he also helped me with the grammar and provided valuable insight as a software developer. Thanks also to my dear friends and siblings for patience, encouragement and providing some down time in situations where I really needed a break.

Finally, I would like to thank my wonderful parents who have trusted in me for all these years and always encouraged my interest towards science and nature.

Meeri Pekkarinen  
May 2018

# Abbreviations

HISAT	Hierarchical indexing for spliced alignment of transcripts
IVT	<i>In vitro</i> transcription
EM	Expectation-Maximization algorithm
LMM	Linear mixture model
LVM	Latent variable model
MAD	Median absolute deviation
MAST	Model-based analysis of single cell transcriptomics
PCA	Principal component analysis
PLS	Partial Least Squares
SC3	Single-Cell Consensus Clustering
SCA	SingleCellAssay (a data object in R)
SCE	SingleCellExperiment (a data object in R)
SLURM	Simple Linux Utility for Resource Management
STAR	Spliced Transcripts Alignment to a Reference
tSNE	T-Distributed Stochastic Neighbor Embedding
UMI	Unique molecular identifier

# Table of Contents

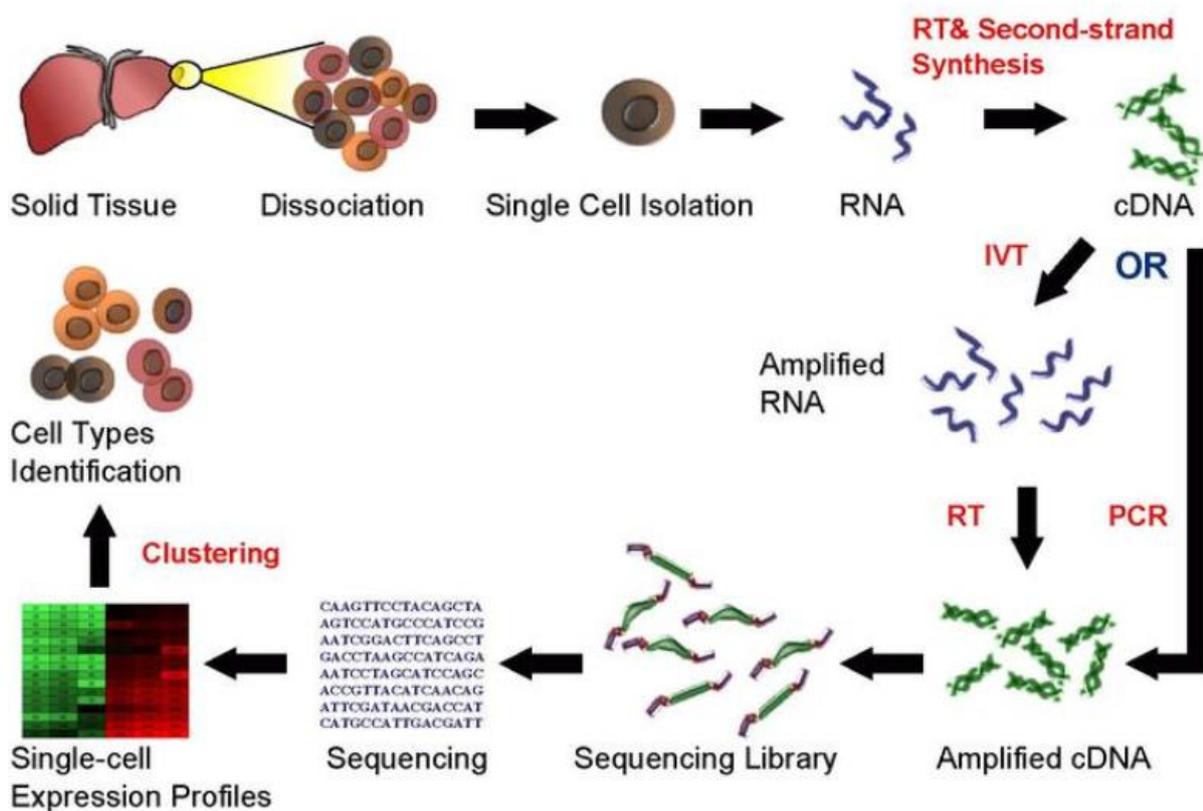
1	Introduction.....	1
2	Literature review.....	3
2.1	Experimental design and quantitative standards.....	3
2.2	Initial quality control and pre-processing of raw reads.....	4
2.3	Gene expression quantification.....	7
2.3.1	Alignment-based approaches.....	7
2.3.2	Alignment-free approaches.....	8
2.4	Quality control and filtering of the expression matrix.....	10
2.4.1	Celloline and cellity.....	12
2.4.2	Scater.....	12
2.5	Normalization.....	14
2.5.1	Principles of global-scaling normalization.....	15
2.5.2	Deconvolution-based scaling normalization and scran-package.....	17
2.5.3	Quantile-regression based normalization and SCnorm.....	18
2.6	Confounding factors.....	19
2.7	Obtaining biological insights.....	20
2.7.1	SC3: Cell type identification by consensus clustering.....	21
2.7.2	Determining cell lineage and differentiation.....	22
3	Objectives.....	24
4	Materials and methods.....	25
4.1	Data.....	25
4.2	Environment and the relevant tools.....	25
4.3	Filtering, normalization and confounders.....	27
4.4	Downstream analyses.....	27
4.5	Comparing pipeline results into the literature.....	27
5	Results.....	29
5.1	The pipeline.....	29
5.1.1	Module 1: Initial quality processing.....	29
5.1.2	Module 2: Quantification, further QC-processing and normalization.....	30
5.1.3	Module 3: Clustering and DE-analysis.....	31
5.2	Initial quality checks and trimming with Module 1.....	31
5.3	Pipeline visualizations for estimating dataset quality before filtering.....	34
5.4	Filtering.....	37
5.5	Checking the features with the highest expression over cells.....	38
5.6	Normalization.....	39

5.7	Confounders .....	40
5.8	Clustering with SC3 .....	41
5.9	Applying tSNE for preprocessed Darmanis .....	45
6	Discussion .....	47
7	Conclusions.....	52
8	References.....	53
	Appendix A: Applying SCnorm for Darmanis data.....	56

# 1 Introduction

Traditional high-throughput RNA sequencing i.e. **RNA-seq** derives genome-wide mRNA expression data from samples consisting of large cell populations. The product is one averaged expression profile to represent the whole biopsy which may easily oversimplify the biological functionality (Bacher & Kendzierski 2016). Thus, the approach is insufficient for studying complex tissues and biological conditions, where the cells have diverse expression profiles (Stegle et al. 2015). **Single-cell RNA sequencing** i.e. **scRNA-seq** aims to solve this problem by recording the expression profiles of the samples that each consist of the amplified mRNA (cDNA) of a distinct cell (Poirion et al. 2016; Bacher & Kendzierski 2016; Stegle et al. 2015). An overview of a typical experiment is presented in Figure 1.

## Single Cell RNA Sequencing Workflow



**Figure 1:** An overview of a scRNA-seq workflow (<https://hemberg-lab.github.io/scRNA.seq.course> 12/16/2017). The cells in a biopsy are gently dissociated and collected in separate pools before extracting the mRNA. After converting RNA into cDNA, the material from each cell is amplified by polymerase chain reaction (PCR) or alternatively, by In vitro transcription (IVT) combined with reverse transcription (RT). Sequencing is followed by complex data processing and downstream analysis to finally obtain biological insights, e.g. identifying the cell-subpopulations.

The progress of scRNA-seq technologies enables diverse new applications. These include for example 1) Recording the diverse expression patterns in embryonic development, 2) Identification of novel cell types and 3) Getting broader insight of transcription control (Arzalluz-Luque et al. 2017; Ofengeim et al. 2017). It also gives new insight for understanding the development and progression of medical conditions such as cancer (Yuan et al. 2017). When the roles of tumor microenvironments can be examined more precisely and the mechanisms behind disease progression and drug resistance are resolved, novel efficient treatments can be developed.

Even though being a powerful approach for biological discovery, the cost is that processing and interpretation of scRNA-seq data is far from trivial (Bacher & Kendzioriski 2016; Poirion et al. 2016; Stegle et al. 2015). The **analysis flow** resembles the traditional RNA-seq data analysis (expression quantification, quality control and normalization followed by downstream analysis for biological insights), but the distinct features of scRNA-seq data require novel approaches. The key challenge in analysis is that scRNA-seq data is **heterogeneous** i.e. there is remarkable level of both biological and technical cell-to-cell variation (Yuan et al. 2017). This makes especially the QC and normalization of the data tremendously challenging. Moreover, the size of the datasets produced by experiments can be enormous: even thousands to tens of thousands samples in a single experiment (Ilicic et al. 2016). The sequencing depth and thus the sizes of sequencing libraries vary from tens of thousands to *millions* of reads per cell (Andrews & Hemberg 2018). Finally, since the field is still quite young, the amount of data and established standards is still marginal. As stated by Andrew&Hemberg, scRNA-seq is not a single method but a collection of protocols designed for various applications that have differing strengths and limitations.

The major focus in this thesis is in the early steps of the analysis flow, especially quality control (QC) and normalization. The objectives are 1) Looking for an optimal way to preprocess scRNA-seq data and assess its quality, 2) Testing and comparing the relevant tools with different publically available datasets and 3) Constructing the analysis pipeline by combining the best approaches and necessary quality metrics. In literature review, some current approaches and relevant insights for scRNA-seq data processing are discussed.

## 2 Literature review

### 2.1 Experimental design and quantitative standards

Even though being outside of the major context of this thesis, it should be noted that experimental design may affect the downstream analysis and data interpretation (Bacher & Kendzierski 2016). For example, the protocol-specific rate of how efficiently the cells are captured as distinct samples and not cell doublets or their multiples may be critical when identifying novel cell types (Andrews & Hemberg 2018). For more detailed discussion of different protocols, see e. g. (Ziegenhain et al. 2017) and (Svensson et al. 2017). Due to the experimental properties, some of the analysis tools are platform-specific. Moreover, there are tools that may be based on certain quantitative standards (Bacher & Kendzierski 2016).

The establishment of **quantitative standards** is still in progress, and there are also differing opinions on their usage. For example, Stegle et al. strongly recommend the usage of the two standards i.e. extrinsic spike-in RNA molecules and unique molecular identifiers (Stegle et al. 2015). On the other hand, they as well as Bacher&Kendzierski mention that even though both have theoretical advantages for normalization and expression estimation, there are some practical challenges and restrictions that have prevented their routine usage (Bacher & Kendzierski 2016).

**Spike-in RNA** refers to synthetic or exogenous RNA that is added into each sequencing library at known concentration, thus working as an **internal control** (Stegle et al. 2015). When used, their sequences should be added into the reference genome or transcriptome prior to mapping (Stegle et al. 2015). Spike-ins help estimate relative differences in RNA content by attributing the differences between the observed and expected expressions of spike-ins to technical errors, which should improve normalization (Bacher & Kendzierski 2016). Normalization with spike-ins is generally done by calculating a cell-specific spike factor that adjusts for the difference, and then applying this factor to endogenous genes (Bacher & Kendzierski 2016).

**Unique molecular identifiers (UMIs)** are short (6-10 nucleotides) and random sequences that are attached to individual molecules of interest before PCR-amplification (Stegle et al. 2015). They make each molecule unique and allow the following of absolute molecular count and further accounting of amplification biases (Bacher & Kendzierski 2016). When UMIs are used, the barcode attached to each read should be removed before mapping (Stegle et al. 2015). In normalization phase, assuming that each cDNA-molecule of the library is observed at least once, the number of UMIs linked to each gene is the direct measure of the number of cDNA

molecules associated with that gene (Stegle et al. 2015). This still does not fully exclude the technical sources of expression variability.

## 2.2 Initial quality control and pre-processing of raw reads

FASTQ-files may contain varying proportions of low-quality reads/sub-reads and synthetic sequences (e.g. UMIs discussed in Chapter 2.1) which are commonly removed before alignment (Bacher & Kendzierski 2016). However, read trimming can be done with several options and stringencies, and currently there seems to be no consensus of the best practice. It was also recently shown that even though read trimming generally improves the mapping rates, aggressive quality-based trimming may distort the expression estimation (Williams et al. 2016).

Some quality filtering of the samples may be done in this early phase of analysis flow, even though the full picture of the quality is obtained only after mapping. The decision of which cells (or reads) to include is not always trivial, even though many preprocessing tools suggest some thresholds for bulk RNA-seq data (Bacher & Kendzierski 2016). The amount of tools specifically developed for scRNA-seq quality control is still small, even though some ready-made pipelines exist (Poirion et al. 2016, Ilcic et al 2016). Suggested individual tools are FASTQC (<http://www.bioinformatics.babraham.ac.uk/projects/fastqc/>, 11/10/2017) or Kraken (Davis et al. 2013), which are used both before and after alignment (Bacher & Kendzierski 2016; Stegle et al. 2015). In general, for example empty sequencing libraries, samples that contain remarkably low amounts of reads as well as the cells that contain suspiciously high amounts of low-quality reads can be discarded prior to expression estimation.

**FASTQC** is an example of generic QC tool which provides diverse within-sample quality information and graphics (see Table 1). It also helps to check whether the low quality samples follow a certain pattern. For example, according to Bacher and Kendzierski, systematic low scores in the beginning of the read may indicate some problem with the run and *should not* be trimmed, while low scores in the read tails indicate a general degradation, in case which trimming may be useful (Bacher & Kendzierski 2016).

**Table 1: Quick review of the modules of FASTQC-tool** (<https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>, 4/11/2018). Quality properties are examined as several modules to check whether the sample passes or causes a warning or a failure according to defined criteria.

<b>FASTQC module</b>	<b>Description</b>	<b>Reasons behind the problems</b>
<b>Per Base Sequence Quality</b>	Plots quality scores across base positions. Warns/fails when lower quartile or the median is below some threshold	Errors may indicate that trimming is needed
<b>Per Sequence Quality Scores</b>	Plots quality score distribution over the reads, showing whether any subset of sequences in the file were problematic. Warns/fails when observed mean quality is below certain threshold	Errors usually indicate a general loss of quality within a run
<b>Per Base Sequence Content</b>	Plots the proportions of A, T, G and C in each base positions. Warns/fails when the difference between A and T, or G and C is greater than certain threshold in any position	Overrepresented sequences, biased fragmentation, Biased composition libraries, too aggressive trimming which causes additional bias
<b>Per Sequence GC Content</b>	Measures the GC content across the whole length of each sequence in a file and compares it to a modelled normal distribution of GC content. Warns/fails if the difference is big enough	Sharp peaks usually indicate a specific contaminant (e.g. adapter). Broader peaks may represent contamination with a different species.
<b>Per Base N Content</b>	Plots the percentage of base calls at each position for which an N was called. Warns/Fails if any position shows an N content greater than certain threshold	General loss of quality, the problematic bin position should be checked for more information.
<b>Sequence Length Distribution</b>	Plots the distribution of fragment sizes in the file. Warns if the reads have differing length and fails if any of the sequences has zero length	Some platforms produce reads with differing lengths. Also trimming affects the lengths
<b>Duplicate Sequences</b>	Counts the degree of duplication for the sequences which first appear in the first 100,000 sequences of the file and plots the relative number of sequences with different degrees of duplication. Warns if the degree is more than 20% and fails if it is more than 50%	Existence of technical (mainly PCR artefacts) and biological duplicates. Problems indicate that the diversity of the library is at least partially exhausted, leading into re-sequencing of the same sequences.
<b>Overrepresented Sequences</b>	Lists the sequence which make up more than 0.1% of the total in the first 100,000 sequences of the file. Warns if such sequences are found and fails if any sequence is found to represent more than 1% of the total.	Easily triggered by small RNA libraries where sequences are not subjected to random fragmentation, and the same sequence is potentially present in a significant proportion of the library
<b>Adapter Content</b>	Warns if any sequence is present in more than 5%, fails if the proportion is over 10%	Libraries with insert sizes shorter than the read length will trigger this module. Failure indicates that the sequences need adapter trimming.
<b>Per Tile Sequence Quality</b>	Shows up if the library contains original sequence identifiers that document the flowcell tile from which each read was originated. Plot shows the deviation from the average quality for each tile, revealing potential problematic positions.	Problems indicate transient or permanent problems such as bubbles or smudges in a flowcell. If majority of the cycles was useful, the file can be used.

A dataset-wide summarization tool for analyzing FASTQC-results is useful, since the potentially large amount of samples make manual checking of each sample impossible or at least time-consuming. An example of this kind of tool is an R-package **fastqcr** (<http://www.sthda.com/english/rpkgs/fastqcr/index.html> 11/10/2017).

**Adapter contamination** refers to the occurrence of experimental add-on sequences that do not provide any biological information (<https://www.ncbi.nlm.nih.gov/tools/vecscreen/contam/>, 12/12/2017). Adapter sequences can be obtained e.g. by FASTQC which detects the overrepresented sequences, or from the sequencing protocol documentations. There are several tools designed for removing this kind of sequences from the raw reads, see e.g. <http://bioscholar.com/genomics/tools-remove-adapter-sequences-next-generation-sequencing-data/> (11/17/2017). One of these tools is **cutadapt**, which can be used to trim adapter sequences, primers, poly-A tails and other types of unwanted sequences (Martin 2011). Cutadapt algorithm calculates optimal alignments between each read and all given adapters/unwanted sequences, and trims the sequence if the **error rate** is below an allowed maximum. The error rate is calculated as the number of alignment errors divided by the length of the matching segment between the read and the adapter.

A wrapper script **TrimGalore!** by Babraham Institute combines FASTQC analysis and cutadapt and is an example of the pipeline that aims to automate the initial quality control procedure ([http://www.bioinformatics.babraham.ac.uk/projects/trim\\_galore](http://www.bioinformatics.babraham.ac.uk/projects/trim_galore) 11/17/2017). It combines both quality and adapter trimming and can be used for any high throughput dataset, both paired and single-end reads. By default, it auto-detects some common adapters (Illumina universal, Nextera transposase and Illumina small RNA adapter sequence), but specific adapter sequences to be trimmed can also be provided. Quality trimming of the read ends based on FASTQC Phred scores can be adjusted. After trimming, the reads shorter than a given threshold can be removed. Unknown bases (Ns) can be removed from either side of the reads.

Krishnaswami et al. mention that deep sequencing tends to produce many duplicate sequences of abundant transcripts (Krishnaswami et al. 2016). These duplicates reduce the ability to detect low-copy transcripts, but cannot be removed, since that would disturb accurate expression quantification. They recommend the evaluation of the sequence duplication proportion so that it can later be used to examine potential impact on low-copy transcripts. A tool that can be used for this is **FASTQ/A Collapser** of **FASTX toolkit** ([http://hannonlab.cshl.edu/fastx\\_toolkit/](http://hannonlab.cshl.edu/fastx_toolkit/), 11/17/2017).

## 2.3 Gene expression quantification

The ultimate goal of **gene expression quantification** is to collect the expression measures of all the genes (or transcripts) in all the samples into one **expression matrix**. In the traditional two-step procedure, the preprocessed reads are first **aligned** against the reference sequence and the mapped reads are then **summarized** with a separate tool to generate the read counts (Engström et al. 2013; Jin et al. 2017; Stegle et al. 2015). Notably, also some **alignment-free approaches** have been recently developed, and they are shown to be remarkably faster than any of the usual aligners, and still have similar or even better accuracy (Patro et al. 2017; Zieleszinski et al. 2017). These tools encompass the tasks analogous to alignment and quantification into a single tool, making the early steps of analysis easier and faster. However, sometimes the exact mapping information is needed in downstream analysis, e.g. when looking for novel transcripts, and in such cases, alignment-free tools cannot be used.

Quantification approach is one of the several factors that may influence on the subsequent analysis and interpretation of the data, since alignment tools differ in performance and accuracy. Overall consensus of an optimal method still does not exist, and there are no approaches designed specifically for scRNA-seq experiments (Jin et al. 2017; Poirion et al. 2016). However, the fast increase in data sizes already challenges both storage and processing capacities of computers (Zieleszinski et al. 2017). Since scRNA-experiments may produce remarkably large datasets, the memory-effectiveness is inevitably a crucial feature when trying to resolve the computational bottleneck in gene expression quantification.

### 2.3.1 Alignment-based approaches

An alignment-based method for transcriptomic data should solve two computationally expensive problems (Dobin et al. 2013). The first is, like in any alignment task, how to accurately align reads containing mismatches and **indels** i.e. insertions or deletions. The second and transcriptomics-specific problem is, how to deal with the reads containing sequences from non-contiguous genomic regions i.e. from various exon-exon boundaries resulting from RNA splicing. Moreover, as a result of genomic evolution there may be multiple identical or related genomic regions that are all transcribed: how to align the reads that map in multiple sites in the genome?

