JUSTUS-LIEBIG-
UNIVERSITÄT
GIESSEN

Bioinformatic management and analysis of single cell RNA sequencing data

Inaugural Dissertation
for the award of the degree of
Doctor rerum naturalium (Dr. rer. nat.)

by

Andreas Hoek

submitted to the
Faculty of Biology and Chemistry

prepared at the
Department for Bioinformatics & Systems Biology

Justus Liebig University Giessen
Giessen, February 2025

1st Supervisor	Prof. Dr. Alexander Goesmann Bioinformatics and Systems Biology Justus Liebig University Giessen
2nd Supervisor	Prof. Dr. Stefan Janssen Algorithmic Bioinformatics Justus Liebig University Giessen

Declaration

I declare that I have completed this dissertation single-handedly without the unauthorized help of a second party and only with the assistance acknowledged therein. I have appropriately acknowledged and cited all text passages that are derived literally from or are based on the content of published work of others, and all information relating to verbal communications. I consent to the use of an anti-plagiarism software to check my thesis. I have abided by the principles of good scientific conduct laid down in the charter of the Justus Liebig University Giessen „Satzung der Justus-Liebig-Universität Gießen zur Sicherung guter wissenschaftlicher Praxis“ in carrying out the investigations described in this dissertation.

Date

Signature

Abstract

Since its development in 2009, single cell RNA sequencing (scRNA-seq) has been a game-changer in understanding cellular biology, offering unprecedented resolution. Over the past decade, advancements in scRNA-seq technology have greatly enhanced throughput, specificity, and cost-effectiveness, leading to profound insights into organ development, tissue heterogeneity, cellular communication, and disease mechanisms.

However, despite its benefits, scRNA-seq introduces new complexities, particularly in bioinformatic analysis. While the landscape of bioinformatic tools has considerably expanded with the rise of next generation sequencing (NGS), not all tools are suited for the processing of single cell data. Unique challenges, such as barcode detection, separating cell RNA from ambient RNA, and normalizing zero-inflated count matrices, necessitate specialized solutions while simultaneously increasing the entry barrier to data analysis. This challenge is further complicated by the growing amount of data and meta data in the field of single cell transcriptomics. In addition to the growing demand for storage capacity, it is becoming increasingly complex for researchers to keep track of available data as well as to organize or share their own experimental data for analysis, making it more difficult to extract the optimal outcome from an experiment. Therefore, standardized storage and data management solutions are crucial to ensure efficient research in the future.

To overcome these challenges, WASP (**W**eb-**A**ccessible **S**ingle Cell RNA-seq Analysis **P**latform) was developed. WASP provides comprehensive preprocessing and downstream analysis, including clustering of cellular populations and identification of marker genes. By providing automated workflows and a user-friendly interface, WASP greatly reduces the entry barrier for researchers, enabling them to perform data analysis independently. Moreover, WASP supports protocols from various manufacturers, offering

researchers flexibility in their choice of single cell platforms. To target the amount of growing scRNA-seq data, WASP has further been integrated into the openBIS research data management platform, facilitating the organization and storage of experimental single cell data as well as WASP analysis results. Leveraging cloud storage and virtual machines provided by the German Network for Bioinformatics Infrastructure (de.NBI), WASP enables direct analysis of single cell data within the openBIS environment, enhancing accessibility and promoting FAIR data practices.

Zusammenfassung

Seit ihrer Enwicklung 2009 hat die Einzelzell-RNA-Sequenzierung (scRNA-seq) das Verständnis der Zellbiologie mit einer bisher unerreichten Auflösung fundamental verändert. Über die letzten zehn Jahre haben Fortschritte in der scRNA-seq-Technologie den Durchsatz, die Spezifität und die Kosteneffizienz bedeutend verbessert, was in tiefgreifenden Erkenntnissen über Organentwicklung, Heterogenität von Geweben, zelluläre Kommunikation und Krankheitsmechanismen gemündet hat.

Trotz dieser Vorteile bringt diese Technologie jedoch auch neue Herausforderungen mit sich, insbesondere im Bereich der Bioinformatik. Zwar hat sich die Anzahl der Bioinformatik-Werkzeuge mit dem Aufkommen der Sequenzierung der nächsten Generation (NGS) deutlich erweitert, jedoch sind nicht alle diese Werkzeuge für die Auswertung von scRNA-seq-Daten geeignet. Neue Herausforderungen wie die Erkennung von Barcodes, Unterscheidung zwischen Zell- und freier RNA aus beschädigten Zellen sowie die Normalisierung von nulllastigen Expressionsmatrizen erfordern spezielle Lösungen und erhöhen die Einstiegshürde in die Datenanalyse. Diese Herausforderung wird durch die stetig wachsende Menge Daten sowie Metadaten auf dem Gebiet der scRNA-seq noch weiter erschwert. Zusätzlich zum wachsenden Bedarf an Speicherkapazitäten wird es für Forschende immer komplexer, den Überblick über verfügbare Daten zu behalten sowie ihre eigenen experimentellen Daten für die Analyse zu organisieren oder zu teilen, was es erschwert ein optimales Ergebnis aus einem Experiment zu erhalten. Daher sind standardisierte Speicher- und Datenverwaltungslösungen von essentieller Bedeutung, um auch in Zukunft eine effiziente Forschung gewährleisten zu können.

Für diese Herausforderungen wurde WASP (**W**eb-**A**ccessible **S**ingle Cell RNA-seq Analysis **P**latform) entwickelt. WASP kombiniert eine automatisierte Vorverarbeitung und anschließende Analyse mit Clustering von Zell-

populationen und Identifizierung von Markergenen mit einer benutzerfreundlichen Oberfläche und senkt die Einstiegshürde für Forschende erheblich. Zusätzlich unterstützt WASP Protokolle verschiedener Hersteller um Benutzenden Flexibilität bei der Wahl der scRNA-seq-Geräte zu bieten. Um der wachsenden Menge an scRNA-seq-Daten gerecht zu werden, wurde WASP in die openBIS-Forschungsdatenmanagementplattform integriert, welche die Organisation und Speicherung von experimentellen Daten sowie von WASP-Analyseergebnissen ermöglicht. Durch die Nutzung von Cloud-Speicher und virtuellen Maschinen, welche vom Deutschen Netzwerk für Bioinformatik (de.NBI) zur Verfügung gestellt werden, ermöglicht die Integration von WASP in openBIS dabei Organisation, Speicherung und eine direkte Analyse von scRNA-seq-Daten innerhalb der openBIS-Umgebung basierend auf FAIR-Prinzipien.

List of abbreviations

3′-UTR	three prime untranslated region
5′-UTR	five prime untranslated region
AEC	alveolar epithelial cell
AI	artificial intelligence
API	application programming interface
APS	adenosine 5´ phosphosulfate
ASCII	American Standard Code for Information Interchange
ATP	adenosine triphosphate
AWS	Amazon Web Services
BCG	Bacillus Calmette-Guérin
BALO	bronchioalveolar lung organoid
BASC	bronchioalveolar stem cell
BAM	Binary Alignment Map
BAT	batch
BCF	bioinformatics core facility
bp	base pair
bwa	Burrows-Wheeler Aligner
CCA	canonical correlation analysis
CD24	cluster of differentiation 24
cDNA	complementary deoxyribonucleic acid

CDS	coding sequence
CLI	command line interface
contig	contiguous genomic sequence
COPD	chronic obstructive pulmonary disease
COVID-19	Coronavirus Disease 2019
CPM	counts per million
CPU	central processing unit
CSS	Cascading Style Sheets
CSV	comma-separated values
DAG	directed acyclic graph
DE	differential expression
de.NBI	German Network for Bioinformatics Infrastructure
dNTP	deoxynucleoside triphosphate
ddNTP	dideoxynucleotide
DNA	deoxyribonucleic acid
DSL	domain-specific language
ELIXIR	European Life Science Infrastructure for Biological Information
ELN	electronic lab notebook
EpCAM	epithelial cell adhesion molecule
EST	expressed sequence tag
ETL	Extract Transform Load

exon	expressed region
FACS	fluorescence-activated cell sorting
FAIR	Findability, Accessibility, Interoperability, Reusability
FISH	Fluorescence *in situ* hybridization
FSC	forward scatter
GB	gigabyte
Gbps	gigabit per second
GEO	Gene Expression Omnibus
GFP	green fluorescent protein
GTF	General Transfer Format
GO	Gene Ontology
GUI	graphical user interface
HPC	high performance computing
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HVG	highly variable genes
IAV	influenza A virus
ID	identity
IMS	immunomagnetic separation
intron	intragenic region
IOPS	input and output operations per second

IP	Internet Protocol
IPF	idiopathic pulmonary fibrosis
IVT	*in vitro* transcription
JSON	JavaScript Object Notation
kb	kilobase
KFO309	Klinische Forschungsgruppe 309
KNN	*k*-nearest neighbor
LDAP	Lightweight Directory Access Protocol
LCM	laser capture microdissection
LIMS	laboratory information management system
LXC	Linux Containers
MAC	message authentication code
MACS	Magnetic-activated cell sorting
MDS	multidimensional scaling
MERS	middle east respiratory syndrome
min	minutes
MNN	mutual nearest neighbors
mRNA	messenger ribonucleic acid
ncRNA	non-coding RNA
NGS	next-generation sequencing
nm	nanometer

openBIS	open Biology Information System
OS	operating system
PC	principal component
PCA	principal component analysis
PCR	polymerase chain reaction
PDF	Portable Document Format
PE	paired-end
PDGFRA	Platelet-derived growth factor receptor A
pi	post infection
PP_i	pyrophosphate
PRIDE	Proteomics Identifications Database
QC	quality control
qPCR	quantitative polymerase chain reaction
RAM	random-access memory
RBS	ribosomal binding site
RIN	RNA integrity number
rMC	lung-resident mesenchymal cell
RNA	ribonucleic acid
RNA-seq	ribonucleic acid sequencing
rRNA	ribosomal ribonucleic acid
SAM	Sequence Alignment Map

SARS	severe acute respiratory syndrome
SARS-CoV-2	severe acute respiratory syndrome coronavirus 2
Sca-1	Stem cells antigen-1
SCGB1A1	Secretoglobin Family 1A Member 1
SE	single-end
SFTPC	Surfactant Protein C
SGE	Sun Grid Engine
SLURM	Simple Linux Utility for Resource Management
SSC	side scatter
SSH	Secure Shell
SMRT	single molecule real time
SMS	single molecule sequencing
SNP	single nucleotide polymorphism
TI	trajectory inference
TR-Mac	tissue-resident yolk sac-derived alveolar macrophages
tRNA	transfer RNA
t-SNE	t-distributed stochastic neighbor embedding
TSV	tab-separated values
3D	three-dimensional
2D	two-dimensional
UCSC	University of California Santa Cruz

UMAP	Uniform Approximation and Projection
UMI	unique molecular identifier
URL	Uniform Resource Locator
USB	Universal Serial Bus
vCPU	virtual CPU
VM	virtual machine
YAML	YAML Ain't Markup Language
YFP	yellow fluorescent protein
ZMW	zero-mode waveguide

Contents

1 Background

> *Science and everyday life cannot and should not be separated.*
>
> **— Rosalind Franklin**
> (Biochemist)

1.1 Protein biosynthesis

One of the central processes of cellular life is protein synthesis. Proteins represent the largest fraction of macromolecules in a cell, function as fundamental building blocks, and play an essential role in almost every task of cellular life [1]. Protein synthesis can be separated into two sub-processes: transcription and translation (Fig. 1). While an organism's deoxyribonucleic acid (DNA) contains the blueprints for proteins in the form of a sequence of nucleobases - also called genes - it is not directly used for generating proteins. Instead, during the transcription process, gene sequences from the DNA are copied into a complementary messenger ribonucleic acid (mRNA) sequence. This process is initiated by binding the ribonucleic acid (RNA) polymerase enzyme to the DNA in a specific region upstream of the gene sequence - also called promoter sequence. Following, the DNA double helix unwinds, and the RNA polymerase begins reading the DNA sequence base by base and generating a complementary chain of nucleotide bases. Compared to DNA the nucleotide thymine is replaced with uracil in RNA sequences. The transcription process is finally stopped when the terminator sequence is reached on the DNA template strand. In many cases, this is dependent on additional proteins (termination factors).

During the second step - translation, the mRNA is bound by the ribosome and the two ribosomal subunits attach to the mRNA molecule. The ribosome reads the mRNA sequence and uses it as a template to generate a chain of amino acids, thereby building the protein. For this step, the ribosome relies upon transfer RNAs (tRNAs) which deliver the amino acids. To ensure incorporation of the correct amino acid, each tRNA

binds to a specific sequence of three nucleotides (also called codon). A specific codon sequence iniates the translation start, also called the start codon, leading to the first tRNA binding to the mRNA. Next, the ribosome moves to the second codon, and the next tRNA binds to this complementary sequence and delivers the second amino acid. Using the enzyme peptidyl transferase, a covalent peptide bond formation between the two amino acids is catalyzed. Following this, the ribosome releases the first tRNA and moves along to the third codon, again calling another specific tRNA with a complementary sequence to the third codon. This process continues until a so-called stop codon is encountered. In this case, no tRNA provides a complementary sequence to the codon, leading to the termination of translation and causing the previously formed polypeptide chain to be released from the ribosome.

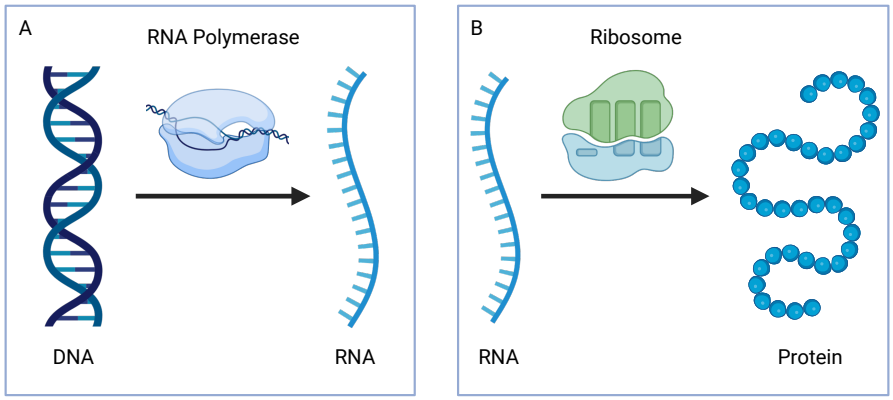


Figure 1: **Protein synthesis start with the DNA.** A) Transcription - RNA polymerase binds one of the strands of the unwound DNA and generates a complementary single-stranded RNA sequence. B) Translation - the ribosomal subunits bind to the RNA and generate a chain of amino acids, the protein, based on triplet sequences of nucleobases. *Figure created with BioRender.com*

The process of utilizing RNA as an intermediate step in protein synthesis provides a variety of advantages:

- RNA can be synthesized fast but also degraded to enable the cell to respond to specific conditions and to regulate gene expression.

- In eukaryotes, a gene can contain intragenic regions (introns) and expressed regions (exons) regions. During the so-called splicing of the pre-mRNA, introns are removed, and different combinations of exons can be merged into the final mRNA, leading to numerous possible proteins that can be synthesized from the same gene sequence.
- Also, in eukaryotes, the transcription occurs in the nucleus, while translation occurs in the cytoplasm. Transporting the mRNA out of the nucleus for translation provides an additional layer of flexibility and control over protein synthesis.

The analysis of all RNAs transcribed by the genome inside a cell or tissue - also called transcriptome - especially of the set of mRNAs, enables detection of differential gene expression, giving a more dynamic description of genetic processes compared to just studying the genome alone. This is very important for the understanding of processes involved in diseases, response to environmental stimuli, and identification of genes belonging to key molecular pathways [2]. It allows a widespread application in biomedical research, *e.g.* cancer research [3], autoimmune diseases [4], or infection-related diseases such as the recently emerged Coronavirus Disease 2019 (COVID-19) [5].

1.2 Sequencing

In 1991, the capture of 609 mRNA sequences from a human brain marked the first study attempting to capture at least a partial cellular transcriptome. For this, commercial complementary deoxyribonucleic acid (cDNA) libraries were chosen and sequenced based on mRNA human brain isolates from the hippocampus and temporal cortex [6]. This approach is also known as expressed sequence tag (EST) analysis, in which cDNAs are synthesized based on mRNAs isolated from a cell. These cDNA libraries are then used to generate short sequences of the cDNAs, which are then sequenced [7]. However, due to limitations of the predominant Sanger sequencing method at that time, this approach was expensive, had relatively low throughput, and did not allow quantitative measure-

ments. Consequently, other techniques have been developed, enabling quantification and enabling a higher throughput [8] [9].

One technology to overcome the mentioned issues are microarrays, consisting of a solid substrate, *e.g.* a glass slide with short nucleotide oligomers fixed on the substrate as probes. In the course of a microarray experiment, mRNA is extracted from the cell, translated into cDNA, and binds to complementary nucleotide probes on the microarray substrate. In addition, transcripts can be labeled fluorescently to use the intensity for measuring transcript abundance. This allows thousands of transcripts to be analyzed simultaneously, saving cost and labor time. However, generation of the nucleotide probes requires prior knowledge of the cell or organism of interest to generate sequences complementary to the targeted mRNAs. Furthermore, this prevents an analysis of the whole transcriptome or all mRNAs inside a cell, as only the sequences of genes complementary to probes will be bound to the substrate [8].

With the advent of the next-generation sequencing (NGS) technologies in the 2000s, mRNA sequencing experienced a resurgence (Fig. 2). By enabling relatively inexpensive sequencing at a massive scale, ribonucleic acid sequencing (RNA-seq) revolutionized the field of transcriptomic analysis [9]. Similar to the sequencing of ESTs, the mRNA initially has to be transformed into cDNA, which is then fragmented into short sequences and sequenced. These fragments typically consist of around 100 base pairs (bps), but depending on the technology, they can range from 30 bps to over 10,000 bps. Due to the high throughput, RNA-seq allows quantification of reads - the analyzed nucleotide sequence - with a lower overall sample volume than microarrays [8]. Furthermore, as this approach can - at least in theory - detect all RNAs inside a cell, it enables analysis of the whole transcriptome instead of only a selected number of genes as in microarrays.

While several companies have developed their own technologies and devices, the NGS market is nowadays almost exclusively dominated by the company Illumina. Their sequencing technology is used to generate more than 90% of the world's sequencing data [11] [12]. The method Illumina uses was developed by the British chemists

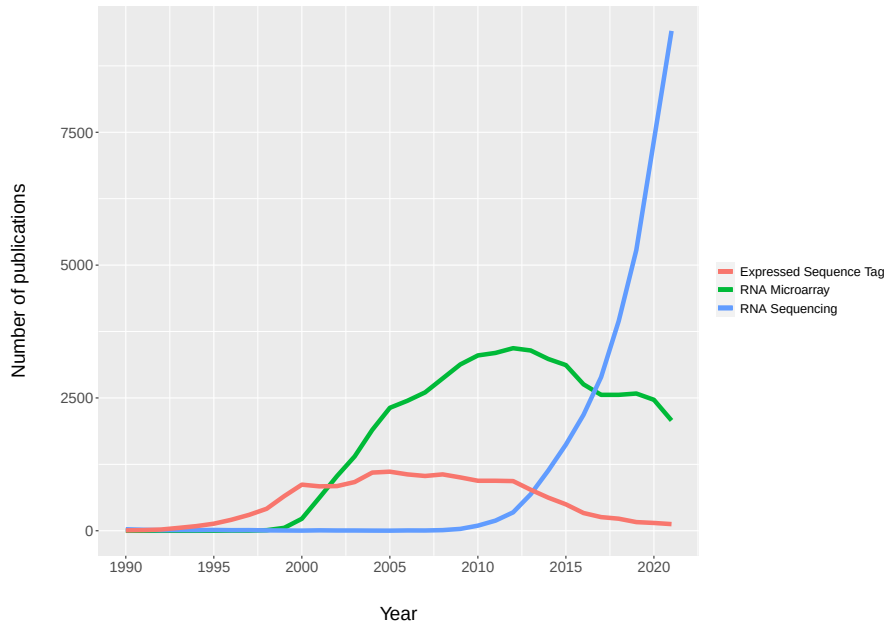


Figure 2: **Transcriptomic methods used in publications.** Number of publications referring to one of the three mainly used transcriptome analysis technologies since 1990 [10].

Shankar Balasubramanian and David Klenerman. To commercially exploit their approach, they founded a company named Solexa, which released its first sequencing device - the Genome Analyzer - in 2006. Solexa was then acquired by Illumina in the same year [13]. To create a sequencing library suitable for Illumina, DNA is fragmented and adapter sequences are ligated to the ends of each fragment. Afterwards, the DNA fragments are distributed onto a glass flow cell, bound to oligonucleotides complementary to the adapter sequences. First, bound DNA is replicated by the so-called 'bridge amplification' method. During this process, DNA strands arch over and hybridize to an adjacent oligonucleotide that is also fixed to the flow cell's surface, and a polymerase chain reaction (PCR) is performed in order to produce multiple copies of the DNA template. As a result, 'bridge amplification' produces about 1,000 clonal molecules from each single DNA molecule [13] [14]. Second, sequencing is initiated with hybridizing a primer complementary to one of the adapter sequences. Also, DNA polymerase and modified deoxynucleoside triphosphates (dNTPs) are added, and the sequencing

process is performed in cycles, with one nucleotide being synthesized during each of them. The modified dNTPs are labeled with different fluorophores which occupy the 3' hydroxyl position, thereby preventing incorporation of more than one dNTP. Incorporated nucleotides are then identified by exciting the fluorophores with a laser and monitoring the signal with a charge-coupled device (CCD). In the last step of each cycle, the fluorescent components are cleaved and, washed away, and the next cycle begins [13] [14] [15].

1.3 Bulk RNA sequencing

The first NGS-based RNA-seq study of an EST library in 2007 [16] marks the beginning of the extensively used analysis method of so-called bulk RNA-seq. Here, the name bulk refers to the fact that a whole tissue or cell population is used as starting material for the following sequencing. Therefore, the sequencing results are able to then give insights into the transcriptome of all sample cells combined. However, as a cell contains various different RNA types, an RNA-seq study might require a specific protocol to obtain the type of RNA which should be analyzed. The typical steps in such an experiment include isolation of and selection of RNA, preparation of a sequencing library, and finally the sequencing itself on an NGS platform (Fig. 3) [17]:

- RNA isolation: As a first step, the RNA must be isolated from the biological sample. For this, the cells must be lysed, RNA separated from other cellular components such as DNA or proteins. After extraction, the quality of the RNA should be measured, *e.g.* by calculating the RNA integrity number (RIN) based on gel electrophoresis and analysis of the ratios of 28S to 18S ribosomal subunit bands [18]. Using samples with a bad quality can negatively influence sequencing and lead to incorrect biological interpretations.

- RNA selection: Typically, this step includes selecting desired RNA species, reverse transcription into cDNA, often amplifying and ligating required adapters to be sequenced with an NGS platform. As mentioned before, RNA-seq enables

analysis of the full transcriptome so selecting the correct RNA species based on the research question is crucial. Generally, a cell contains a variety of different types of RNAs [17]:

- ribosomal ribonucleic acid (rRNA)
- precursor mRNA
- mRNA
- various classes of non-coding RNA (ncRNA)

The highest fraction of RNA species in most cell types consists of rRNA with over 95% in most cell types. As a result, rRNA transcripts are typically removed from the sample as their presence would reduce the fraction of other RNA molecules and thus impact detection of other RNA species of interest. This can be achieved by actively removing or depleting rRNAs with commercial kits. Another option is to enrich a specific RNA type selectively. In the case of the typical gene expression analysis experiment, mRNAs can be selected based on their polyadenylation. As mRNA molecules in eukaryotes contain a poly-A tail consisting of multiple adenosine monophosphates, they can be bound to complementary poly-T oligonucleotides - consisting of numerous thymine monophosphates - attached to a substrate such as magnetic beads [17] [19].

- Library preparation: After RNA species selection, the sample can be sequenced. Even though the library preparation depends on the NGS platform, most steps are quite universal. The isolated RNA molecules have to be fragmented into smaller pieces. This can be achieved by physical methods such as sonication, enzymatic treatments with *e.g.* the enzyme RNAse II, or chemical treatment such as heat. In the next step, the RNA is conversed into cDNA, which increases the stability of the molecule and is necessary as NGS devices are limited to sequencing DNA molecules. Following, sequencing adapters are ligated to the generated cDNA molecules. Adapters vary depending on the sequencing protocol: in the case of single-end (SE) sequencing, the reads will only be generated from one end, either

3′ or 5′. On the contrary, paired-end (PE) sequencing generates reads from both ends of the molecule with a predetermined length, generating two reads per template. While SE sequencing is usually selected due to lower cost, PE sequencing retains information about detected isoforms of expressed genes [20]. Finally, the constructed sequencing library consisting of the modified cDNA reads is amplified using PCR and sequenced [17] [19].

An exception to the aforementioned steps are the so-called third generation sequencing devices from companies such as PacBio and Oxford Nanopore, which offer direct sequencing of RNA molecules, including generation of long-reads of up to 26,000 bp [21] compared to the typical 75 - 125 bp from short-read sequencing such as Illumina-based devices [19].

After successful preparation and sequencing, the reads can be extracted from the NGS device and bioinformatically analyzed.

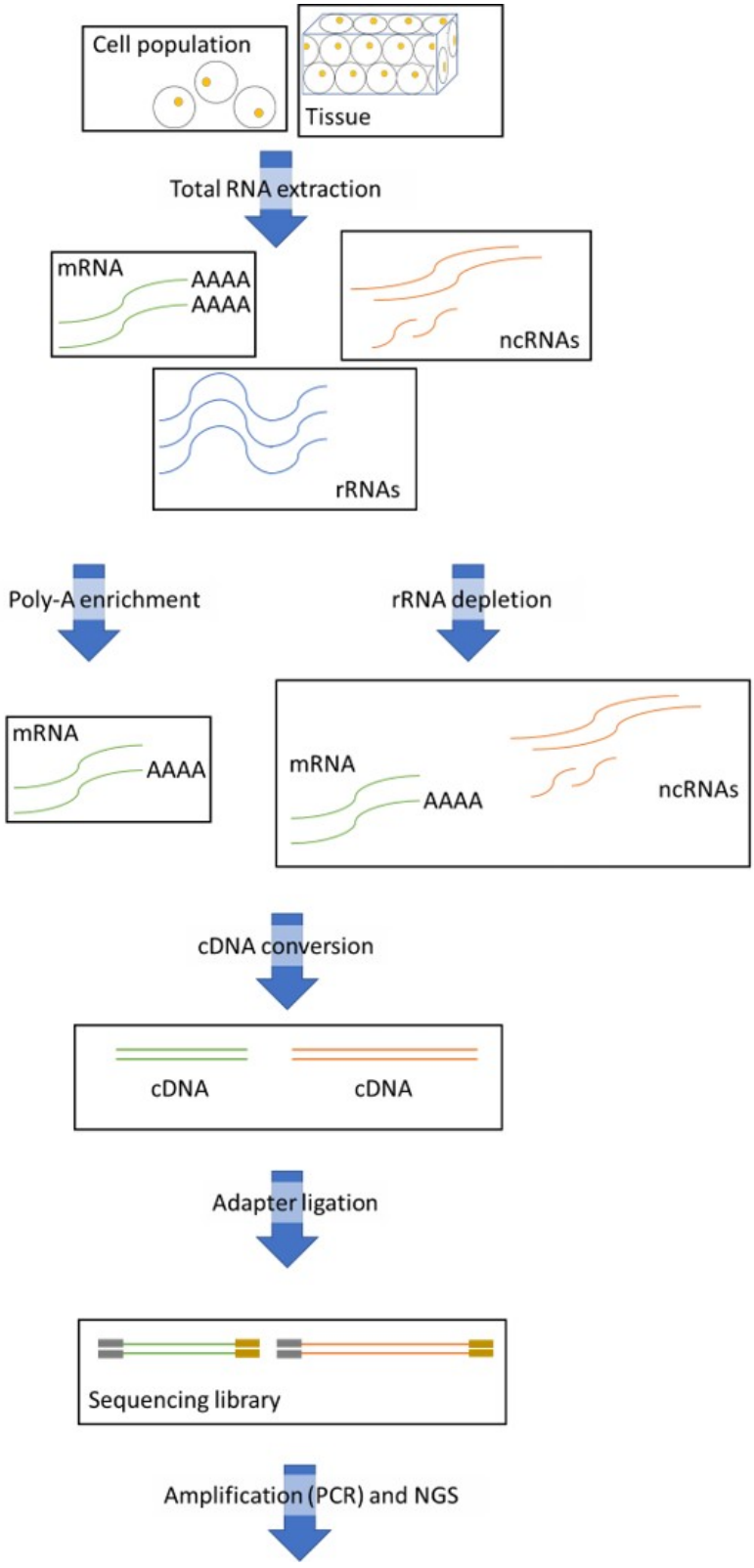


Figure 3: **Main steps conducted in a bulk RNA-seq experiment.**

1.4 Single cell RNA sequencing

While bulk RNA-seq proved to be a revolution to the field of transcriptomics and gave researchers a powerful technique to gain a deeper understanding of *e.g.* gene expression processes and their impact on diseases, it lacks an essential feature: this approach relies on a bulk of cells, *e.g.* a whole tissue, but this also means that the obtained information represents the gene expression averaged over the entire sample. This represents a major issue as it has been shown that heterogeneity does not only occur between tissues but also between cells inside a tissue or with similar phenotypes [22]. The transcriptomic changes inside a cell might be related to changing stimuli as well as mRNA synthesis and decay. Also, tissues comprise a variety of different cell types, that might exclusively express a gene. However, in bulk analysis studies this exclusive expression pattern may be falsely identified as a co-expression with other genes over all cells. Thus, it is not possible to distinguish between *e.g.* different cell types showing different gene expression patterns. This means that especially rare cell types with a low abundance in a sample are not visible in the data and results will be dominated by cells with higher abundances. An illustrative example for this is the development of cancer in *e.g.* a human body. Even though a typical human body consists of approximately 37.2 trillion cells, a single cell can be sufficient to lead to the downfall of the entire organism [23].

In order to overcome the limitations of bulk sequencing, new protocols and techniques have been developed to increase the resolution up to a single cell level - the fundamental building block of every living organism. The first experiment that was successfully analyzed on a single cell level was published in 2009 with a transcriptome study on blastomeres from a four-cell-embryo stage of mice [24]. This was achieved by modification of a single cell cDNA amplification method used for microarrays. As a first step, a single cell was manually selected using microscopy and lysed afterwards. In order to yield a sufficient amount of input material for the sequencing process, the reverse transcription step was prolonged from 5 minutes (min) to 30 min as well as the PCR step was increased from 3 min to 6 min. The generated cDNA was then frag-

mented by sonication and sequenced with the SOLiD platform [24]. Even though some steps of single cell RNA-seq experiments are similar to bulk RNA-seq, others required entirely new approaches. In general, a single cell RNA-seq experiment can be separated into these major steps:

1. Cell isolation
2. Amplification
3. Sequencing
4. Analysis

1.4.1 Cell isolation

The first step in performing single cell RNA-seq is the separation and isolation of cells from tissues or a cell culture. This step is crucial to ensure that each cell can later be processed, its RNA sequenced and analyzed individually. Over time, multiple methods have been developed with a range of advantages and disadvantages [25] [26].

Serial dilution is based on the Poisson distribution and is the simplest method which works by diluting a cell population multiple times. As an advantage, this step can be performed using hand pipettes or pipetting robots with, *e.g.* 96-well plates resulting in a convenient and low-cost operation (Fig. 4). However, on the downside, erroneous cell separation may lead to wells containing multiple cells. Furthermore, this method is time-consuming, provides only a low throughput, and cannot filter target cells [25].

Micromanipulation describes the approach of selecting a cell from a cell suspension via microscopy and using a capillary pipette to suck up the cell (Fig. 5). The advantage of this method is the possibility to inspect the cellular morphology and coloring before selecting a cell, thus enabling the filtering of cells. However, the drawback of this method is also the time-consuming manual labor, a low throughput of cells, and possible cell damage due to mechanical shearing [25] [27].

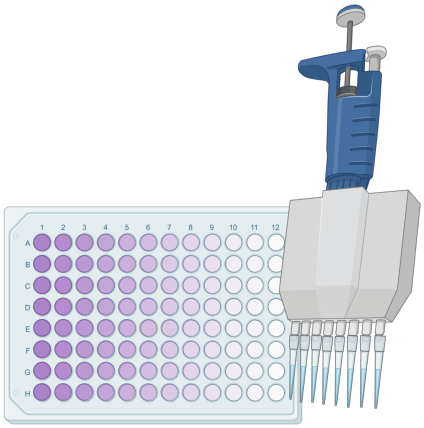


Figure 4: **Single cell separation by dilution of a cell population in a 96-well plate.** Process starts with the highest cellular concentration on the left side with increasing dilution for each column to the right side. *Figure created with BioRender.com*

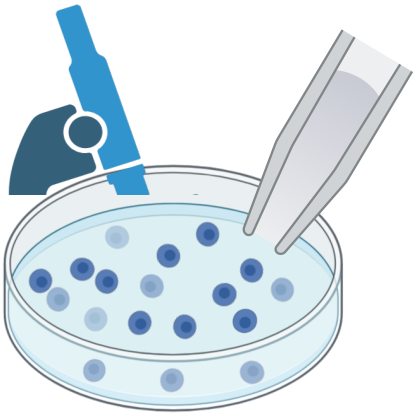


Figure 5: **Single cell separation using a microscope and a pipette.** Cells are mixed in a solution and can be selected using a microscope (left) and a micropipette (right). *Figure created with BioRender.com*

Fluorescence-activated cell sorting (FACS) is a separation technique mainly based on flow cytometry. Suspended cells are flowed through a device under pressure, generating a constant stream of cells passing one or multiple lasers. The resulting light scattering enables measuring of cell size and granularity (Fig. 6). Furthermore, detection of specific cells of interest is possible using fluorescent markers, *e.g.* transfection and expression of fluorescent proteins or staining with fluorescent dyes or antibodies

against known markers. The advantages of this method are a high throughput as well as high accuracy in selecting and separating cells of interest. Disadvantages are its requirement for a large sample size and the limited fluorescence signal intensity which might lead to losing specific cell types, especially in case of low abundance cells. Furthermore, cells might get damaged due to the rapid liquid flow in the device or the fluorescent dyes [26] [28].

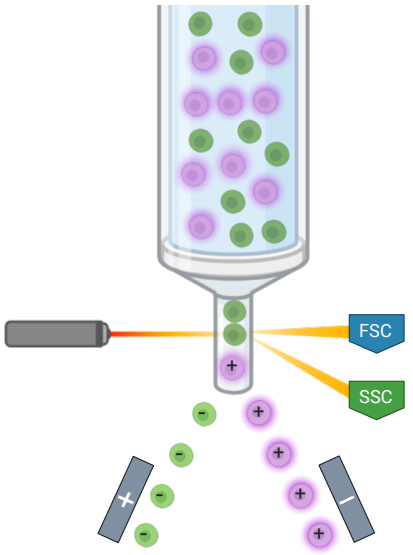


Figure 6: **Single cell separation using a flow cytometry system.** Cells are shown in green and purple and flow as a suspension through the device. A laser is used to measure the fluorescence of cells after treatment with specific fluorescent proteins or antibodies. Depending on fluorescence, cells are charged and then separated based on their charge. Two optical detectors measure light scattered by cells, forward scatter (FSC) can be used to discriminate between cells based on their size, and side scatter (SSC) can be used to analyze internal complexity of the cell such as granularity. *Figure created with BioRender.com*

Immunomagnetic separation (IMS) utilizes magnetic beads with antibodies binding to cell surface antigens of interest. This results in connecting only cells with the corresponding surface marker to the beads, enabling separation of cells of interest (Fig. 7). Compared to FACS, IMS benefits from lower costs and less instrumental needs. However, depending on the phenotype of interest, selection might be quite complex.

Also, a known surface marker is required to apply this technique, and compared to FACS it is only possible to select positive or negative marker expression, but not based on the expression level of the marker. Another disadvantage of IMS are possible alterations of phenotype or viability of the cell due to a possible continuous binding or engulfing of bead components after separation [26] [29].

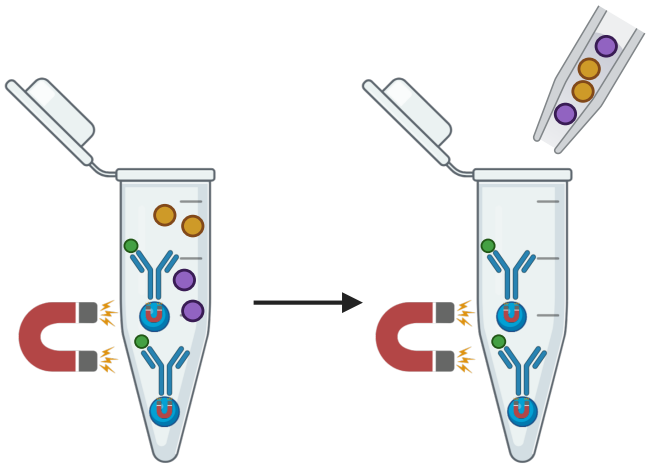


Figure 7: **Single cell separation using magnetic antibodies.** Magnetic antibodies (blue) bind to the cell's (purple, yellow, green) surface antigens of interest. By using a magnet, antibodies with their bound cells remain in the reaction tube while the supernatant containing other cells can be removed with *e.g.* a pipette. *Figure created with BioRender.com*

Laser capture microdissection (LCM) describes cutting cells out of tissue samples using a laser beam. The laser is coupled to a microscope enabling the user to select a region of interest. Following the cutting process, an extraction process is carried out to fix the cells, *e.g.* by melting a transparent membrane cover onto the cut-out region (Fig. 8). A major advantage of this method is the preservation of a spatial location of cells and manual selection of a specific region in the sample. Shortcomings of LCM include high cost, a low throughput of cells, and possible damage to cells as a result of the cutting process. This might lead to loss of chromosomal fragments or damaged nuclei of adjacent cells. Also, this method is less suitable for single cell transcriptomic studies as LCM often leads to RNA degradation within the sample [26].

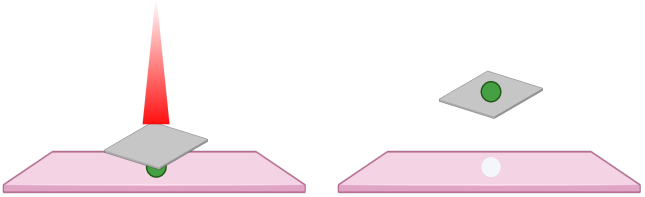


Figure 8: **Single cell separation using laser capture microdissection.** The cell of interest (green) is microscopically selected in a tissue (pink) and cut by a coupled laser beam (red). Following this, a medium such as a transparent membrane cover is melted onto the cut-out region to extract the cell. *Figure created with BioRender.com*

Microfluidic platforms separate cells based on properties such as diameter or surface antigens. For this, the cells are processed in small volumes of fluids in channels of small dimensions of 10 - 100 micrometers. A widely used commercial implementation was the microfluidic chip C1 from Fluidigm, containing the fluidic channels in which cells are captured. After capturing, cells can be readily processed for sequencing library generation on the chip, including cell lysis, reverse transcription of mRNA into cDNA, and amplification. Due to the micro reaction volumes used in this method, a high throughput with low reagent consumption is achieved, leading to reduced cost. Furthermore, accuracy and efficiency of the amplification process benefit from the small volumes, and cell contamination rates are reduced. However, these benefits come with strict equipment requirements including high-priced devices. Other drawbacks include an increased input volume of at least 1,000 cells and a homogenous cell size limit [26] [30].

Microdroplet-based microfluidics is a different approach in the field of microfluidics devices. Instead of using, *e.g.* a chip, single aqueous droplets containing lysis buffer are dispensed in a continuous oil phase. These droplets are brought together with a flow of the sample cells and beads, leading to encapsulation of cell and bead into droplets (Fig. 9). Subsequently, the cells are lysed and their RNA is released into the droplet. Using poly-T oligonucleotides bound to the beads, mRNA from the cell is bound with its poly-A tail and prepared for the next analysis steps.

Compared to older microfluidic systems such as chips, even lower volumes are required for cell preparation, leading to a much higher number of tens of thousands to millions of cells which can be analyzed in one run at reduced cost. This higher sample number allows the analysis of also rare cell types in a larger sample. However, similar to, *e.g.* chip-based systems, a higher number of input cells is required. Furthermore, droplets might accidentally capture two cells that will be processed together or even capture no cell at all but instead free mRNA from a cell damaged inside the device leading to false results. Also, analysis is limited to 3' ends of the RNA instead of full transcripts. Finally, the reduced processing cost is met with expensive devices [30] [31].

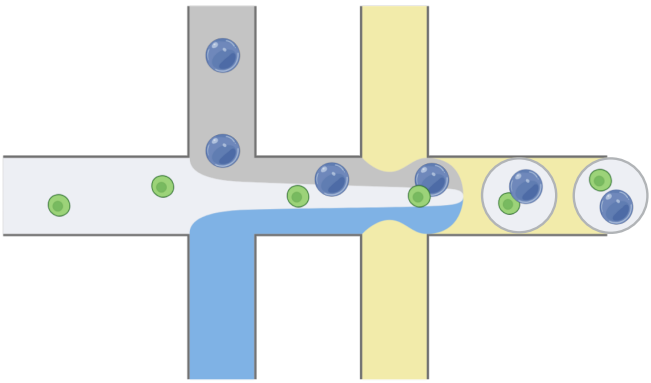


Figure 9: **Single cell separation using a microfluidics device.** The cell suspension (green) flow is mixed with beads (blue dots) and lysis buffer, finally flowing into an oil phase. As a result, droplets consisting of lysis buffer containing cells and beads are formed. *Figure created with BioRender.com*

Reversible hydrogel utilizes liquid-gel-liquid transitions to separate and process single cells. For this, cells in a suspension are labeled with a chemical linker. In the second step, the cell solution is mixed with a solution containing beads in a gelation tube. Homogenization with a micropipette leads to collisions and thus coupling of an individual labeled cell with an individual bead. Afterwards, the new mixture is diluted into a hydrogel solution containing thermal-sensitive polymers. The gelation process is then performed by incubation on ice, leading to immobilization of the bead-cell complexes. In the next step, lysis buffer is added, which diffuses into the hydrogel, enables lysis of the cells, and results in the release of mRNAs that are bound to poly-T sequences of

the coupled bead. Finally, the hydrogel is degelated, beads can be extracted and the mRNAs processed.

A great benefit of this method is its independence from specific instruments or devices. All of the previously mentioned steps can be performed using standard laboratory equipment, enabling to be carried out at variable sample collection points, possibly reducing cellular stress and processing delays necessary in case of sample transfer. Furthermore, no up-front investment in possible expensive devices is necessary. Compared to microfluidics devices, it is also less restricted by larger cell sizes and scalability for larger sample sizes might be achieved by modifications of the gelation device. However, even though first comparisons to commercial droplet devices such as the Chromium series from 10x Genomics look promising, this technology has been developed recently and might require more dedicated benchmark experiments to compare its quality to established technologies. Also, while in theory it is possible to increase scalability, it has been currently only tested and is limited to capturing 10,000 cells at best, which is less compared to throughput rates of current microdroplet devices [32].

Barcodes and unique molecular identifiers (UMIs) are oligonucleotide modifications aiming to increase throughput and reduce an amplification bias that some captured mRNA sequences might be amplified more often than others. They are typically employed with technologies yielding a high throughput and using beads or microparticles to capture mRNAs such as microdroplets or reversible hydrogel.

- Barcodes are oligonucleotide sequences consisting of multiple DNA bases A, C, G and T. In case of the original Drop-seq protocol, barcodes had a length of 12 bp and were generated by running 12 cycles of split-and-pool synthesis leading to 4^{12} possible barcodes. As every unique barcode sequence is added to only one bead, all mRNAs captured by one bead can be identified with the same barcode, thus enabling to process and sequence a high number of cells in one run or device reducing cost and massively increasing throughput [31] [32].

- UMIs are also randomly generated oligonucleotide sequences, similar to the barcodes. However, their sequence is usually shorter than the barcode sequence, and in the case of the original Drop-seq protocol, UMIs had a length of 8 bp. In contrast to the barcodes, they are not unique for each bead, but instead unique for each mRNA fragment captured by the bead. This enables to tackle a possible amplification bias. So instead of counting the number of sequences identified for a gene, the UMI sequence is counted instead in the later analysis [31].

Barcodes and UMIs are usually flanked by a primer sequence identical on all beads that is used for PCR and sequencing and a poly-T oligosequence consisting of multiple thymine bases, that is complementary to the poly-A sequence of the mRNAs. Hence, a typical bead is then coated with a high number of oligonucleotides containing a primer sequence similar to all other beads, a barcode sequence which is similar to all barcodes of oligonucleotides on the same bead, an UMI sequence which is unique for each oligonucleotide on a bead and finally, a poly-T sequence (Fig. 10). Depending on the protocol or commercial implementation, the oligonucleotides might contain some additional sequences that are used for analysis purposes. Also, the length of primer, barcode, UMIs and poly-T sequence might vary between different implementations [31].

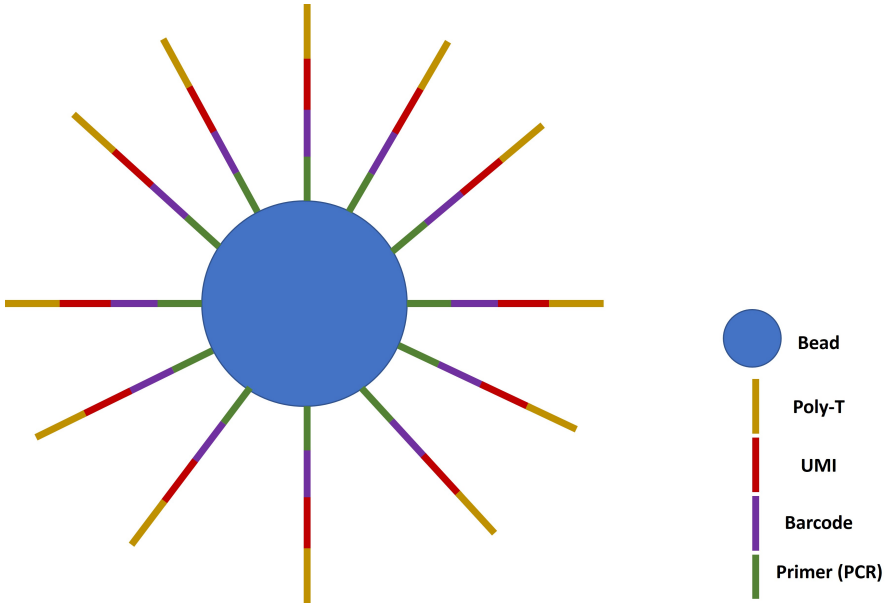


Figure 10: **Scheme of a bead or microparticle with attached oligonucleotides used to capture mRNAs in single cell experiments.** The primer sequence is identical across all beads, barcode sequence is identical for all oligonucleotides on the same bead, UMI sequence is unique for each oligonucleotide and poly-T sequence consists of multiple thymine bases. mRNAs from a lysed cell are bound to the poly-T part with their complementary poly-A sequence. Thus, each captured mRNA is assigned a unique UMI and a barcode identical to all mRNAs from the same cell. [31]

1.4.2 Amplification

As the amount of RNA extracted from a single cell is not sufficient for library preparation and sequencing. While a typical RNA-seq experiment requires a microgram amount of input RNA, a single cell contains only about 10 pg of total RNA and 0.1 pg of mRNA. Thus, the necessary amount of input RNA would require around a million cells as input [33]. However, this issue is avoided by amplification of the input material coming from single cell experiments. As a first step, the captured mRNA is turned into cDNA by reverse transcription [26]. Following this conversion, two different approaches are commonly used for amplification of the input RNA material:

- *in vitro* transcription (IVT)-based amplification. This was the first method used for successful amplification of RNA and is based on the RNA polymerase of the T7 bacteriophage infecting *Escherichia coli*. In the initial step, the reverse transcription of the captured mRNA into cDNA is performed using a specifically designed primer. This primer contains a binding sequence, *e.g.* a poly-T sequence, a promotor sequence required by the T7 polymerase and optionally other sequences such as barcodes or sequencing adapters. After binding to the poly-A tail of the mRNA, the reverse transcription reaction takes place and a double-stranded cDNA containing the T7 promotor sequence is generated. Following, the IVT takes place and the T7 polymerase transcribes multiple RNA copies of the cDNAs leading to a linear amplification rate. Finally, these copies are converted back into cDNA so that they can be sequenced [33] [34].

- PCR-based amplification. This method is ubiquitously used in modern molecular biology and medical science to amplify small amounts of DNA in short time. In the first step, the target DNA - or in case of single cell transcriptomics, cDNA - that should be amplified is mixed into a tube together with the four nucleotide bases adenine, thymine, cytosine and guanine. Furthermore, primers - short DNA fragments with a sequence complementary to the target DNA - and a DNA polymerase are added. In the next step, the double stranded DNA is separated by heating up the mixture above the melting point of the DNA strands. Afterwards, the temperature is lowered, allowing the primer sequences to bind to the separated DNA fragments, also called the hybridization or annealing phase. Finally, the temperature is increased again, which enables the DNA polymerase to bind and extend the primer sequences by adding nucleotides, leading to a newly generated strand, complementary to each of the previously separated strands and completing one PCR cycle. Thus, the number of DNA fragments is doubled in each cycle leading to exponential amplification of the input [35] [36]. Depending on the research question, analysis of the full length transcript might be required, *e.g.* in case of isoform detection.

As the standard sample preparation PCR protocol leads to a 3′ end bias for each mRNA sequence, isoform detection might not be possible in case of long sequences. For this, modifications of the PCR protocol have been developed such as the template-switching approach. The idea is to use the Moloney Murine Leukemia Virus reverse transcriptase due to its property of adding non-templated nucleotides to the generated cDNA strand when reaching the mRNA's 5′ end. These nucleotides, mostly cytosines, are then used to pair another primer complementary to the added non-templated nucleotides. This primer then leads to a template switch in the reverse transcription process, resulting in transcription of the other cDNA strand with both strands including PCR primer binding sites at both 3′ and 5′ end. Thus, full-length cDNA transcripts are enriched during the PCR process. However, in case the reverse transcription process does not reach the 5′ end, these partially transcribed sequences will be lost [33].

In general, both methods are applied in current protocols used for single cell RNA sequencing. However, they differ in amplification rate, product length and specificity. While IVT only produces a linear amplification rate, PCR outputs an exponential amount of amplified sequences, thereby leading to a faster processing time of the samples. Also, PCR can be used with a lower amount of input RNA than IVT and is also able to produce longer fragments of more than 1 kilobase (kb) in length while IVT-based amplification is often limited to less than 1 kb and needs to be modified for longer transcripts. Furthermore, IVT is more biased towards the 3′ end of genes which might be problematic when isoforms should be detected. An issue that is addressed by using PCR instead of the template-switching method. However, PCR-based amplification also comes with disadvantages, as it is more error-prone for accumulation of primer-dimers and other non-specific products, especially on later cycles. Furthermore, exponential amplification might interfere with the expression quantification of the mRNAs and mask the original differences in gene expression between the original cells. In this regard, usage of UMIs should help to prevent this issue [33] [37].

1.4.3 Sequencing

Following the amplification process, single cell RNA-seq data is processed in a similar way as bulk RNA-seq data. Library preparation is performed according to the sequencing device and the sequencing process itself is typically carried out on Illumina devices [38] [39]. However, sequencing depth requirements are different when compared to bulk analysis. In general, the required number of reads per cell depends on multiple factors such as the single cell protocol used - leading to *e.g.* full-length or only 3' end sequences, the research question, or the complexity of the sample. For example, cell-type classification based on a mixed population is possible with as few as 10,000 to 50,000 reads per cell, depending on cell type and cell state [40]. However, if the sample consists of closely similar cell types, sequencing depth might have to be increased. For example, it requires a higher number of reads to differentiate between different T-cell types than to differentiate between *e.g.* T-cells and neural cells. However, due to economical aspects, read depth might be limited as well, especially in case of a large sample. It might also be feasible to increase the number of cells in combination with reduced sequencing depth to identify cell populations with a low frequency [38].