There is a broad amount of alignment tools to choose from: for example Engström et al. evaluated as many as 26 mapping protocols based on 11 programs and pipelines (Engström et al. 2013). A common feature for all alignment-based programs is that they look for

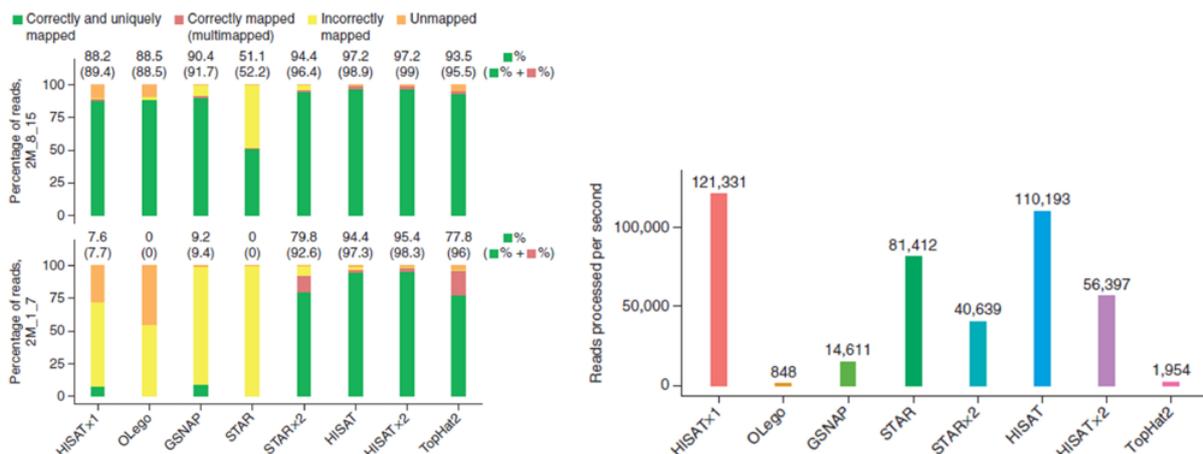
correspondence of individual bases that are in the same order in the sequences they are comparing (Zielezinski et al. 2017). They assume that every character of the sequence can be categorized at least into match or mismatch, and most approaches also recognize indels.

Based on the comparative analysis by Engström and colleagues, **STAR** (Spliced Transcripts Alignment to a Reference, see (Dobin et al. 2013)) seemed to outperform most of the other approaches based on different performance benchmarks. On the other hand, Kim et al. later showed that **HISAT** (Hierarchical Indexing for Spliced Alignment of Transcripts) was the most efficient when using two simulated datasets, as seen in Figure 2 (Kim et al. 2015). However, it should be noted that sequence alignment depends on multiple a priori assumptions about the evolution of the sequences, and these with other parameters are user-defined and therefore more or less artificial (Zielezinski et al. 2017). Modifying parameters, as well as using another aligner program with potentially differing scoring system may lead in very different results, and this makes the choice of best or most accurate approach quite challenging.

After the alignment, the gene counts are generated with the tool of choice, for example HTseq or cufflinks (Bacher & Kendzioriski 2016; Ilicic et al. 2016). Useful functionality may also be provided by the aligner, for example STAR has the option “quantMode” which generates the read counts. However, the layout of these files may differ from the commonly used alternatives, and may require some additional formatting.

### 2.3.2 Alignment-free approaches

Recently it has been found out that exact alignments of the reads against the reference genome are not crucial for the reliable estimation of transcript levels (Zielezinski et al. 2017). Instead,



**Figure 2: Comparison of some most common RNA-seq aligners (Kim et al. 2015). Two simulated datasets were used to measure mapping quality and the alignment speed, respectively.**

the reads can be compared against the **reference transcriptome** and find the transcripts from where the queries have *most probably* originated from (Jin et al. 2017; Srivastava et al. 2016). Most of these kind of **alignment-free algorithms** are either **word-frequency based** methods that calculate the frequencies of subsequences (**k-mers**) of a defined length, or **information-theory based** methods that evaluate the informational content between full-length sequences (Zielezinski et al. 2017).

Programs **kallisto** (Bray et al. 2016; <https://pachterlab.github.io/kallisto>, 11/9/2017) and **Salmon** (Patro et al. 2017; <http://salmon.readthedocs.io/en/latest/salmon.html> 11/9/2017) are examples of recently developed alignment-free approaches for expression quantification. They require much less computational capacity than traditional aligners, and are thus particularly useful with large datasets. In addition to expression quantification, alignment-free methods have also been developed for other tasks such as variant calling (Zielezinski et al. 2017).

**Pseudoalignment** algorithm kallisto builds a deBruijn graph from the k-mers of the reference and, for each read, uses that graph to define an **equivalence class** i.e. a multi-set of transcripts associated with the read (Bray et al. 2016). From these pseudoalignments, the transcript abundances are then quantified with a likelihood function which is iteratively optimized with the **Expectation-Maximization (EM)** algorithm. Moreover, the uncertainty of abundance estimates can be quantified by bootstrapping, and a model for correcting sequence-specific bias can be used. The program requires only the k-mer length and the mean of the fragment length distribution for expression quantification. K-mer length applied for the reference must be set large enough to avoid mappings of random sequences but short enough for error-robust mapping. Recently, the developers have updated kallisto with a mode specifically developed for scRNA-seq experiments (<https://pachterlab.github.io/kallisto/singlecell.html>, 4/8/2018).

**Quasi-mapping** algorithm Salmon is also based on k-mers, and is built on statistical (Bayesian) inference procedure which enables the building of a probabilistic model of the experiment to quantify the reads (Patro et al. 2017). The position and orientation of all mapped fragments are tracked and combined with the calculated abundances, resulting in **per-fragment conditional probabilities**. According to Patro et al, Salmon is the first available method which has sample-specific models for positional, sequence-specific and GC-dependent biases. Similar to kallisto, the uncertainty of abundance estimates can be assessed. In addition to **the quasi-mapping mode**, salmon also has the **alignment-based mode** than can be used to quantify the expression

based on the mapping results (BAM-files) of some other aligner. However, also with this mode the reads has to be originally aligned against a transcriptome, not a genome.

The principles of kallisto and Salmon are similar, even though underlying algorithms are different. The final output for both contains the counts presented as **transcripts per million (TPM)**, meaning that the feature counts are first divided by the length of each feature in kilobases, and the results are then divided by a sample-specific “per-million” scaling factor. Moreover, a set of additional useful information e.g. bootstrapping and bias correction information and mapping metadata is provided.

## 2.4 Quality control and filtering of the expression matrix

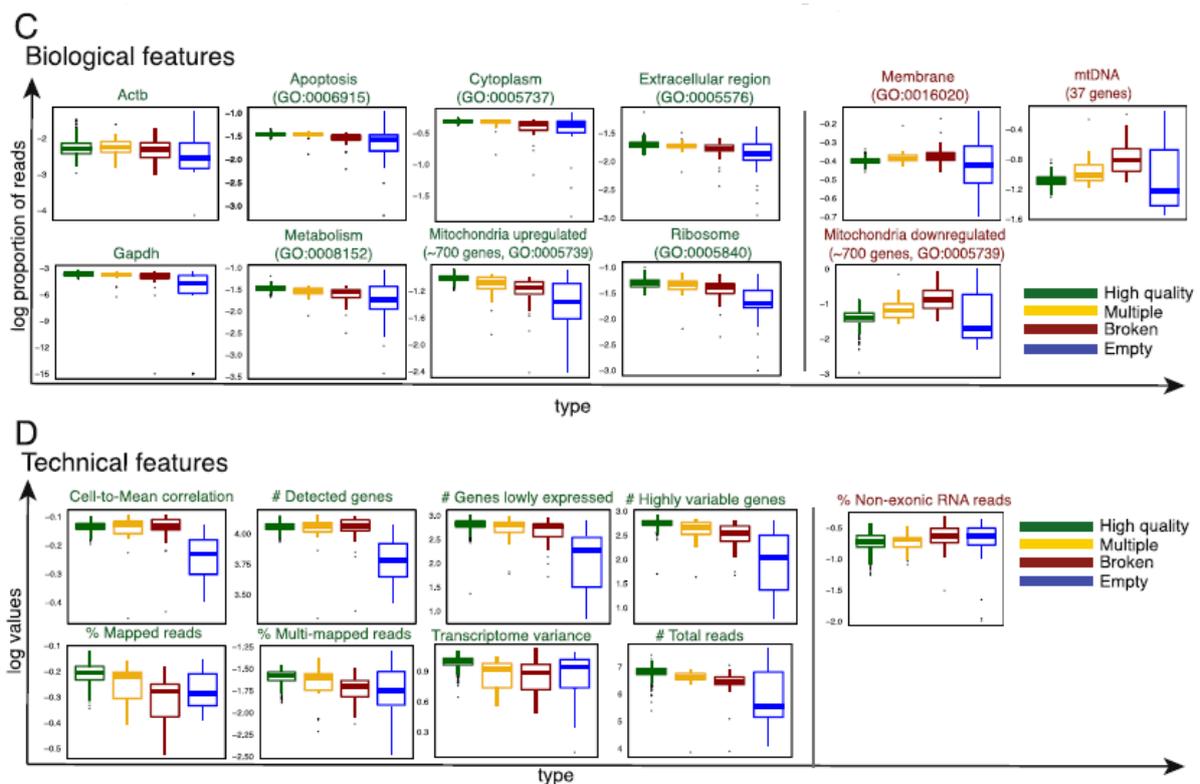
The final product of gene expression quantification is the **expression matrix** containing  $n$  features and  $p$  samples together with various metadata of the mapping. Based on this data, the low-quality samples should be removed from subsequent analysis. **Low-quality** in scRNA-seq samples refers to 1) stressed/broken cells with degraded RNA or aberrant expression patterns, 2) samples that contain material from more than one cell (sometimes referred as **doublets**) and 3) empty samples resulted from either failed cell isolation or failed/premature cell lysis (Bacher & Kendzioriski 2016; Ilicic et al. 2016).

Ready-made pipelines for QC exists, but they are not necessary well generalized: they may be platform-specific or compatible only with certain tools (Poirion et al. 2016). As an alternative, there are couple of toolkits for tailored analysis, providing approaches not only for QC but also further steps in analysis flow. Examples are **Seurat** (Satija et al. 2015) and **scater** (McCarthy et al. 2017). Each approach typically has a different representation for the data. For example, Seurat stores the data in its own *Seurat*-class, while scater uses *SingleCellExperiment*-class by Lun & Risso (<http://bioconductor.org/packages/release/bioc/html/SingleCellExperiment.html>, 5/3/2018).

Examining the technical features, like how well RNA was captured and amplified from each cell, is extremely important in scRNA-seq experiments (Bacher & Kendzioriski 2016; Stegle et al 2015). One this kind of quality metric is the **fraction of mapped reads** back to genome: low rates may indicate RNA-degradation, contamination or inefficient cell lysis. If spike-ins were used, the ratio of the number of **reads mapped to endogenous RNA** to the number of **reads mapped to extrinsic spike-ins** should be calculated. If this ratio is small (i.e. there is a high proportion of reads mapped to spike-ins), the capturing of the decent proportion of mRNA-content of the cell has potentially failed.

The factors that should be taken into account in scRNA-seq QC are very comprehensively discussed by Ilicic et al 2016. They investigated both biological and technical features that differ in normal and low-quality cells (see Figure 3). For example, they showed that the low-quality cells have higher average gene expression in specific functional GO-categories and also noisier expression profiles in these categories. Genes relating to GO-terms *Cytoplasm*, *Metabolism* and *Membrane* were generally downregulated in broken cells, while *Mitochondrially encoded genes* and *Mitochondrially localized proteins* seemed upregulated. The loss of cytoplasmic content of damaged cells before the experimental cell lysis and resulting increase in the relative proportion of mitochondrial content is suggested as explanation for mitochondrial upregulation.

Aevermann et al. have also investigated approaches for quality classification of single-cell and single-nuclei RNA-seq samples, and found differing results compared to Ilicic et al. (Aevermann et al. 2016). According to their random forest -based analysis, they suggest that metrics such as the ratio of trimmed reads over raw reads and percent of duplicated reads could be useful as well. They concluded that since there are probably more than one types of failing samples, the advanced machine learning methods might be the solution to learn the different



**Figure 3: Summary of interesting biological and technical features that define sample quality (Ilicic et al. 2016).** GO-categories labelled with green indicate upregulation in high-quality cells, while red labelling indicates upregulation in low-quality cells.

patterns they follow. All in all, it seems that there is still much to learn of scRNA-seq QC. Some current tools and approaches to deal with QC is next discussed.

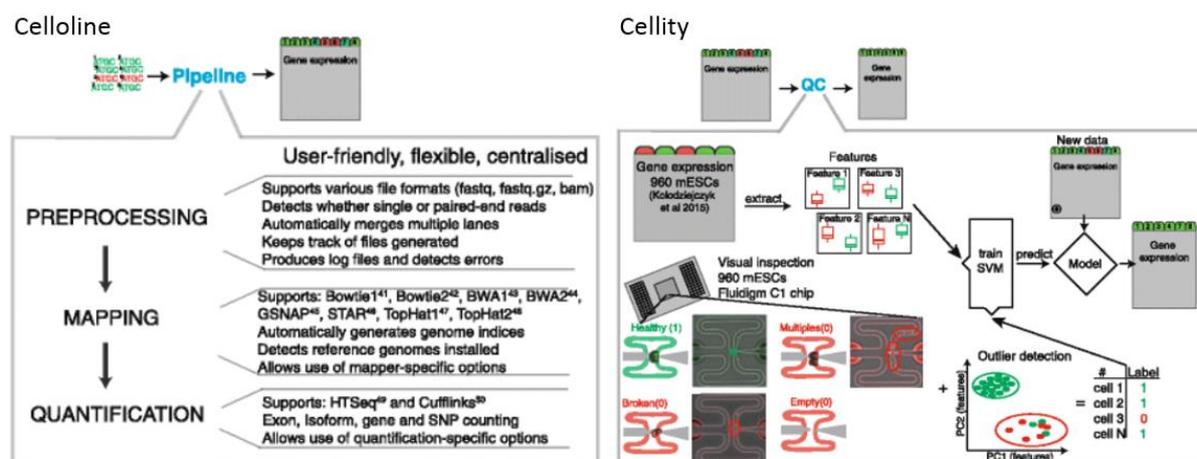
#### 2.4.1 Celloline and cellity

**Celloline** is one of the few quite generalized pipelines designed specifically for the early processing steps scRNA-seq data (Ilicic et al. 2016). It is a python pipeline that combines preprocessing, mapping and quantifying and has the capacity to process large datasets (see Figure 4). Currently supported mapping algorithms are *bowtie*, *bowtie2*, *STAR*, *BWA*, *gsnap* and *salmon*, and supported quantifying tools are *tophat*, *htseq-count* and *cufflinks* (<https://github.com/Teichlab/celloline>, 12/13/2017).

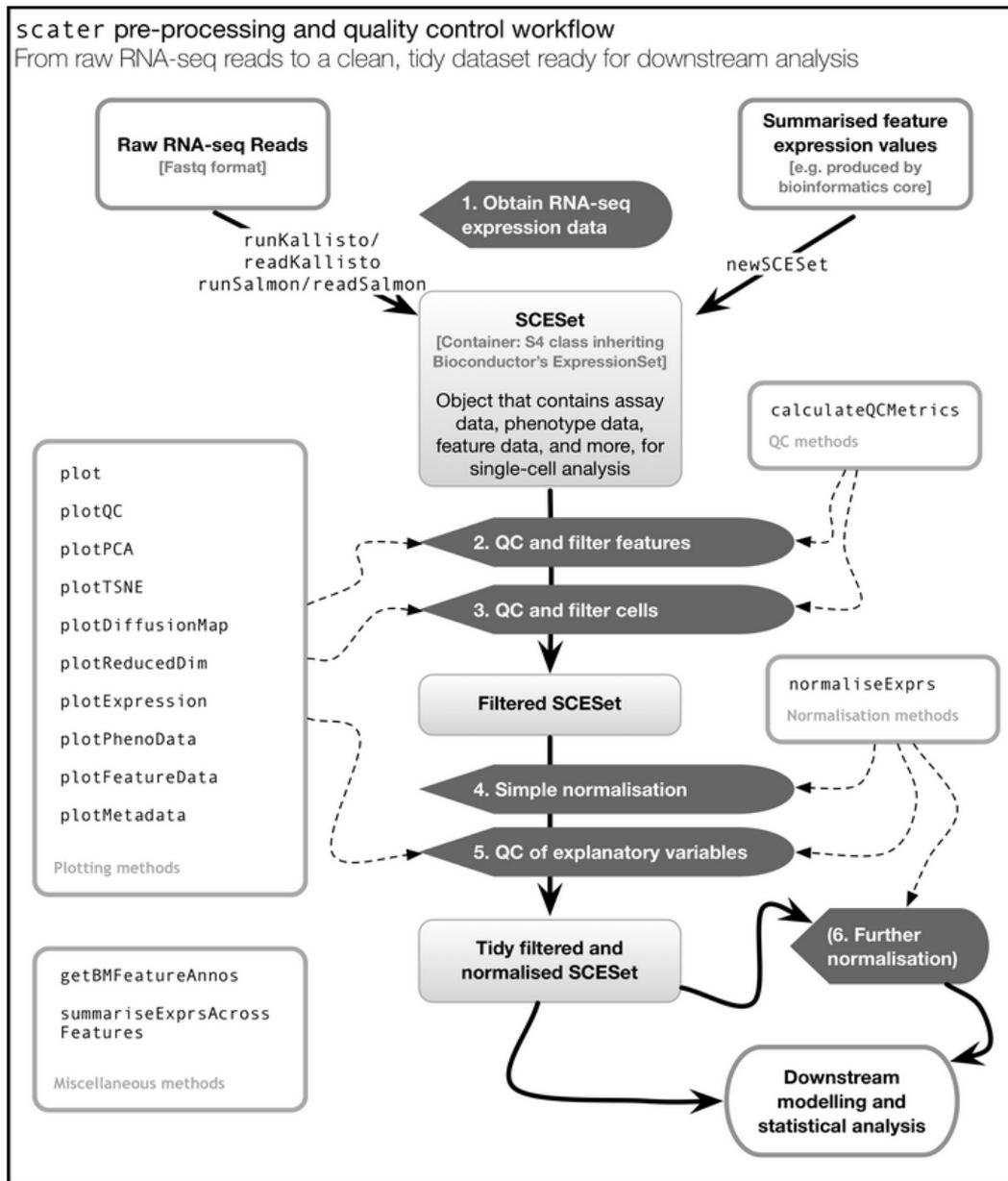
The output of celloline consists of two parts: gene expression matrix and read statistics matrix (Cells x Metrics), which can then be further analyzed e.g. with R-package **cellity** (<https://github.com/Teichlab/cellity>, 12/13/2017) which was developed together with celloline. Cellity contains the functionality for **support vector machine (SVM)** based quality classification of the cells.

#### 2.4.2 Scater

R/Bioconductor package scater is a recent tool that contains a platform and diverse set of functions for scRNA-seq data quality control, visualization and normalization (McCarthy et al. 2017). An overview of the functionality provided by scater is seen in Figure 5. Scater provides approaches to conduct all the most relevant steps in scRNA-seq preprocessing and monitoring data quality with the diverse plotting methods.



**Figure 4: Classification and quality control of cells by celloline and cellity (Ilicic et al. 2016).** The pipeline implements the early processing steps for scRNA-seq data in a generalized approach, and the R-package cellity can then be used for quality control. Low-quality cells or reads are marked in red.



**Figure 5: Single-cell pre-processing and quality control with scater (McCarthy et al. 2017).** The R-package provides diverse functions that can be combined into a single workflow. Note that SCESet-object is replaced by SingleCellExperiment-object in current version.

Especially useful are the functions for reading the data produced by Salmon and kallisto from list of directories, and summarize them all into one object. Scater even has the wrapper functions for running both Salmon and kallisto in R. The data is stored in a **SingleCellExperiment** (SCE) object (**SingleCellExpressionSet/SCESet** –class in earlier versions) which is specifically designed for scRNA-seq data, and transcript read counts can be easily summarized into gene counts within the workflow. Even though the package is built strongly compatible with the alignment-free methods, a new SCE-instance can be also built manually to store the quantification results of other approaches

(<http://bioconductor.org/packages/release/bioc/html/SingleCellExperiment.html>,5/3/2018).

Some additional metadata, for example annotations, can also be retrieved by scater.

After calculating diverse cell-specific and sample-specific QC-metrics (many of them being similar as defined by Ilicic et al 2016), uninterested features and low-quality cells can be filtered out. The features of interest may depend on the experiment, and should require case-by-case consideration. However, the genes that are constitutively silent across all cells i.e. **systematic zeros** is safe to drop already before normalization (Lun et al. 2016a).

Scater provides two main approaches for filtering low-quality cells. Outliers for a given metric (e.g. library size or proportion of mitochondrial RNA/spike-ins, mapping rate etc.) can be determined based on user-adjusted **median-absolute deviation (MAD)** thresholds. MAD is the average distance between each data point and the mean and describes the variation in a data set. Alternatively, an automatic outlier detection using multivariate normal methods can be used: basically, the **principal component analysis (PCA)** is applied for the given set of QC-metrics, and the outliers found by this approach should represent low-quality cells with abnormal library characteristics (McCarthy et al. 2017). Poor-quality cells may also sometimes form a second, distinct cluster (Stegle et al. 2015).

Scater provides some simple data normalization approaches (TPM, CPM, FPKM) for **scaling normalization** to remove biases caused by differences in sequencing depth, capture efficiency and other factors between samples, and **batch effect correction** (McCarthy et al. 2017). However, most scaling normalization methods are originally designed for bulk-RNA-seq data, and thus relying only to scater approaches may be insufficient. Scater is compatible with many tools specifically designed to account for scRNA-data normalization.