A higher sequencing depth might be required to tackle the so-called 'dropout' problem, as the capture efficiency of poly-A mRNAs varies between protocols and experiments with 10% - 40%. As a result, the mRNAs for some expressed genes might not be detectable, especially in case of low expression levels. However, increasing sequencing depth is not able to circumvent this issue as the number of mRNAs inside a single cell is limited to around 300,000 transcripts per cell [40]. Due to the capture rate, a very high sequencing depth would just lead to an increase of sequence PCR duplicates [38] [40]. In general, it seems that all established single cell protocols approach saturation with a depth of approximately 1,000,000 reads. Also, the majority of genes can already be detected with a depth of 500,000 reads [38] [40].

While the majority of single cell RNA-seq experiments uses NGS technology for sequencing, some studies have also applied third generation sequencing with Oxford Nanopore devices. Ideally, this could reduce sequencing cost and lab workflow com-

plexity due to portable sequencing devices while also enabling full transcript coverage instead of 3' end coverage only. However, the higher error rate compared to Illumina sequencing is a major obstacle when working with UMI and barcode sequences [41] [42].

1.5 Bioinformatic analysis of transcriptomic data

As with bulk RNA sequencing, read data from NGS sequencing is obtained in form of a FASTQ file. This file format is commonly used for storing and sharing read sequence data in conjunction with quality values, which denote the probability of a wrongly identified nucleotide. A sequence entry inside a FASTQ file spans four lines [43] (Fig. 11):

1. The first line is a title line that starts with '@' followed by a sequence identifier along with optional description and information about the sequencing device.

2. The second line contains the sequenced nucleotide bases as letters.

3. The third line starts with a '+' and marks the end of the sequence line, optionally it can include a repeat of the first line (except for the initial '@' character).

4. The final line contains the sequence quality information encoded as so-called Phred score, originally used in the eponymous software for Sanger sequencing base calling [44] [45]. This score Q is calculated based on the probability of a base calling error P_e as following:

$$Q = -10x \log_{10}(P_e)$$

   Resulting quality values are then stored in form of American Standard Code for Information Interchange (ASCII) symbols in the range of ASCII 64-126 [43]. As a result, each quality value is associated with one nucleotide base defining the estimated probability of error for its detection - or base calling - during sequencing. Thus, this line must contain the same number of symbols as the second line.

```
@Sequence_ID:HMKNHBGX2:1:11101:18377:1048
ATCCCTAACTCAGACACACTCTGCAAGTAATGTGCACTTGC
+
AAAAAEEEEEEEE/EEEEEEAE6AE</EEEE/EEE/EEEAE
```

Figure 11: **Example of a typical sequence entry in a FASTQ file.** The first line contains sequence identity (ID) and technical information, the second line the called nucleotide bases, the third line the '+' separator and the fourth line quality scores for each base.

Even though some steps and the software used for the bioinformatic analysis of single cell data are similar to bulk RNA-seq, the majority of the analysis needs to be modified or custom-tailored. Generally, the analysis can be separated into pre-processing and downstream analysis. While pre-processing focuses on sequence quality control and transition of the sequence data into a gene expression matrix, downstream analysis focuses on gaining biological insights

1.5.1 Pre-processing of single cell data

Pre-processing starts by assessing the sequencing quality of the FASTQ file. A popular tool for quality control of NGS sequencing data is the software FASTQC (Babraham Institute, `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`). This tool processes and visualizes the quality scores for the complete read file resulting in various visualizations enabling an easy assessment of the sequencing quality. In case of low quality bases, reads can either be trimmed to remove just a few bases or be discarded in total. Also, in some cases the reads contain data from adapter nucleotides used for technical reasons which also need to be removed. However, sequence additions such as barcodes or UMIs must not be removed. Similar to quality control, adapter removal can be done with software originally developed for bulk data such as cutadapt [46] or Trimmomatic [47] [30].

In the next step, the remaining sequenced reads have to be aligned - or mapped - to a reference genome. This step is necessary to select only reads of interest, *e.g.* only reads mapping to the genome of the analyzed organism such as mouse or human.

Ideally, sequencing data should only include matching sequences, but due to contamination or technical artifacts, a lot of noise might be introduced in the data. Furthermore, in some cases such as a virus-host study where infected cells are analyzed, this step allows to separate reads between the host and virus genomes [48]. Again, this step can be carried out with popular batch data tools such as the Burrows-Wheeler Aligner (bwa) [49] or STAR [50]. In order to use the reference-based approach, a reference genome is required onto which the sequenced reads are subsequently mapped. Multiple databases such as RefSeq [51] or the University of California Santa Cruz (UCSC) Genome Browser [52] provide these references for public download, especially for humans and model organisms such as *Mus musculus* or *Danio rerio*. However, in case a reference genome is not available or a more unbiased analysis of, *e.g.* isoforms is planned, it is also possible to use specific tools such as RNA-Bloom for reference-free *de novo* assembly of single cell RNA-seq data [53].

Afterwards, a quantification step is performed where the aligned reads are allocated to corresponding annotation features based on their alignments' position. In case of the typical mRNA or gene expression analysis of single cell data, only reads mapping to gene-coding regions on the reference genome should be taken into account. In a different scenario, where isoform detection should be performed, it is important to verify and quantify which reads align to which exonic sequence. Another challenge might be identification and quantification of reads mapping to multiple target sequences. In case of PE sequencing, read pairs might also map to the same gene, but at different locations, which could result in multiple hit counts. Specialized tools are available for this task, which were originally developed for the processing of bulk RNA-seq experiment data, such as, *e.g.* featureCounts [54] or htseq [55]. In addition to the mapping results, feature counting requires a genome annotation matching the used reference genome. This annotation contains information on which genomic positions which features are localized and is usually obtained together with the reference genome file.

Furthermore, in case UMIs have been used, these need to be extracted from the counted features to perform a deduplication of PCR duplicates of the same original

Table 1: **Example of a gene expression matrix.** The rows are corresponding with genes, columns with barcodes and the matrix cells contain the number of identified UMIs per gene per barcode.

	Cell_BC_1	Cell_BC_2	Cell_BC_3
Gene_1	8	2	0
Gene_2	22	9	12
Gene_3	2	0	0

mRNA fragment. As UMIs are usually employed in combination with cellular barcodes, it is therefore necessary to identify the presumed cell of origin for each read - also called demultiplexing - resulting in a gene count matrix. This matrix or table consists of rows corresponding to genes and columns corresponding to barcodes (Table 1). Presumably, each barcode represents a different cell and the contents of the count matrix contain the number of identified reads or UMIs per gene per barcode. This task is performed with tools tailored for UMIs and barcodes such as UMI-tools [56] or zUMIs [57].

1.5.2 Downstream analysis of single cell data

The generation of the gene expression matrix marks the end of the pre-processing phase and is the starting point for downstream analysis. Similar to the pre-processing, a quality control step is mandatory in the beginning of the analysis to make the data suitable for statistical assessment. As most protocols, especially high throughput protocols such as the popular microfluidics-based systems, do not allow optical assessment of each cell, quality metrics have to be calculated. For this, multiple metrics can be extracted from the gene expression matrix and used for quality assessment:

- Number of counts, *i.e.* UMIs per barcode
- Number of expressed genes per barcode
- Fraction of mitochondrial counts per barcode

In general, outliers with a low number of fragments or UMIs, a low fraction of expressed genes, or a high amount of mitochondrial mRNA counts for a barcode might indicate a dying or damaged cell. If the cell has been dead before processing, the amount of expressed genes or mRNAs is likely reduced and this cell should therefore be excluded from subsequent analysis steps. A high proportion of mitochondrial gene counts often appears in damaged cells, as their cytoplasmic mRNA leaks out of the cell, but mitochondrial mRNAs are retained due to the mitochondrial membrane [58]. Opposed to low counts, outliers with a high count of mRNA fragments indicate so-called doublets - the capturing of two or multiple cells instead of a single cell. This occurs *e.g.* in microdroplet-based systems, when two cells are accidentally encapsulated by one single droplet. Following, both cells are lysed, their mRNAs are captured as if they were originating from a single cell and combined with the same barcode. Once demultiplexing is performed based on the barcode sequence, reads from both cells appear as reads originating from the same cell and thus have to be removed to avoid false conclusions in the downstream analysis steps. In order to achievei this, apart from removing outliers with high counts, some specific tools try to detect doublets based on other metrics, *e.g.* gene expression patterns [59] [60]. Another typical problem in droplet-based experiments is the presence of barcodes that do not correlate with actual cells. The reason for this lies in free mRNA fragments - also called ambient RNA - introduced by *e.g.* damaged cells, floating around in the channels of the device and getting encapsulated by droplets containing beads. As with lysed cells, the mRNA fragments are bound to the beads and get assigned with a cellular barcode and a UMI. Due to the Poisson-limiting concentration of droplets, only a small amount of 1 - 10% of beads is combined with a cell and its RNA inside a droplet, while the majority is only exposed to ambient RNA. A common method to select barcodes related to reads from cells is the knee plot. For this, all barcodes are sorted in descending order based on the number of reads or UMIs per barcode. The cumulative fraction of reads is then plotted and the inflection point of the curve calculated (Fig. 12). All barcodes above the inflection are considered to represent cells and thus retained, while barcodes below the inflection point are

discarded as potential ambient RNA [31]. However, these quality metrics always have to be thoroughly evaluated after performing additional downstream analysis steps, as they can vary between data sets and analysis steps. For example, a very heterogeneous cell sample might contain larger and smaller cells, thus leading to higher and lower read counts of expressed genes or reads without them being caused by doublets. Therefore, it is advisable to initially start with less stringent thresholds, repeat the analysis with more stringent parameters and investigate the effects on the analysis [59].

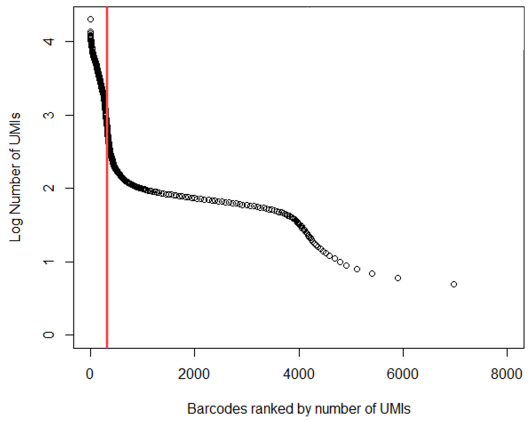


Figure 12: **Example of a typical knee plot.** In this plot, all barcodes are ranked descending by the number of their UMIs, with the x-axis showing the number of barcodes and the y-axis the logarithmic UMIs count. The red line marks the inflection point of the curve, indicating all barcodes to the left of the red line as real cells while barcodes to the right of the line are considered to be background noise, which should be filtered.

After the removal of low quality barcodes, the count matrix has to be normalized. This is required as unwanted biological or technical effects such as sequencing depth or capture efficiency might lead to different count depths even for identical cells. In order to be able to compare the gene expression values between cells, their count data has to be adjusted. While some bulk RNA-seq methods have been applied to single cell data, it is recommended to apply modified or specifically developed normalization methods instead. This is due to single cell specific variations such as dropouts of genes due to the

comparably low RNA amount of cells in combination with the low capture efficiency. As a result, some genes appear with zero counts in many cells, leading to difficulties for bulk RNA-seq normalization tools. A commonly applied technique is a modification of the bulk-based counts per million (CPM) normalization. For this, normalization is performed by calculating a size factor proportional to the count depth and applying it to all counts of the expression matrix [59]. However, this method assumes that all cells initially contained a similar number of mRNA molecules and depth changes are solely caused by either sampling or technical effects. Therefore, some modifications to the CPM approach have been introduced, such as excluding genes accounting for 5% or more of the cell's total counts [61]. Another adaptation is performed by the software Scran, which applies a linear regression over genes after pooling the cells to calculate the size factor to avoid technical dropout effects [62]. Apart from these linear normalization approaches, non-linear normalization methods have been developed, as well. Those methods are possibly able to account for more complex variation in the data sets. Especially when working with single cell protocols prone to batch effects or larger count depth variations, such as plate-based samples, non-linear methods are assumed to perform better [63] [64]. Typically, these methods are based on parametric modeling of count data and they combine external correction of technical or biological effects with count-depth normalization. One approach is to fit a negative bionomial model to the expression matrix with addition of technical covariates such as read depth and counts per gene to fit the model parameters. Afterwards, the model residuals are taken as normalized quantification of the expression data [65]. Also, the optimal normalization method might vary between data sets, for example full length transcript single cell data might benefit from normalization methods that include gene length in contrast to 3'-end-based protocols. In order to achieve an equal weight for all genes in the subsequent downstream analysis steps, some normalization methods scale gene counts in addition to the cell counts. One prominent example for this approach is implemented by the single cell software suite Seurat [66]. In general, normalization usually trans-

forms the gene expression matrix into log(x+1) which eases statistical data evaluation by preventing values of zero. This introduces a variety of benefits:

- The canonical way of measuring gene expression changes requires distances between log-transformed expression values which is achieved during normalization [59].
- Noise reduction by reducing the mean-variance relationship of single cell data which is higher compared to bulk [67].
- The distributions of log-transformed gene expression values are closer to a normal distribution which is expected by the majority of downstream analysis tools [59].

Even though this normalization aims to remove sampling effects, further unwanted variability possibly skews the downstream analysis. Therefore, it is useful to further correct the data for other technical or biological covariates, partially similar to non-linear normalization described above. A correction method that is frequently employed to address biological variability targets the effects of the cell cycle. Usually, cells of a sample are in different stages of their cell cycle, thereby leading to possibly unwanted noise when being compared to each other. The easiest solution for this is the application of a linear regression against a cell cycle score used in the Seurat [66] or Scanpy [68] single cell suites. Other tools rely on more complex mixture models, *e.g.* scLVM [69] and f-scLVM [70]. Both of these approaches require known marker genes related to the cell cycle in the analyzed organism to calculate a cell cycle score. However, apart from reducing biological noise, a cell cycle correction can also lead to masking important biological signals. For example, a proliferating cell population could be detected based on a specific expression of cell cycle genes. Also, dependencies between different biological processes might unexpectedly be skewed by this approach. So this correction should be used with caution and requires enhanced biological knowledge about the organism of interest. The models used for biological regression can also be used in combination with technical covariates to account for further technical noise in the data. Typical covariates used for technical regression are count depth or sample batch.

However, correcting for multiple effects, *e.g.* biological and technical covariates, both should be included in one unified correction step to account for a possible dependence between covariates [59].

Another great challenge in RNA-seq data sets are batch effects. Whenever cells are processed in different environments such as different laboratories, harvested at different time points, processed on different days or possibly just sequenced in different lanes, their transcriptome might be influenced. Especially when comparing different experimental data sets, batch effects can hide underlying biological effects of interest. Ideally, batches are completely avoided through a clever experimental design. One approach is the pooling of all cells across experimental conditions and samples. This is possible using technologies such as cellular barcodes, or also cell tagging [71]. Nevertheless, if batch effects cannot be avoided, tools like ComBat [72] can be used which aim at reducing such effects by applying a linear model correcting gene expression mean and variance data with batch information [59]. However, correcting batch effect between data sets with different compositions, more complex solutions are required, as a linear correction such as ComBat is not able to distinct accurately between biological differences due to a batch effect and biological differences due to different cell types or states. Hence, integration of different data sets requires more specialized non-linear solutions. A variety of tools has been developed for this challenge, such as Seurat canonical correlation analysis (CCA) [66], mutual nearest neighbors (MNN) [73], Harmony [74], and several more. These typically perform a dimensionality reduction and project cells of the merged data set into a reduced space. In a subsequent step, cells sharing similar expression profiles between batches are then corrected in a way that similar cells show a common distribution in the reduced space [75]. As before, the ideal solution often varies between data sets and it should be noted that normal batch effects should preferably be corrected with linear approaches such as ComBat to prevent over-correction, and to apply non-linear approaches for integration of different data sets [59].

After correction of the gene expression matrix, the data set has to be streamlined in order to gain biological insights. This is necessary due to the very large dimension of the data sets. For example, a human single cell RNA-seq data set usually contains tens of thousands of genes, in extreme cases up to approximately 19,000 genes [76]. Furthermore, when using high-throughput single cell protocols, this number of genes is multiplied with hundreds or hundreds of thousands of cells. In addition, as mentioned before, many of the detected genes lead to zero counts in most cells - dropouts. This results in multiple problems for the subsequent analyses: usually the cells are clustered to expected cell types based on their gene expression patterns. However, in this large dimensional space, the differences are very small, making it difficult to distinguish between cell types. Also, calculation of such large-dimensional data sets poses a significant computational burden and finally is almost impossible to be visualized in a way understandable for researchers. Therefore, a common filtering step consists of the selection of highly variable genes (HVG). Those genes are a subset of the whole data set, comprising typically between 1,000 and 5,000 genes that can be assumed to contain the most 'information' about the data set. Selection is typically performed by binning genes based on their mean expression. Then, genes with the highest variance-to-mean ratio are selected as HVG for each bin. Examples for HVG selection are implemented in the single cell tool suites Seurat and Scanpy [59].

Even though HVG selection leads to a large data set reduction, the filtered matrix still contains hundreds to thousands of genes times hundreds to hundreds of thousands of cells. Therefore, further dimensionality reduction is required in order to reduce the data set to its most essential dimensions. This leads to less noise, highlighting biological differences and enables a useful visualization of the results. It has also previously been shown that single cell data and cellular expression profiles can be accurately described in a much lower dimensional space than genes present in the matrix [77]. Generally, dimensionality reduction can be performed in a linear or a non-linear way. The latter way, however, is accompanied by a loss of interpretability, therefore making it more suitable for visualization only [59]. The most common approach for linear summarization of

single cell data sets is principal component analysis (PCA) [78]. PCA creates principal components (PCs) - new uncorrelated variables - that maximize variance in the data set. Following, the top N PCs are selected that exhibit the highest variation in the data set. Selection of N is usually performed using the heuristic elbow plot approach, plotting the PCs in decreasing order of their variance ratio on the x-axis and the variance ratio value on the y-axis. The component marking the flattening of the elbow curve marks the number of relevant PCs. Alternatively, the permutation-based jackstraw method is used, which calculates PCA scores for random genes from the permuted random subset. Those scores are then compared with observed PCA scores resulting in a p-value denoting significance of each gene's association with each PC [31] [79]. Due to its consistency of distances in the reduced space, PCA results are useful to further investigate the performance of previous steps, such as quality control (QC) and normalization or importance of genes in the data set. However, PCA captures data structure in fewer dimensions not as well as non-linear approaches making those more popular for visualization [59].

A very common method for visualizing single cell RNA-seq data is t-distributed stochastic neighbor embedding (t-SNE) [80]. For this, t-SNE calculates pairwise similarities between data points - cells - which are then transformed into probabilities indicating likelihood of two cells being neighboring cells based on their expression profile. In the next step, t-SNE constructs another probability distribution for each data point in a lower dimensional space, usually 2D or 3D. The cells are initially randomly distributed in the lower dimensional space, and t-SNE's algorithm then iteratively adjusts each points position with the aim to minimize the mismatch between high and low dimensional distributions. Compared to PCA, this preserves local similarities in a better way, but at expense of the global structures. This, however, leads to visualizations highlighting cluster or subcluster structures such as cell types. On the downside, t-SNE is computationally demanding, leading to long calculation times for very large data sets. Also, the choice of its perplexity parameters strongly influences the number of detected clusters.

As an alternative, Uniform Approximation and Projection (UMAP) [81] has gained popularity in recent years. UMAP starts by constructing a neighborhood graph in which each data point - cell - is connected to its nearest neighbors - here cells with most similar expression patterns. The distance metric for neighbor determination is usually an Euclidean distance. An important part is the 'fuzzy simplical complex', which describes the likelihood of two points being connected by extending a radius outwards from each point. If the radii of two points overlap, they are connected and considered a cluster of, *e.g.* the same cell type. The mentioned 'fuzzyness' is obtained by decreasing the likelihood of two points being connected when the radius grows. Following, the data is mapped into a lower dimensional space and UMAP's algorithm minimizes the difference between pairwise distances of connected cells in the high dimensional space and their distance in the low dimensional space. Compared to t-SNE, UMAP preserves local, but also global structures while also providing a faster calculation and better scalability for larger data sets [59] [82]. Dimensionality reduction also concludes the preparation steps of the expression matrix and is then followed by analysis steps aiming at gaining biological insights into the samples.

Usually, the first biological-driven analysis step is organization of cells into clusters. Here, a cluster corresponds to a group of cells with a similar identity, typically their cell type, based on their gene expression profiles. A common starting ground for the clustering is to calculate a distance matrix between each cell's gene expression profile. This can be achieved by calculating Euclidean distances based on the PCA reduced expression space. Clustering of cells in a distance matrix can be described as a typical unsupervised machine learning problem. Cells should be organized in clusters by minimizing distances inside a clusters or by identifying dense regions of many cells with similar expression patterns. A fundamental approach for this is the popular *k*-means clustering algorithm [83] which allows to divide all cells into *k* clusters. Input for this algorithm might either be a Euclidean distance matrix or a correlation-based distance matrix. For this, *k* centroids are spread, then all cells are assigned to their closest centroid. In the next step, the centroids are moved into the center of all its belonging cells,

and cells are then re-assigned to their closest centroid. This can be repeated multiple times to improve the optimal cluster assignment for all cells. In order to apply this algorithm, a parameter k is required, which represents the number of clusters, *i.e.*, cell types and is typically unknown. This means that identifying an optimal k value requires an elaborate heuristic approach by repeatedly performing the clustering with varying fixed expected cluster numbers [59]. Another approach are graph-based community detection algorithms. At first, a graph representation of the data is generated using a k-nearest neighbor (KNN) [84] approach, which connects each data point to its k nearest neighbors based upon Euclidean distances between their expression profiles. In the resulting graph, cells are connected based on their expression similarity with densely connected regions of a higher number of similar cells. Using community detection methods, these regions are recognized as possible clusters or cell types. This leads to faster calculation times than clustering, as only neighboring cell pairs belonging to the same cluster have to be considered. Furthermore, the current standard algorithm for detecting clusters in single cell data, the Louvain algorithm [85], is a KNN graph-based approach. For community detection, Louvain groups cells into clusters if they have more links between them than expected from the number of links the cells have in total. Also, the algorithm features an optimization modularity function with a resolution parameter, which allows users to change the scale of cluster partitioning, thereby changing the number and size of detected clusters. Both of the commonly used single cell analysis tools Seurat and Scanpy rely on Louvain-based clustering [59].

The generated clusters with their assigned cells now have to pass an additional validation step. This is required as clustering algorithms are always able to separate data points into clusters, but possibly without any meaning, especially when being used with custom parameters. Therefore, the expression signature of proposed clusters can be used to verify the existence of a correlation with a biological meaning. This approach is often denoted as cell type identification, as in assigning all clusters to a known cell type or cell identity. Depending on the resolution and size of the data set, this can describe different levels of information. For example, the identification of cells as 'T cells'

might already be a sufficient annotation in one type of study, while other researchers or experiments that aim to identify subtypes of T cells and require a higher resolution in order to distinguish between subtypes such as CD4+ and CD8+ T cells. This, however, is largely depending on the experimental setup, sample composition and biological effects, as *e.g.* cell cycle processes can lead to different clustering results. A common approach to assign meaningful identities to cells is performed by identifying genes that are expressed specifically within each cluster, so-called marker genes. These genes are usually up-regulated inside cells of a cluster compared to cells from other clusters and can be identified by performing a differential expression (DE) analysis. Based on the expectation that marker genes have a strong differential expression effect, statistical tests such as the Wilcoxon rank-sum test or the *t*-test can be used to rank genes by their differential expression between their cluster and the other clusters. The top genes obtained via this analysis are then compared to known marker genes from literature or databases such as the Human Cell Atlas [86] or the Mouse Ageing Cell Atlas [87]. Alternatively, the approach can be reversed and instead of selecting marker genes from the data set, gene expression for genes of interest can be analyzed to identify possible cell identities in different clusters. However, marker genes should always be critically questioned, as both genes and clusters are determined based on the same underlying expression data. Hence, marker genes with significant p-values might also be identified via random clustering. This problem especially arises when cell identities are with unsupervised clustering algorithms. In addition, the sample composition influences results as well, as samples with low cellular diversity do not exhibit an accurate background gene expression, thereby leading to a bias in marker gene detection [59]. Automated approaches for cluster annotation have been developed with tools such as scmap [88] or Garnett [89]. These methods directly compare gene expression profiles of known reference cell identities to cells from the sample data set. Matching annotations are then transferred to the sample data set, enabling a simultaneous cluster assignment and annotation. However, this approach should be used with caution, as cell type and cell state differ between different experimental conditions and might

therefore lead to false assignments. Also, this approach is only suitable for a few model organisms, because it requires a well curated set of cell identities with corresponding marker genes. In general, clustering and annotation is an iterative approach requiring exhaustive validation based on biological knowledge. Thus, an automatically annotated data set should always be further validated [59].

As mentioned previously, DE testing is performed to identify differentially expressed genes in different cells or clusters. Apart from annotation, this is also of interest to identify differentially regulated genes of identical cell clusters between multiple conditions. Differential expression analysis itself originates from bulk RNA-seq, where it is applied to analyze gene variation between samples. In single cell experiments, sample sizes are typically much higher, while on the other hand, challenges such as technical noise, gene dropouts and cell-to-call variability are introduced. Therefore, several tools specifically modified for single cell DE testing have been developed. Nonetheless, several studies comparing tools used for bulk as well as single cell RNA-seq studies suggest that bulk RNA-seq tools, when combined with gene weights, might outperform tools specifically developed for single cell data [90]. Therefore, popular tools such as DESeq2 [91] and edgeR [92] could be suitable for single cell data sets as well. However, addition of weights to bulk testing comes with more computational burden, especially when working with large data sets. A single cell-specific alternative with better performance is the tool MAST [93], which is able to account for dropouts and to analyze gene expression changes dependent on condition and technical covariates. Another tool with faster performance is the limma-voom [94] package, which is a modified version of the bulk DE analysis package limma [95]. Nevertheless, independent of the selected tool, it is important to prevent incorporation of confounding covariates into the DE testing model. For example, sample and condition covariates are usually confounded, as a single sample is typically obtained under only one condition. Confounding covariates, however, would make it difficult or impossible to identify the underlying causes of the differential expression [59].

This concludes the most common analysis workflow for single cell RNA-seq data. However, as should be apparent by now, in many cases the analysis has to be, at least partially, repeated multiple times, since *e.g.* clustering results are subject to change, normalization needs to optimized, or due to marker gene expression changes. Whenever analysis steps are re-calculated, all subsequent following steps that rely upon the outcoming data have to be re-calculated as well, and finally results such as cluster identities or marker gene expression always have to be validated for their respective biological meaning.

1.5.3 Optional analysis steps of single cell data

Depending on the experimental setup and underlying research questions, further analysis steps might be performed with the previously generated and analyzed data set:

- Trajectory inference (TI) is an additional analysis step to describe continuous processes [96]. This is useful, as cellular diversity and heterogeneity is a continuous process rather than a discrete condition such as the defined cluster identities. In this regard, the RNA-seq data is more like a snapshot of all biological processes. TI-based methods aim to minimize changes in transcriptional patterns between neighboring cells and thus create one or multiple paths through the cellular space. Cells are then ordered along these paths by their expression similarities, capturing processes such as transitions between cell types, differentiation processes including branching or biological reactions to *e.g.* treatment. This ordering is described as a so-called pseudotime ordering which describes developmental time of the cells, beginning with a root cell cluster [59] [97]. First tools implementing TI have been developed around 2014 with Monocle [96] and Wanderlust [98] with continuous development leading to other popular tools such as Slingshot [99] or PAGA [100] and up to 70 TI tools already in 2018. Available tools differ a lot in their computational performance and are differently well suited depending on the trajectory structure. For example, some tools perform better with simple structures such as linear or just bifurcating paths while other tools are better suited for com-

plex trajectories such as multifurcating or tree-like paths. In general it is useful to compare pseudotime results of multiple tools. Furthermore, the trajectory paths only describe transcriptional similarity, so biological meaning has to be validated as well [97].

- RNA velocity is another analysis step focusing on dynamic cellular processes. The concept is based on dynamics in mRNA maturation, as gene expression first leads to an increase of pre-mRNAs that are unspliced, hence contain intron sequences and subsequently an increase in mature or spliced mRNAs. By analyzing the abundance of spliced and unspliced mRNAs in cells, it is possible to infer models for each gene about its velocity [101]. This allows to predict the cell's direction and movement inside the cellular space and how the transcriptomic state of the cell is expected to change. Positive velocity therefore indicates up-regulation of genes, described by a higher abundance of unspliced mRNAs followed by spliced transcripts. Negative velocity on the other hand indicates down-regulation of genes. Both velocities are combined to predict the future state of a cell. Currently, two main models are used - velocyto [101] and scVelo [102]. While velocyto introduced the concept of RNA velocity, scVelo introduced an alternative model with a more dynamic approach on expression kinetics compared to a steady-state idea of velocity assuming a common splicing rate across genes [103].

- Gene set analysis enables to group differentially expressed genes into sets or groups. This is helpful to interpret the possibly very long lists of genes differentially expressed between clusters. Therefore, DE expressed genes are grouped based on their role in biological processes. This is done by using databases containing labels for biological processes with associated genes, such as Gene Ontology [104], KEGG [105], or Reactome [106]. While gene set analysis is often performed in bulk data as well, a more single cell related method focuses on using paired gene labels for ligand-receptor analysis. Ideally, interaction between cell clusters can be analyzed based on expression of receptors and their regard-

ing ligands. Again, this method relies on using curated labels describing known ligand-receptor pairs such as CellPhoneDB [107] [59].

1.5.4 Current state of single cell software solutions

The advent of single cell RNA-seq introduces an unprecedented resolution for the understanding of biological processes and cellular heterogeneity that revolutionized the field of transcriptomics. This is not only emphasized by awarding single cell sequencing as Nature Method's 'Method of the year 2013' [108]. Also, many high ranking publications from various fields undermine the tremendous potential single cell sequencing brings with it. Numerous studies can be found for a variety of organisms and fields such as planaria [109], zebrafish [110], and frogs [111], but also in very specialized studies in the medical field, for example in characterizations of tumors [112], drug development [113], or the modeling of diseases in organoids [114]. The popularity of single cell RNA-seq has been ever increasing since its development, which means that the amount of generated data increased drastically over the years. In addition to the number of studies, the throughput of single cell methods has been upscaled a lot since its first application in 2009 leading to experiments with up to hundreds of thousands of cells [115]. Another great example for the growing amount of analyzed cells are the previously mentioned cell atlases for human or mouse (Section 1.5.2). Thus, this not only demonstrates the great potential and opportunities this technology provides, but also implies that a steadily growing amount of data has to be stored and analyzed. The analysis is, in general, performed following the workflow outlined in Section 1.5.1 and 1.5.2. However, this leads to variety of challenges for researchers. First of all, for each of the pre-processing and downstream analysis steps, a variety of tools is available, making it difficult to determine an optimal workflow. Furthermore, a significant challenge in bioinformatic analysis lies in the usability of the tools. The pre-processing workflow, for instance, consists of mostly command line interface (CLI)-based tools requiring a Linux-based operating system (OS). Even though the tools might be used on other systems such Windows or MacOS using additional software, users still need to be

able to navigate on a CLI and be familiar with the installation of multiple dependencies and software packages via this method. Also, running the tools, selecting specific parameters and storing in- and outputs of each step might be challenging for researchers without bioinformatic knowledge. Furthermore, many outputs of these tools are only text-based, *i.e.* users have to implement customized scripts to validate the quality and results of all pre-processing steps. Another issue during pre-processing is the variety of different barcode and UMI schemes. Since the read structure can differ significantly, there is a need for tailored demultiplexing and UMI counting for almost every different platform available. This problem is partially addressed by multiple softwares such as ddSeeker [116], or STARSolo [117]. However, those are often limited to one protocol or need modifications which might not support the full requirements for more complex experimental protocols such as ddSEQ.

Tools employed for downstream analysis steps often provide methods to visualize results and to assess the quality of their outputs. However, similar to the pre-processing, users have to select tools and take care of providing the output of one step as a correct input for a subsequent analysis step. Thanks to the development of software suites taking care of multiple or most typical downstream analysis steps, this process is less challenging for the user. The most popular tool suites in this case are Seurat, scater and Scanpy. However, as Seurat and scater are developed as R packages and Scanpy is developed as a Python package, their usage is restricted to these programming languages and text-based interaction in general. Therefore, these tools require a certain level of technical knowledge and understanding of software and dependency installation on each OS as well.

Several tools like Single Cell Explorer [118], Granatum [119] or ASAP [120] have chosen an approach of implementating a graphical user interface (GUI) to enable simplified operation. However, these tools come with several disadvantages. ASAP and Granatum are both web based services running on external servers, which might be problematic when working with sensible data. Furthermore, both tools require a gene expression matrix as input data, thus the initial pre-processing has to be performed by

the user in advance. Single Cell Explorer, on the other hand, can be run locally and also provides a pre-processing step. However, pre-processing is limited to 10x-based data, as it internally relies upon 10x software. Furthermore, it requires a variety of Linux-based software dependencies that need to be installed, which again introduces a hurdle for non-bioinformaticians.

Further options providing full workflows are tools developed by companies providing single cell solutions, such as Illumina BaseSpace, 10x CellRanger [121] or Scipio Cytonaut. However, these solutions are limited to their respective barcode and UMI schemes. BaseSpace and Cytonaut are also web based cloud solutions that require payment, while also being effectively a blackbox as their code is not publicly accessible. CellRanger, on the contrary, is freely available, with code publicly available on Github, while also running offline but similar to Single Cell Explorer requires installation on a Linux-based OS.

1.6 Software platforms for sequence data management

A rising problem in the field of bioinformatic analysis in general, including single cell RNA-seq, is an efficient management of generated data. Due to the advent of NGS, declining cost for sequencing and the rise of technologies enabling an even higher scalability, the amount of generated data sets has massively grown in the past years and will likely continue to do so. While, in theory, more experimental data sets should also enable scientists to improve studies by reusing and sharing data, this introduces a variety of hurdles in reality. In the field of transcriptomics, multiple technologies have been developed and replaced each other, leading to heterogeneous data sets (Fig. 2). While some properties are shared among data sets originating from different technologies, *e.g.* FASTQ-based sequencing data files, other required properties do indeed change, especially meta data entries. Also, depending on the experimental setup, meta data underlies significant changes, *e.g.* experiments comparing different treatments or cell types. Furthermore, modern biomedical and biological analysis is often based on a more systems biology-based approach, utilizing experimental data generated on mul-

tiple levels. For example, genomics, transcriptomics, and proteomics data of the same cell type, tissue or organism are combined to achieve a better understanding of the 'big picture'. This enables researchers to understand and model a condition, *e.g.* a disease in different resolutions, from the single cell up to the whole organism. Also, it allows to characterize a condition with more parameters which can not be captured by only using one technique. Another issue is the growing amount of required data storage with an ever growing amount of data. It is expected that biology will likely overtake current data-heavy disciplines such as astronomy and big data generating services such as YouTube or Twitter [122] [123]. To ensure permanent storage of these large amounts, multiple web-based repositories have been created that also allow to share research data among scientists worldwide. Some of the most commonly used repositories for biological data are domain-specific, such as Genbank for genomics [124], Gene Expression Omnibus (GEO) for transcriptomics [125], and Proteomics Identifications Database (PRIDE) for proteomics data sets [126]. Other purpose-specific platforms focus on diseases such as cBio for cancer [127] or AutDB for autism spectrum disorders [128]. Additional databases focus on an organism such as the human cell atlas [86], or on many different biological analysis targets such as methylation, metabolic pathways, regulatory elements, or protein-protein interactions [129]. This leads to the problem that heterogeneous data sets or data types might not meet the criteria of these specific platforms, thus preventing their deposition. As a result, general-purpose repositories have been developed, such as Zenodo [130], FigShare [131] or EUDat [132]. These platforms accept a wide range of data types with different data formats and origins. While these repositories typically contain the raw data and/or analysis results, in many cases they do not include or even accept deposition of the used software or mathematical models used with the data. Sometimes, those are uploaded to multipurpose repositories as well, but these databases lack standardized formats and meta data schemes. Therefore, databases and search engines specifically focusing on bioinformatics tools or systems biology models have been developed. Examples are the platforms bio.tools [133], supported by European Life Science Infrastructure for Biological

Information (ELIXIR) and BioModels [134], for software and models, respectively. Furthermore, as many bioinformatic analyses comprise multiple tools or computational steps, some platforms focus on storing full workflows such as nf-core [135]. The resulting variety of decentralized platforms with heterogeneous data sets and meta data in different formats introduces complexity and obstacles in the discovery and reusability of the data, especially when working with data that is not covered by a domain or purpose-specific database, *e.g.* a less commonly studied organism or a rare disease. Hence, the search for a suitable data set in this case requires identification of an appropriate platform and corresponding search tools. The search itself then requires a variety of different parameters, *e.g.* omics domain, organism, tissue, sequencing technology, data type, condition or specific genes. Identification of such parameters highly depends on the accessibility of meta data and information about the data set on the corresponding platform. Even after a matching data set has been identified, further questions arise, such as how the data is licensed, how the data set can be obtained or downloaded, or what format it is provided in. Especially when the data and the software, pipeline or model are not stored in conjunction, users need to verify if the data is compatible to their software or which steps are required to transform the data into a suitable format. As a result, manual identification of appropriate data sets is a timely process which often takes up weeks or even months. Taking into account the previously mentioned large and growing amount of data, it is clear that this task would significantly benefit from machine or computer-based approaches that can easily be scaled up, *e.g.* by using artificial intelligence (AI) for data mining and analysis [136]. However, while humans are capable to identify and interpret a wide range of different contextual information, computer-based approaches require standardized meta data formats and structures in order to successfully process data [137] [138]. In order to overcome these obstacles and to reduce existing hurdles in data discovery and reusability, a group of academic and private stakeholders met in 2014 to design a set of principles for data submission. The draft principles developed in this meeting are referred to as the Findability, Accessibility, Interoperability, Reusability (FAIR) guiding principles [137]:

- **F**indability
  - (Meta) data are assigned a globally unique and persistent identifier
  - Data are described with rich meta data
  - Meta data cleary and explicitly include the identifier of the data it describes
  - (Meta) data are registered or indexed in a searchable resource
- **A**ccessibility
  - (Meta) data are retrievable by their identifier using a standardized communications protocol
  - The protocol is open, free, and universally implementable
  - The protocol allows for an authentication and authorization procedure, where necessary
  - Meta data are accessible, even when the data are no longer available
- **I**nteroperability
  - (Meta) data use a formal, accessible, shared, and broadly applicable language for knowledge representation
  - (Meta) data use vocabularies that follow FAIR principles
  - (Meta) data include qualified references to other (meta) data
- **R**euseability
  - (Meta) data are richly described with a plurality of accurate and relevant attributes
  - (Meta) data are released with a clear and accessible data usage license
  - (Meta) data are associated with detailed provenance
  - (Meta) data meet domain-relevant community standards

These principles do not specify a certain technology or implementation, but rather describe characteristics which should be taken up by *e.g.* repositories, databases or tools

to aid human and machine-based discovery and reusability of data sets. Furthermore, the FAIR characteristics are domain-independent and include not only the data itself, but also associated meta data or non-data resources such as workflows that were applied. To ensure a low entry barrier, FAIR principles are minimally defined and separable from each other. Thus, it is possible to achieve different degrees of FAIR-ness for a data object, even if the full data might not be available, *e.g.* in case of sensitive or personally-identifiable data. Since its formulation, multiple platforms from different scientific fields have been working on providing their data according to FAIR such as the protein-specific data bases UniProt [137] and ProteomicsDB [139] from biological research, the US Materials Genome Initiative [136] from material science or the Cherenkov Telescope Array data base [140] from astronomical science [137]. While the spread of FAIR principles simplifies and speeds up data identification and thus aids future research projects, some issues still remain. As mentioned before, data is usually stored separately from used or compatible software and workflows and even though FAIR reduces the search complexity, it still represents a time-consuming task to connect data and software. This task remains especially challenging for researchers without bioinformatic knowledge and/or access to a corresponding infrastructure, as many tools or workflows have a variety of complex software dependencies and often require some sort of workstation or high performance compute infrastructure to process larger data sets. Furthermore, most platforms do not cover the full flow of data generation, analysis, and reporting until its publication. Finally, many of the data repositories only provide a public submission of the data. Thus, experimental data which has not been fully analyzed and is not ready for publishing or personally identifiable data, *e.g.* patient samples, can not be uploaded to these platforms. However, as research is often performed in consortia or at least with multiple labs or institutes, it would definitely be useful to share data in a FAIR way among involved researchers. Otherwise, the data needs to be copied between multiple computers or sites, leading to a variety of possibly different intermediate results that require to be synchronized between collaborators

and results in an unnecessary expenditure of time and an inefficient research structure. Therefore, there is need for a platform which provides:

- private data storage with access control to share experimental data sets with selected collaborators
- combined storage of data sets and suitable software or workflows
- interface to directly run software or workflows with data sets inside the platform
- full life cycle coverage of data sets from generation, analysis until publication
- compliance to the FAIR principles

Currently, a variety of platforms are available fulfilling these requirements, although installation, deployment and administration of these platforms does require a certain degree of informatics knowledge. Also, integration of workflows or software varies in its complexity and sometimes even requires out-of-box solutions. Following are three of the most common data management and processing platforms in biomedical sciences. [141]

Galaxy is a web-based open-source platform started 2005 as the Galaxy project [142]. The concept includes a server and a GUI which is accessed via a web browser. Users are able to upload their own data as well as accessing popular data repositories. Furthermore, Galaxy supports data management, sharing of data and results and provides a large number of established workflows and supported bioinformatic tools, installed in the so-called tool shed. It also includes a workflow editor with GUI which allows users to create their own workflows based on supported tools. These workflows can be published and shared, enhancing reproducible analysis. In addition to the usage of workflows, Galaxy provides a variety of visualization frameworks for general or more specialized data representation. The Galaxy community provides a variety of publicly available instances, including privacy guarantees for private or patient-based data. These instances are often supported by national compute infrastructures and the Galaxy community further provides training courses. Finally, it is also possible

to download and deploy a Galaxy instance on local premises to provide the highest level of data privacy [142] [143] [141].

LabKey server is a web-based open-source platform developed originally in 2006 as Computational Proteomics Analysis System [144]. In 2011, the software has been renamed to LabKey Server and now functions as a biological data management platform [145]. LabKey Server is a commercial software but a free community edition is available. Similar to Galaxy, the software runs centrally as a server and users can access it via a GUI in a web browser. In addition to data management, it also enables sharing results with collaborators due to fine-grained data access controls. LabKey Server also includes out-of-box data visualization tools and a variety of features focusing on clinical research and studies, such as clinical sample management and de-identified data publishing. Furthermore, it includes an electronic lab notebook (ELN) and laboratory information management system (LIMS) features, providing *e.g.* digital experiment documentation, sample management and tracking, or direct connection and data integration to laboratory devices. However, many of these features are not included in the free community version but instead require a monthly payment per user for one of the commercial versions. In contrast to Galaxy, LabKey also does not directly focus on workflows with bioinformatic tools, but instead provides access via an application programming interface (API) to the stored data. This allows to create custom modules which can include workflows and tools [145] [141].

open Biology Information System (openBIS) is an open-source software framework, developed since 2011 by the ETH Zurich [146]. The development was founded as a reaction to the challenges of growing size and diversity of data sets in systems biology. Similar to Galaxy and LabKey, openBIS runs as a central server structure and can be accessed as a GUI via web browsers. Additionally, openBIS provides a set of CLI tools resulting in a variety of different clients. Data can be imported from multiple measurement platforms, *e.g.* computers or laboratory instruments using dropboxes, which correspond to directories on a file system that are constantly monitored for new incoming data. By depositing data into a dropbox, this triggers the Extract Transform

Load (ETL) process, resulting in the creation of a new data set within openBIS, including its annotation with extracted meta data. Similar to LabKey, openBIS also enables fine-grained data access structures, allowing to assign users to different roles to grant or revoke access to data sets. This can also be used to achieve public access to (meta) data, *e.g.* for publication of experimental results. Furthermore, openBIS also provides simple of out-of-the-box visualization features for *e.g.* Hypertext Markup Language (HTML) files and a variety of download options for stored (meta) data such as Excel tables or a bulk export via specific clients. In the years after its initial publication, openBIS has been extended with new features, and since 2016, the software also provides an ELN and LIMS [147]. While openBIS itself does not offer a great variety of workflows and adapted tools like Galaxy, it does offer a variety of very open APIs with a focus of easy integration into existing solutions or custom data processing pipelines [146] [147] [141].

These above examples demonstrate that the increased need for data management and processing platforms has indeed been recognized and addressed over the past decade. In comparison, all of the three presented platforms provide a more or less fine-grained data management system for (meta) data, including the implementation of the FAIR principles. They also provide user-specific access control and can be run on local premises, including a GUI accessible via web browser. The Galaxy platform is more focused on direct integration workflows and tools, and it does not include features such as an ELN or LIMS which can be essential for everyday laboratory work, especially in a clinical environment. Back in 2013, a LIMS implementation for Galaxy has been published, but it never became an official extension and is no longer available [148]. LabKey and openBIS provide both an ELN as well as a LIMS, which helps users in better tracking or the integration of meta data. Also, an automated upload from laboratory equipment such as microscopes or sequencers reduces the complexity of data integration. In direct comparison, LabKey and openBIS appear similar in many features, however, many functionalities within LabKey are only available in its commercial version. Compared to Galaxy, LabKey and openBIS require a higher implementation effort in order to create workflows or adapt analysis tools so that they can be

used directly within the platform via its GUI. However, openBIS is providing a more open API which directly aims in its integration into existing workflows compared to the custom module structure in LabKey [141].

1.6.1 Current openBIS system and workflow registry

At the University Giessen, the openBIS system is implemented, maintained and extended by the bioinformatics core facility (BCF). The implementation is built on the software framework qPortal which uses openBIS as a backend and the Java-based Liferay as frontend interacting with openBIS via API calls (Fig. 13). This includes data storage for raw data, a PostgreSQL database for meta data management and an application server which enables users to browse and manage data and corresponding meta data. Furthermore, the openBIS data model, which is based on hierarchically ordered levels can be customized to be applicable for specific experiments. In addition to the Liferay frontend, qPortal implements multiple web applications developed in Java and the open-source framework VAADIN, so-called portlets. These can be accessed via Liferay and provide a set of functionalities such as a 'Project Browser' or 'Project Wizard'.

Data upload into openBIS can be accomplished in multiple ways, with the openBIS dropbox enabling automated import of even large data sets. This is implemented as an ETL routine for each dropbox utilizing Jython scripts. Jython represents a Java implementation of Python, enabling to combine Python scripts and software with the Java platform and Java classes. These scripts take care of registering raw data and corresponding meta data. This process can also be extended using additional external scripts. Alternatively, new data can also be uploaded via the Liferay frontend using a web browser. In this case, data is imported based on the 'Project Wizard' portlet which allows upload of raw data and meta data. Furthermore, the 'Project Wizard' guides users through creation of openBIS entities such as experiments, samples and data sets by providing further information such as sample type, organism, treatment and many others. Raw data and meta data is finally connected to data sets and moved to internal data storage. The openBIS model enables access right management for the top level

workspaces which can include multiple projects, experiments samples, and data sets. Meta data corresponding to data sets can be either attached to samples or experiments and in addition to structured meta data storage, users may also upload own files for meta data description. Users can be registered using an Lightweight Directory Access Protocol (LDAP) server enabling login to Liferay, and subsequently, openBIS. Based on the user's roles, access to corresponding data and meta data is granted, thus enabling a fine-grained data shareability between multiple projects.

In order to provide bioinformatic analysis directly from within openBIS, a custom workflow registry is developed and implemented by Sven Griep. It is written in the programming language GO and provides an API accessible via Hypertext Transfer Protocol (HTTP) requests. To enable a broad usage of different bioinformatic workflows, the registry supports popular workflow engines like Nextflow and Snakemake. Furthermore, usage of Conda and Singularity for running software containers, such as Docker, is supported. While users select data sets and workflows in the Liferay frontend using the 'Project Browser' portlet, the workflow registry directly accesses selected data sets from the openBIS storage. The workflow is subsequently executed on the BCF's Simple Linux Utility for Resource Management (SLURM) high performance computing (HPC) cluster enabling scalability for large data sets and a timely analysis (Fig. 13). Finally, results generated during workflow execution are stored inside openBIS and connected with the used entity such as a data set or experiment. Using the 'Project Browser' portlet in the Liferay frontend, users can access the workflow's results. Currently, only a limited number of result visualizations is supported, focusing on images, plain text files and static HTML documents.

Currently, the openBIS instance includes the following user roles:

- BCF members function as admins for the system, including its development and maintenance. In addition, they are able to directly register and add new data sets to openBIS by deploying data into an openBIS dropbox (Fig. 13).
- Administrative users function as admins for research groups, laboratories or locations of a research consortium. They can either send data virtually or physically,

in case of large data sets to BCF members or use the Liferay frontend to interact with the openBIS system. Furthermore, they are able to define new access groups for openBIS entities (Fig. 13).

- Users are other researchers from a consortium that are able to interact with openBIS via the Liferay frontend to upload (meta) data, download data, or also interact with projects by *e.g.* executing workflows. Furthermore, they can modify access controls for their uploaded data.
- Service-admins develop and maintain the workflow registry and ensure its connection to the SLURM HPC cluster to ensure workflow execution (Fig. 13).

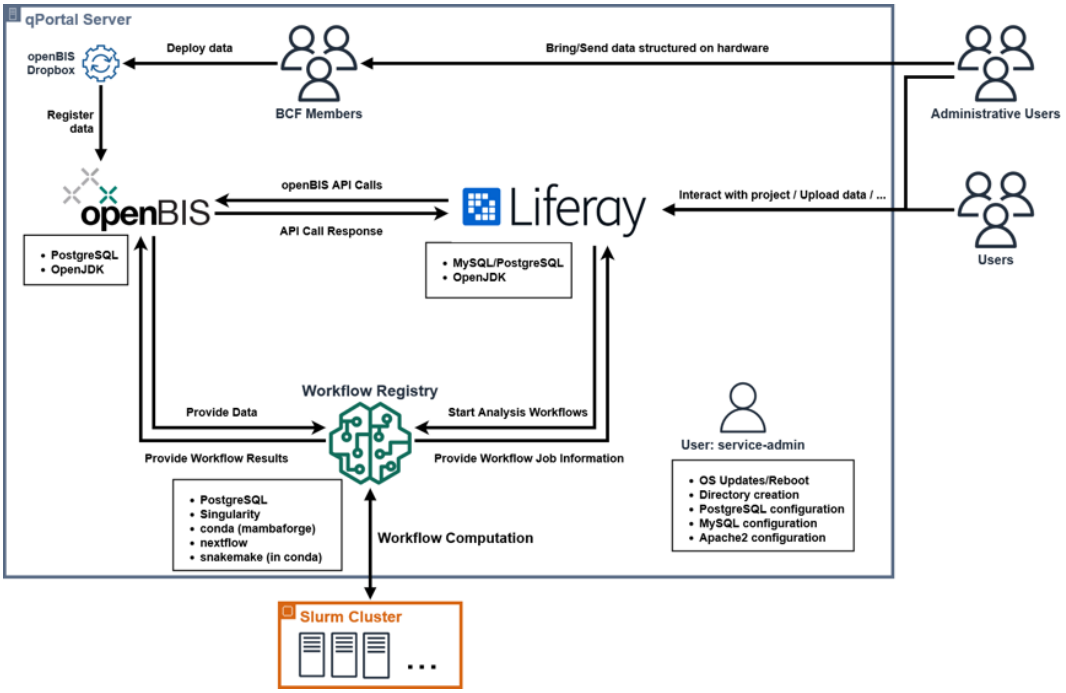


Figure 13: **Structure of the openBIS implementation setup by the BCF for research consortia.** Different components are connected via arrows showing the corresponding interaction. Furthermore, different user roles are shown to visualize how the system can be accessed. Generally, the Liferay module serves as a frontend portal accessible via a web browser. This interacts with the openBIS software itself and also enables users to run analysis workflows. These workflows are executed via the workflow registry which directly accesses data from the openBIS system, executes each workflow on the SLURM HPC cluster and writes analysis results back to the openBIS system, where these are then accessible by the user via the Liferay frontend. Figure provided by Jannis Hochmuth and Frank Förster.