## 2.5 Normalization

**Normalization** refers to the scaling of the data so that the information of the samples can be reasonably compared as well as correcting uninformative biological and technical variation in data to ease the detection of interesting signals. Simple normalization for sequencing depth can be done e.g. by calculating TPM (Bacher & Kendzioriski 2016). However, in addition to this kind of within-sample normalization, between-sample normalization is essential for downstream analysis.

Normalization has major effect on the interpretation of the data. When compared with bulk experiments, the contents of scRNA-seq matrix is heterogeneous and significantly sparser and noisier, which sets special challenges and makes the traditional normalization methods

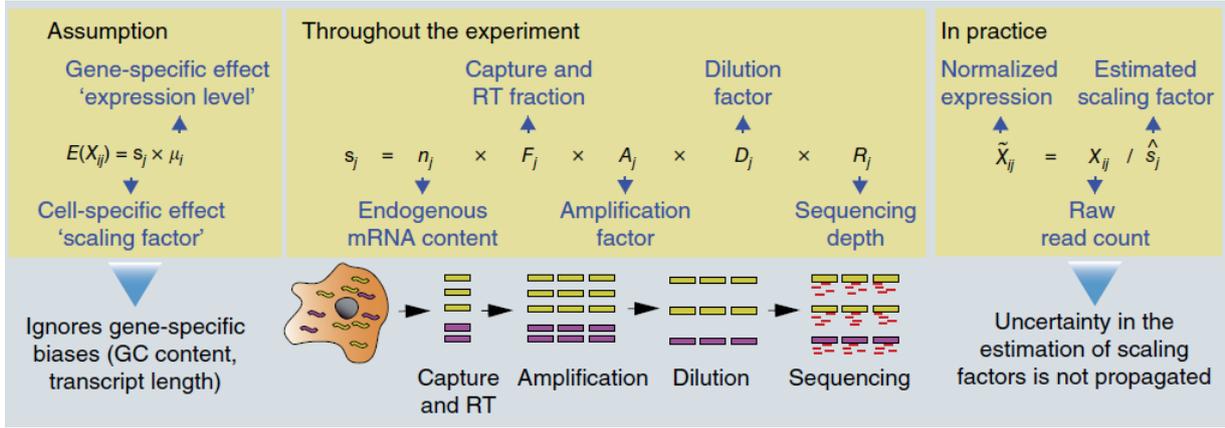
potentially misleading, because the assumptions they make of the data are not necessarily valid (Bacher & Kendzioriski 2016; Stegle et al. 2015; Vallejos et al. 2017). Regardless, the same methods are widely used (Vallejos et al. 2017). The challenging features of scRNA-seq data are next discussed more precisely, and some approaches designed to deal with them is then reviewed.

**Sparsity** i.e. the high proportion of zeros in expression matrix arises for both biological and technical reasons. For example, certain genes may not be expressed in different cell types or because of the transient states of the cells (Vallejos et al. 2017). Alternatively, it is possible to lose the signals of the genes that are actually expressed during the experiment: some transcripts may be lost e.g. in reverse transcription or PCR amplification phases, or in detection (Yuan et al. 2017). These are called **dropout events**.

The origins of **noise** are likewise various and their sources may be difficult to detect (Bacher & Kendzioriski 2016; Vallejos et al. 2017). For example, when PCR-amplifying the scarce starting material obtained from each cell before sequencing, **amplification biases** may distort the original expression profile (Yuan et al. 2017). As a result, some transcripts that have optimal amplification properties may appear abundant and vice versa, even though their proportions were originally rather different. Experimental design, e.g. changes in culturing, capturing and sample preparation may lead in technical variation between cell populations, referred as **batch effects** (Stegle et al. 2015; Yuan et al. 2017). Sequencing platform may also produce certain **lane effects**.

### 2.5.1 Principles of global-scaling normalization

In short, global-scaling approaches are **between-sample normalization** methods that produce the normalized expression counts by dividing the raw read count of each cell by the estimated **scaling factor** (a.k.a. **size factor**) (Stegle et al. 2015; Vallejos et al. 2017). They are calculated with different methods depending on the normalization algorithm, and there is no consensus of the best approach (Vallejos et al. 2017). Even though being common approach in scRNA-seq, many global-scaling algorithms make assumptions that do not perform well with sparse scRNA-matrix, since the algorithms were originally developed for bulk RNA-seq. However, some scRNA-seq specific approaches have recently been developed, for example **deconvolution based size factor calculation** by Lun et al., which is described more precisely in chapter 2.5.2, and **BASiCS** (Bayesian Analysis of Single-Cell Sequencing data) (Vallejos, et al. 2015), which relies on spike-ins.



**Figure 6: Global scaling normalization for scRNA-seq data sets (Vallejos et al. 2017).**  $X_{ij}$  is a random variable representing the read count of a gene  $i$  in cell  $j$ . In order to normalize the expression levels, the scaling factor  $s_j$  needs to be estimated for each cell.

Vallejos et al. quite nicely visualized (Figure 6) and explained all the features affecting to the size factor  $s_j$  in scRNA-seq experiments, and which ideally should be taken into account (Vallejos et al. 2017). For simplicity, only the homogeneous cell population is examined, even though the interpretation should be valid also for heterogeneous populations, assuming that most of the genes are not differentially expressed and the number of upregulated and downregulated genes is roughly equal.

As seen in Figure 6, the scaling factor  $s_j$  should adjust for factors such as endogenous mRNA content of the cell ( $n_j$ ) depending from e.g. cell state before the lysis, the differing capture and RT efficiency ( $F_j$ ) after the lysis and variability in amplification efficiency ( $A_j$ ). Moreover, since a fixed number of reads are distributed between genes, the samples are subsequently diluted by a cell-specific factor  $D_j$ . Vallejos et al present two options to  $D_j$  interpretation. The first is **library quantification** (see equation 1), which aims to set the  $D_j$  so that a library contains the same number of molecules from each cell. In this case

$$D_j = m / (n_j F_j A_j) \quad (1)$$

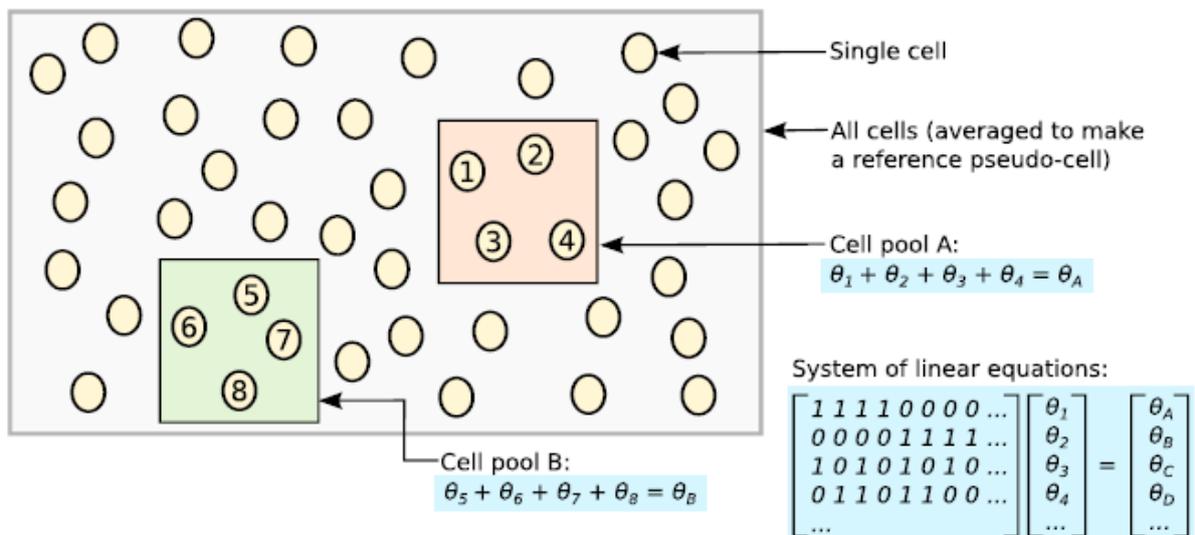
where  $m$  is the desired number of molecules per cell. Without library quantification,  $D_j$  is the proportion of amplified molecules used to prepare the sequencing library. Finally, the **sequencing depth** i.e. the number of sequenced reads per molecule from each cell ( $R_j$ ) varies stochastically.

## 2.5.2 Deconvolution-based scaling normalization and scran-package

As a result of the diverse nature of scRNA-seq experiments (i.e. several different platforms, whether or not spike-ins and/or UMIs are used etc.), it is hard to find a generalized method for normalization (Vallejos et al. 2017). Lun et al. aimed to develop a method that is not dependent on spike-ins and could be applied to all datasets (Lun et al. 2016a). This kind of method should recognize **stochastic zeros** that are found in actively expressed and thus informative genes due to sampling stochasticity, and **semi-systematic zeros** where the gene is silent only in certain cell sub-population(s) (Lun et al. 2016a). They developed a **deconvolution method** by combining the three commonly used scaling-normalization methods originally developed for bulk RNA-seq (DESeq, TMM and size factor normalization) and aiming to correct the assumptions that are not valid with scRNA-data. A visual description of the method is seen in Figure 7.

There are five key steps in the deconvolution method by Lun et al. First, a pool of cells is defined by clustering. Second, the expression values are summed across the pool. Third, the pool is normalized against an average reference built from the averaged expressions of the whole dataset. Next, the same is repeated to different pools of cells to construct a **linear system**. Finally, the pool-based size factors are deconvolved to their cell-based counterparts.

The method is available as a function in Bioconductor/R-package **scran** (<http://bioconductor.org/packages/release/bioc/html/scran.html>, 2/3/2018). Scran implements a



**Figure 7: Deconvolution-based normalization (Lun et al. 2016a).** Instead of using same size factor across the samples for scaling, the cells are first clustered and an optimal factor is calculated for each of the clusters with summing expression values ( $\theta_i$ ), normalizing against a reference cell and constructing linear system of equations. Cell-specific factors are finally deconvolved from the pool-based factors.

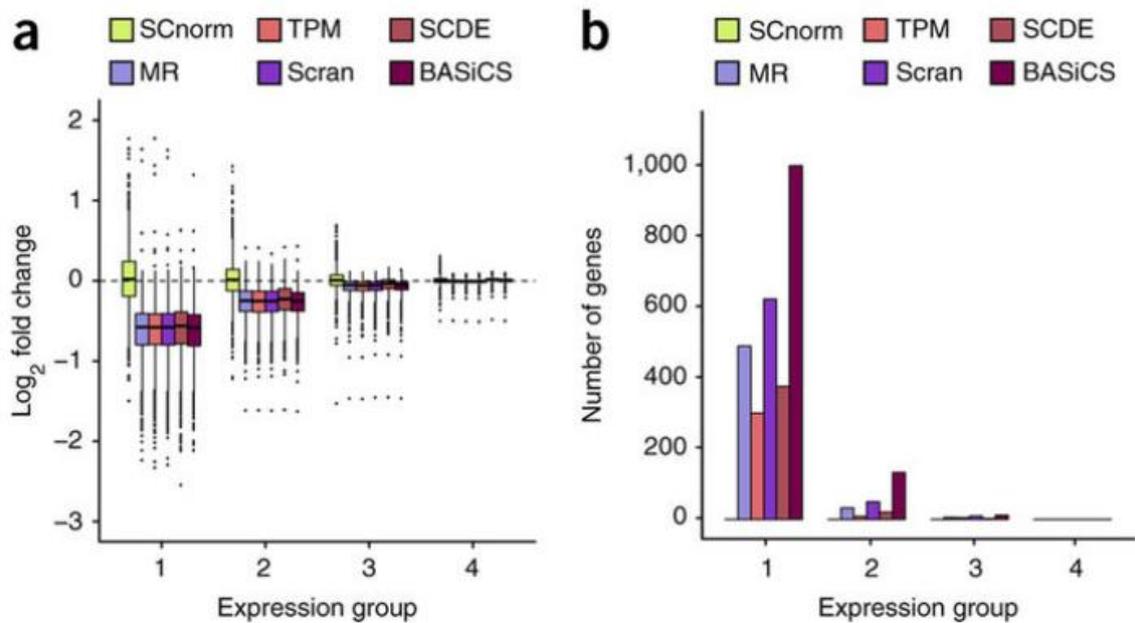
variety of low-level analyses of scRNA-seq data, such as normalization of cell-specific biases, assignment of cell cycle phase and detection of highly variable and significantly correlated genes.

### 2.5.3 Quantile-regression based normalization and SCnorm

Bacher et al. state that global-scaling approaches, also the ones designed to scRNA-seq data, are still insufficient to accommodate with the **count–depth relationship** in scRNA-seq data (Bacher et al. 2017). By this they refer to the systematic variation in the relationship between transcript-specific expression and sequencing depth. It cannot be explained accurately by a single scale factor common to all genes in a cell, since the relationship is not necessarily common across the genes. This is why scaling-factor normalization may lead to **overcorrection** for weakly and moderately expressed genes or **undernormalization** of highly expressed genes.

To solve the problem, they developed novel between-sample normalization method **SCnorm** (<https://bioconductor.org/packages/release/bioc/html/SCnorm.html>, 2/3/2018) which uses quantile regression to estimate the dependence of transcript expression on sequencing depth for every gene. Genes with similar count-depth relationship are pooled, and the second quantile regression is used to estimate scale factors within each subgroup. Finally, the within-group adjustment for sequencing depth is performed using the estimated scale factors to provide normalized estimates of expression. In addition to this kind of between-sample normalization, SCnorm package also implements some within-sample normalization approaches to adjust for gene-specific biases.

Bacher et al evaluated SCnorm by applying it for both simulated and real datasets, and the results were compared with five other normalization methods (see an example in Figure 8). Some remarkable differences were shown. According to the results, Bacher et al. suggest that the former normalization algorithms easily lead in false detections of DE-genes in downstream analysis. However, since the performance of any tool is typically affected by the properties of the data, the results may also vary. Bacher et al. recommend the usage especially with the data having groups with significantly differing count-depth relationships. SCnorm-package contains a function *plotCountDepth()* for the visualization of this relationship in detected expression groups which can also be used in the evaluation of normalization results.



**Figure 8: Log-fold changes (panel a) and numbers of DE-genes detected by MAST (panel b) after normalization by different tools (Bacher et al. 2017).** In panel b, genes are divided into four equally sized expression groups based on their median among nonzero un-normalized expression measurements, and results are shown as a function of expression group.

## 2.6 Confounding factors

Biological experiments always have sources of unwanted variation called **confounding factors**. Batch effects and other technical factors are most typical confounders, and depending on the biological interests of the experiment, also some biological factors may distort the interpretation of the data, if not taken into account (Stegle et al. 2015). For example, when detecting subpopulations in a heterogeneous tissue, the cells in differing stages of the cell cycle may affect the clustering, and thus the cell cycle effects are considered as confounders. On the other hand, experiments that aim to **pseudotemporal ordering** i.e. the construction of differentiation paths from a cell population, may consider cell cycle effect as a signal of interest (Bacher & Kendziorski 2016).

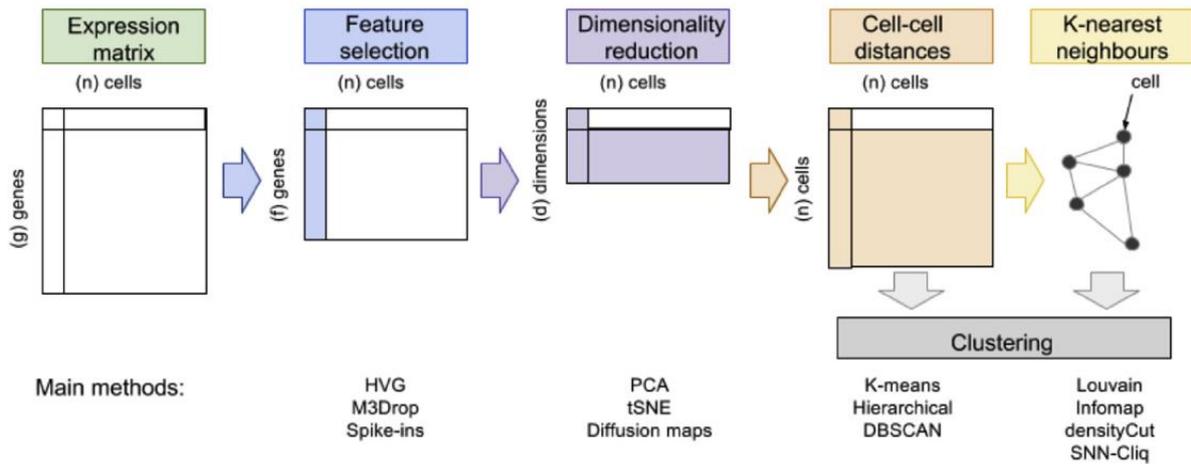
Chen & Zhou reviewed some statistical unsupervised and supervised methods recently developed to infer confounding factors (Chen & Zhou 2017). **Supervised methods** are application specific and restricted only to experiments where the primary variable of interest is known. An example of this kind of tool is **OEFinder** developed for Fluidigm C1 platform, which is designed to correct the expression variation correlated with capture sites with small or large plate output IDs, named as the **ordering effect (OE)** (Leng et al. 2016).

**Unsupervised methods** are more generalizable and can be further divided into two subcategories (Chen & Zhou 2017). The first one includes approaches such as PCA and **linear mixed models (LMM)** which treat all genes similarly. In contrast, the methods of the second category divide the genes into a *control set* (consisting of e.g. spike-ins or certain cell cycle genes) and a *target set*, which are treated differently. The confounding factors are inferred from the control set, which contains only genes known to be free of the effects of interest, and is then used to remove the confounding effects in the target set. An example of this kind of tool is **scLVM** (single-cell latent variable models) (Buettner et al. 2015). A recent tool **scPLS** (single cell partial least squares) by Chen & Zhou combines approaches from both categories of unsupervised methods.

In conclusion, due to the diverse nature of confounders and experiments, there is not any universal tool for the problem. For example scater-package can be used for simple **batch effect correction**, utilizing the PCA (McCarthy et al. 2017). Also scran package has a PCA-based function for de-noising the matrix. In PCA-based approach, the principal components of each gene in each sample are extracted to represent the confounding factors (Chen & Zhou 2017). These are then treated as covariates whose effects are further removed from gene expression levels. For more sophisticated methods, the tool to be used needs case-by-case consideration.

## 2.7 Obtaining biological insights

After normalization, the analysis flow is branched depending on what are the biological interests of the experiment and, what are the confounding factors. The goal may be e.g. cell type identification, cell type characterization or construction of gene regulatory networks, and Stegle et al. state that bulk RNA-seq tools can be generally used for these analyses (McCarthy et al. 2017; Stegle et al. 2015). There are also already several statistical methods and software tools developed for obtaining this kind of insights specifically from scRNA-seq data (see e.g. Table 1 in (Bacher & Kendziorski 2016)). Most of them assume that the data is already preprocessed and normalized, while some provide built-in normalization approaches.



**Figure 9: Workflow for cell type identification after normalization (Andrews & Hemberg 2018).** Feature selection for normalized expression matrix is followed by dimensionality reduction, distance calculation and clustering

The downstream analysis flow, e.g. **cell-type identification** presented in Figure 9 further consists of smaller tasks which can be completed in varying, and not necessary mutually exclusive approaches. In this chapter, couple of interesting applications that implement the downstream analysis flows are shortly discussed as a final part of the literature review.

A general issue in most downstream analyses is, that the high resolution of scRNA-seq experiments leads in high **dimensionality** e.g. number of features (rows) in expression matrix. Even though this kind of resolution is very powerful property for biological discovery, it comes with the cost called “**curse of dimensionality**” (Andrews & Hemberg 2018). The higher dimensionality, the harder to find significant differences between the samples i.e. distinguish the differences between populations from the variability within a population. Moreover, the computational capacity needed to process the data is correlated with dimensionality. **Feature selection** i.e. dropping out the “uninformative” genes and **dimensionality reduction** i.e. projecting the data into a lower dimensional space by prioritizing certain properties are main approaches to deal with the problem. According to Andrews&Hemberg, it is a common practise to apply several of them in a single analysis flow.

### 2.7.1 SC3: Cell type identification by consensus clustering

Identification of cell populations is an example of unsupervised clustering problem, which is one of the main fields in machine learning and also among the most popular goals of scRNA-sequencing (Andrews & Hemberg 2018). The amount of possibilities how to divide a large set of cells into different clusters is enormous, and in order to find a solution in a feasible time, all clustering algorithms make certain assumptions and approximations of the data. For example k-means clustering algorithm which is a part of **SC3-method** presented by Kiselev et al.

assumes the clusters to be roughly similarly sized and spherical (Andrews & Hemberg 2018; Kiselev et al. 2017). Obviously, when a clustering algorithm is used for data that does not follow these assumptions, the result is not reliable, and thus there is no general solution for clustering. Several approaches are more precisely reviewed by Andrews&Hemberg.

**K-means** clustering looks for k clusters that are found based on k **centroids** i.e. points that represent the centers of each clusters. The first centroids are artificially picked, after which the algorithm iteratively calculates the average of the points that are near to each centroid, and sets these average values as the new centroids. When the centroids stay stable, the solution is found. Naturally, the clustering results may differ depending on how the original centroids were chosen. SC3 aims to increase the classification accuracy by **consensus approach** where k-means clustering is repeated several times and the outcomes are collected into a matrix which summarizes how often each pair of the cells is located into a same cluster (Kiselev et al. 2017).

SC3 clustering by Kiselev et al. has five key steps: gene filtering, distance calculations (Euclidean, Pearson and Spearman), transformation of distance matrices, k-means clustering and computing a consensus matrix by cluster-based similarity partitioning algorithm (CSPA). Finally, the consensus matrix is clustered by hierarchical agglomerative clustering, and inferred into user-defined amount (k) of clusters. Each step has some adjustable parameters, which are, as default, set to the values found optimal by running the algorithm with six “gold-standard” datasets. According to Kiselev et al., these default values seemed to perform well when testing them with six additional public datasets, and finally, when applying it to a clinical data. SC3 is available as a Bioconductor/R package and is compatible with scater. Naturally, the increased classification approach comes with some computational cost, but computation can be sped up by parallelization.

### 2.7.2 Determining cell lineage and differentiation

In sequencing experiments, cells are potentially captured in unsynchronized stages of differentiation and cell cycle phases. This may be problematic for many scRNA-seq downstream analyses, but also enables insights that are not possible with bulk experiments. An example of novel downstream direction is **pseudotemporal ordering** i.e. computational reconstruction of cell differentiation path from the data, and also other dynamic cellular processes such as proliferation may be targets of interest (Bacher & Kendzioriski 2016; Trapnell et al. 2014). When differentiation states of the cells are known, it is possible to e.g. predict the

dynamic genetic networks in larger scale biological processes such as organogenesis, repair and disease (Guo et al.2017).

The first remarkable method for determining differentiation pathways was **Monocle** by (Trapnell et al. 2014), and since then, several other approaches have been developed. According to Bacher & Kendziorski, most of them perform some type of dimensionality reduction followed by algorithms from **graph theory**, aiming to order cells so that the distance between them, determined by gene expression, is minimized. Guo et al. state that the methods that pseudotemporally classify cells based merely on transcriptome similarity require external knowledge, e.g. time information, cell identity or marker gene expression in order to determine the start and end points of dynamic processes, and thus cannot be always applied. As a solution, they developed a novel algorithm **SLICE** (Single Cell Lineage Inference Using Cell Expression Similarity and Entropy), which is based on the calculation of single-cell entropy (**scEntropy**) and the perception that the entropy inversely correlates with cell differentiation state (Guo et al. 2017). Recently, also (Teschendorff & Enver 2017) have confirmed the potential of entropy in studying cell differentiation.

Entropy can be seen as a measure of cellular heterogeneity: low entropy corresponds to narrow, well-defined gene expression patterns with strict regulatory constraints while high entropy corresponds to broad, diverse patterns of expression with weaker regulatory constraints (Guo et al. 2017). Thus, ‘entropy’ in this context refers to the multiple potentials or uncertainty in a biological system instead of e.g. the noise or disorder in gene expression. In short, SLICE calculates scEntropy by computing pairwise gene functional similarities based on Gene Ontology annotations, and then clusters the genes based on their functional similarity. These clusters are then used together with calculated expression similarity clusters to construct a cell network, in which the neighboring cells are grouped into cell clusters, and local minimums within individual cell clusters are identified as relatively stable cell states in the network. An R implementation of the algorithm can be found in <https://github.com/xu-lab/SLICE>.

### 3 Objectives

This thesis has three main objectives. First, gain insight of scRNA-seq analysis flow and especially find the optimal ways to preprocess the data and assess its quality. Second, test the relevant tools as the first phase of the computational part. Third and most importantly, construct an analysis pipeline by combining the best approaches and necessary quality metrics. The reliability of the pipeline should also be estimated, mainly by testing the pipeline with more than one publically available scRNA-seq datasets.

In this kind of experiment, where the pipeline is one of the main goals, a central issue is, how to define the “relevant” or “best” approaches for the pipeline. The amount of available tools for different data analysis steps is remarkable and is continuously increasing. Tools specifically designed to scRNA-seq experiments are picked whenever possible, since they are the major interest in this thesis. Moreover, since the experimental standards are not established, the methods that are not strictly dependent on spike-ins or UMIs is preferred over the more specific methods. Finally, the compatibility between the tools in different phases should be taken into account.

In the scope of this thesis, there are clearly limited resources for comprehensive testing and comparison of several tools with various parameters in different phases of the analysis flow. Thus, the “best approach” that is finally chosen into the pipeline does not necessarily mean that it is the best method of all available approaches. Instead, it may be for example a familiar, commonly used tool (e.g. FASTQC), or a tool that seems to have some favorable or interesting properties either in general or in single-cell specific point of view. For interested readers, the comparison and validation of different tools are discussed broadly and more systematically in other research articles already available or in press.

## 4 Materials and methods

### 4.1 Data

The two publically available scRNA-seq datasets used for building and testing the pipeline are presented in Table 2. The data was accessed through NCBI archive (<https://www.ncbi.nlm.nih.gov/gds/>). Only parts of the full **Camp dataset** consisting of 770 samples was used, while all 466 samples in **Darmanis dataset** was originally loaded. However, one of the Darmanis samples had a truncated fastq-file and was thus excluded from further analysis.

**Table 2: Properties of the used data.** Source organism for both datasets is *Homo sapiens*.

Name	Tissue	Accession	Cell types	Samples	Article
<b>Darmanis</b>	Brain	GSE67835	9	465	Darmanis et al. 2015
<b>Camp</b>	Liver	GSE81252	4	247	Camp et al. 2017

**Gencode** sequences of release 27 was used as **reference transcriptome** ([ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode\\_human/release\\_27](ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_27), 11/6/2017). The transcriptome file containing only protein-coding sequences (*gencode.v27.pc\_transcripts.fa*) was preferred over the full set of transcripts (*gencode.v27.transcripts.fa*). An alternative reference containing also spike-in RNA sequences was constructed by concatenating the sequences of a common set of external RNA controls developed by the **External RNA Controls Consortium (ERCC)** with the transcript file. These ERCC-sequences can be downloaded from <https://tools.thermofisher.com/content/sfs/manuals/ERCC92.zip>.

### 4.2 Environment and the relevant tools

Computational environment and capacity was provided by **CSC** – IT Center for Science, Finland (<https://www.csc.fi/>), and the work was done in *Taito-supercluster*. The resource management system SLURM used by Taito (<https://research.csc.fi/taito-batch-jobs>, 4/10/2018) affected slightly into the structure of the pipeline and may thus also affect the portability.

Many of the tools discussed in literature review were also included into final pipeline. Table 3 presents the tested and chosen tools together with the reasoning. When designing the data-analysis steps, especially the step-by-step workflow presented by Lun et al. together with the course material by Hemberg lab provided solid base to build on (Lun et al. 2016b; <https://hemberg-lab.github.io/scRNA.seq.course/> 4/19/2018).

**Table 3: Choosing the tools into pipeline.**

Analysis step	Tools tested	Tools chosen	Notes/arguments
<b>Initial quality processing</b>	FASTQC TrimGalore! Celloline	FASTQC TrimGalore! (fastqcr)	Usability and popularity weighted most. It turned out that Celloline (Ilicic et al. 2016) was not currently actively maintained and was tricky to configure.
<b>Alignment &amp; quantification</b>	STAR Salmon	Salmon	Memory effectiveness, processing speed, novel and interesting approach.
<b>Quality control</b>	Scater	Scater	Scater (McCarthy et al. 2017) provides both platform and diverse functions for scRNA-seq analysis. Salmon output straight compatible with Scater.
<b>Normalization</b>	scran SCnorm scater-CPM	scran SCnorm	Single-cell specificity. Unfortunately SCnorm (Bacher et al. 2017) proved a bit unstable and has quite many parameters that should be adjusted, and thus only scran (Lun et al 2016a) was finally used.
<b>Confounding factors/noise</b>		Scater scran	Lack of simple generic approaches. Scater and scran were used to <i>collect</i> information about potential factors, but they cannot be corrected without supervising. A scran-function <code>denoisePCA()</code> was tested.
<b>Cell type identification &amp; characterization</b>	scran SC3 SCDE MAST	SC3 MAST	SC3 (Kiselev et al. 2017) easily compatible with scater. SCDE (Kharchenko et al. 2014) was first planned for use, but proved too slow for large datasets. Thus, MAST (Finak et al. 2015) was chosen instead.

The paired **fastq-files** were extracted from the SRA-raw files using **fastq-dump** program of **SRAToolkit** ver 2.8.0 (<https://www.ncbi.nlm.nih.gov/sra/docs/>, 4/11/2018). The quality of fastq-files was estimated with **FASTQC (ver. 0.11.2)** and, when relevant, the initial quality processing was performed with **TrimGalore! (ver. 0.4.5)**, with the guidelines presented by developers in <https://www.bioinformatics.babraham.ac.uk>. The reads were quantified against the reference transcriptome using **Salmon v0.9.1** (Patro et al. 2017)

**R (ver. 3.4.1)** with several **Bioconductor (ver. 3.6)** packages was used to perform downstream steps including filtering, visualization, normalization and clustering (<http://bioconductor.org/>). The most relevant packages are listed in Table 3, but in addition, *SingleCellExperiment*-package and (only with MAST-part) *SingleCellAssay*-package are essential since they provide the data storage and manipulation platforms for the tasks. Please refer to the documentation of each package for full listing of dependencies.

### 4.3 Filtering, normalization and confounders

Three main approaches for **filtering the cells** were tested. First, the automatic outlier detection with **PCA** provided by Scater, second, the **feature-specific** i.e. “manual” approach and finally, the **interception** of the two. Scater default settings were mostly used, but the examined variable set was replaced with the following variables: "total\_counts", "total\_features", "pct\_counts\_Mt" and "pct\_counts\_top\_200\_features". The six feature-specific filters were developed based on literature research:

- *Library size* i.e. number of reads: MAD-threshold (Lun et al. 2016b)
- *Feature amount* i.e. number of detected genes: MAD-threshold (Lun et al. 2016b)
- *Mitochondrial gene proportion*: MAD-threshold (Ilicic et al 2016, Lun et al. 2016b)
- *Spike-in proportion* [only if spikes were used]: MAD-threshold (Lun et al. 2016b)
- *Mapping rate*, MAD-threshold
- *Existence of the housekeeping genes* GAPDH and ACTB: drop the cells with no expression (Ilicic et al 2016, Camp et al 2017)

The **genes** were filtered based on the following criteria:

- No expression in any of the cells
- No annotations found
- High enough average expression over the cells ( $> 1$ ) (Lun et al. 2016b)
- Genes showing detectable expression in more than n cells ( $n=10$ ) (Lun et al. 2016b)
- The default filters of SC3 and MAST-packages (after normalization)

The normalization was performed by calculating the size factors with scran and then applying the normalization-function of scater, as described in the step-by-step workflow (Lun et al. 2016b).

### 4.4 Downstream analyses

Clustering was done with the SC3-package, and the classification accuracy between the predicted and provided labels (if available) was checked with different settings. The classification accuracy was calculated with the R-function `adjustedRandIndex()`. **sc3-heatmaps** visualizing the classification and the detected marker genes was plotted for data. MAST package was used to characterize the differentially expressed genes between the groups. MAST analysis was run to the clusters automatically detected with SC3 and, if no clear control group existed, cluster 1 was used as contrast to which the other groups were compared.

### 4.5 Comparing pipeline results into the literature

tSNE-clustering (see Darmanis et al 2015, especially Fig.1A and the Supplementary material) was tested for Darmanis data pre-processed with the pipeline to compare the result with the published one. Quality trimming with discarding reads shorter than 50 bp was followed by

quantification, QC and normalization as described above, but without cell filtering, since Darmanis et al. had used all cells in their analysis. Genes of the resulting normalized matrix were further filtered with SC3 default settings. Pearson distances of the cells were calculated with SC3, and given as an input for tsne-function of tsne-package (<https://github.com/jdonaldson/rtsne/>). Other parameters for tsne were k=3, perplexity=50 and epoch=50. Moreover, epoch\_callback function was given as

```
function(x, y){ plot(x, t='n'); text(x, labels=colnames(SCE)) }
```

The plot3d-function of rgl-package (<http://rgl.r-forge.r-project.org/>) was used for visualization.

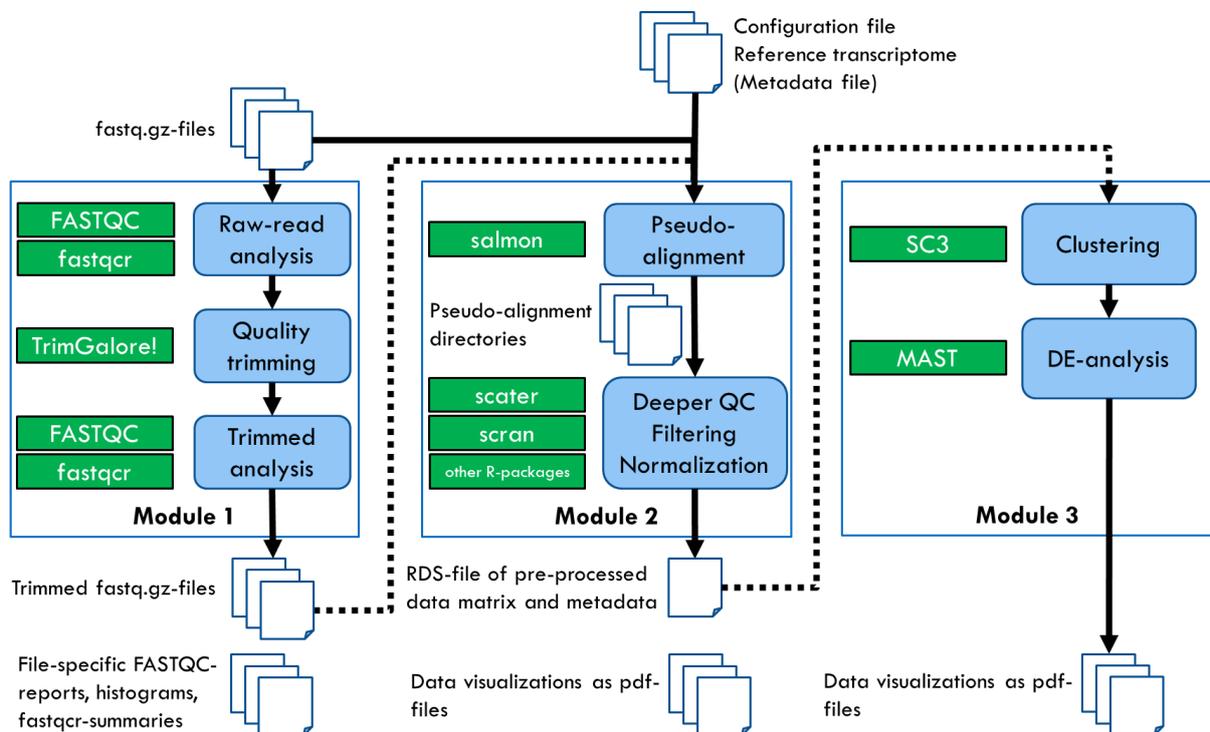
# 5 Results

## 5.1 The pipeline

Instead of one pipeline, the analysis flow was finally implemented as three individually launched modules presented in Figure 10. Even though full automation might be possible at some point, the literature research strongly suggested that scRNA-seq data analysis requires some supervised checkpoints during the whole process. **Module 1** processes the raw-read files and produces initial quality metrics for both original and trimmed read-files. **Module 2** quantifies the transcripts, generates an expression matrix, collects various information and quality metrics and performs filtering and normalization. The major purpose of **Module 3** was to implement simple downstream analysis steps and visualizations for observing the effect of two previous modules in the data. The functionalities of each module is next described more precisely.

### 5.1.1 Module 1: Initial quality processing

The purpose of Module 1 is to evaluate the initial quality of the data and the effect of trimming on library quality. It takes the raw-read files and couple of user-adjusted parameters as input



**Figure 10. The three modules of the pipeline.** Sub-phases are marked as blue rectangles, and the green rectangles show the major tools used. Module 1 is not necessary for running Module 2. Module 2 should be quite independent of the experimental goals, while Module 3 is more specific

and produces quality-trimmed read files together with file-specific and experiment-wide quality information of both raw and trimmed fastq-files as output. Currently Module 1 has only three adjustable parameters : *<quality trimming threshold>* **-q** and *<read length discarding threshold>* **-l** required by TrimGalore together with the decision of whether the *<summary tables>* created by fastqcr are saved into disc or not (**-s**). Otherwise the default settings of the tools are used. In future, further flexibility should be added.

Module 1 automatically produces a self-explanatory directory hierarchy under the result directory given by user. The output information consists of file-specific FASTQC-reports and experiment-wide quality information, such as listing of samples that have marked as passed, failed or caused warnings and measures of GC-content, duplication rate and read length. The script is launched with the command

```
./scripts/doFastqcAndFastqcrSummarizationForRawAndTrimmedFiles.sh \  
-i <raw read directory> \  
-o <result directory> \  
-q <quality trimming threshold> \  
-l <read length discarding threshold> \  
-s <TRUE/FALSE>
```

Separating Module 1 tasks from the quantification module enables supervised adjustment of the data before quantification: one may decide to process the reads in different or additional approach, e.g. dealing with duplicates, or remove problematic samples prior quantification. Moreover, since the quantification is done by Salmon, the trimming might not be necessary at all, and one may choose to use raw reads instead of trimmed files. Salmon maps the *k-mers of the reads* into reference transcriptome, and the sequences that are not included into reference transcriptome are simply not quantified. Nevertheless, FASTQC-analysis for raw reads followed by fastqcr-summarization might provide some additional information for data-analysis.

### 5.1.2 Module 2: Quantification, further QC-processing and normalization

To deal with the growing parameter space and providing more flexibility, Module 2 was implemented with a configuration file. It enables the indexing of the reference transcript with adjustable *<k-mer length>* and running salmon for paired read files with different parameters. Finally, it launches a configured R-script that summarizes the data from sample-specific pseudo-alignment directories into a single SCE-element and manipulates it according to instructions of configuration file. The relevant steps are summarization to gene counts, fetching annotations, collecting various quality metrics and filtering the cells and genes based on this information. Moreover, the cell-cycle phase is predicted with the *cyclone()*-function of the

scran-package, and also other potential explanatory variables are detected (but not corrected!). Finally, the expression matrix is normalized with deconvolution-based normalization.

The output of Module 2 consists of several pdf-visualizations of data properties and the RDS-file of processed data that can be restored as a SCE element in R if needed. This file includes also the un-normalized counts so that normalization may be later performed again with different approach. It also self-documents the steps that have been performed for the data (See the Bioconductor documentation for SingleCellExperiment for further information). On the other hand, the R-script can also be used individually by providing the salmon result directory and other parameters as an input:

```
Rscript scripts/doAutomatedQC.R <salmon_dir> <metafile>  
<cell_name_column> <nmds_threshold> <sample_QC_filter>  
<min_cluster_size> <threads>
```

The adjustable parameters for R-script are currently the *<cell-filtering approach>* (“none”, “pca”, “manual”, or “both”), the size of MAD (median absolute deviation) *<threshold used for cell filtering>* and whether the scaling factors for normalization are calculated by *<pooling similar cells>* or not. Moreover, there is possibility to add a metafile-table (e.g. SRA-run table), which may prove useful in downstream analysis. The script was developed to automatically detect the possible control features (mitochondrial genes and spike-ins) and taking the spikes into account in filtering and normalization.

### 5.1.3 Module 3: Clustering and DE-analysis

Module 3 is the simplest of the three modules, and is implemented as a single R-script. It applies SC3 for the user-defined range of clustering, or if the range is not given, the optimal clustering is automatically detected. It then uses MAST (Model-based Analysis of single Cell Transcriptomics) for fitting a model for filtered data and then calculates both fold-change hurdle values and log-likelihood ratios which are then used to find and order DE genes. By default, it uses condition 1 as a contrast to which compare all other groups, but the contrast can be changed with a parameter.