1.6.2 openBIS encryption extension

Apart from maintaining the openBIS instance, the BCF and members of the Goesmann group extended the system with an encryption feature. This was performed with the aim to ensure secure storage of sensitive data, *e.g.* patient-derived data. In order to combine this feature with the fine-grained access control for different levels such as experiments or raw data, encryption should always be applied to each data set. The encryption method is based on the Crypt4GH specification defining a file container

providing encryption of genetic sequencing material leaving the underlying data itself unchanged [149]. For this, a Crypt4GH file consists of a header and data portion, separable into data blocks, containing the sequencing information. Both parts are encrypted based on the same encryption method but, using different keys. Access to the data is controlled by header packets which are encrypted using a recipient's public key and the writer's private key. For this, each header packet contains the packet length, the used encryption method, the writer's public key, a random sequence, called Nonce, in the corresponding data and a message authentication code (MAC) which can be found within the data block as well. The MAC is based on a shared key to be used as integrity check to prevent decryption without a successful validation by the recipient. A successful decryption of a header packet enables the recipient to access the data portion and to modify the header segment to *e.g.* provide access to another user. Separability of the data portion into data blocks further enables random access into the encrypted file. In addition, Crypt4GH is designed to support streaming reads and writes [149].

Therefore, the integration of Crypt4GH within openBIS enables users to encrypt their data sets and to add or remove access to/from other researchers by adding or removing corresponding header packets. Also, this prevents access to data sets by *e.g.* administrative users with more permissions as the data can only be decrypted when the user is added to the header segment. In order to simplify this process for users, a JavaScript implementation of Crypt4GH, called Crypt4GH-JS [150], was developed to combine key management with user login within openBIS. Furthermore, the workflow engine is integrated into the key management to ensure secure processing of raw data. If an authorized user requests a workflow, the header is decrypted and completely replaced with a new header consisting of the user's secret key and a randomly generated public key from the workflow registry. Subsequently, the data set is moved to a newly spawned virtual machine (VM) containing the corresponding private key to the previously used public key, enabling to decrypt the data set inside the VM. After all processing steps have been performed, the data set and results are encrypted again using the random private key and the public keys from all users that previously had access

to the data set. Thus, after moving the data back into the openBIS system, previously authorized users are able to decrypt data set and results.

2 Motivation and goals of this work

Although single cell RNA-seq is an exciting new technology, providing an uprecedented resolution and numerous novel insights, the current bioinformatic software landscape exhibits specific pitfalls. While the general analysis of single cell RNA-seq data remains a complex task requiring researchers to install and use a variety of different tools, some software solutions at least partially combine multiple analysis steps into a software suite. However, some of these suites such as Granatum or ASAP run on external servers, making them unsuitable to upload sensitive data such as sequence data derived from patient samples. Others like Seurat or scanpy do require a certain bioinformatic knowledge adding an entry barrier for non-bioinformatician researchers. This also accounts for many tools only available for Linux-based OS. In addition, many tools only cover the pre- or post-processing of single cell RNA-seq data or might be limited to data generated from devices by a specific manufacturer. However, research consortia or collaborations usually consist of a heterogeneous composition of researchers from different disciplines. An example for this is the consortium of the Clinical Research Unit 309, referred to in this dissertation under its German name Klinische Forschungsgruppe 309 (KFO309). This group focuses on analysis of infectious respiratory diseases, resulting lung injury and its pathobiology in order to develop novel therapeutic strategies. The research team consists of a heterogeneous group of biologists, physicians, technical personnel and bioinformaticians. Still, the hurdles of complex software solutions limit data analysis to be only performed by a bioinformatics group. This however unnecessarily slows down and leads to an increased complexity of possible data analysis. Furthermore, as the research team spans over multiple institutes, data is generated in multiple locations and with devices from multiple manufacturers requiring different compatible software solutions.

Therefore the first aim of this work was:

- Development of a single cell RNA-seq analysis platform with easy usability, covering pre- and post-processing steps without restriction to a single manufacturer protocol.

As mentioned in section 1.6, there is a need for data management and workflow platforms, especially for an efficient scientific progress in smaller or specialized research consortia. The KFO309 is a perfect example for this challenge, as their members are located in three sites - Justus Liebig University Giessen, Philipps University Marburg and Max Planck Institute Bad Nauheim, are working with a variety of different omics technologies and generating sensitive data. In order to provide a software solution that allows storage and sharing of data between the KFO309's various members, the BCF deployed an openBIS instance. Furthermore, this openBIS instance has been extended by Sven Griep with a so-called workflow repository, enabling users to run bioinformatic workflows directly from the web-based GUI. However, this requires the development or adaption of bioinformatic tools and workflows to be usable with the repository.

Therefore the second aim of this thesis was:

- Integration of the software platform mentioned as first aim into the openBIS workflow repository, enabling KFO309 scientists to perform single cell RNA-seq analysis of data stored within the openBIS system.

In general, both aims focus on lowering the entry barrier for all KFO309 scientists to perform a single cell RNA-seq analysis, including an efficient visualization and shareability of the results.

3 Implementation

3.1 WASP: A versatile web-accessible single cell RNA-seq processing platform

The first thesis aim of this work was the development of a software platform for single cell RNA-seq analysis for KFO309 researchers. A need for such a platform is originating from analysis requests made by KFO309 researchers. Even though the consortium is separated into multiple smaller subprojects, many experiments are based on transcriptome analyses. These are often performed via cell culture or cells harvested from *Mus musculus*, and provide a detailed insight into *e.g.* gene expression changes during viral infection or recovery. Furthermore, exploiting single cell resolution, it is possible to shed further light onto how each cell type is affected during infection or recovery processes in order to identify possible treatment targets. In addition to state comparison, cell type detection, and also characterization based on single cell RNA-seq is an essential process to validate experimental setups and models. In order to provide an appropriate analysis, a first workflow for single cell RNA-seq data was established. This comprised a variety of different manual tool executions and scripts in different programming languages. Finally, a number of single cell-specific software packages were integrated to generate result visualizations. While this process supported the typical analysis of transcriptomic single cell data sets, it lacked automated processing steps and also required more advanced bioinformatic knowledge. Thus, researchers with a focus on other fields were unable to perform data analysis on their own. This resulted in a high amount of communication to be required between KFO309 researchers and the bioinformatic counterpart. Especially analysis steps that had to be re-calculated multiple times, such as clustering (see section 1.5.2), or detection and highlighting of upregulated genes, led to an unnecessary complex and inefficient analysis process. This resulted in the concept of a software platform for single cell transcriptome data analysis, which automates and standardizes as many processing steps as possible while further providing a GUI enabling easy usability for researchers regardless of their bioinfor-

matic expertise. In general, the concept was aimed at providing all necessary analysis steps that are part of pre-processing and downstream analysis, described in detail in sections 1.5.1 and 1.5.2. However, the concept is accompanied by a number of challenges:

- Pre-processing at least partially requires Linux-based software
- Analysis of eukaryotic sequencing data, *e.g.* from *Mus musculus* requires a high amount of computer system memory
- Processing of droplet-based single cell data substantially benefits from parallel processing as hundreds of thousands of barcodes need to be validated
- Due to the varying structure of barcode and UMI sequences, reads originating from devices of different manufacturers each require a tailored algorithm
- Many tools used in pre-processing steps do not exhibit visual output, making it difficult to assess whether an experiment has been successful
- Selection of barcodes belonging to droplets containing RNA from cells instead of ambient RNA
- Downstream analysis packages are used by writing programming code which also includes specific parameters

In accordance with the goal of simplifying the user experience and offering software with a low entry barrier, the workflow was split into multiple modules which together make up the 'Web-Accessible Single Cell RNA-seq Processing Platform' - WASP [151]. The different modules are described in more detail in the next subchapters.

3.1.1 WASP: Pre-processing of single cell data

Single cell RNA-seq data analysis, especially the pre-processing of droplet-based protocols, involves a large amount of input data. Usually, hundreds of millions of reads including hundreds of thousands of barcodes result in a file in the double-digit gigabyte

range, even for comparably small data sets. The pre-processing of this data requires the files to be passed to multiple different tools. Furthermore, some intermediate results have to be passed on from one tool to another. Therefore, manual processing of this data represents a time-consuming task which negatively impacts reproducibility. These challenges are not limited to single cell or even transcriptomic data, but rather apply almost ubiquitously to any kind of modern day biological data. Especially since the advent of NGS sequencing and the continuing cost reduction, omics experiments have a growing volume of input data and usually require multiple tools to be applied. Furthermore, the increased amount of data requires using a workstation or even HPC cluster to enable successful data analysis. In order to provide a more sophisticated method to ensure a timely, reproducible and scalable data analysis, workflow engines have been developed. Their main task is to provide an abstract layer that takes care of data processing and movement, managing task dependencies and resource allocation. Additionally, some tools also provide runtime reports, error management, user authentication and data security mechanisms. Also, workflow engines often support APIs to interact with processed data and to integrate different methods of software deployment [152], [153]. The Galaxy platform, mentioned in chapter (1.6) is an example for a workflow engine providing a GUI for workflow creation [142]. Other popular platforms with a focus on bioinformatic data processing are Nextflow [154] or Snakemake [155]. Both of these systems are built upon a domain-specific language (DSL) which is used to define a workflow. For this, the user describes all analysis steps in DSL-based code in the form of a text file which is then interpreted by the workflow engine. An advantage of this is the portability of these workflows as they only require the workflow engine to be installed instead of a whole server, as it is necessary *e.g.* in case of Galaxy. Both platforms further support an integration into existing HPC environments such as Sun Grid Engine (SGE) [156] or SLURM [157] and are able to orchestrate task submission to such a system [154], [155].

In order to realize an efficient and reproducible pipeline for the pre-processing phase in WASP, Snakemake was chosen as workflow engine. A main reason for this choice

was the great similarity between Snakemake's DSL and the Python programming language's syntax, as some analysis steps of WASP have already been developed in Python [155]. Thereby, it reduces the expense of adding another programming language as well as providing the possibility to directly embed Python code within a Snakemake. The workflow itself is defined in a so-called 'Snakefile', which contains rules defining the individual steps that have to be performed. Every rule specifies a name, input and output files including wildcard support and a run statement, *e.g.* a shell command or a Python code fragment. The executed commands then generate some type of output from the specified input files. An example for this might be a mapping step, which contains required input files such as reference genome and sample file, define which output files have to be created when the rule was executed correctly and a run command which executes the alignment tool. As a Snakefile usually consists of multiple rules, the workflow is implied by adding output files of one rule as input files for another rule. Whenever Snakemake is executed, it constructs a directed acyclic graph (DAG) representing the planned execution of the rules specified within the Snakefile. Each executed rule, also called job, represents a node and directed edges connecting nodes represent output files from one job needed for the other connected job. Thus, a path in the DAG stands for serially executed sequence of jobs (Fig. 14). This also means that disjoint paths inside the DAG can be executed independently which also allows a parallel execution of these jobs. For example, a Snakefile running with multiple samples can execute the workflow for each sample in parallel to achieve a faster analysis. Snakemake is then able to orchestrate as many job executions in parallel as possible according to a threshold of available central processing unit (CPU) cores and system memory. Thresholds can but do not have to be specified by a user, and Snakemake optimizes the execution to a maximum of efficiency. Also, it is possible to define specific requirements such as threads or memory for each rule to prevent an execution without the necessary free resources. Therefore, a Snakemake workflow exhibits a high degree of scalability from single-core systems up to HPC environments [155]. The pre-processing Snakemake workflow of

single cell RNA-seq data in WASP contains a number of rules explained in more detail below.

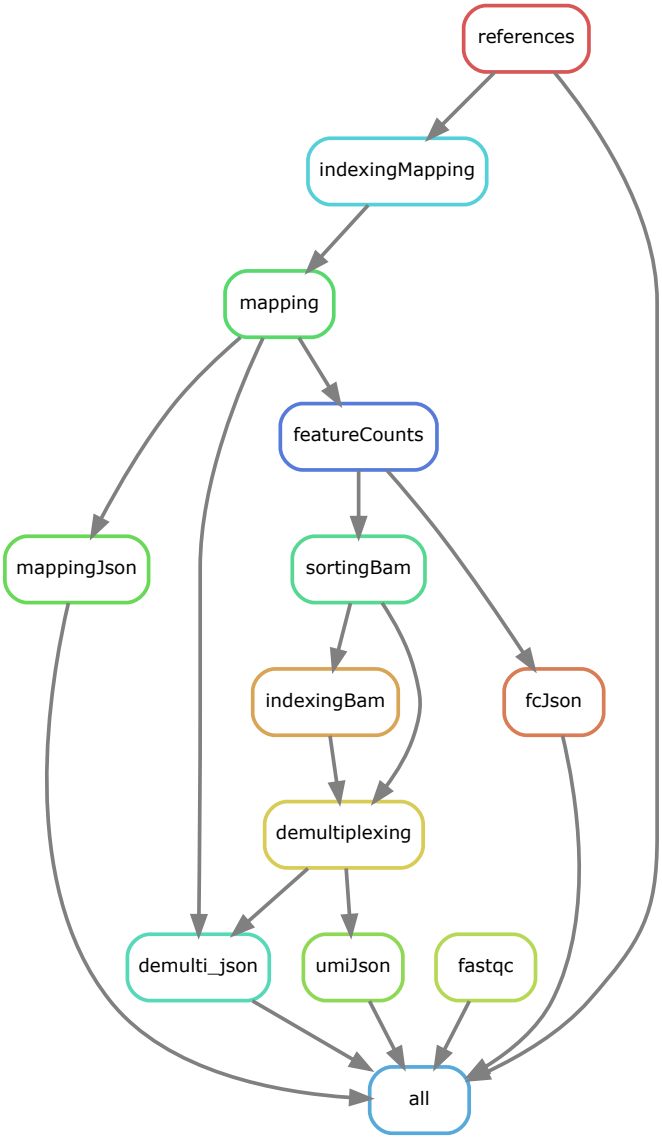


Figure 14: **Scheme of the DAG used for the Snakemake workflow of WASP including all executed analysis steps (rules).** Rules are shown as colored ovals, arrows show the direction of data flow between the rules and represent the dependencies between rules. The rule 'all' defines all outputs in the Snakefile that need to be generated during a successful pre-processing workflow execution.

Directory structure: In order to run properly, the workflow requires a specific structure of directories which include the necessary input files. The folder structure is designed as shown in Fig. 15.

```
project_directory
├── Snakefile
├── wasp.sh
├── Reference/
│   ├── reference.fa
│   └── reference.gtf
├── Samples/
│   ├── Sample_1_R1.fastq
│   ├── Sample_1_R2.fastq
│   ├── Sample_2_R1.fastq
│   └── Sample_2_R2.fastq
└── Scripts/
    ├── demultiMetrics.py
    ├── demulti_umi.py
    ├── fcMetrics.py
    ├── gene_counting_json.py
    ├── mappingMetrics.py
    ├── mapping_json.py
    ├── read_transform10x.py
    ├── splitBarcodes.py
    ├── umiMetrics.py
    ├── whitelist_10X_3M-february-2018.txt
    └── whitelist_10X_737K-august-2016.txt
```

Figure 15: **Tree visualization of a WASP project directory and associated files.** This structure is required by the WASP pre-processing Snakemake workflow. Apart from manual creation, users can directly obtain this structure by cloning WASP from the Github repository. Sample files and reference genome and related annotation have to be added by the user.

The **project_directory** is the directory in which all project and workflow data is stored. This includes the **Snakefile** containing the workflow with its rules and a shell script - **wasp.sh** - which is used to start WASP. As transcriptome analysis usually involves a mapping and a quantification step, **genome reference** and **annotation** files are required. The reference file contains the nucleotide genomic sequence of the analyzed organism and is expected to be in FASTA format. Furthermore, a matching annotation

file in the General Transfer Format (GTF) format is necessary, which contains information about the gene structure of the according reference genome, *e.g.* which parts of the genome are introns or exons. Both formats are standard formats used in bioinformatics and can be obtained in various databases and are deposited inside a directory named **Reference**. The **Samples** directory is used to store all sample files in FASTQ format containing the sequenced transcriptomic data. Finally, the **Scripts** directory contains a variety of Python scripts used during different analysis steps. Also, barcode whitelists for 10x-based data are stored inside this directory. Although this structure could be created by the user, the easiest way would be to download WASP from its public Github repository. This directly generates the above described structure and only requires the user to add genome reference, annotation, and the sample sequencing files. Finally, WASP can be started using the designated shell script with the following steps:

Quality control: As mentioned in section 1.5.1, the initial step in the workflow is performing a quality control of the sample data. This is realized in WASP using the FASTQC tool which generates a variety of metrics based on the FASTQ sample files, such as sequencing quality per base, GC base content or overrepresented sequences. After the analysis, FASTQC summarizes all metrics in the form of an HTML report.

Mapping: Independently of the quality control, the mapping of sample reads onto the reference genome is initiated. The mapping is realized using the software STAR [50]. However, before the mapping itself begins, STAR needs to generate reference genome indices. These are necessary for STAR to speed up the mapping process and need to be generated for each genome individually. After indexing, the mapping is conducted for each sample's FASTQ file which outputs a Binary Alignment Map (BAM) file, a binary representation of the Sequence Alignment Map (SAM) file format [158]. This file consists of a header section containing information about the entire sample such as sample name, sample length and alignment method and an alignment section containing information about successfully mapped reads such as read name, sequence and quality as well as custom tags [158]. Furthermore, STAR also outputs text files

with information about mapping statistics, such as number of mapped and unmapped sequences along with reasons why reads could not be mapped.

Feature counting: While the generated BAM only contains the successfully mapped reads along with some information, it is not filtered for reads overlapping with features nor does it include a quantification step. For this, WASP uses the tool featureCounts [54] to filter BAM results and only retain and count reads mapping to exons within the reference genome. This step therefore requires the user-provided genome annotation as well as the BAM file obtained during the mapping step. A filtered BAM file is generated along with a text file containing statistics on how many reads have been successfully assigned to features or why reads have not been assigned, similar to the STAR statistics output.

Split and barcode correction: The next step is an intermediate step between feature counting and the final UMI quantification and the demultiplexing based on barcodes. This is necessary, as these following steps require either barcode or UMI in a specific format that is not provided in the currently generated files. Also, due to the requirement to support protocols of multiple manufacturers resulting in different UMI and barcode schemes, this step standardizes all files to be compatible with the tools used in further analysis steps. For some manufacturers or protocols, it is mandatory to sanitize results from invalid barcodes. As each protocol or manufacturer uses an individual scheme of barcode and UMI sequences, the user needs to provide the used protocol at the start of WASP. Based on this parameter, reads of the sample are treated accordingly. During the development of WASP, a variety of protocols have been added to cover the most common protocols in general and also those used within the KFO309. However, the variety of different protocols still introduces a complexity to this task:

- The first added scheme aims on processing data generated with the SureCell protocol used on the ddSEQ device from the companies BioRad and Illumina. Reads generated with this protocol contain three barcode sequences (BC1, BC2, BC3) with a length of six nucleotides each. Furthermore, BC1 and BC2 and BC2 and BC3 are separated by two linker sequences, each with a defined nucleotide se-

quence with a length of fifteen nucleotides. BC3 is then followed by the triplet sequence ACG and the UMI sequence with a length of eight random nucleotides which is followed by a second triplet sequence GAC. Furthermore, the whole sequence can be flanked by a random sequence in the beginning and a poly-T sequence at the end (Fig. 16). The processing of these reads also requires some of these sequences to be checked for erroneous bases with a maximum Hamming distance of one, to account for possible sequencing errors. Also, BC1, BC2 and BC3 need to be compared with a manufacturer-provided whitelist containing 96 valid barcode sequence variations. If these criteria are not met, the corresponding read must be discarded.

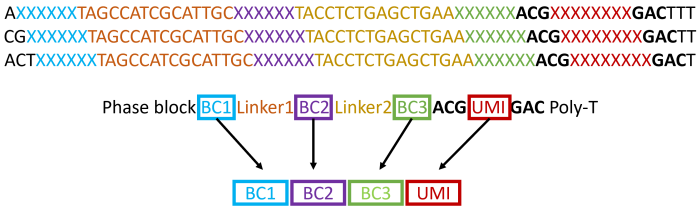


Figure 16: **Barcode and UMI scheme of SureCell ddSEQ-based single cell reads from BioRad and Illumina.** All three barcode (BC1, BC2, BC3) sequences with a length of six nucleotides each have to be extracted as well as the UMI sequence with a length of eight nucleotides for further processing. Barcodes need to be checked against a manufacturer-provided whitelist. Also, linker sequences and the triplets flanking the UMI sequence are specified and need to be validated.

- The second added protocol aims on processing data generated with the Chromium device from the company 10x. Here, the read structure is less complex and contains a barcode sequence of 16 nucleotides followed by an UMI sequence with a length of either 10 or 12 nucleotides for the v2 (Fig. 17 A) and v3 (Fig. 17 B) protocols, respectively. Barcode sequences also need to be compared with manufacturer-provided whitelists which differ between the v2 and v3 protocols. Reads without matching barcodes must be discarded similar to the SureCell approach.

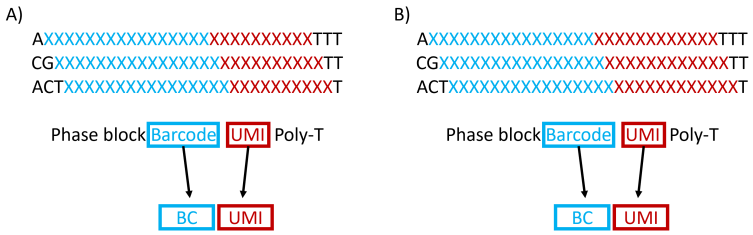


Figure 17: **Barcode and UMI scheme of Chromium-based single cell reads from 10x.** The read design is less complicated compared to SureCell (Fig. 16) with only one barcode sequence with a length of 16 nucleotides directly followed by a UMI sequence. Depending on the used chemistry, v2 A) or v3 B), the UMI's length is 10 or 12 nucleotides, respectively. Barcode and UMI have to be extracted and barcodes need to be validated against a manufacturer-provided whitelist.

- The third added protocol aims on processing data generated with the Nadia device from Dolomite. Barcode and UMI sequence are used similar to the original Drop-Seq protocol from 2015 [31]. Thus, reads contain a barcode sequence with a length of 12 nucleotides followed by an UMI of 8 nucleotides (Fig. 18). Compared to the previous two protocols, the manufacturer does not provide an official whitelist for barcode sequences.

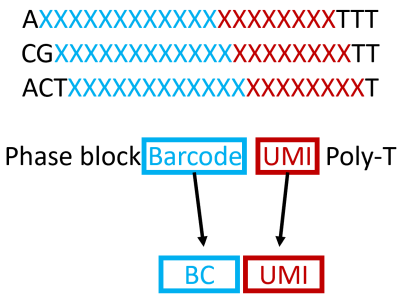


Figure 18: **Barcode and UMI scheme of the original Drop-seq protocol from 2015 for single cell reads, also adapted by Dolomite.** The read design is less complicated compared to SureCell (Fig. 16) with only one barcode sequence with a length of 12 nucleotides directly followed by a UMI sequence with a length of 8 nucleotides. Barcode and UMI have to be extracted, but a manufacturer-provided whitelist is not available.

- The fourth added protocol aims on processing data generated with the Asteria device from Scipio. This protocol provides a read schema including a barcode sequence with a length of 12 nucleotides followed by an UMI sequence with a length of 13 nucleotides (Fig. 19). Similar to the original Drop-Seq protocol, no whitelist is provided by the manufacturer.

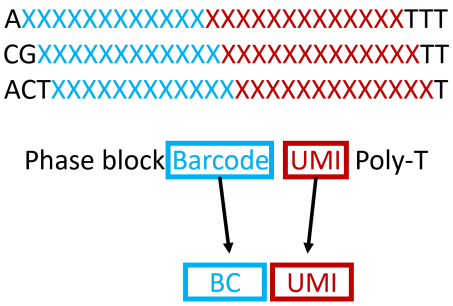


Figure 19: **Barcode and UMI scheme of the Asteria protocol for single cell reads from Scipio.** The read design is less complicated compared to SureCell (Figure 16) with only one barcode sequence with a length of 12 nucleotides directly followed by a UMI sequence with a length of 13 nucleotides. Barcode and UMI have to be extracted, but a manufacturer-provided whitelist is not offered.

At first, the resulting BAM file obtained from feature counting in the previous rule is split into chunks of 1,000,000 reads each. This is achieved using samtools [54] and the GNU coreutils split tool. File separation is used because the following barcode cleanup step requires a series of compute-intensive String matching operations, Hamming distance calculations and multiple file writing processes that can be performed more efficiently in parallel to reduce the analysis time for each sample. Each of these chunk files is combined with the original file's header and is ordered and indexed with samtools for faster processing. After this, the barcode sanitation using a custom WASP Python script is initiated. Based on the provided protocol, all reads are processed according to the manufacturers specifications. For this, barcode and UMI are extracted from each read's header, if applicable sequences are checked against provided whitelists, error-corrected and valid reads subsequently written to a new BAM file. Barcode and UMI

sequences are added as custom tags to each entry, allowing direct access in subsequent analysis steps.

UMI quantification: The filtered output BAM files are then collected and merged into one file per sample. Using samtools, the newly created file is indexed again. Subsequently, the UMI quantification is initiated, but while featureCounts is able to count all reads matching to one gene or exon, it does not account for UMI sequences used. However, as mentioned in section 1.4.1, UMIs are used to deal with the PCR amplification bias and thus UMIs should be counted instead of reads. For this step, WASP utilizes the software UMI-tools [56] which generates a tab-separated values (TSV) file containing three columns with gene name, a cellular barcode and the number of UMIs counted for the gene in the corresponding cell.