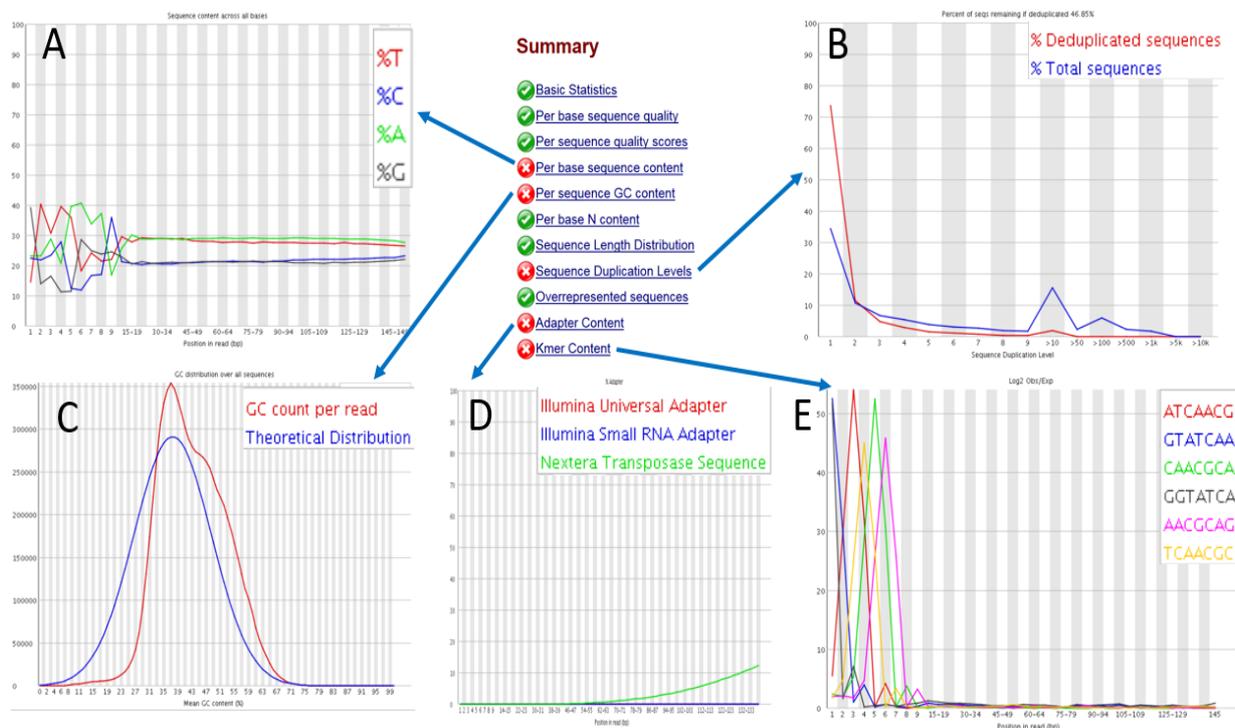
## 5.2 Initial quality checks and trimming with Module 1

Fastqcr-summary information for Darmanis and Camp raw data produced by the Module 1 are collected in Table 4. The FASTQC-modules that were considered problematic with both datasets were "Per base sequence content" "Per sequence GC content" and "Sequence Duplication Levels". Moreover, Darmanis had some adapter contamination and Camp data had minor problems with Per-tile sequencing quality. See Table 1 for a quick review for FASTQC

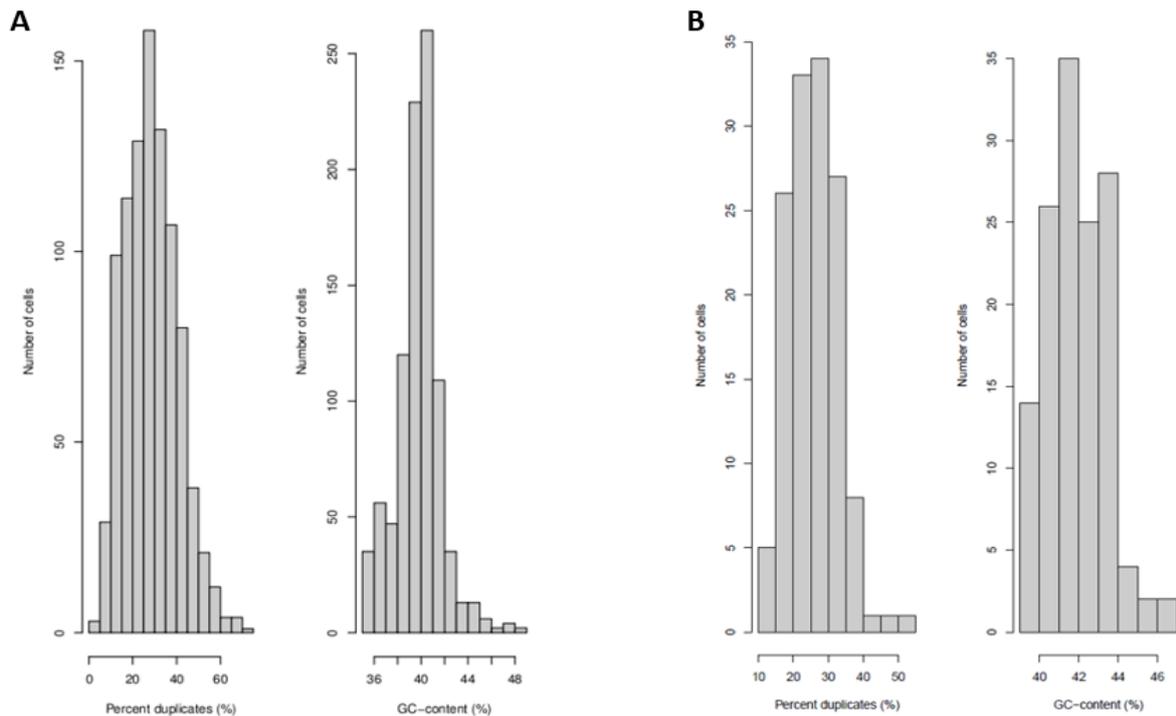
**Table 4: Comparison of Fastqcr-summary tables for Darmanis and Camp raw data. FASTQC-modules which caused fails in the samples are marked red.**

FASTQC module	Darmanis				Camp			
	nb_samples	nb_fail	nb_pass	nb_warn	nb_samples	nb_fail	nb_pass	nb_warn
Adapter Content	930	21	774	135	554	0	554	0
Basic Statistics	930	0	930	0	554	0	554	0
Overrepresented sequences	930	0	572	358	554	0	264	290
Per base N content	930	0	930	0	554	0	554	0
Per base sequence content	930	927	0	3	554	554	0	0
Per base sequence quality	930	1	925	4	554	0	529	25
Per sequence GC content	930	424	110	396	554	96	167	291
Per sequence quality scores	930	0	930	0	554	0	554	0
Per tile sequence quality	930	0	296	2	554	72	372	110
Sequence Duplication Levels	930	51	501	378	554	2	416	136
Sequence Length Distribution	930	0	930	0	554	0	554	0

modules. Figure 11 visualizes some sample-specific problematic features and Figure 12 shows the dataset-wide histograms of duplicate rates and GC-contents of the dataset.



**Figure 11: An example of FASTQC (version 0.11.2) results for the raw sample SRR1974690\_1 in Darmanis-dataset. As seen in the middle panel, 5 metrics-modules were failed. Panel A: Sequence content (%T, %C, %A, %G) against the base position revealing some bias in first 10 base positions. Panel B: Sequence duplication levels (%). According to the figure header, only 46,85%.sequences would remain if deduplicated. Panel C: GC-distribution against the mean GC-content showing the peak probably explained by the duplication level. Blue curve represents the theoretical and red curve the observed distribution. Panel D:Adapter content (%) against the base position showing Nextera transposase contamination in last base positions. Panel E: K-mer content (as log2 Obs/Expr) against the base position, showing repeated sequences in first base positions. The k-mer content was marked passed after trimming.**



**Figure 12: Histograms of duplicate rates and GC-contents for untrimmed Darmanis (A) and Camp (B) presented as an examples of Module 1 output.** In both datasets, the duplication rates seem to be quite high, but Darmanis shows slightly higher rates than Camp. Moreover, Darmanis shows quite many samples whose GC-content differs from the one expected in human

Table 5 demonstrates how the problems with adapter content and per-base sequence quality were corrected by trimming. Also the amount of samples that passed module “Overrepresented sequences” and “Sequence duplication levels” was increased. Dealing with the remaining problems is not trivial, and is currently not implemented in the pipeline.

**Table 5: Effect of quality trimming (-q 20, -l 50) into Darmanis set: Minor problems with adapter contamination and Per base sequence quality were corrected, but problems with Per base sequence content and Per sequence content were increased.**

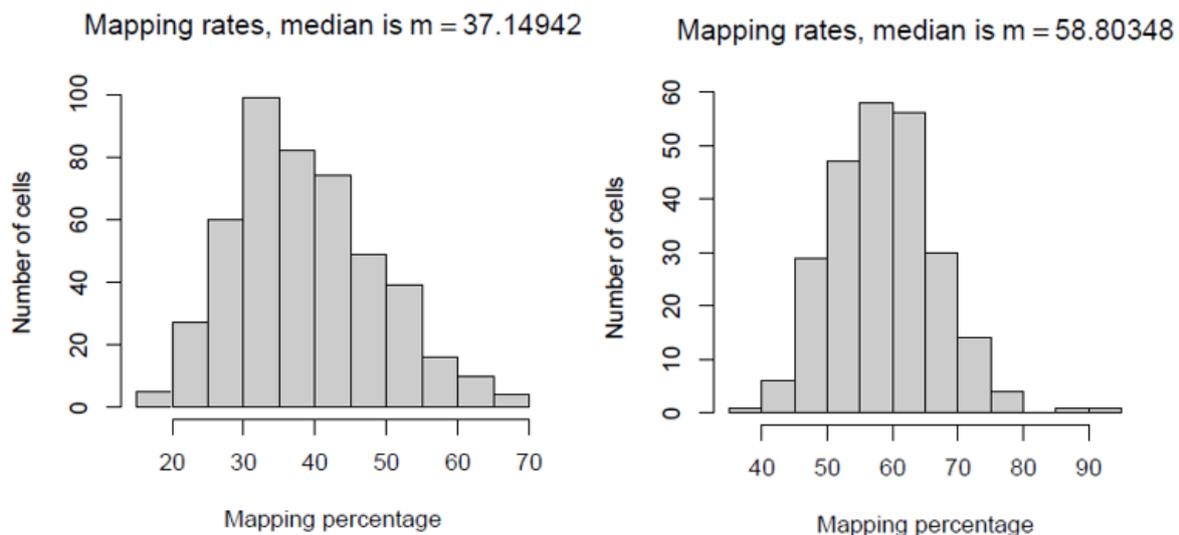
FASTQC module	Darmanis raw				Darmanis trimmed			
	nb_samples	nb_fail	nb_pass	nb_warn	nb_samples	nb_fail	nb_pass	nb_warn
Adapter Content	930	21	774	135	930	0	930	0
Basic Statistics	930	0	930	0	930	0	930	0
Overrepresented sequences	930	0	572	358	930	0	764	166
Per base N content	930	0	930	0	930	0	930	0
Per base sequence content	930	927	0	3	930	930	0	0
Per base sequence quality	930	1	925	4	930	0	930	0
Per sequence GC content	930	424	110	396	930	456	101	373
Per sequence quality scores	930	0	930	0	930	0	930	0
Per tile sequence quality	930	0	296	2	930	0	298	0
Sequence Duplication Levels	930	51	501	378	930	50	525	355
Sequence Length Distribution	930	0	930	0	930	0	0	930

### 5.3 Pipeline visualizations for estimating dataset quality before filtering

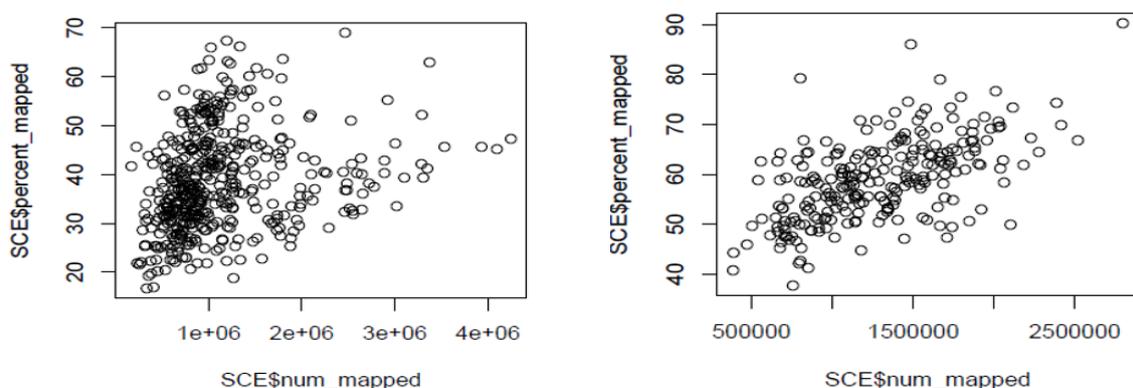
Pipeline produces several visualizations of data properties before and after filtering for estimating the quality of the datasets. The following features were considered most informative:

- Mapping rate
- Relationship between the mapping rate and the number of mapped reads
- Library size i.e. total numbers of reads
- Feature amount
- Proportion of control features e.g. mitochondrial genes and spike-in sequences (if used)
- Scatter PCA plot of (a given subset of) quality features, which also visualizes potential outliers

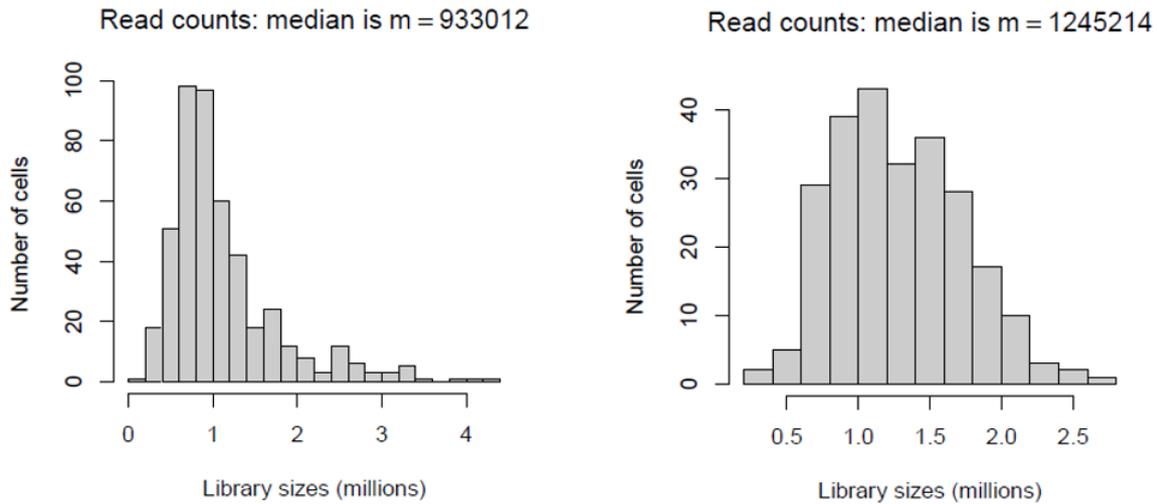
The comparison of outputs obtained for the two test datasets is presented in Figures 13 to 19, demonstrating the differing properties of the datasets. Excluding Figure 19, Darmanis data is aligned against the full transcriptome that resulted in slightly better alignment rates compared to protein-coding only transcriptome. Camp data is aligned against protein coding transcriptome completed with the spike-in sequences.



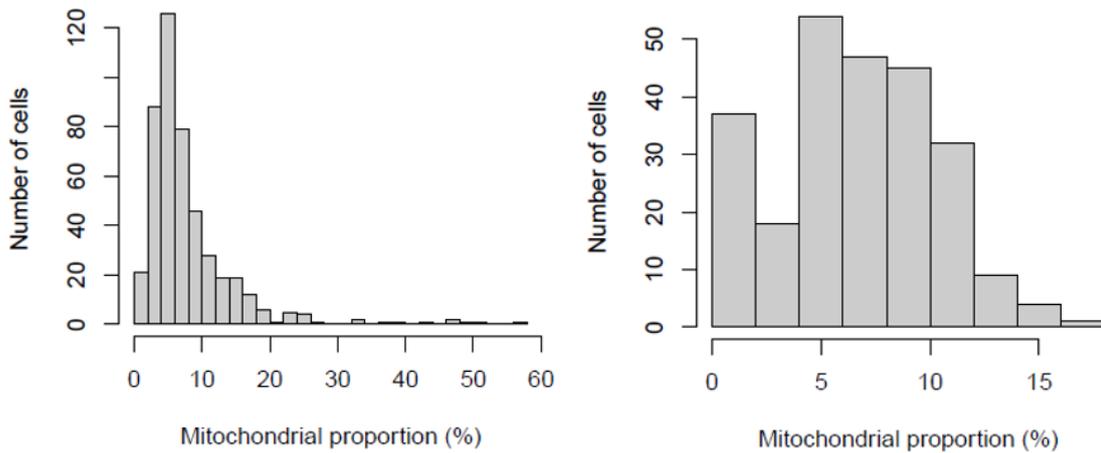
**Figure 13: Histograms of mapping rate for Darmanis (left) and Camp (right). Camp shows higher mapping rates, which was the trend regardless of the used reference transcriptome.**



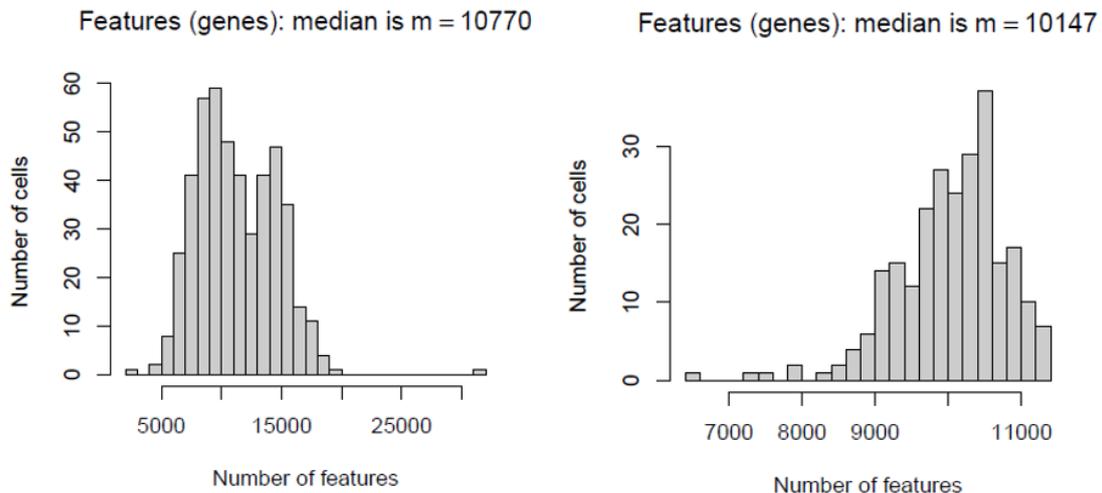
**Figure 14: Mapping percentage vs. Number of mapped reads in Darmanis (left) and Camp (right). Separation may indicate problems e.g. in paired fastq-file organization (see <http://www.nxn.se/valent/2017/9/1/low-mapping-rate-1-unsorted-fastq-pairs>, 5/4/2018)**



**Figure 15: Histograms of library sizes for Darmanis (left) and Camp (right).** Darmanis data is skewed while Camp shows clear Gaussian trend. Remarkably big library size may indicate bad quality (cell doublet)

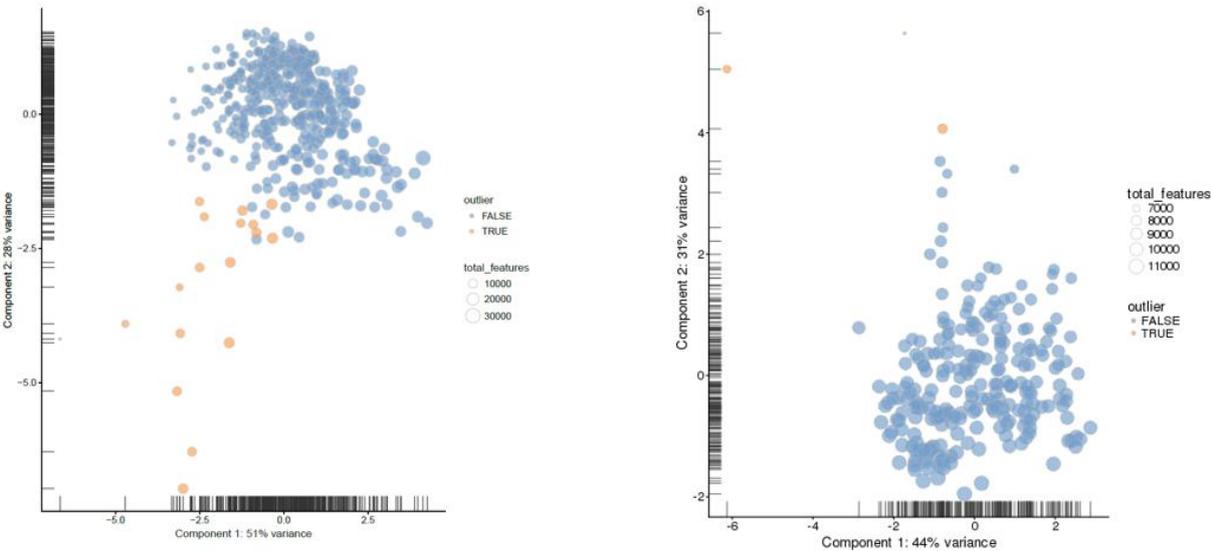


**Figure 16: Histograms of the proportion of mitochondrial genes for Darmanis (left) and Camp data (right).** Some Darmanis cells had remarkably large proportions of mitochondrial genes, which may indicate bad quality (apoptosis, premature cell lysis)

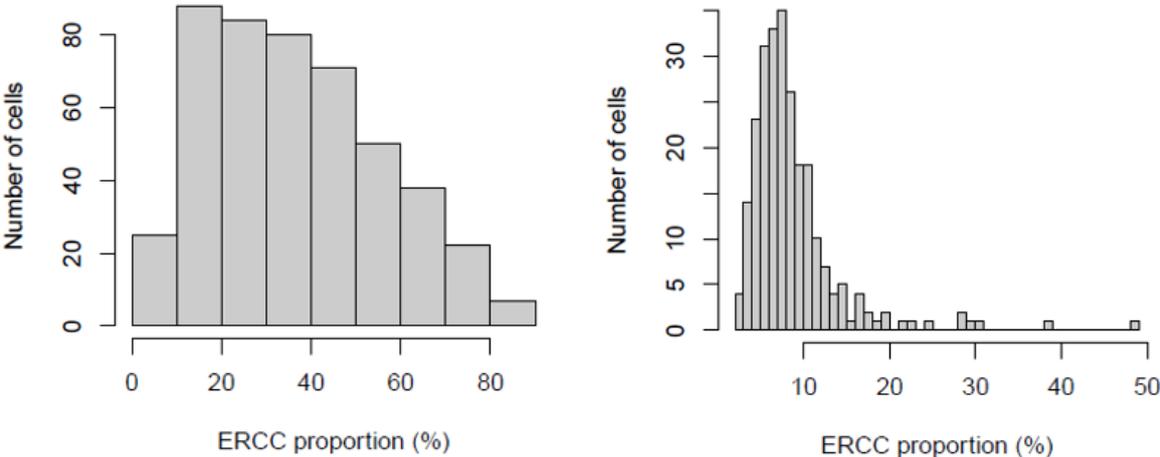


**Figure 17: Histograms of feature amounts for Darmanis (left) and Camp (right).** Both the trends and the ranges of the feature distribution are different, but medians are quite near to each other

According to the original articles, Camp data was produced with spike-ins but whether this was the case also with Darmanis data, remained unclear. For curiosity, the effect of quantifying Darmanis-data against the same spike-in containing reference used for Camp was tested as well, and surprisingly, high proportions seemed to hit into ERCC-genes, as seen in Figure 19.



**Figure 18:** Scatter PCA-plot and outlier detection for Darmanis (left) and Camp (right). Set of variables examined: "total\_counts", "total\_features", "pct\_counts\_Mt", "pct\_counts\_top\_200\_features". 23 outliers were found in Darmanis and 3 outliers were detected in Camp. The points are sized by the feature amounts and predicted outliers are colored with red.



**Figure 19:** Histograms of spike-in RNA proportions for Darmanis (left) and Camp (right). For Camp, the spike-ins were expected to see but for Darmanis, it was not clear. However, several sequences seemed to map into the ERCC-sequences also in Darmanis data, and in suspiciously high amounts.

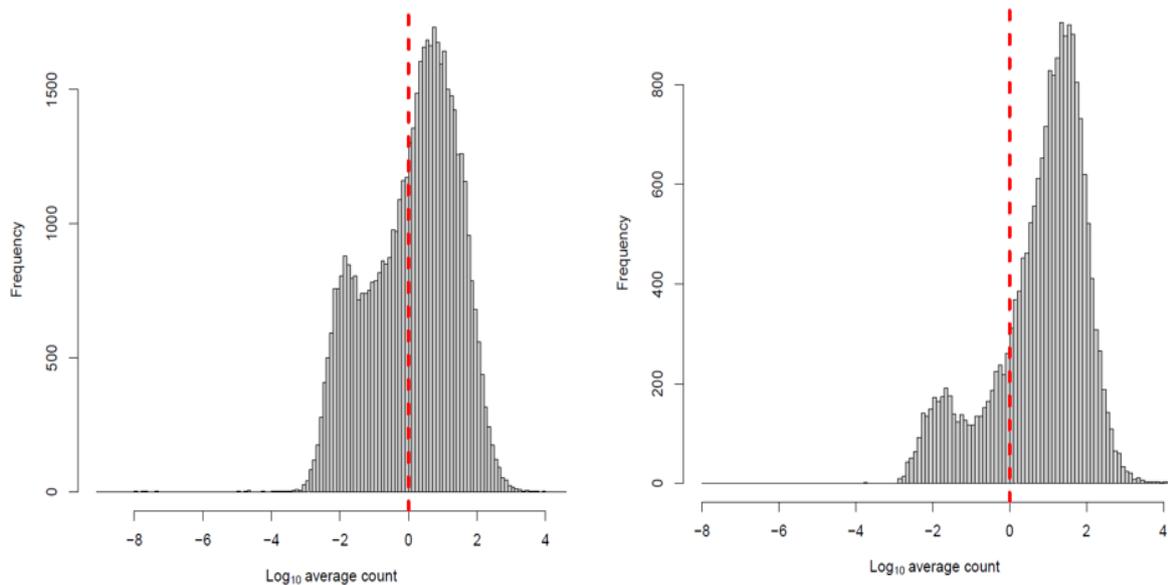
## 5.4 Filtering

To demonstrate the effect of cell filters, the filtering with five different settings for both datasets were run and the removed cells were calculated. The results are presented in Table 6. For feature-specific i.e. manual filters, three MADs-thresholds were used.

**Table 6: Filtering of Darmanis (D) and Camp (C) with different settings.** With manual filters (i.e. filters based to MADs only), the cells to be removed could be categorized according to the problematic property (library size, number of detected features, proportion of mitochondrial and spike-in genes, housekeeping genes and mapping rate, respectively), while the remaining filters show only the total amount and percentage of cells removed. Note that Housekeeping-filter actually just checks the expression of the two housekeeping genes GAPDH and ACTB.

	Filter	ByLibSize	ByFeature	ByMito	BySpike	Housekeeping	ByMapping	Tot.removed	Pct.removed
D	3 MADs	4	1	45	0	8	1	59	12.69 %
	4 MADs	0	1	29	0	8	0	39	8.39 %
	5 MADs	0	1	16	0	8	0	25	5.38 %
	PCA	-	-	-	-	-	-	23	4.95 %
	PCA & 3 MADs	-	-	-	-	-	-	23	4.95 %
C	3 MADs	2	6	0	18	0	1	29	11.74 %
	4 MADs	0	3	0	10	0	0	13	5.26 %
	5 MADs	0	2	0	7	0	0	9	3.64 %
	PCA	-	-	-	-	-	-	3	1.21 %
	PCA & 3 MADs	-	-	-	-	-	-	2	0.81 %

In the Darmanis data, the proportion of the mitochondrial genes was clearly the most “problematic” feature detected by MADs-based filtering. Camp cells showed such a spike-in distribution that the filtering was most aggressive with BySpike-check. Note that Housekeeping-filter is not affected by MAD-threshold since it only checks the expression of



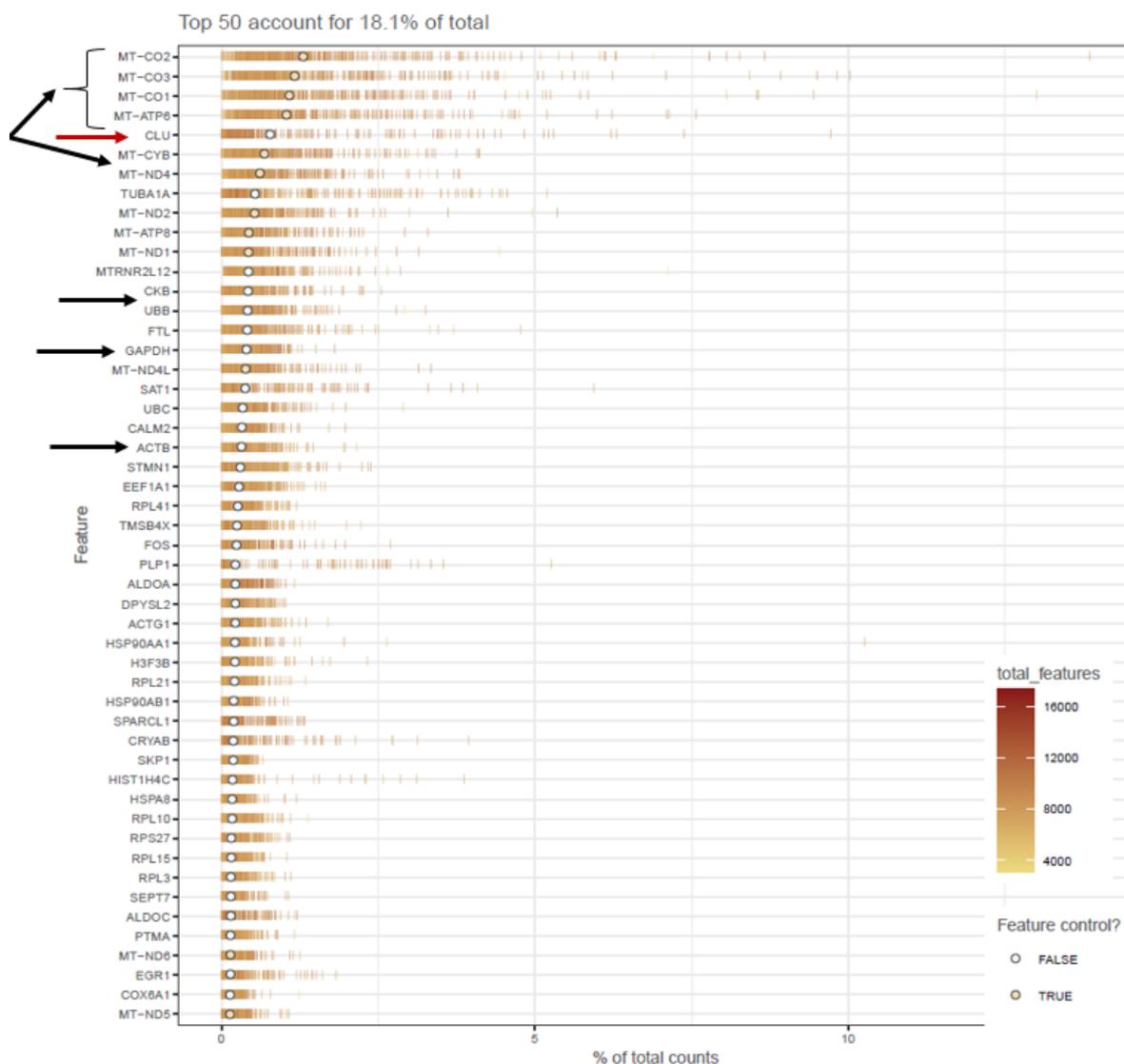
**Figure 20: Checking the suitability of gene filter that removes the features having average expression less than 1 for datasets.** Note that x-axis is presented as logarithmic scale. The filter is shown as a dashed red line. In both cases, the features with low expression are effectively removed, but the filter is a bit more aggressive for the data in right panel.

the two housekeeping genes, GAPDH and ACTB. Darmanis dataset had 8 cells where no expression for these genes was detected.

Figure 20 presents the effect of gene-filter, when keeping only the features having average expression in counts-matrix more than 1. The peak of moderately expressed genes on the right side of the dashed line is remained while the features with lower expressions are discarded. This kind of figure is suggested by Lun et al. 2016b to check the suitability of the filter.

### 5.5 Checking the features with the highest expression over cells

In order to quickly get more information about the dataset expression patterns, the scatter-function that plots the expression of most expressed genes (top 50 by default) was added into



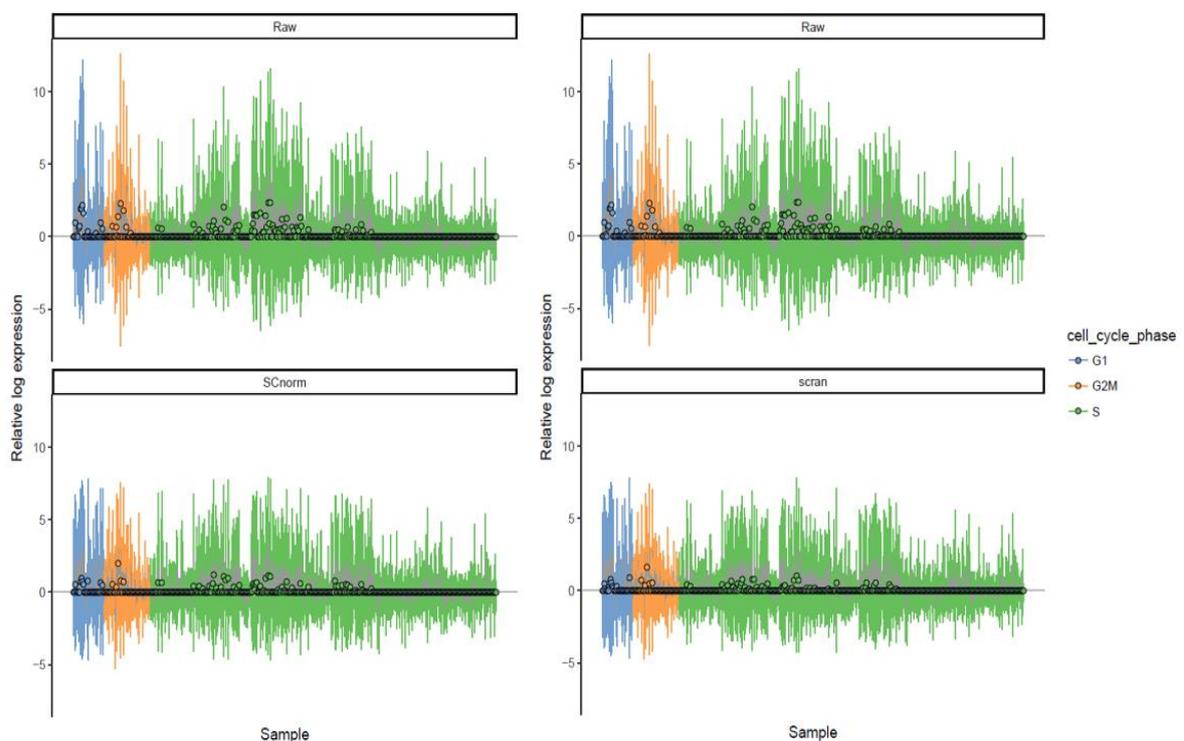
**Figure 21: Feature-specific percentages of highly expressed genes in the unfiltered Darmanis-dataset.** Circles show the median expression over all cells, while the vertical bars represent sample-specific values.

pipeline. Most of the top 50 genes should be housekeeping genes and mitochondrial genes (<https://bioconductor.org/packages/release/bioc/vignettes/scater/inst/doc/vignette-qc.html>, 5/6/2018). An example figure with unfiltered Darmanis data is seen in Figure 21.

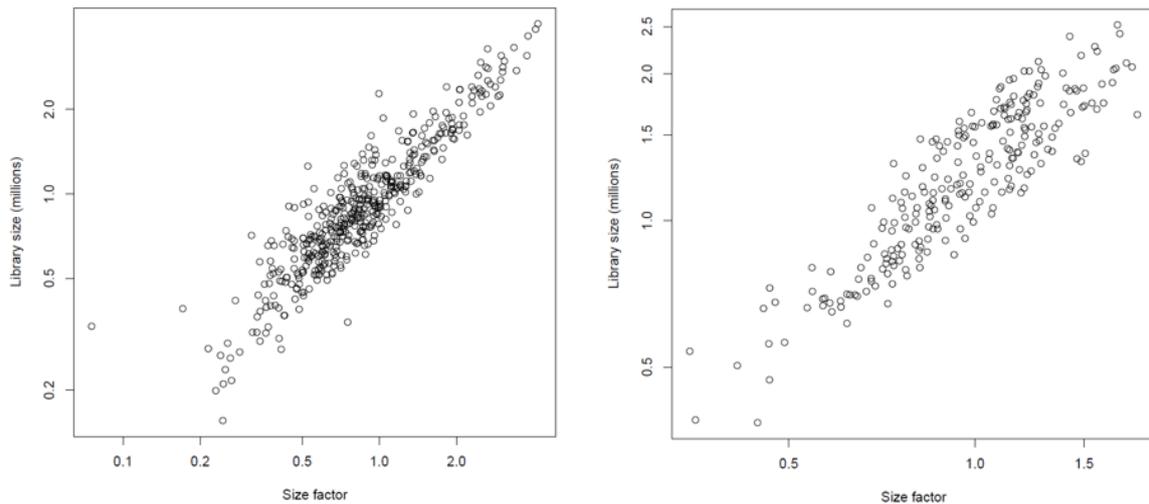
It can be seen that many top features are genes essential for cellular respiratory chain such as MT-CO-genes (Mitochondrially Encoded Cytochrome C Oxidases) and MT-ND-genes (Mitochondrially Encoded NADH:Ubiquinone Oxidoreductases) (<http://www.genecards.org> 4/22/2018). Both housekeeping genes GAPDH and ACTB are also shown, as well as other conserved genes such as CKB and UBB. However, for example CLU (clusterin) might have something to do with cellular stress conditions and apoptosis (<http://www.genecards.org> 4/22/2018).

## 5.6 Normalization

Two single-cell specific normalization approaches were tested during the pipeline development. The **RLE-plots** (Relative log expression) for raw and normalized data with SCnorm and scran are presented in Figure 22 to visualize the major effects of normalization into the data.



**Figure 22 RLE-plots (Relative log expression) for Darmanis-dataset: SCnorm (left) and scran (right).** Each line represents the IQR (the 25th or 75th percentile) range of the expression and the median is shown with a circle. No outliers are shown in this “minimal” plotting style. For visualization purposes, the samples are colored by predicted cell cycle phase. The data was aligned against *pc\_transcripts*. Used min-cluster size was 100.



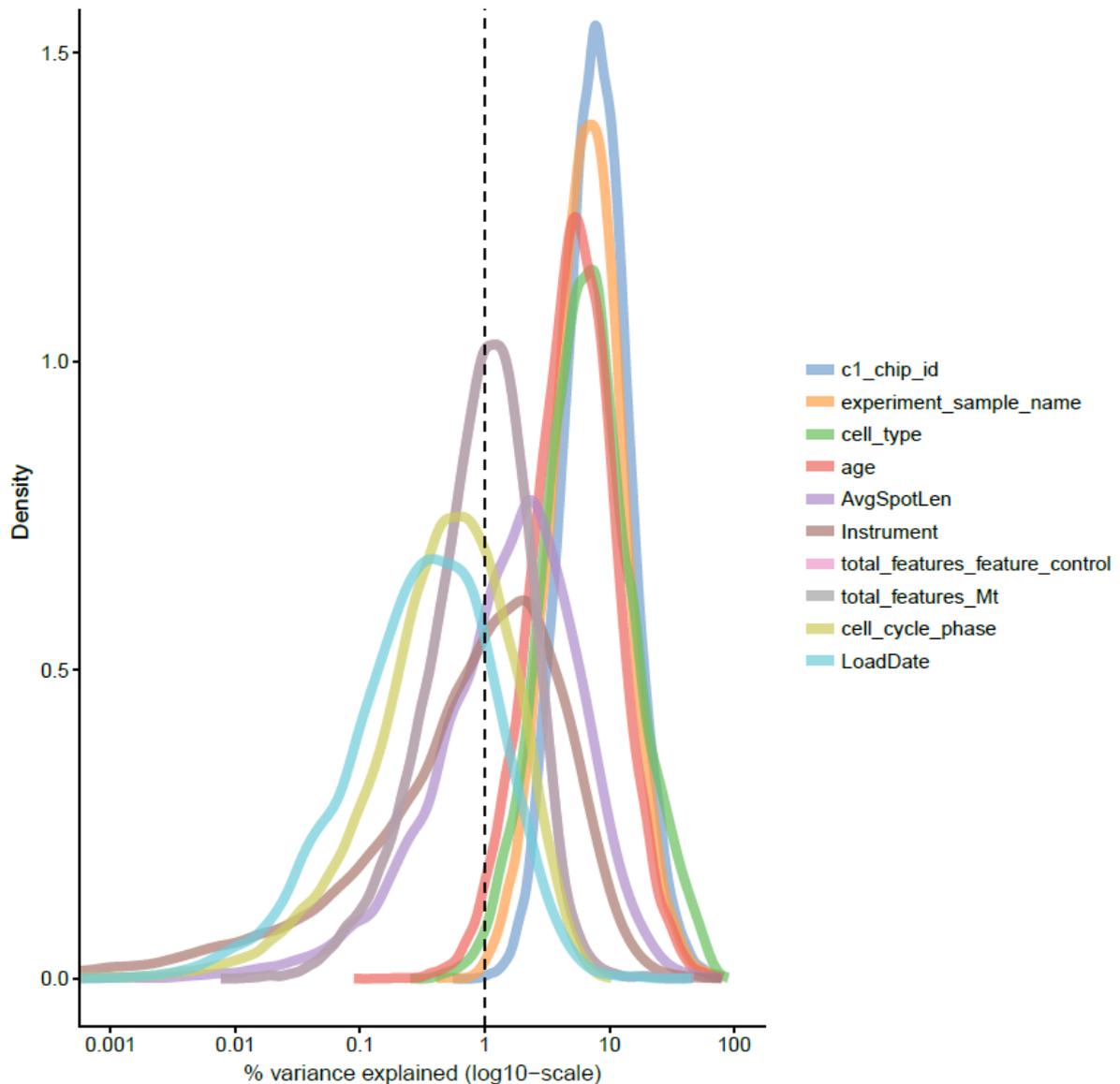
**Figure 23:** Scatter plots of deconvolved size factors vs. library sizes for Darmanis (left) and Camp data (right) in logarithmic axis. The more linearity, the more homogeneous cell types and vice versa.