Demultiplexing: As the generated TSV file contains counts for all cells aggregated, but later analysis steps focus on single cells, the data needs to be demultiplexed. Thus, the aim of this step is to separate the file into one file for each cell containing the counted UMIs per gene for this cell, or in other words, a gene expression matrix. This is performed using a custom Python script within WASP that reads the TSV file generated by UMI-tools, caching all barcodes with gene counts and finally writing a new file for each barcode. In order to prevent writing a high number of barcodes (usually more than hundreds of thousands of single files) in one directory directly, WASP creates folder structures with sub folders named after the first two nucleotide possibilities. Inside these folders, this is repeated so that barcode files are separated based on their first and second nucleotide sequence and again separated based on their third and fourth nucleotide sequence. Finally, after all gene expression matrix files have been created, these are zip-archived and compressed using the open-source software Info-ZIP.

Quality statistics: Finally, a variety of quality metrics is generated which spans over the different steps that have been performed. For this, WASP runs a number of tailored Python scripts which take in the text files from mapping, feature counting, UMI quantification and sanitized BAM files.

In order to provide a standardized format for later visualization, WASP aggregates the important information from these files and stores them inside JavaScript Object Notation (JSON) formatted output files.

So after a WASP pre-processing run has been performed, the user is presented with a results directory containing the following data:

- Gene expression matrix ZIP archive containing a file for each detected barcode
- Quality metrics JSON files
- Read quality metric results as HTML file from FASTQC

To sum up, the Snakemake workflow takes care of the previous mentioned challenges from section 3.1:

- Pre-processing at least partially requires Linux-based software
- Analysis of eukaryotic sequencing data, such as *Mus musculus*, requires a high amount of memory
- Processing of droplet-based single cell data benefits substantially from parallel processing as hundreds of thousands of barcodes need to be validated
- Due to a varying structure of barcode and UMI sequences, reads originating from devices of different manufacturers each require a tailored algorithm

The workflow is conducted in Linux, supporting all current required bioinformatics software for a single cell RNA-seq pre-processing analysis. An overview of the major steps performed, including the used software can be obtained from Fig. 20. Furthermore, Snakemake provides features to scale WASP to any given system enabling a timely and efficient analysis of large data sets. Although, a minimum amount of memory is required when working with eukaryotic data sets. Also, compute-intensive steps have been adapted to be performed in multiple simultaneous processes by splitting

large files into smaller chunks. Finally, WASP supports the most common droplet-based protocols currently in use. For this, a number of tailored algorithms have been implemented processing data from different manufacturers or devices.

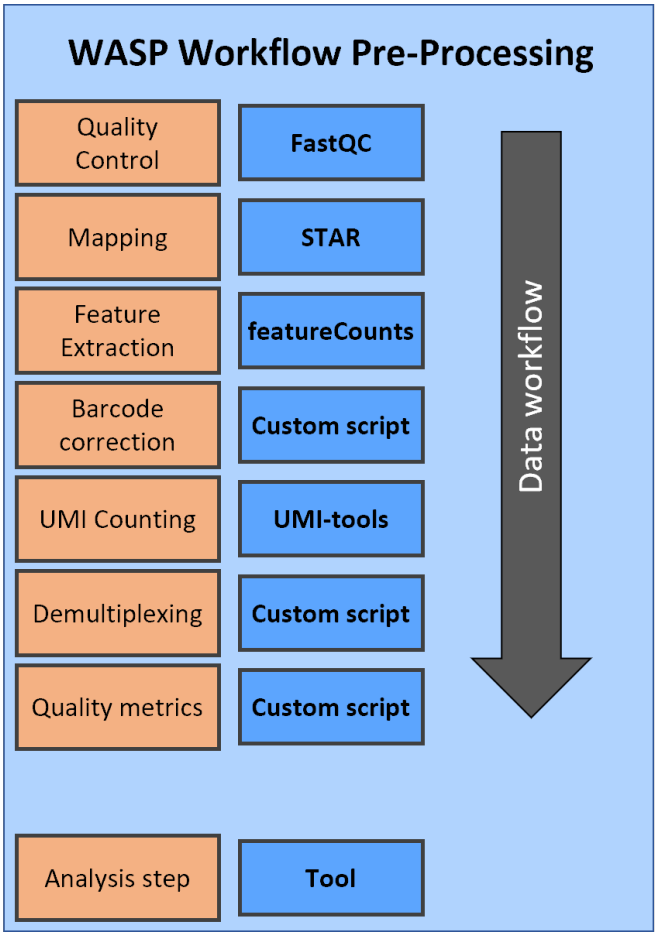


Figure 20: **Major analysis steps performed within the WASP pre-processing workflow.** Each step is visualized as an orange rectangle with the software used in a blue rectangle. While some steps utilize existing bioinformatics software, steps with WASP as used tool refer to tailored Python scripts developed specifically for the corresponding task.

3.1.2 WASP: Pre-processing visualization

While the pre-processing does simplify the analysis and ensures a standardized processing, the results are almost impossible to understand for users without advanced

bioinformatic knowledge. As mentioned in section 3.1.1, the workflow emits gene expression matrices and results mostly stored as JSON files. Therefore, a visualization of the data is crucial for researchers using WASP to verify a successful pre-processing. Also, this helps to identify possible pitfalls or low quality of input data, *e.g.* due to errors during the laboratory sample processing. Furthermore, the following downstream analysis steps require a gene expression matrix containing all barcodes belonging to cells of interest. As mentioned in section 1.5.2, droplet-based single cell RNA-seq is burdened with a high ratio of false-positive cells due to ambient mRNA fragments inside the devices. Thus, without filtering, a massively bloated gene expression matrix would be created, containing tens of thousands of gene entries (lines) for more than hundreds of thousands of barcodes (columns). Even though quality filtering during downstream processing likely removes a lot of these false-positive barcodes, this would still lead to an increase in computation time. Also, some false-positive barcodes might still remain in the processed data leading to erroneous results. In order to solve both of these issues, this intermediate step is performed which connects pre-processing and downstream analysis. One option could be to extend the Snakemake workflow to provide visualizations to the JSON files in form of image or HTML files and also an automated detection and filtering of false-positive barcodes. However, this would prevent users from changing the selected number of barcodes which is necessary if the automated detection fails. For example, if the researchers know a maximum limit for the captured cells based on manufacturer or protocol-specific information, they were unable to set this limit accordingly. Also, depending on other quality metrics such as mapping or feature identification rate, it can also be necessary to remove some barcodes before further processing is performed. This information, however, is not available before running the pre-processing, which would require at least a partial re-computing of the workflow introducing unnecessary complexity. A better option consists of a solution providing dynamical visualizations and data filtering, directly showing how the quality metrics change based on the user's input. In combination with a GUI, this also reduces the entry-barrier for non-bioinformaticians as Snakemake is a CLI-based software.

In order to fulfill these requirements, the software package Shiny based on the programming language R was chosen (https://www.R-project.org/). R Shiny is an open-source package which provides a web framework to build web applications (https://shiny.posit.co/). The code is mostly written in R, so it is possible to execute R-based code and functions on the backend, while at the same time providing a dynamic frontend. For this Shiny translates the R code into HTML, Cascading Style Sheets (CSS) and JavaScript providing a state-of-the art web platform. Shiny also enables to connect input values that can be given or modified by the user with R code in the backend. Thus, it is possible to provide dynamic content which constantly adapts based upon the user's interactions. Another advantage is Shiny's independence from a specific OS.

The implementation within WASP was therefore performed based on the R Shiny package with the addition of various other R packages. Using the shinydashboard package, a modular web page is created as frontend, showing menu options on a column of the page (http://rstudio.github.io/shinydashboard/). This menu can be dynamically extended based on performed calculations, thus allowing users to select between the results of different analysis steps. The remainder of the applications' page is used to either provide descriptions and buttons to guide the user to upload data or modify parameters, as well as visualizing analysis results.

After starting the Shiny application, the user is presented with a welcome page in a web browser giving an introduction on how to use the software. This page also contains a button which then executes an upload dialog. The access to files on the users host system is implemented in WASP using JavaScript and the R package shinyjs. Initially, the output from the pre-processing workflow needs to be imported and data is validated for both, completeness and consistency. This prevents uploading of incomplete results or incompatible files. After successful validation, WASP provides an overview of all samples and enables the user to select between different analysis steps:

Sample Summaries: This page provides a read-based overview about all analyzed samples based on the previously generated JSON files. Therefore, all sample names are presented as a table with their number of reads, barcodes and the fraction of reads that have been classified as valid if a barcode whitelist was provided. Furthermore, mapping statistics and featureCount statistics are presented in the same concept. Here, users are presented with the number of reads that have been mapped, and also reads that have been successfully assigned to features for each sample. Furthermore, unmapped and unassigned reads are categorized by reasons responsible for filtering out these reads, *e.g.* if a read was mapped to multiple loci or features or could not be mapped to the reference genome at all. All statistics are also visualized with corresponding bar plots providing an easy assessment of the overall data set quality (Fig. 21).

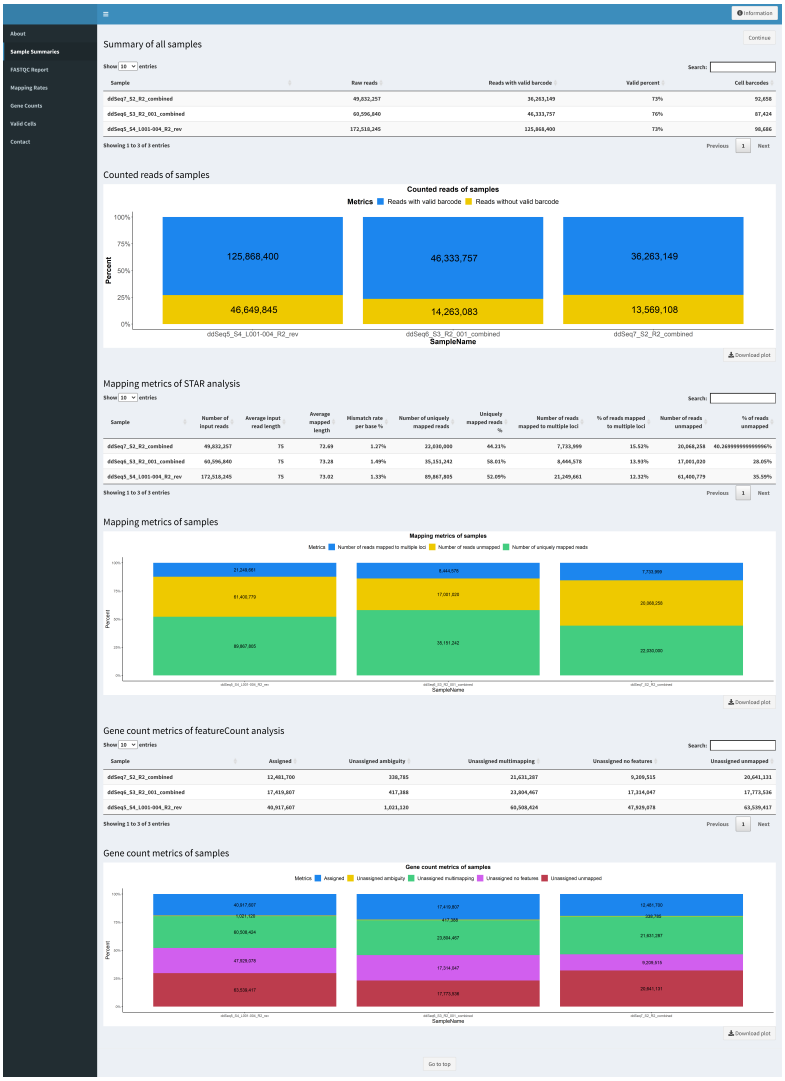


Figure 21: **Summary page of WASP's pre-processing visualization Shiny application.** This page provides a variety of quality metrics for all processed samples. The first table shows read-based information for each sample with rows corresponding to the samples and columns corresponding to the different metrics. In addition, the first bar plot shows the amount of reads with and without valid barcode. The second table provides an overview about the mapping statistics for each sample, such as the number and amount of successfully mapped reads. As above, a bar plot is used to provide visual information about the fraction of mapped and unmapped reads broken down according to category. The third table provides information about how many of the mapped reads per sample have been assigned to genomic features. This is again visualized below with a bar plot showing the amount of reads assigned or unassigned broken down by category. Furthermore, users are able to sort tables by the different columns, download bar plots and navigate to different analysis metrics using the menu on the top left side.

FASTQC Report: This page provides summarized reports about quality control of the reads or sequencing in general. Compared to the summary page, the results are presented separately for each sample, allowing the user to switch between samples by selecting the corresponding tab. As the analysis was performed using FASTQC, WASP integrates the generated output images from FASTQC and adds some additional information as a table on the top of the page.

Mapping Rates: This page provides a more detailed overview about the mapping quality performed with STAR. Compared to the Sample Summaries page, the mapping rate is shown as a stacked bar plot for each barcode individually, beginning with the barcodes with the highest number of reads in decreasing order. Users can select different categories such as mapped and unmapped reads, which are dynamically added or removed from the bar plot. Furthermore, it is possible to directly draw a box inside the bar plot to investigate the included barcodes in more detail. Also, a slider enables the user to select the maximum number of barcodes which should be displayed, including an option to directly select the predicted number of barcodes belonging to real cells. An additional barplot visualization displays the frequency of mapped reads. On the bottom of the page, an interactive table is included, presenting each barcode with its corresponding number of mapped and unmapped reads, allowing users to search for a specific barcode or change the order of each column (Fig. 22).

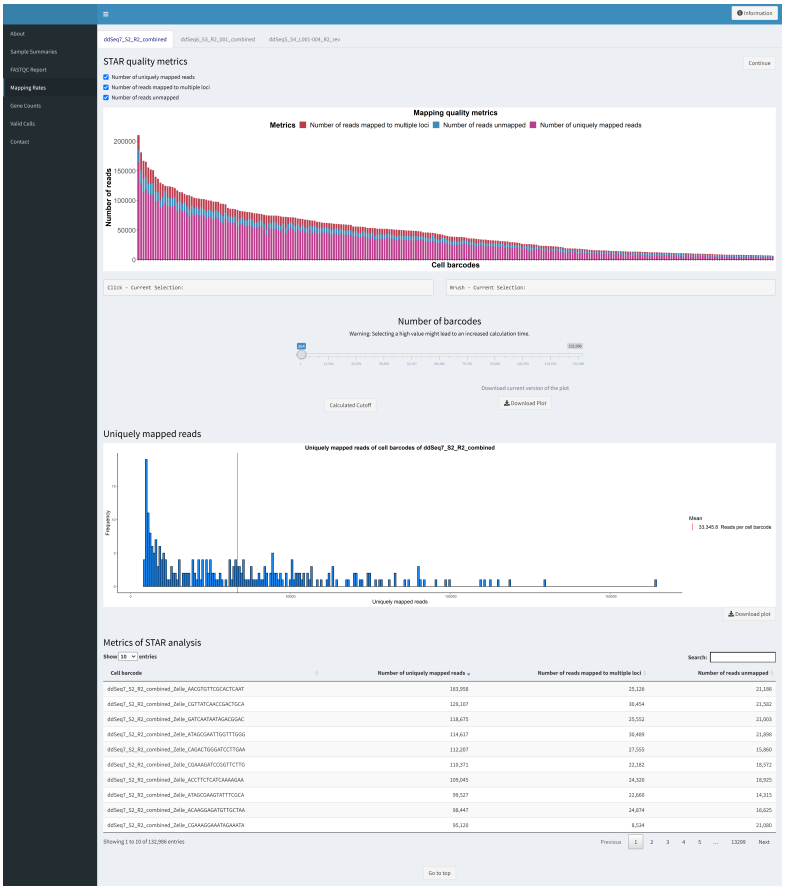


Figure 22: **Mapping results page of WASP's pre-processing visualization Shiny application.** This page provides a variety of quality metrics separated by tabs for each sample. A stacked bar plot is used to visualize the fraction of successfully mapped reads and unmapped reads separated by category. Users can select and deselect categories using check boxes resulting in a dynamical re-calculation of the plot. Below, users can select a cutoff value of barcodes to be used, ideally correlating with the number of real cells. If this value is changed, plots are re-calculated according to the newly selected number of barcodes. The second plot on the page provides information about the frequency of mapped reads per cell. Finally, a table sums up results for each barcode with the corresponding number of mapped and unmapped reads including the option to sort and search the table, *e.g.* for a specific cell. Users are also able to download bar plots and navigate to different analysis metrics using the menu on the top left side.

Gene Counts: This page provides a more detailed overview about the feature assignment performed with featureCounts. Its structure is similar to the Mapping Rates page. Users are provided with a dynamical stacked bar plot showing the assigned and unassigned features per barcode starting with the highest number of reads. The plot can be modified by selecting the different categories of assigned and unassigned features, including the option to investigate a subset of barcodes by drawing a rectangle around the samples of interest within the plot. The barcode cutoff value can manually adapted by the user with a slider element. Similar to the mapping rates page, frequency of assigned reads is displayed via a bar plot and an interactive table provides information about all detected barcodes.

Valid Cells: This page provides a summary of quality metrics including the final cutoff selection. This component of WASP does not only aim to provide users with an interactive and easy to interpret quality visualization, but also implements a gene expression generation necessary for further downstream analysis steps. Therefore, the user is presented with a visualization of the knee plot (section 1.5.2 and Fig. 12) containing all barcodes in descending order by their UMI counts on the x-axis with the logarithmic count of UMIs on the y-axis. WASP then calculates the first inflection point of the graph as the recommended threshold to be used for barcode filtering. Thus, barcodes with a lower number of UMIs are recommended to be discarded while the other barcodes are retained for the gene expression matrix. However, users can either directly select another threshold or also modify the minimum amount of expected cells, *e.g.* if the protocol used specifies a minimum or exact number of cells. This directly interacts with the threshold calculation resulting in a re-calculation of an inflection point leading to the same or higher number than the minimum of expected cells. With this option, users can directly correct a low turning point which might be erroneously calculated as too low in rare cases. Also, when a minimum number of cells is known, *e.g.* due to the used device or protocol, it can be used to aid with the correct threshold detection.

Furthermore, the page includes an interactive pie chart displaying the percentage of mapping rates and feature assignment rates for the selected number of barcodes only.

Whenever the user changes the threshold of barcodes, the plots are dynamically recalculated, providing a direct feedback on how the selection influences quality metrics. Similar to the Mapping Rates and Gene Counts pages, this page also includes an interactive table displaying each barcode with the corresponding number of UMI and gene counts. Finally, users are able to export and download a gene expression matrix per sample, containing all selected barcodes as columns and the number of UMI counts per gene as lines in form of a comma-separated values (CSV) file. In addition to the gene expression matrix, it is also possible to download a text file containing parameters of Snakemake when the Snakefile is included in the results directory as well as the selected barcode cutoff (Fig. 23).

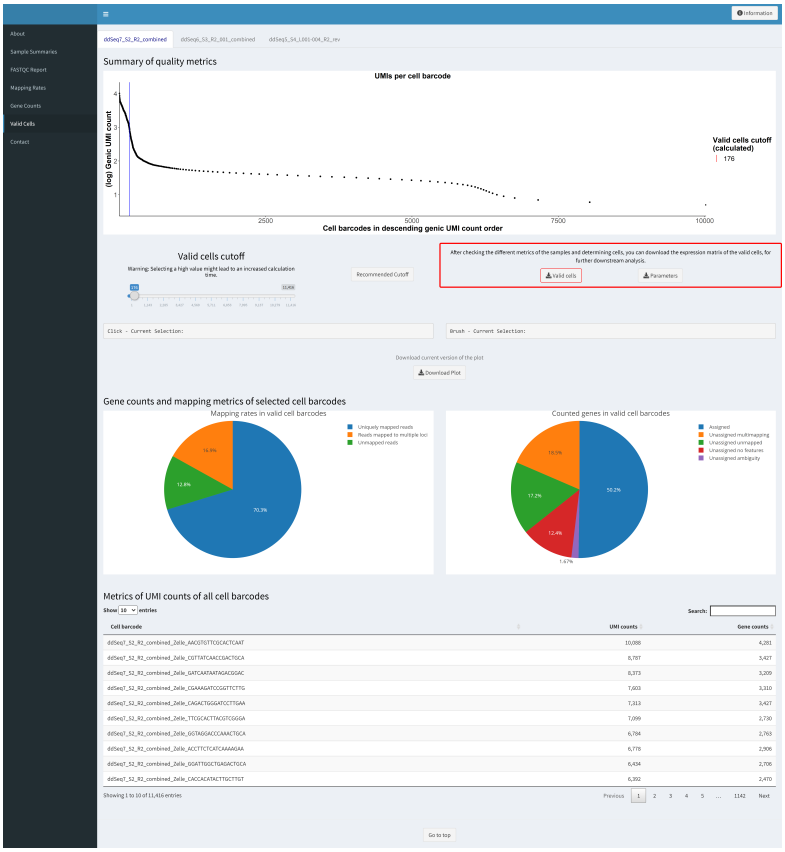


Figure 23: **Cell selection page of WASP's pre-processing visualization Shiny application.** This page represents the last step of the pre-processing visualization. The knee plot on top of the page shows the first inflection point of the curve which is expected to correlate with the cutoff value separating barcodes belonging to real cells from barcodes belonging to ambient RNA. Furthermore, users can change the cutoff value and also select a recommended value from WASP. Below, pie charts show mapping rates and feature rates for all selected barcodes. This plot is dynamically re-calculated when the cutoff value above is changed. Finally, a table on the bottom of the page shows UMI and gene counts per barcode. As before, users can search tables for specific barcodes and sort by different columns, download all plots and navigate to different analysis metrics using the menu on the top left side. Also, users can obtain a gene expression matrix as CSV file including all barcodes from the selected threshold.

In conclusion, the WASP pre-processing visualization Shiny application provides solutions for the challenges mentioned in Chapter 3.1:

- Many tools used in pre-processing steps do not exhibit visual output
- Selection of barcodes belonging to droplets with cells instead of ambient RNA

The burden of text-based quality metrics output only from various tools during pre-processing is removed by numerous visualizations using R Shiny. Furthermore, due to the interactivity of the web application, users can directly switch between different metrics for an easy quality assessment of their data set. All generated visualizations can be downloaded in their current state, based on the users selection of parameters. Also, the application aids users in filtering barcodes that likely belong to ambient mRNA fragments, thus false-positive hits. Still, users are able to adapt default values with direct feedback on how their decision impacts the data set metrics. In order to support a reproducible analysis in an FAIR manner, the applied parameters can be exported. Finally, a gene expression matrix of selected barcodes is generated for further downstream analysis.

3.1.3 WASP: Downstream analysis

The third module of WASP aims at providing the typical downstream analysis described in section 1.5.2. A huge variety of different tools is mentioned within that chapter with the vast majority of single cell-specific tools being developed in R. However, these tools or packages usually do not provide a GUI and require the user to transfer output from one package as input to another package. These requirements could theoretically be met by creating a Snakemake workflow with rules for each analysis step, connecting data flow between different packages. However, this would result in a static analysis workflow being inappropriate for steps such as clustering that do require multiple re-calculations with the option to change and optimize parameters. Generally, a parameter change at any stage would be unnecessarily complex while still not providing a GUI, thus requiring to add a visualization application afterward.

Instead, as R Shiny completely supports R packages to be run in the backend and their results displayed on the web frontend, a Shiny-based web application represents a more sophisticated solution. Therefore, this module of WASP was developed in R Shiny similar to the pre-processing visualization part. In addition to the possibility to use R modules natively and connect data flow between packages in a very similar way to developing an analysis script in R, this also introduces synergies in WASP as some parts like the design structures can be used in both Shiny applications using the shinydashboard package, simplifying the usage of both modules for users. Therefore this implementation removes OS limitations and enables users to modify parameters with a dynamic re-calculation of plots and result tables. This is of particular importance for processes with a demand for re-calculations, like the previously mentioned clustering of cells as a prime example.

When running the WASP application, the user is presented with a welcome page giving an introduction about the features of the app. The user is then guided to an upload page for the input data. In addition to the upload options, exemplary schemes for the upload files are shown to guide users to generate compatible input. While the previously generated gene expression matrix from WASP can directly be used, this also allows usage of this WASP module with an externally generated gene expression matrix, thus expanding the possible user base. Apart from the mandatory gene expression matrix, it is also possible to provide an optional annotation file containing barcodes or cell IDs with corresponding information, *e.g.* an already known cell type or sample origin. This is useful to check the data set for batch effects when analyzing cells from multiple experiments or conditions. Both files have to be provided in the CSV format. When the user has uploaded at least a gene expression matrix file, the data is verified and if the uploaded data is valid, the user is presented with additional options to start the analysis.

For this, two analysis types are possible:

- Automatic mode performs a full analysis run including all steps described in one piece. All parameters are set to a default value without the option to influence any analysis step. This mode might not generate the best possible result, but is rather meant to provide a very easy and fast overview about the whole data set.
- Manual mode performs the same set of analysis steps as the automatic mode but pauses after each step, allowing the user to change parameters before continuing or to re-calculate from a previous step by selecting the desired step in the menu on the left side, similar to the pre-processing visualization app.

Regardless of the type of analysis selected, the following steps are performed:

Filtering: Although the gene expression matrix should already be filtered for false-positive samples, another round of quality control is required. This is necessary as even real cells might be of low quality regarding their number of UMI counts or detected genes. For this, the uploaded gene expression matrix is converted into an object from the class SingleCellExperiment based on the eponymous R package [159]. This is essentially a form of data container introducing conventions for RNA-seq-based data including how cells and genes should be internally represented and provides methods for storing dimensionality reduction results. In addition to filtering out whole cells, WASP also provides the option to filter out genes with low abundance, for example genes that were only detected in a very low fraction of cells or that are only represented by a very low number of UMIs. The cutoff values can be modified by slider elements or also as direct text input (Fig. 24). Furthermore, multiple visualizations present each distribution for the whole data set as well as the selected cutoff values (Fig. 24). After a threshold has been set, the user can continue to the next step which initiates removal of cells below the selected threshold.