Both approaches produced the normalized values in similar range: the un-normalized range of RLE (showing expression values approximately from -7 to 13) was scaled into the range between -5 and 7. Figure A in Appendix A further demonstrates, how SCnorm detected separately normalized conditions in Darmanis data, indicating that count-depth relationship normalization could be useful for this dataset. Figure 23 presents the relationship between the library size and scran size factors.

The scatter plot of size-factors vs. library sizes helps to estimate whether the calculated size factors are consistent with the biological expectations (Lun et al. 2016b). For a dataset consisting a single cell type, the trend should be linear, and for the dataset that potentially contains heterogeneous cell types, like for Darmanis and Camp, there should be dispersion which was also the case, as seen in Figure 23.

## 5.7 Confounders

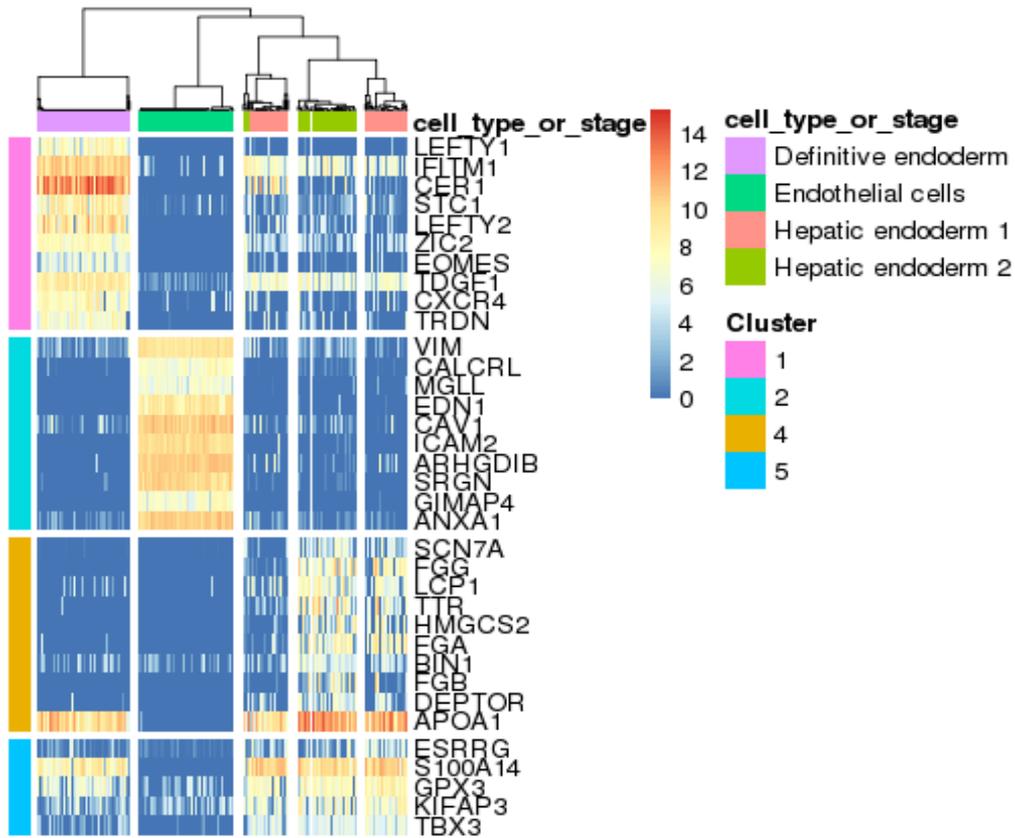
Pipeline detects “**potential explanatory variables**” simply by selecting and plotting the variables having 2-20 unique values over samples. An example figure for Darmanis, visualizing the detected variables with scatter, is seen in Figure 24



**Figure 24: Potential confounders detected in Darmanis data by the pipeline.** Each line corresponds to one factor and represents the distribution of  $R^2$  values across all genes. Note that these factors were detected among features having 2-20 unique values over the dataset, and supervising is needed to determine whether they truly are explanatory or confounding factors.

## 5.8 Clustering with SC3

SC3 was tested by both pre-defining the amount of clusters that should be found and by letting the algorithm detect suitable number of clusters automatically. With the used input data, SC3 seemed to reach better classification accuracy with automatic detection, and easily produced warnings of small clusters when providing pre-defined amount of clusters. The algorithm classified Camp input data into 5 cell types which were close of the provided 4 labels (Figure 25).

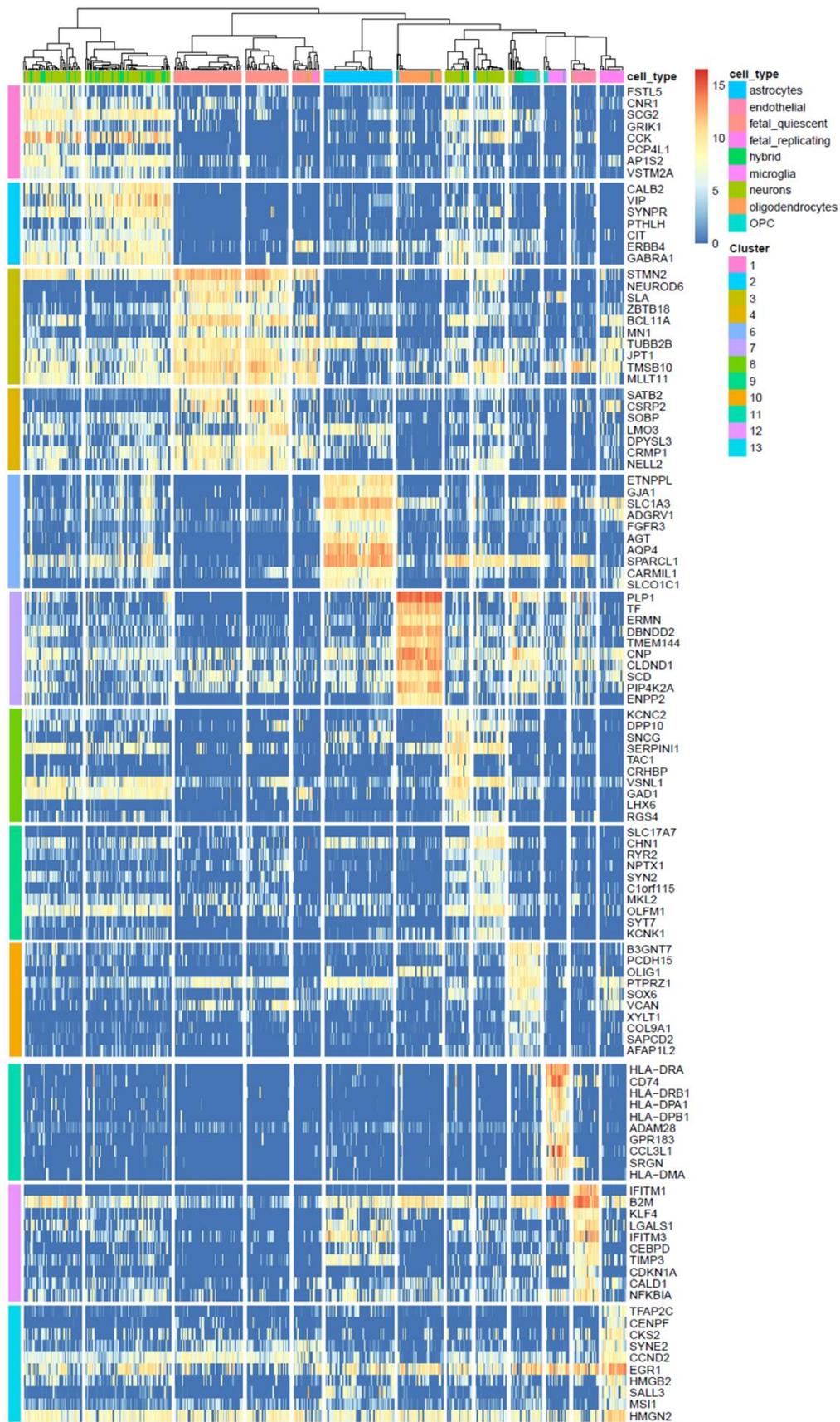


**Figure 25 Automatic cluster detection and marker gene identification with sc3 for Camp data.** The used input data contained 247 cells from two hepatic endotherms (HE), definitive endoterm (DE), and endothelial cells (EC). sc3 detected 5 different cell types and performed well with the DE- and EC-cells which were perfectly assigned into their own clusters. Making distinction for HE-cells was more difficult.

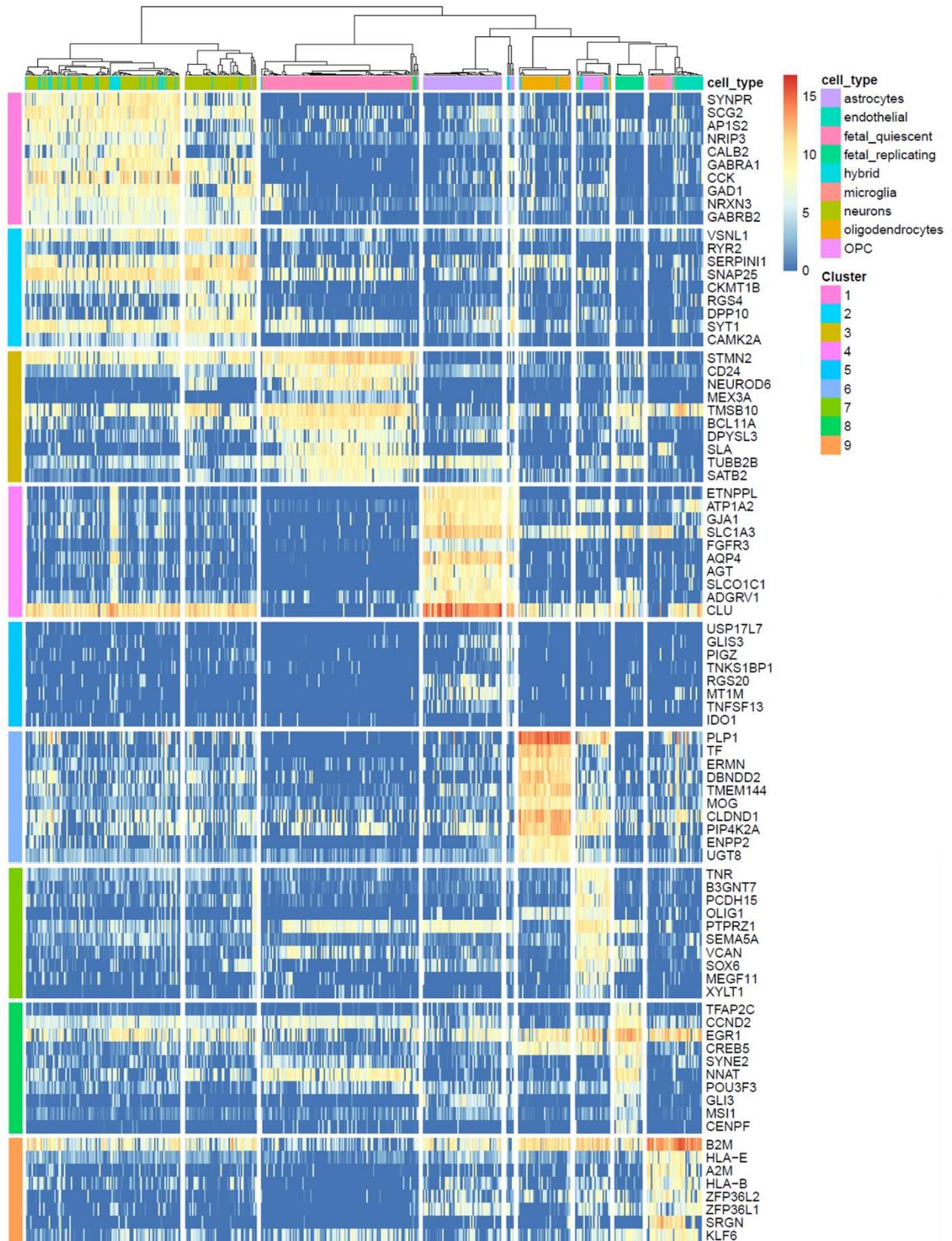
For Darmanis data, 13 distinct cell types were automatically detected, which was clearly higher than the 9 pre-assigned labels of the dataset. Table 7 shows the predicted vs. pre-assigned clusters and Figure 26 shows the heatmap of marker genes.

**Table 7: Pre-assigned cell types of Darmanis data and how they were divided into 13 clusters by sc3.** The input data consisted of 465 pre-processed cells and the genes were filtered with the default settings of sc3.

	1	2	3	4	5	6	7	8	9	10	11	12	13	row sums
<b>astrocytes</b>	1	0	0	0	0	58	1	0	2	0	0	0	0	62
<b>endothelial</b>	0	0	0	0	0	0	0	0	0	0	0	20	0	20
<b>fetal_quiescent</b>	0	0	57	36	17	0	0	0	0	0	0	0	0	110
<b>fetal_replicating</b>	0	0	0	0	5	0	0	0	0	0	0	0	20	25
<b>hybrid</b>	10	24	0	0	0	0	2	1	1	6	1	0	0	45
<b>microglia</b>	0	0	0	0	0	0	1	0	0	1	14	0	0	16
<b>neurons</b>	36	48	0	0	1	0	0	19	23	3	0	1	0	131
<b>oligodendrocytes</b>	2	0	0	0	0	0	34	0	0	2	0	0	0	38
<b>OPC</b>	0	0	0	0	0	0	0	0	0	14	4	0	0	18
<b>column sums</b>	49	72	57	36	23	58	38	20	26	26	19	21	20	



**Figure 26: Automatic cluster detection and marker-gene heatmap with sc3 for Darmanis data.** The used input data contained 465 preprocessed samples from nine neural cell types which were assigned into 13 groups.



**Figure 27: Clustering of Darmanis data with sc3-algorithm when the number of clusters was expected to be 9.**

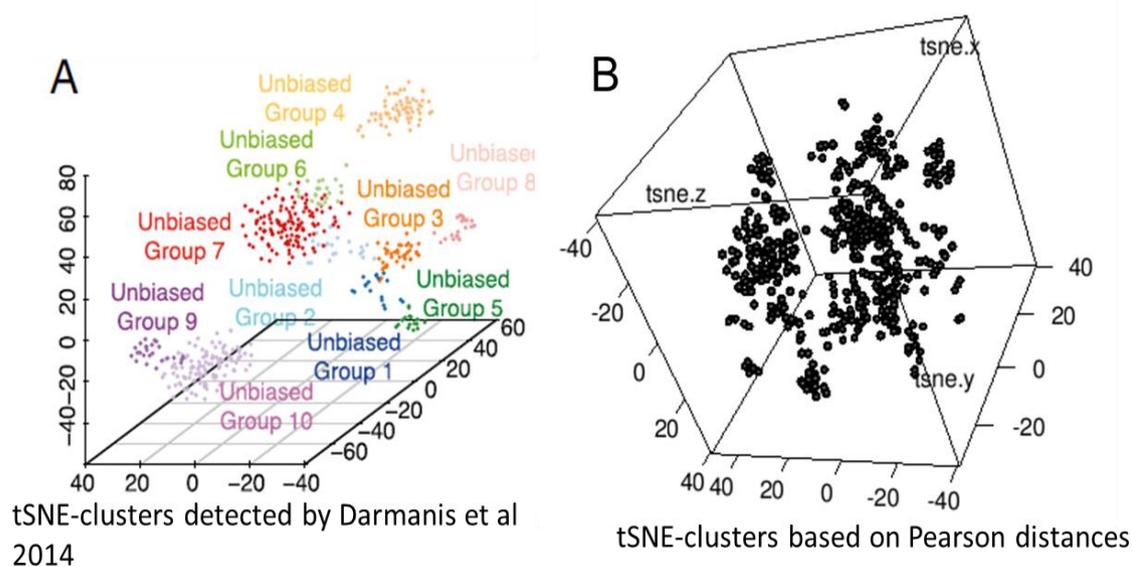
**Table 8: Pre-assigned cell types of Darmanis data and how they were divided into 9 clusters by sc3. The input data consisted of 465 pre-processed cells and the genes were filtered with the default settings of sc3.**

	1	2	3	4	5	6	7	8	9	row sums
astrocytes	0	1	0	0	60	1	0	0	0	62
endothelial	0	0	0	0	0	0	0	0	20	20
fetal_quiescent	0	0	108	2	0	0	0	0	0	110
fetal_replicating	0	0	0	4	0	0	1	20	0	25
hybrid	2	34	0	0	0	2	6	0	1	45
microglia	0	0	0	0	0	1	1	0	14	16
neurons	40	84	0	1	1	0	4	0	1	131
oligodendrocytes	1	1	0	0	0	34	2	0	0	38
OPC	0	0	0	0	0	0	14	0	4	18
column sums	43	120	108	7	61	38	28	20	40	

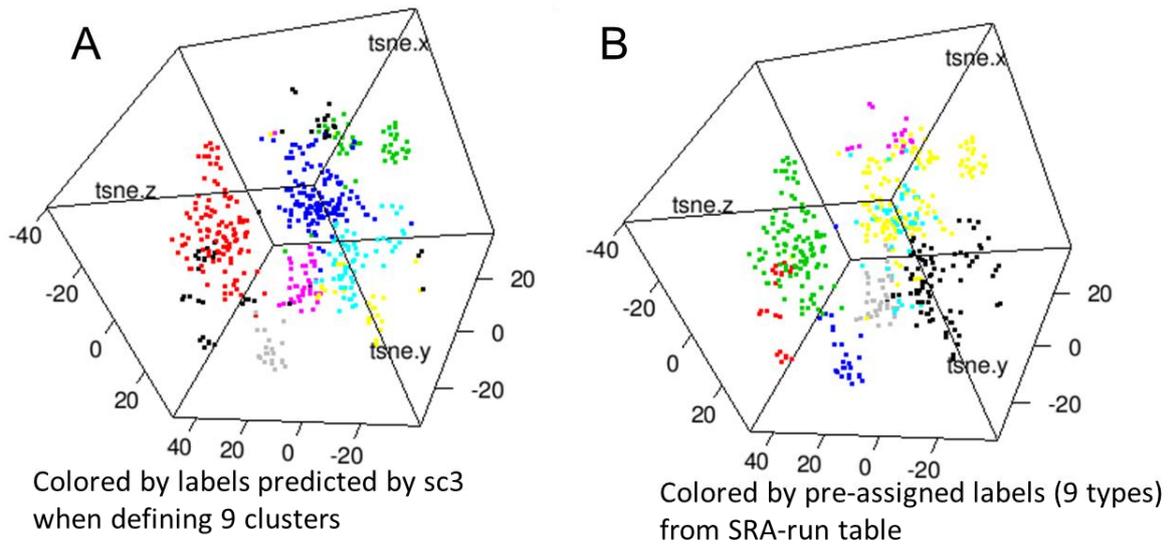
Darmanis data seemed to contain features that were challenging for sc3-algorithm. Especially hybrid and neuron cells were highly dispersed into multiple clusters. Astrocytes, oligodendrocytes, microglia and OPC-cells were separated quite well. The algorithm was also run with pre-defined amount of clusters that was set to the number of provided labels (9). The heatmap is presented in Figure 27 and the result is further checked in Table 8

### 5.9 Applying tSNE for preprocessed Darmanis

The comparison of the original clustering by Damanis et al. (panel A) with the reproducing trial (Panel B) are presented in Figure 28. Roughly 7 or 8 bigger clusters and 2 or 3 small ones seems



**Figure 28: tSNE (T-Distributed Stochastic Neighbor Embedding) visualizations. Panel A represents the result by Darmanis et al. while Panel B shows the embedding produced with Pearson distances (SC3-package). In both plots, X- Y and Z coordinates show the tSNE-coordinates based on Pearson distances**



**Figure 29** Assigning predicted and provided labels into tSNE visualization to compare clusters. X- Y and Z coordinates show the tSNE-coordinates based on Pearson distances

to occur. The reproducing trial produced a different scaling, probably due to differing preprocessing methods.

For visualization purposes, cells were also colored with sc3-predicted labels as well as the provided labels, as seen in Figure 29. By this it is easier to see how the clusters are separated. Interestingly, sc3 seems to have assigned most of the potential outlier cells into same black cluster (Panel A), while in panel B they are assigned into separate clusters. Provided labels show some dispersion between yellow and light blue clusters.

## 6 Discussion

The nature of this work was highly technical, for the major objective was to investigate approaches for scRNA-seq analysis and make a pipeline for low-level analysis of the data. A three-step pipeline utilizing various command line tools and Bioconductor R-packages was implemented. This pipeline performs preprocessing, transcript quantification, filtering, normalization and simple downstream steps. Most importantly, it produces both visual and statistical information for estimating various general features and quality properties of the data. The pipeline was successfully used to process two public scRNA-seq datasets, and it also clearly pointed out the differences between the datasets. However, many pitfalls – both analytical and technical – were encountered and some of them still need solutions.

The early quality processing of the libraries remains quite un-established in the literature. First, trimming is used with various practices to increase mapping rates, but the effect and necessity when using novel alignment-free approaches (See chapter 2.3) for gene expression quantification is a curious question. Intuitively, since the adapter sequences are not part of the reference transcriptome, Salmon should be robust to them and adapter trimming is more problematic issue with traditional alignment protocols. During the thesis, it was indeed noted that the mapping rate was similar for both untrimmed and trimmed samples. Second interesting question is, how to deal with the high duplication rates and GC-contents measured by FASTQC (see Figure 12). Since the protocols for bulk-experiments and single-cell experiments are different, these quality issues may originate for different reasons.

Dealing with the quality issues considering duplicated reads, GC-bias and sequence content bias is not trivial in any experiment. First, de-duplicating the samples with high duplicate rates can distort the data (<https://sequencing.qcfail.com/articles/libraries-can-contain-technical-duplication/>, 5/6/2018 ). Second, the origin of GC-bias may be due to the duplication or can also be affected by the existence of non-human sequences ([http://bioinforma-core.org/index.php/9th\\_Discussion-28\\_October\\_2010#Wrong\\_library](http://bioinforma-core.org/index.php/9th_Discussion-28_October_2010#Wrong_library), 5/6/2018). Finally, the sequence content bias (see Figure 11A) that intuitively would be easy to fix by trimming the first 10 base positions, would not be a good solution as explained in <https://sequencing.qcfail.com/articles/positional-sequence-bias-in-random-primed-libraries/> (5/6/2018).

In general, one remarkable challenge in quality analysis is that the concept of good quality related to certain features may vary depending on the data. For example, the comparison of the

two datasets with the pipeline shows that Darmanis data contained quite many cells with remarkable mitochondrial feature proportion. The high proportion of mitochondrial genes may be indicative for e.g. apoptosis and thus bad quality, but this may not be a valid assumption if the analyzed cell type is naturally rich with mitochondria. Mitochondria are shown to play a vital role in neural functions (Picard & McEwen 2014) and brain tissue also requires lot of energy, suggesting that the high mitochondrial proportion might be quite normal.

Another quite clear difference between the datasets was that Darmanis library size histogram (Figure 15) shows more cells with remarkably high library sizes compared into the median. This may again be explained by some biological phenomena, e.g. the potentially large size of neurons (the larger cytoplasm, the more gene expression etc.). Or, it might be that some of these samples actually contain material from more than one cells, or that the sequence content of some cells was more favorable for amplification. It may even be possible that the samples have been contaminated with DNA or other RNA-types than mRNA, mainly rRNA and tRNA. Or, maybe ERCC-control usage had some effect to this.

The concept of good quality may also vary depending on the used methods. For example, one quite common metrics for sample quality in any sequencing experiment has been the mapping rate. As seen in Figure 15, both datasets show much lower mapping rates than usually aimed with traditional aligning protocols. One clear reason for this is that the quantification is done against the reference transcriptome and not against the genome, like in many alignment protocols. Thus, it is very probable that samples contain sequences, for example novel isoforms, that are not mapped since the corresponding sequences simply do not exist in the reference. On the other hand, samples may also contain DNA or RNA contamination that decreases the mapping rate for the same reason. Finally, for example Bacher & Kendzierski mention that the mapping rates typically are lower in scRNA-seq (Bacher & Kendzierski 2016).

The quality histograms of Chapter 5.3 together with the filtering summary in Chapter 5.4 demonstrate that MAD-based filtering should be done very carefully and adjust the filtering thresholds depending on the data. The current implementation in the pipeline for feature-specific filtering is probably inconveniently strict for it applies the same threshold for all features. Thus, before choosing the filter, the forms of the distributions of the relevant features should be checked and do some background investigations into biological phenomena potentially affecting to the features. A plot by scater which visualizes the top 50 genes based on expression frequencies over the data (Figure 21) helps to gain insight of the dataset.

The alternative for feature-specific filtering is the PCA-based approach, which takes several features into account at once. In the pipeline, the default feature set by scater was overwritten by using the set of other features (currently including "total\_counts", "total\_features", "pct\_counts\_Mt" and "pct\_counts\_top\_200\_features"), since the default settings are partially based on spike-ins. In general, PCA is less aggressive but depending on used feature set, different cells can be marked as outliers. Pipeline does not currently show, which was the final criteria by PCA to drop a cell, but it would be interesting to see, how much the results of PCA-filtering and feature-specific filtering overlap.

Gene filtering was performed with several approaches and different levels. In current implementation, the systematic zeros and the genes that were not annotated are removed before any further processing. It was also noted that in annotation phase, multiple ENSG-identifiers sometimes mapped to the same gene, and filtering of this kind of genes was also considered. In current implementation, these genes with their ENSG-identifiers are kept, while uniquely annotated genes are named with their symbolic names. This may not be an ideal way to deal with this kind of duplicates, but since their amount was minor, they should not disturb the downstream analysis too much. Before normalization, the genes with uninterestingly low tpm counts are removed. The effect of this filter is seen in Figure 20. The used threshold (average expression  $< 1$ ) seemed to be suitable for the two datasets, but depending on the data it may be too strict or too gentle. Thus, this is one of the parameters that should be added into configuration file.

Two scRNA-seq specific approaches, SCnorm and scran were tested in normalization phase. SCnorm was not able to normalize Camp data in the pipeline implementation, but produced quite similar normalized values for Darmanis when compared with scran and was also followed by similar sc3-clustering (13 cell types). However, the computation took a long time, and the parallelization approach that clearly is advantageous for bigger dataset, seemed to contain a flaw. Because of the problems, SCnorm is currently excluded and scran is used as primary normalization approach. All in all, even though SCnorm may not be optimal for pipeline usage due to its complexity, it would be interesting to see if its effects on DE-detection is as remarkable as reported in Bacher et al 2017.

It turned out that SC3 method detected more clusters from the data than reported for the datasets. The difference was only small for Camp data, but more remarkable for Darmanis data, in which especially neurons and hybrid cell types seemed to be hard to recognize. Moreover,

tSNE-clusters showed clear variation, but this was expected, since the algorithm has stochastic nature and even the results for one dataset are different in each run (Andrews & Hemberg 2018). Also the scaling of the axes was different, which might result from differing normalization approaches: Darmanis et al used CPM, while the pipeline produces deconvolution-normalized values. The number of clusters in tSNE-plot is however near the expected 10 clusters detected by Darmanis et al.

The differences in cell type identification may be explained by preprocessing and normalization approaches, but it was noticed that both SCnorm-normalized and CPM-normalized expression matrices also led into 13 cell types. The assumptions that the algorithm makes of the data may not be valid, but SC3 might also be more sensitive method than the one used by Darmanis et al: fetal quiescent cells seemed to consist of 3 sub clusters, and also neuron cells were the most abundant in couple of other sub clusters. Finally, since even one point can affect into the clustering result, it is possible that the sample whose fastq-files were truncated and which was thus excluded, was somehow deciding for the clustering.

The nature of technical issues is as complex as the analytical ones discussed above. One challenge in this kind of complex data analysis is, that the parameter space that potentially affect to the analysis is vast, and it may also affect the robustness of the tools. In this thesis, some of the important parameters were investigated, but the pipeline also relies on complex Bioconductor functions that tend to have so many parameters that it is very difficult to make the pipeline as flexible as these functions truly are. For example, SCnorm-method demands the adjusting of multiple parameters in order to work with different kinds of data, which was one of the reasons it was finally excluded. Thus, even though a fully automated pipeline would be ideal, the varying features of the datasets and distinct biological objectives of the experiments often lead in situation, where the interactive approach is more reasonable and reliable.

The parameter space problem above is related to the concept of generalizability, which is a central issue in pipeline development. As soon noticed in this thesis, generalizability has quite an endless amount of levels from sample properties into tool compatibility and into environmental features. Solving all of them is more than can be addressed in a single thesis, but it is important to be aware of relevant features. Some insights are summarized in Table 9.

**Table 9: Summarization of the example features affecting into pipeline generalizability**

	<b>Currently</b>	<b>Insights/Possible development in future:</b>
<b>Organism</b>	Human	Quite easily changed to support mouse, trickier (but possible) for other organisms (some scater and scan functions currently support only human and mouse)
<b>Platform/protocol</b>	Unknown	Currently, paired end sequencing libraries expected by Module 1. Effects of a platform into downstream results hard to define: some of the used Bioconductor functions may be adjusted based on biases typical to certain platforms.
<b>Data set size</b>	Unknown	Not tested. Probably not very informative for small datasets (under 50 cells). The pipeline uses parallelization and performs well with the data sets containing hundreds of cells. Module 1 is slow because of the trimming. Some Bioconductor functions are potentially slow with the datasets containing thousands of cells.
<b>Quantification approach</b>	Salmon	Updating Module 2 R-script and providing expression matrix and annotations as possible inputs → data generated by any aligner could potentially be analyzed with the script. Scater is also compatible with another pseudo-aligner, kallisto
<b>Environment</b>	SLURM-based	Sh-scripts dependent on the environment, since parallelization was implemented based on SLURM and used whenever possible. R-scripts should be environment-independent
<b>Implementation</b>	shell, R	Replacing shell parts with python for clarity and flexibility (e.g. Snakemake)

Additional key aspects of any pipeline or tool are usability and the quality of documentation. Many variations of these features were faced during the project, and it also affected to the decision of whether or not to use a given tool in thesis. “Good documentation” is also partially subjective, and is easily left out of major focus. Usability of this pipeline was tried to maintain by building the script files hierarchically and using clear naming policies with the variables, scripts and functions. Moreover, Module 2 which has the broadest parameter space, was updated to be launched with configuration file. In this file, the purposes of each adjustable parameter is explained. In future, it would probably be ideal to define the parameters of all modules in one (or module-specific) configuration file.

## 7 Conclusions

Single-cell RNA-sequencing is a still developing field of transcriptomics that aims to track the molecular fingerprints of the cells from highly complex, multidimensional data. Not only the complexity of the data but also the different protocols for conducting scRNA-seq experiments together with the novelty and fast development of the analyzing approaches set many challenges for data analysis and pipeline design. In this thesis, the approaches for analyzing the scRNA-seq data were investigated and constructed a three-step pipeline for low-level data analysis, including quality control, normalization and simple downstream steps.

The pipeline was successfully used to process two public scRNA-seq datasets and the objectives of the thesis were met quite well, even though some of the originally planned tools was not tested. A single tool may have quite a steep learning curve, and in this thesis, one of the challenging parts was to combine several new tools that were not yet been used in literature very much. Implementation was also done with quite a minor former experience of similar tasks. Thus, it is clear that more testing is needed for validating the procedure, and some of the used solutions may require further consideration and updates.

All in all, even though a fully automated pipeline would be ideal, the diverse features of the datasets and distinct biological objectives of the experiments often lead in situation, where the interactive approach is more reasonable and reliable. However, since this pipeline produces quite a diverse visual and statistical information of the dataset, the parts of the pipeline are potentially useful for at least routine processing and checking of the dataset. Even in the situation where the pipeline is too inflexible for a given dataset, the information it is providing should be helpful for picking suitable tools and thresholds for more sophisticated methods.

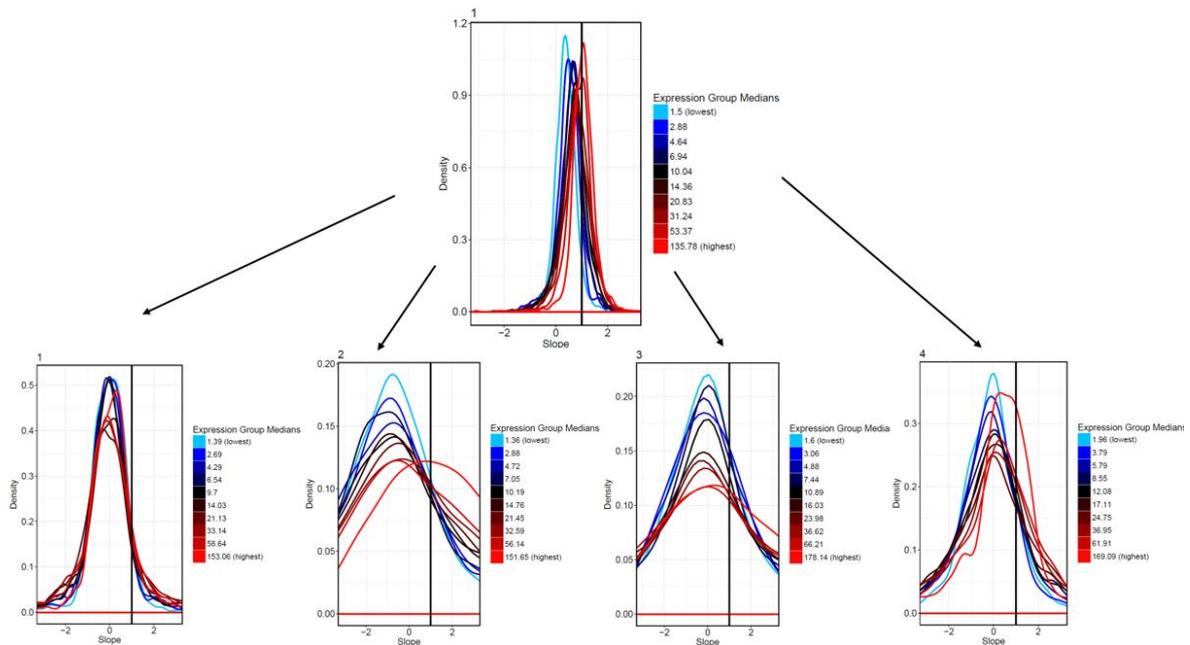
## 8 References

- Aevermann B, McCarrison J, Venepally P, Hodge R et al. Production of a preliminary quality control pipeline for single nuclei rna-seq and its application in the analysis of cell type diversity of post-mortem human brain neocortex. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* 2016;22:564–575.
- Andrews TS, Hemberg M. Identifying cell populations with scRNASeq. *Molecular Aspects of Medicine* 2018;59:114–122.
- Arzalluz-Luque Á, Devailly G, Mantsoki A, Joshi A. Delineating biological and technical variance in single cell expression data. *International Journal of Biochemistry and Cell Biology* 2017;90:161–166.
- Bacher R, Chu L, Leng N, Gasch AP et al. SCnorm: Robust normalization of single-cell RNA-seq data. *Nature Methods* 2017;14(6):584–586.
- Bacher R, Kendzioriski C. Design and computational analysis of single-cell RNA-sequencing experiments. *Genome Biology* 2016:17(1).
- Bray NL, Pimentel H, Melsted P, Pachter L. Near-optimal probabilistic RNA-seq quantification. *Nature Biotechnology* 2016;34(5):525–527.
- Buettner F, Natarajan KN, Casale FP, Proserpio V et al. Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. *Nature Biotechnology* 2016;33(2):155–160.
- Camp JG, Sekine K, Gerber T, Loeffler-Wirth H et al. Multilineage communication regulates human liver bud development from pluripotency. *Nature* 2017;546(7659):533–538.
- Chen M, Zhou X. Controlling for confounding effects in single cell RNA sequencing studies using both control and target genes. *Scientific Reports* 2017;7(1).
- Darmanis S, Sloan SA, Zhang Y, Enge, M et al. A survey of human brain transcriptome diversity at the single cell level. *Proceedings of the National Academy of Sciences of the United States of America* 2015;112(23):7285–7290.
- Davis MPA, van Dongen S, Abreu-Goodger C, Bartonicek, N et al. Kraken: A set of tools for quality control and analysis of high-throughput sequence data. *Methods* 2013;63(1):41–49.
- Dobin A, Davis CA, Schlesinger F, Drenkow J et al. STAR: Ultrafast universal RNA-seq aligner. *Bioinformatics* 2013;29(1):15–21.
- Engström PG, Steijger T, Sipos B, Grant G et al. Systematic evaluation of spliced alignment programs for RNA-seq data. *Nature Methods* 2013;10(12):1185–1191.

- Finak G, McDavid A., Yajima M, Deng, J, et al. MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. *Genome Biology* 2015;16:278.
- Guo M, Bao EL, Wagner M, Whitsett JA et al. SLICE: Determining cell differentiation and lineage based on single cell entropy. *Nucleic Acids Research* 2017;45(7).
- Ilicic T, Kim JK, Kolodziejczyk AA, Bagger FO et al. Classification of low quality cells from single-cell RNA-seq data. *Genome Biology* 2016;17(1).
- Jin H, WanY, Liu, Z. Comprehensive evaluation of RNA-seq quantification methods for linearity. *BMC Bioinformatics* 2017;18(Suppl 4):117.
- Kharchenko PV, Silberstein L, Scadden DT. Bayesian approach to single-cell differential expression analysis. *Nature Methods* 2014;11(7):740–742.
- Kim D, Langmead B, Salzberg SL. HISAT: A fast spliced aligner with low memory requirements. *Nature Methods* 2015;12(4):357–360.
- Kiselev VY, Kirschner K, Schaub MT, Andrews T et al. SC3: Consensus clustering of single-cell RNA-seq data. *Nature Methods* 2017;14(5):483–486.
- Krishnaswami SR, Grindberg RV, Novotny M et al. Using single nuclei for RNA-seq to capture the transcriptome of postmortem neurons. *Nature Protocols* 2016;11(3):499–524.
- Leng N, Choi J, Chu L, Thomson JA et al. OEFinder: A user interface to identify and visualize ordering effects in single-cell RNA-seq data. *Bioinformatics* 2016;32(9):1408–1410.
- Lun ATL, Bach K, Marioni JC. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biology* 2016a;17(1).
- Lun ATL, McCarthy DJ, Marioni, JC. A step-by-step workflow for low-level analysis of single-cell RNA-seq data *F1000Research* 2016b;5:2122.
- Martin M. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnetjournal* 2011;17(1):10–12.
- McCarthy DJ, Campbell KR, Lun ATL, Wills QF. Scater: Pre-processing, quality control, normalization and visualization of single-cell RNA-seq data in R. *Bioinformatics* 2017;33(8):1179–1186.
- Ofengeim D, Giagtzoglou N, Huh D, Zou C et al. Single-cell RNA sequencing: Unraveling the brain one cell at a time. *Trends in Molecular Medicine* 2017;23(6):563–576.
- Patro R, Duggal G, Love MI, Irizarry RA et al. Salmon provides fast and bias-aware quantification of transcript expression. *Nature Methods* 2017;14(4):417–419.
- Picard M, McEwen BS. Mitochondria impact brain function and cognition. *Proceedings of the National Academy of Sciences of the United States of America* 2014;111(1):7–8.

- Poirion OB, Zhu X, Ching T, Garmire L. Single-cell transcriptomics bioinformatics and computational challenges. *Frontiers in Genetics* 2016;7:163
- Satija R, Farrell JA, Gennert D, Schier AF et al. Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology* 2015;33(5):495–502.
- Srivastava A, Sarkar H, Gupta N, Patro R. RapMap: A rapid, sensitive and accurate tool for mapping RNA-seq reads to transcriptomes. *Bioinformatics* 2016;32(12):192–200.
- Stegle O, Teichmann SA, Marioni JC. Computational and analytical challenges in single-cell transcriptomics. *Nature Reviews Genetics* 2015;16(3):133–145.
- Svensson V, Natarajan KN, Ly LH, Miragaia RJ et al. Power analysis of single-cell RNA-sequencing experiments. *Nature Methods* 2017;14(4):381–387.
- Teschendorff AE, Enver T. Single-cell entropy for accurate estimation of differentiation potency from a cell's transcriptome. *Nature Communications* 2018;8:15599.
- Trapnell C, Cacchiarelli D, Grimsby J, Pokharel P et al. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nature Biotechnology* 2014;32(4), 381–386.
- Vallejos CA, Marioni JC, Richardson S. BASiCS: Bayesian analysis of single-cell sequencing data. *PLoS Computational Biology* 2016;11(6)
- Vallejos CA, Risso D, Scialdone A, Dudoit S et al. Normalizing single-cell RNA sequencing data: Challenges and opportunities. *Nature Methods* 2017;14(6):565–571.
- Williams CR, Baccarella A, Parrish JZ, Kim CC. Trimming of sequence reads alters RNA-seq gene expression estimates. *BMC Bioinformatics* 2016;17(1).
- Yuan GC, Cai L, Elowitz M, Enver T et al. Challenges and emerging directions in single-cell analysis. *Genome Biology* 2017;18(1).
- Ziegenhain C, Vieth B, Parekh S, Reinius B et al. Comparative analysis of single-cell RNA sequencing methods. *Molecular Cell* 2017;65(4):631–643.
- Zielezinski A, Vinga S, Almeida J, Karlowski WM Alignment-free sequence comparison: Benefits, applications, and tools. *Genome Biology* 2017;18(1).

## Appendix A: Applying SCnorm for Darmanis data



**Figure A: Testing SCnorm for Darmanis data (Density vs. Slope).** The estimation of 10 expression groups for un-normalized data shows that there is minor deviation between the groups. The algorithm detected 4 different conditions from the un-normalized data. The algorithm normalized each detected condition separately, aiming to scale all 10 expression group modes within 0.1 of zero in each condition. The normalized count-depth relationships of each condition are shown.