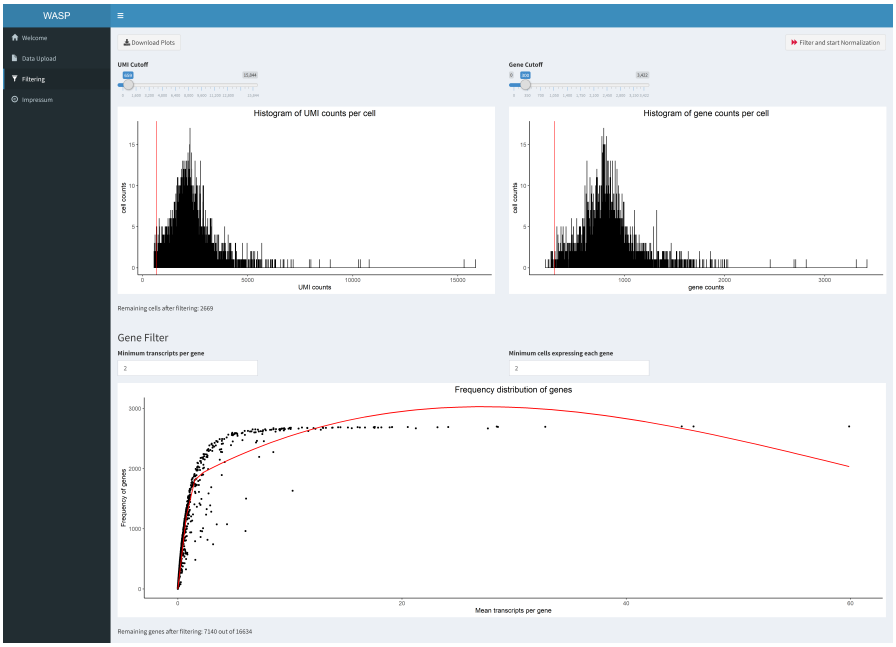


Figure 24: **Visualization of multiple quality control metrics for the downstream analysis.** Two histograms show distribution of UMI and gene counts per cell with a red line representing the cutoff value. This value can be changed via the slider elements above each plot. Below is distribution visualizing correlation of transcripts per gene and number of genes for each cell. To aid identification of trends, a smoothed regression line in red is added to the plot. Threshold values can be modified using the text boxes above the plot. Furthermore, users can navigate between the different analysis steps with the menu on the top left side. Also, plots can be downloaded using the button on the top left side and the next analysis step can be initiated using the button on the top right side.

Normalization: Following the cell filtering, the remaining data is normalized using the NormalizeData method from the Seurat R package. This is performed by dividing the counts for each gene per cell by the total counts for that cell followed by multiplication with a scale factor of 10,000. Finally, each result is natural-log transformed. The normalized data is subsequently used to perform the following analysis steps based on the Seurat package. As described in section 1.5.2, the high number of genes - usually up to tens of thousands of genes - is analyzed for HVGs, as these describe the relevant differences between cells. Also, the data is scaled to prevent domination of only a few highly expressed genes.

Furthermore, a PCA is conducted on the scaled data to identify genes that are main drivers for the variation inside the data set. The results are then visualized:

- The Elbow Plot shows each PC and its corresponding standard deviation, which results in a curve resembling a human elbow. This is very useful in determining the dimensionality of the data set, *i.e.* how many PCs are significant and should be included for the following analysis steps. When choosing too few PCs, important data might be lost, however choosing too many PCs can introduce noise to the data, thereby skewing the results. The elbow plot provides a heuristic approach to this task as the elbow - the point when the curve begins to flatten - should be chosen (Fig. 25).

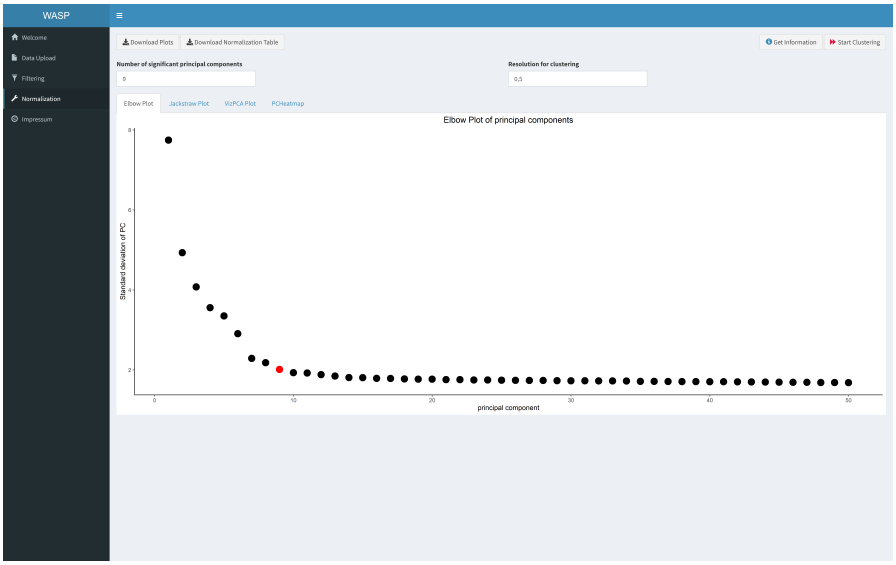


Figure 25: **Elbow plot visualization for significant PCs selection in WASP's downstream analysis Shiny application.** Each dot represents a PC, ordered descendingly by their standard deviation which is shown on the y-axis, while the number of each PC is shown on the x-axis. While the first PCs describe the highest amount of variation of the data set, with increasing PC number the variation decreases and instead noise is increased. Thus, the point where the 'elbow' flattens is recommended to be used as a cutoff. WASP marks this as a red dot and thus recommends to use the first PCs up to and including this value. Above the plot are buttons to download analysis data and plots, tabs to switch to different visualizations for PC selection as well as input fields to select the number of PCs and the resolution value for the clustering which is initiated by the button on the top right side of the page. Furthermore, users can navigate between the different analysis steps with the menu on the top left side.

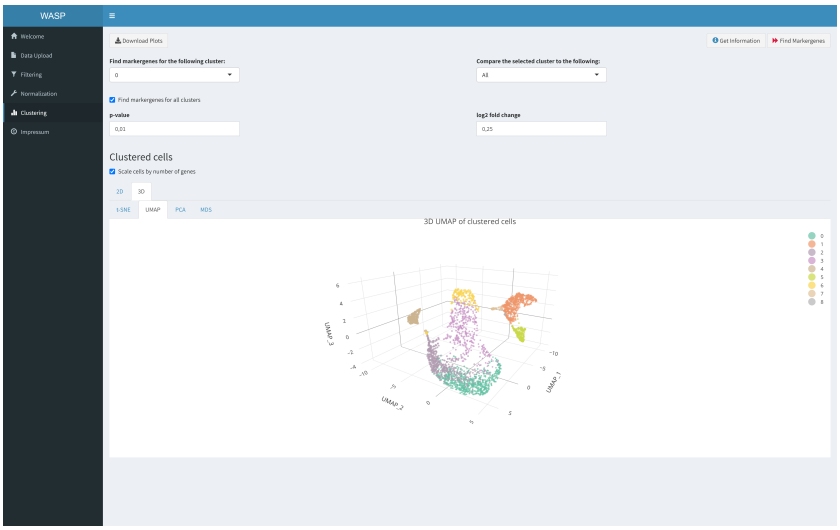
- The Jackstraw Plot is the result of a re-sampling test, permuting random subsets of 1% of the data with a re-calculated PCA which resembles a null distribution of feature scores. This procedure is then repeated, resulting in PCs showing an enrichment of low p-values features being recommended for selection.
- VizPCAPlot shows the top 12 PCs and the genes which represent the main drivers in these components.
- PCHeatmap is another visualization for main drivers of PCs, showing heatmaps with the top cells and genes responsible for the PC variation.

Based on these results, users can select a number of expectedly significant PCs. Especially the Elbow Plot is very easy to interpret and WASP additionally calculates the point where the curve begins to flatten, *i.e.* the recommended cutoff. Additionally, users can inspect the other plots as well to take a deeper look into the data down to the gene level to also use biological information for the PC selection. Finally, the number of significant components can be modified, the standard selection is based on WASP's computed recommendation. Also, as continuing to the next step initiates the clustering, users have to select a resolution parameter for this process. The default value for this is 0.5 and basically defines the granularity of clusters, thus users can increase or decrease the value to separate data into more or less clusters, respectively.

Clustering: The clustering process in WASP is based on the Seurat package and therefore is separated into two smaller steps. In the first step, the previously selected PCs are used to construct a KNN graph based on euclidean distance in the PCA space. Subsequently, edge weights between each two cells are adjusted according to the overlap both share in their local neighborhood - also known as Jaccard similarity. In the second step, the clustering itself takes place based on modularity optimization techniques. Here the Louvain algorithm is applied with the goal to optimize the modularity, a quality function. This essentially measures density of connections within a cluster compared to the density of connections within a random distribution. For this, the algorithm first assigns each cell to its own cluster to calculate an initial modularity and then iteratively merges or moves individual nodes (cells) between clusters to improve the overall modularity. The previously specified set resolution parameter influences the determination of cluster sizes and abundance to produce smaller fine-grained or fewer and larger clusters. After individual node updates, the algorithm performs optimization at a larger scale, treating clusters as single nodes instead of cells. The algorithm then iterates between cell-level and cluster-level optimization steps until an optimal modularity score has been achieved, thus representing all clusters with their corresponding nodes, *e.g.* cell types with corresponding cells.

Clustering results are presented in the form of various plots based on a dimensionality reduced visualization of the data set. These visualizations include t-SNE, UMAP (Fig. 26), PCA, and multidimensional scaling (MDS) as two-dimensional (2D) and three-dimensional (3D) visualizations. The 2D plots are generated using the scater R package and include additional information such as the cell size (gene counts) as well as meta data, if an annotation file was provided (Fig. 26a). 3D plots are generated with the plotly R package and provide a mouse overlay exhibiting further information for each cell (26b).

(a)



(b)

Figure 26: **Visualization of identified clusters with dimension-reduced UMAP plots.** (a) 2D UMAP plot showing all identified clusters as dots of different colors where each dot stands for a cell and the color represents the assigned cluster. Additionally, the dot size correlates with the gene counts for each cell and in case of available meta data, different symbols apart from dots are used for *e.g.* different cell types or samples. (b) 3D UMAP plot showing all identified clusters as dots of different colors where each dot stands for a cell and the color represents the assigned cluster. Above each plot, different tabs for other dimensional reduction-based plots are shown as well as the option to switch to the 2D or 3D version, respectively. Furthermore, the dot size correlation to gene counts can be disabled and other parameters for marker gene selection can be modified using text boxes. The menu on the left side allows navigation to other analysis steps and buttons on the top and top right side allow download of visualizations and initiation of marker gene analysis.

Before continuing to marker gene detection, users can modify the following parameters:

- Find marker genes: This enables users to select a specific cluster of interest to be analyzed for marker genes. However, it is also possible to select 'none', so that only marker genes for each cluster without specific comparison are calculated.
- Compare to selected cluster: In combination with selection of a specific cluster, it is possible to detect genes specifically separating the previously selected cluster from this one. It is also possible to select all other clusters to be compared to.
- p-value: Maximum probability that the null hypothesis - both populations are the same - is true. Thus, a lower p-value indicates a more significant difference between populations.
- log2fold change: Minimum log-scaled fold difference of a gene between the two clusters to be considered as a marker gene.

Marker genes: Subsequently, marker genes are detected and results shown with a variety of visualizations which can be accessed via multiple tabs:

- **Heatmap Top Markers** shows a heatmap in which clusters are separated into blocks in which each column corresponds to a cell and each row to a gene. The coloring of each cell indicates an increase (yellow) or decrease (purple) of the gene expression for each gene within this cell compared to either all other clusters or to a specifically selected cluster (Fig. 27).

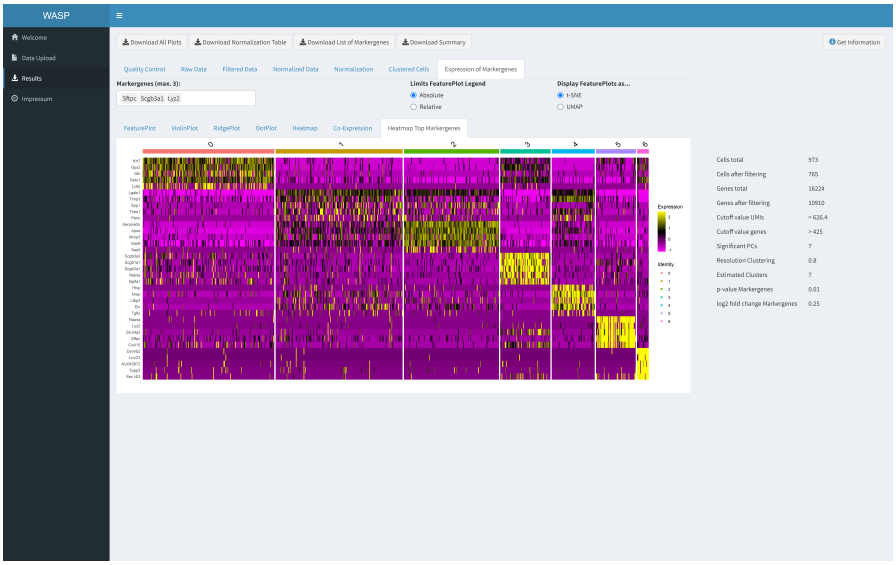


Figure 27: **Visualization of top up-regulated marker genes for all identified clusters.** Clusters are represented by numbers and color bars - identical to the cluster colors in other visualizations such as UMAP - above the heatmap visualization. Each column stands for a cell within the corresponding cluster while each row corresponds to the gene mentioned on the y-axis. Expression values of log2fold change are marked color-coded with negative values in purple to positive values in yellow. Above the visualization tab are different elements enabling figure modification for other plots. Furthermore, users can navigate between the different analysis steps with the menu on the top left side. Also, plots and data tables can be obtained using the buttons on the top side. On the right, used values for each analysis step are presented.

- **Expression of Marker Genes - FeaturePlot** shows a t-SNE and UMAP plot of the data set in which an up-regulated gene expression for each cell is indicated on a blue to red scale which can be chosen either in absolute or relative log-scale. Furthermore, users can enter or select gene names to generate the plot for a gene of interest.

- **Expression of Marker Genes - ViolinPlot** shows a violin plot for the selected genes providing information about the distribution of the gene expression up-regulation in each cluster.

- **Expression of Marker Genes - RidgePlot** shows a ridge plot for the selected genes providing information about the distribution of the gene expression up-regulation in each cluster.

- **Expression of Marker Genes - DotPlot** displays each cluster as a dot, with the dot size indicating the fraction of cells expressing the selected genes and the coloring with a blue to red scale indicating the average expression level of the gene for this cluster.

- **Expression of Marker Genes - Heatmap** shows a heatmap with similar design to the top markers heatmap but with the selected genes instead.

- **Expression of Marker Genes - Co-Expression** displays the data set as t-SNE or UMAP plot but in contrast to the previous FeaturePlot visualization, expression of two genes is combined. Thus, a green or blue color indicates that only the first or second gene is expressed, and a turquoise color indicates that a cell highly expresses both genes.

- **List of Marker Genes** provides an interactive table showing all detected marker genes including various analysis data such as the average log fold change for a gene in a specific cluster, the p-value or fractions of cells expressing the gene in either of the compared groups, *e.g.* cluster 1 vs all other clusters. In addition, the table can be sorted by category and searched for specific genes.

Results: Finally, WASP aggregates generated visualizations, enabling users to review results of each performed analysis step which can be exported in Portable Document Format (PDF) format along with the list of marker genes. Furthermore, a summary of selected parameters and quality metrics can also be exported to support reproducibility.

In conclusion, the third module of WASP provides a robust downstream analysis workflow incorporating typical analysis steps mentioned in Chapter 1.5.2 and visualized in Fig. 29.

Also, the mentioned challenge from 3.1 have been taken into account:

- Downstream analysis packages are used by writing programming code which also includes specific parameters

The implementation of this module as a Shiny application enables an easy and native implementation of various single cell-specific R packages. Furthermore, the web application is designed to integrate important parameters as input elements in the Shiny GUI, which enables easy access to users unfamiliar with writing or developing R code. As this module contains a large number of visualization-aided analysis steps, and biological insights, special attention was paid during software development to the export of all parameters used and visualizations ready for publication. Finally, the website menu enables a quick transition between different analysis results including their recalculation with modified parameters.

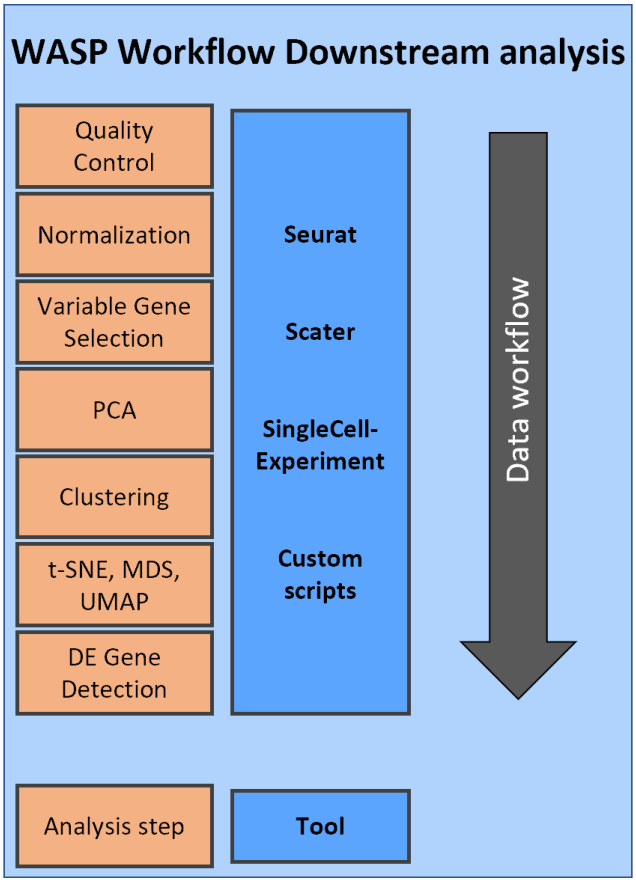


Figure 28: **Major analysis steps performed within the WASP downstream workflow.** Each step is visualized as an orange rectangle with the software used in a blue rectangle. Compared to the pre-processing workflow (Fig. 20), the steps often comprise multiple tools instead of a clear separation for each analysis. This is due to the interoperability of internally used R objects, enabling to use multiple R packages in a synergistic manner.

3.1.4 WASP: Distribution

As a main concept of WASP is the possibility to use the software on-premise to be suitable for being used with sensitive data, a variety of distribution options have been considered. In order to provide the best possible analysis results, components from multiple domains (R, Bioconductor, Python, Snakemake) have been used for the implementation of WASP. While the different domains provide established distribution solutions, they are limited to their own ecosystem, hence none of these provides a suit-

able solution for a distribution of WASP. Generally, all modules and corresponding scripts are maintained via Github (`https://github.com/andreashoek/wasp/`), but in order to distribute WASP efficiently, additional solutions were evaluated. A distribution method which can be considered closest to a one-fits-all solution is based on Docker. Docker is a framework which provides software containers using OS-level virtualization (`https://www.docker.com/`). The concept is based on Linux Containers (LXC), which basically describes a method to allow sandboxing of processes from each other. Therefore, LXC uses a variety of features from the Linux kernel, especially user namespaces and control groups (cgroups) [160]. Namespaces allows to create separate environments (namespaces) including, *e.g.* own process IDs, users, filesystem including a root directory. Thus, processes outside a namespace are not visible or accessible to processes inside a namespace. However, the processes appear to run in a normal Linux system and share the same kernel. The cgroups feature allows grouping of processes, limiting their resource consumptions (*e.g.* CPU and random-access memory (RAM)) [160]. Each Docker container provides an isolated environment encapsulating applications and dependencies together and running processes in isolated user spaces while still using the underlying kernel of the OS. Therefore, containers are usually lightweight and executed without much overhead as they do not have to include an entire OS. Docker uses its own objects such as images, containers, networks or volumes. Images are templates including instructions for creating a container. Usually, an image makes use of another image, *e.g.* a basic Ubuntu image, and contains additional features. Consequently, a container is an instance of an image that executes the included application. In addition to commands given in the template, a container can be modified at its start. Furthermore, it is possible to pause, stop, move and delete a container, connect the container to one or more networks and attach storage to the container.

Therefore, a Docker container for each WASP module was created which is publically available at the Dockerhub registry (`https://hub.docker.com/r/andreashoek/wasp_prepro_pipe`,

https://hub.docker.com/r/andreashoek/wasp_prepro_shiny,https://hub.docker.com/r/andreashoek/wasp_postpro_shiny). Furthermore, the image files that were used for container creation are publically available in WASPs Github repository as well to enable users to build or modify own WASP containers.

However, even though Docker provides an out-of-the-box solution, not all users are able to access Docker. This might be due to security concerns on distributed systems or users might not be authorized to install Docker. Also, Docker is available for Windows, but its installation and usage might not be feasible for non-experienced users. Therefore, alternative distribution options were implemented with different solutions for Snakemake and Shiny.

The Snakemake workflow requires a variety of different bioinformatic tools and Python packages, which a user would have to install manually to make sure that they are accessible within Snakemake. In order to simplify this process for users, the usage of Conda is recommended. Conda is an open-source CLI-based package management system available of Linux, Windows, and MacOS, that creates and manages isolated software environments (https://anaconda.org/anaconda/conda). While originally developed for Python packages, Conda nowadays enables users to search a variety of different repositories, called channels, providing a comprehensive collection of essential Linux CLI-based tools as well as bioinformatics software and Python packages. Selected tools and dependencies can be installed in a specific directory on the host system, also called a Conda environment. Creating a Conda environment then searches the specified packages in provided channels and installs all tools including their required dependencies inside the environment. Following the installation, the environment can be activated to directly access all installed tools and packages from within a terminal session. This encapsulation also enables users to easily install multiple versions of tools and packages in parallel. Therefore, WASP provides a configuration file in the YAML Ain't Markup Language (YAML) format including all necessary tools and Python packages, and their corresponding channels. Thus, users only need to have Conda or the

alternative package manager Mamba (https://github.com/mamba-org/mamba) installed to automatically create an environment for WASP's pre-processing workflow.

As the second and third modules of WASP are based on R Shiny, the installation procedure generally is less complex. Users only need to install R itself and the required R packages which can be installed in R directly. A list of packages can be found at the beginning of the Shiny application file. Furthermore, as R Shiny is not limited to a Linux-based environment, a standalone version for Windows was developed to provide an installation-free usage. This is done by using a portable R version bundled with all necessary packages which can simply be copied to other systems as it does not use any system-wide path specifications. To run the Shiny modules, a batch (BAT) file was created which can be directly executed within Windows, starting WASP and automatically opening the web page on the system's default web browser. All files have been ZIP archived and thus only need to be downloaded, extracted and the BAT file executed. The Windows versions can be obtained via Github (https://github.com/andreashoek/wasp) and are currently stored within the German Network for Bioinformatics Infrastructure (de.NBI) cloud.

Although running WASP on local premises provides the highest level of data privacy, some researchers might prefer to use the software as a service. This could be due to a lack of suitable hardware or the need to run the software from multiple devices without having to download WASP multiple times. For this, a publically usable version of WASP has been created based on the open-source framework Shinyproxy. Shinyproxy is a Java Spring Boot-based web application with a focus on providing web pages with a GUI to run Shiny apps (https://www.shinyproxy.io/, https://github.com/openanalytics/shinyproxy). This includes user authentication and allows to run multiple Shiny apps for multiple users in parallel while ensuring fully isolated workspaces for each application. Whenever a user selects to run a Shiny application, the corresponding application is spawned as a Docker container and the user is forwarded to the Shiny session. Thus, Docker compose, a tool to define and manage multi-container applications, was used to spawn Shinyproxy as a container,

providing a web page for users to log in with credentials and select the two WASP Shiny applications. After selection, Docker compose manages the creation and life cycle of each corresponding WASP container, including its network connection. Hence, after the WASP session is terminated, the container and all uploaded data is automatically removed ensuring data privacy. This approach has also been adapted for other scientific Shiny applications as well [161].

3.1.5 WASP: Summary

In conclusion, WASP represents a software platform providing a comprehensive analysis of droplet-based single cell RNA-seq data. The software comprises three modules separating the analysis into pre-processing, evaluation of pre-processing with cell calling and finally downstream analysis. The implementation of the pre-processing is based on the workflow engine Snakemake, thereby enabling scalability and portability to various HPC systems. To support a broad range of droplet-based data, a tailored processing of barcode and UMI schemes for multiple manufacturer-specific protocols has been implemented. Pre-processing visualization and downstream analysis modules are implemented as R Shiny apps enabling integration of single cell-specific R packages in combination with a web browser accessible GUI. Furthermore, both modules provide a wide range of interactive visualizations and enable users to modify analysis parameters without being familiar with R code generation or bioinformatic knowledge. Separation of downstream processing in an individual module expands the potential user base, as externally generated gene expression matrices can be analyzed, as well. All generated visualizations can be exported as publication ready images as well the used parameters to facilitate reproducible analysis in regard to FAIR principles.

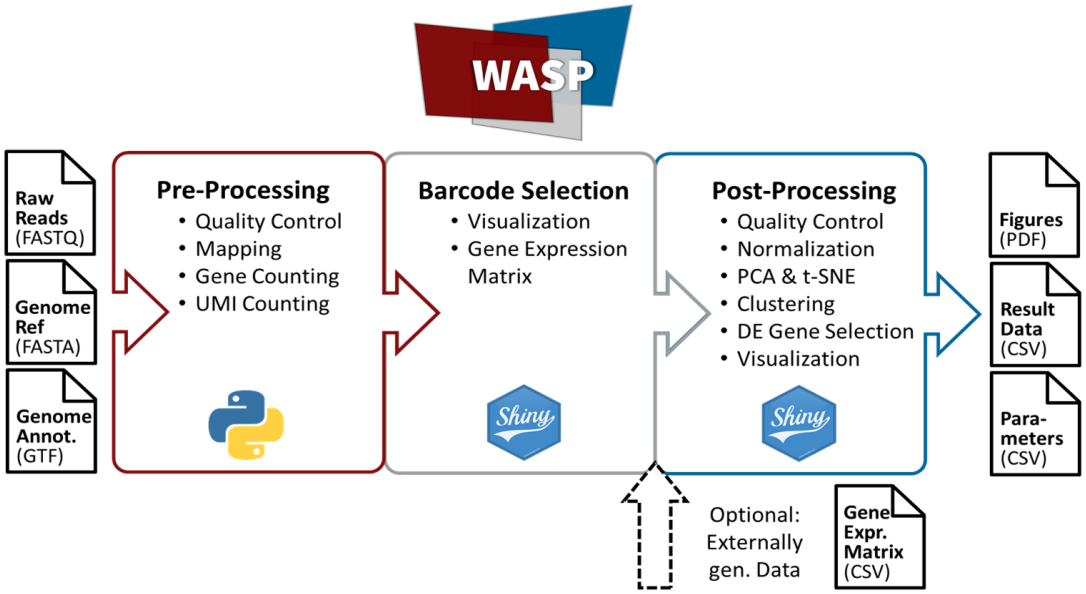


Figure 29: **Overview of WASPs different modules.** Pre-processing is performed using Python and the workflow engine Snakemake and requires sample data as FASTQ files and a genome reference in FASTA format with a matching annotation in GTF format. The resulting data from this step is then visualized based on an R Shiny application which also generates a gene expression matrix in CSV format containing all barcodes assigned to suspected cells. Finally, the generated gene expression matrix or an alternatively obtained gene expression matrix is uploaded in the third WASP module. This module is also implemented as an R Shiny application and provides a variety of general and single cell-specific analysis steps providing insight into the biological meaning of the data. Furthermore, users can export results in multiple formats: publication-ready visualizations as PDF format, result tables and used parameters for FAIR-based data analysis as CSV format.

In order to enable the analysis of sensitive data, a variety of distribution options for WASP are implemented. All modules and source code is publically released via Github (`https://github.com/andreashoek/wasp`). For a simplified accessibility, Docker is used to provide a containerized version of WASP which includes each module with its required dependencies. Furthermore, a Conda environment YAML file definition is included to provide a simple installation of all dependencies necessary to run the pre-processing workflow on Linux-based OS. Both Shiny applications are available as standalone versions on Windows, which only require a web browser as a prerequisite.

3.2 Single cell analysis workflows with openBIS

The second thesis aim was the integration of the previously introduced single cell RNA-seq analysis software - WASP - into an existing data management platform. Although WASP already provides user-friendly operation by integrating a GUI for as many processing steps as possible, the pre-processing workflow is still based on CLI execution. This is necessary as the utilized tools require a Linux OS. Also, when working with eukaryotic organisms such as *Mus musculus* or *Homo sapiens*, the alignment step requires a higher amount of system memory, making execution of the pre-processing on a standard laptop or laboratory computer unfeasible. Hence, WASP already provides a variety of advantages and simplifies pre-processing for users with access to appropriate systems. However, this is not the case for many researchers or lab members, and when running large-scale studies or multiple samples, access to an HPC system could noticeably speed up the analysis. In addition, data storage poses a challenge in this regard, as researchers generate a large amount of data (sets) that benefit should be accessible for analysis and comparisons across different laboratories and sites. This problem has already been described extensively in sections 1.6 and 2. Also, in regard to FAIR principles, it is the most preferred option to bundle data sets and workflows including their results and offer the option to directly run analysis from within the data storage platform. As previously mentioned in chapter 2, the BCF provides a modified openBIS system including a module to run workflows with stored data. Thus, the integration of WASP as an analysis platform for transcriptomic single cell data in openBIS meets the previously mentioned criteria and aims to simplify reproducible and efficient single cell data set analysis even further.

3.2.1 Integration of WASP into openBIS

The limited visualization possibilities mentioned above (section 1.6.1) stand in contrast to the interactive Shiny modules of WASP. Even though these modules are accessible via a web browser, they focus on providing interactive results for users, which can not be delivered by a static HTML page. Therefore, an integration of WASP into openBIS is

desirable to provide interactive usability. However, the openBIS implementation does not provide execution of software frameworks outside of the workflow registry. Thus, it is possible to run the WASP pre-processing workflow as it is Snakemake-based and therefore supported by the workflow registry. However, its results can only be stored as files inside openBIS but not visually accessed. One way to work around this limitation is to download all data from the pre-processing locally to continue with the analysis locally. This would be a functional alternative and especially makes sense for the downstream processing to promote usability for externally pre-processed data. However, users would benefit from accessing at least the pre-processing Shiny module directly from openBIS to be able to perform a full pre-processing workflow with GUI support. Also, this further lowers the entry-barrier to perform single cell RNA-seq analysis for non-bioinformaticians as they do not have to switch between different systems.

In order to achieve a direct usage of WASP inside openBIS, the pre-processing workflow was first integrated into it. As mentioned before, the workflow registry already supports execution of Snakemake workflows and is capable of handling Conda environments. Therefore, this part of WASP was easily integrated in cooperation with Sven Griep. This integration also enables to overcome the remaining entry-barrier of WASP - the CLI-based usage of the pre-processing module. As the workflow registry comes with its own GUI, users are able to select all required parameters via a dropdown menu, select data sets directly in the project browser and start the analysis via a button. Following such an analysis, resulting pre-processing files are stored inside the openBIS system, connected to the original experiment and are displayed in the project browser. In order to provide the mentioned interactive visualization, pre-processing was extended and an external modified WASP pre-processing module was implemented.

This pre-processing extension now includes another archive step which collects all results inside a ZIP archive and is then followed by an upload step (Fig. 30a). This enables to not only store the result data from WASP in openBIS, but also performs an upload into an S3 object storage bucket located inside the de.NBI cloud. This concept is built upon an Ubuntu-based VM located in the de.NBI cloud running Docker

and an open-source nginx web server. Whenever the upload step is reached, a custom Python script is used within Snakemake to perform an HTTP GET request to receive a S3-Uniform Resource Locator (URL) for data upload. The nginx server on the VM redirects this request to a custom Python script generating a new random ID sequence. Furthermore, utilizing Amazon's open-source Boto3 Python package (`https://github.com/boto/boto3`), this ID is combined with a pre-signed URL providing temporarily limited access to an S3 bucket. After receiving the URL, the upload is executed within Snakemake using the HTTP POST method. Furthermore, an HTML document is generated which contains a redirect link including the received ID corresponding to the uploaded data. When the pre-processing workflow is finished, the HTML file is stored among all other result files in the openBIS system. As openBIS does natively support HTML files, users can select this file in the openBIS project browser and thereby follow the redirect link (Fig. 30a).

Subsequently, the request is received and redirected via the nginx server in the VM to a second Python script. This uses the received ID and again the Boto3 library to generate a temporary URL pointing to the corresponding data set. In addition, the script starts a modified WASP Shiny Docker container which uses the temporary URL to download and extract the data set. This container is then presented in the users web browser. Instead of presenting the user an upload button for WASP results, the software directly presents the Shiny web application with the data set corresponding to the ID. In summary, the user simply opens the HTML file and is automatically redirected to a remote WASP Docker container interactively presenting results from the pre-processing workflow (Fig. 30b). Similar to running the Shiny application locally, the user can continue to perform a WASP analysis described in section 3.1.2.

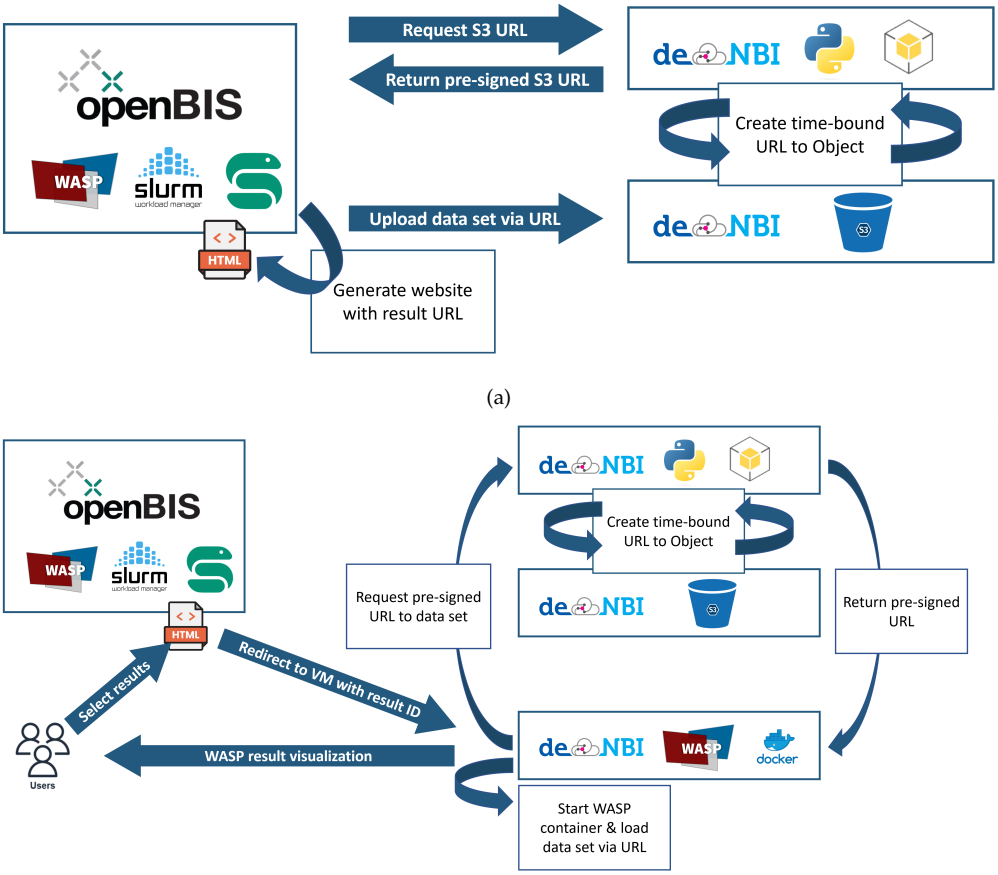


Figure 30: **Schematic overview of the WASP integration in openBIS.** (a) During the final steps of the pre-processing workflow performed via the workflow registry within openBIS, the Snakemake workflow sends a request for an S3 URL to a VM in the de.NBI cloud. Using the boto3 package, a time-bound URL with a random ID providing access to an S3 bucket is generated. This pre-signed URL is returned and used to upload the pre-processed WASP. Finally, the URL with the ID is written to a HTML document and stored in openBIS. (b) Users can select the previously generated HTML file to analyze the results in an interactive WASP session. For this, the HTML document redirects the user from openBIS to a VM in the de.NBI cloud. This request also provides the ID for the data set of interest which is used to request a temporary pre-signed URL providing access to the data set in the S3. Subsequently, a Docker container with the WASP pre-processing Shiny application is spawned, the data set imported into the container using the S3 URL and the user gets redirected to the WASP GUI.

3.2.2 WASP openBIS integration summary

In conclusion, the integration of WASP's pre-processing into the openBIS workflow registry provides a user-friendly interface for this step, thereby making the sophisticated features of the WASP pre-processing workflow available to all users of the openBIS instance. Also, as openBIS is directly connected to the HPC infrastructure maintained by the BCF, users are not required to provision own compute resources. Additionally, users are able to use the sophisticated data storage and management features provided by openBIS and can directly perform a single cell RNA-seq analysis from within this system, extending FAIR data analysis possibilities. This is achieved by the extension of the WASP Snakemake workflow to perform a duplicated data upload into the de.NBI S3 storage enabling to use remote WASP instances to deliver visualization and further analysis steps overcoming the limited possibilities to explore results within openBIS. Using the Boto3 library and custom Python scripts, the process of data upload is fully automated and its use of random IDs prevents third parties from unauthorized data access. Finally, integration of data access into an HTML document allows users to directly perform further interactive WASP analysis steps from within the openBIS web frontend. Thus, the WASP integration into openBIS provides an additional step towards providing researchers with a comprehensive platform for secure and FAIR sequence data storage, management and analysis with a low-entry barrier.

4 Results and practical application

WASP was designed as an easy solution to provide researchers with the appropriate means to conduct single cell RNA-seq analyses. A need for such a solution originally arose from bioinformatic analysis requirements based on experiments carried out as part of the KFO309 research consortium. The successful application of the WASP platform is demonstrated in this thesis using two examples.

For the first example in chapter 4.1, WASP was used for a ddSEQ-based single-cell analysis derived from *Mus musculus* lung organoids within the KFO309. The second example in chapter 4.2 shows the application of WASP in a research project focusing on the Asteria-based single cell analysis of *Galleria melonella* as an insect infection model for the lung disease tuberculosis.

In addition to the direct application of WASP, an example in chapter 4.3 demonstrates the openBIS integration of WASP using a 10x-based single-cell RNA-seq data set, which enables both the storage of experimental data and meta data and the direct analysis of this stored data.

4.1 Single cell analysis of *Mus musculus* bronchioalveolar lung organoids

The impact of respiratory tract diseases for humanity drastically showed in the recently emerged COVID-19 pandemic which resulted in a total of 663,640,386 cases and 6,713,093 deaths worldwide as of January 21, 2023 [162]. Other notable examples of pulmonary diseases caused by coronaviruses include severe acute respiratory syndrome (SARS) and middle east respiratory syndrome (MERS) [163] and the less severe common cold, caused by coronaviruses as well as a variety of other viruses. These diseases affect almost every person each year [164]. Further airborne diseases include influenza, caused by various influenza viruses, and a variety of bacteria-related pulmonary infections, which can lead to pneumonia, often as a secondary infection following a virus-based disease [165]. In addition to infectious diseases, chronic respiratory disorders such as asthma, idiopathic pulmonary fibrosis (IPF), and chronic obstructive

pulmonary disease (COPD) introduce another significant health issue accounting for approximately 8% of global mortality [166], thereby making this the 4th leading cause of deaths worldwide [167]. Finally, lung tumors increase the burden of lung-related morbidity and mortality, thus making research an important topic for global health.

Research of the lung introduces a variety of cellular and structural complexities. Within an organism, the lung is relatively inaccessible and further is subjected to continuous movement [168]. In addition, the cellular composition comprises more than 40 different cell types [169], while other organs such as the stomach or liver solely consist of four or five main cell types, respectively [170, 171]. These cell types are further organized in multiple regions in different proportions embedded in a 3D structure. Thus, previous study approaches involving *ex vivo* tissues do not provide accurate dynamic information [168]. Other approaches, *e.g.* using a 2D based cell culture of lung cells, do provide opportunities for studying a variety of clinical conditions or drug responses of these cells [168]. However, this method lacks an appropriate organ-like structure including its cellular composition. Therefore, these approaches exclude specific cell types, structures and cell-to-cell interactions leading to a limited model for studying lung disease mechanisms. For example, influenza or severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) infections may induce inflammation and thus damage and cell death to epithelial lung tissue, which can subsequently lead to a reduced resistance against other pathogens [172, 173]. Another example is the effect of smoking on epithelial lung integrity, which leads to epithelial remodeling playing an important role in COPD [174]. Further, the disease mechanism of COPD [175], IPF [176] and specific lung cancers [177] is fundamentally influenced by lung-infiltrating immune cells. However, these examples require a more complex study model than a 2D cell culture system to shed light on the involved biological processes, and thus, identification of treatment targets.

An approach to overcome these limitations to some extent and to implement a more sophisticated analysis model is the generation of so-called organoids. These can be defined as *in vitro* 3D structures based on different types of stem cells, which form

structures resembling an organ when cultured appropriately [178]. In the past decade, organoids have been successfully cultured for multiple tissues including colon [179], stomach [180], liver and pancreas [181], and more recently, the lung [182]. The respiratory system comprises upper and lower respiratory tracts. The upper respiratory tract includes the nasal cavity as well as larynx and pharynx, while the lower respiratory tract includes the trachea, primary bronchi, bronchioles and alveoli. The epithelium covering the upper respiratory tract consists of basal cells, mucus-producing goblet cells, and ciliated cells facilitating mucociliary clearance. The tracheal epithelium mainly consists of basal, goblet and ciliated cells, as well, but further includes club and tuft cells, which are fulfilling protective roles such as repair after injury [183], and chemosensory, neuronal and immunological functions [184], respectively [185]. The epithelium in the bronchioles comprises the same cell types as the tracheal epithelium, but is based on more cuboidal-shaped cells with shorter cilia and additonal contains pulmonary neuroendocrine cells modulating immune processes through secretion [186] [185]. In contrast to the previous tissues, alveolar epithelium primarily consists of alveolar type I and type II cells. Type I cells cover around 95% of the alveolar surface and play a crucial role for the gas-exchange function in the lung [187]. Type II cells secrete surfactant and can further differentiate into type I cells following injury repair [187]. Therefore, respiratory-based organoids can be classified into three different types according to their cellular composition [188]:

- Airway organoids resemble upper respiratory tract tissue as well as lower respiratory tract tissues except alveoli. Thus, these organoids typically consist of basal, ciliated, goblet, tuft and pulmonary neuroendocrine cells [188].
- Alveolar organoids resemble alveolar epithelium, consisting of type I and type II alveolar epithelial cells (AECs) [188].
- Lung organoids comprise cells from airway and alveolar tissue and thus resemble a more complete model of the lung [188].

These organoids can be grown from different cells obtained from either adult tissue, embryonic progenitor cells or induced pluripotent stem cells, which are reprogrammed adult somatic cells altered through induced genes and factors [188] [189]. In a first step, the tissue is either enzymatically or mechanically digested to extract cells from the extracellular matrix. Following, cells have to be filtered based on *e.g.* specific cell markers using FACS or Magnetic-activated cell sorting (MACS). Filtered cells then need to be cultured to facilitate cellular differentiation and growth of the organoid structures. One of the most common culture methods is the usage of Matrigel, which is based on the basement membrane matrix secreted by Engelbreth-Holm-Swarm mouse sarcoma cells [188] [190]. This serves as a sort of artificial extracellular membrane containing growth factors and enabling attachment and growth of the filtered cells [190]. Other methods include an air-liquid interface using a permeable membrane to enable contact between cells and liquid medium and self-assembled spheres, where cells are seeded onto ultra-low attachment plates [188]. These methods have been utilized to create a variety of different organoid approaches. For example, isolated basal cells can be grown to so-called tracheospheres consisting of basal and ciliated luminal cells [191]. Co-culturing of AEC type II cells with Platelet-derived growth factor receptor A (PDGFRA) expressing lung mesenchymal cells on the other hand enables growing of alveolar-like structures exhibiting AEC type I and II cells [192], [193]. A higher variety in the organoid structure has also been achieved by combining lung epithelial stem cells with bronchioalveolar stem cells (BASCs) and lung endothelial cells [194]. Although these approaches already provide a more complex structure to a distinct level and enable an improved model for research, they still reflect limited morphological and cellular properties of a complete lung. Thus, many approaches only contain a sample of the cellular composition such as lung epithelial cells and sometimes mesenchymal cells, but these organoids also do not resemble the complexity of bronchioalveolar structures. Furthermore, cells of myeloid origin residing in the lung-tissue, crucial mediators for lung development, immune response, and tissue regeneration are missing.

4.1.1 Generation and bioinformatic analysis of bronchioalveolar lung organoids

The following section describes a study, alongside which WASP was developed and successfully applied for the first time; the contents of this section are based on the original publication describing the results (Vazquez-Armendariz *et al.*, [195]) without further explicit attribution.

To overcome the aforementioned limitations of lung organoid approaches, the KFO309 has dedicated a project to facilitate development of a more complete lung model. A major motivation for this project is to obtain an organoid model which sheds further light on regenerative processes of lung tissue in response to viral infections such as influenza. In this regard, a previous study already showed that a specific population of distal lung epithelial cells enriched with lung epithelial stem/progenitor cells reveals a high proliferative potential in reaction to influenza A virus (IAV)-induced injury [196]. The epithelial cell population is further characterized based on a distinct surface protein profile exhibiting a high expression level of epithelial cell adhesion molecule (EpCAM), a low expression level of cluster of differentiation 24 (CD24) and the presence of Stem cells antigen-1 (Sca-1), or in short: $EpCAM^{high}CD24^{low}Sca\text{-}1^{+}$ [197]. Furthermore, this population proliferates and forms organosphere structures when cultured with multiple growth factors in Matrigel culture and initiates lung tissue repair and even generation of AEC type I cells when transplanted into IAV-injured murine lungs, exhibiting stem cell properties [197]. This ability further depends on a subpopulation of BASCs, which characteristically co-express the genes Secretoglobin Family 1A Member 1 (SCGB1A1) and Surfactant Protein C (SFTPC). These BASCs enable differentiation into club cells and AEC type I and II, contributing to bronchiolar and alveolar epithelial tissue repair following injury. Based on this prior knowledge, a new type of lung organoid should be established.

In order to develop a protocol for generation of so-called bronchioalveolar lung organoids (BALOs), the necessary cells need to be isolated. For this, leukocyte and endoethelial cell-depleted lung homogenates from wild-type *Mus musculus* were used. Based on FACS sorting, lung epithelial stem/progenitor cells with the expression pro-

file EpCAM^{high}CD24^{low}Sca-1^{+} were isolated. In addition, a subset of lung-resident mesenchymal cells (rMCs), defined by absence of EpCAM and presence of Sca-1 - EpCAM^{-}Sca-1^{+}, was isolated as well. Both isolates were subsequently cultured together in Matrigel for organoid growth. A successful organoid formation was observed with development of central bronchiolar-like structures after 10-11 days and development of peripheral alveolar-like structures after 21 days. Validation via fluorescent imaging also showed positive staining of bronchiolar-like structures for the club cell marker SCGB1A1 and alveolar-like structures for the AEC II marker SFTPC [195].

Apart from successful BALO formation, making up about 80% of all organoids, two more types of organoids have been observed during culturing: bronchiolospheres, a phenotype based on composition of large tube structures making up 14% of formed organoids, and alveolospheres, a phenotype based on composition of compact saccular structures, making up 6% of all organoids, with both phenotypes resembling the less complex organoids mentioned before [195]. In order to further understand the development of multiple phenotypes, two different populations of EpCAM^{high}CD24^{low}Sca-1^{+} cells with integration of different fluorescence proteins were selected. One cell population expresses tdTomate-red, while the other population expresses green fluorescent protein (GFP), resulting in red or green organoids only. Thus, observed organoids were derived from clonal expansion of a single cell instead of self-assembly of multiple cell types which would result in colorfully mixed phenotypes [195].

To characterize the cell type(s) responsible for the different organoid phenotypes, the used EpCAM^{high}CD24^{low}Sca-1^{+} cells were analyzed for different epithelial markers. As mentioned before, BASCs have repair abilities following lung injury and thus are able to differentiate into bronchiolar and alveolar cells. Therefore, cells were tested for expression of a club cell marker SCGB1A1 and an AEC type II marker SFTPC by extracting cells from double reporter mice. Thus, expression of SCGB1A1 is coupled to red fluorescent protein mCherry and expression of SFTPC is coupled to yellow fluorescent protein (YFP). The analysis resulted in a distribution of around 95% of the cells being positive for club cell marker and negative for AEC II marker (SCGB1A1^{+}SFTPC^{-}),

about 5% of the cells being positive for both markers (SCGB1A1$^+$SFTPC$^+$) and less than 0.5% negative for club cell marker and positive for AEC II marker (SCGB1A1$^-$SFTPC$^+$). Subsequently, the formed organoids were analyzed for their marker composition as well, resulting in a very different distribution. After 21 days of culturing, the largest fraction of organoids with 80% were represented in the form of mature BALOs, which consisted of SCGB1A1$^+$SFTPC$^+$ cells. Bronchiolospheres presented 11% of organoids and consisted of SCGB1A1$^+$SFTPC$^-$ cells, while alveolospheres accounted for 9% of organoids and grew from SCGB1A1$^-$SFTPC$^+$ cells. This result was validated by separating cells with flow sorting according to their expression of both marker genes and subsequent culturing. Mature BALOs were only obtained from the population with SCGB1A1$^+$SFTPC$^+$ expression pointing out their BASC phenotype, while SCGB1A1$^+$ or SFTPC$^+$ single positive cells did not grow into BALOs but instead developed exclusively into bronchiolospheres or alveolospheres, respectively. Further, BALO formation was analyzed regarding mCherry and YFP expression during development. In the early-stage BALO development, cells positive for both markers showed a uniform distribution with a decreasing number of double-positive cells resulting in a small fraction remaining after 8 days. After 21 days, SCGB1A1 expression concentrated on central branches which are surrounded with alveolar-like structures that show SFTPC expression. Presence of BASCs was further investigated by digesting successfully formed BALOs and culturing the obtained cells together with freshly sorted rMCs. As a result, new BALO formation was observed, although with a lower frequency indicating that cultured BALOs still include cell/progenitor cells with the potential to develop into bronchiospheres and alveolospheres. The possibility to create new BALOs from previous generations was retained even with additional passages. Finally, SCGB1A1$^+$ SFTPC$^+$ cells were labeled using the LacZ gene encoding for β-galactosidase, allowing to visualize presence of these cells throughout BALO development. By detecting LacZ activity using the specific substrate X-gal, presence of SCGB1A1$^+$SFTPC$^+$ BASCs was detected in distal BALOs regions after more than 60 days of organoid culture.

After this first molecular characterization of BALOs and BASCs, a more comprehensive analysis of the organoids was planned with a focus on their cellular composition. In order to achieve a unbiased and high resolution, mature BALOs grown for 21 days were digested and processed using single cell RNA-seq. This was performed using the ddSEQ cell isolator from BioRad in combination with the SureCell protocol from Illumina. As mentioned above in section (3.1.1), the SureCell ddSEQ protocol constitutes a droplet-sequencing approach with high throughput processing of cells. For processing, cells were mixed with lysis buffer, barcoding beads and reverse transcription reagent. This mixture was turned into an emulsion using the ddSEQ isolator resulting in cell lysis and capture of mRNAs to barcoded beads within the droplets. Following, reverse transcription turned captured mRNA into cDNA which was subsequently purified and complemented by second-strand synthesis. In the next step, cDNA was turned into libraries for Illumina sequencing which required fragmentation with a tagment enzyme and following amplification of the fragments. Lastly, the libraries were cleaned up, pooled and sequenced using an Illumina NextSeq 500 system.

Following the sequencing, raw reads of three samples - named 'ddSeq5', 'ddSeq6' and 'ddSeq7' - were uploaded to Illumina's BaseSpace platform and a first analysis was conducted using the SureCell analysis pipeline as implemented within BaseSpace. This workflow includes mostly pre-processing steps: mapping, feature extraction and demultiplexing of cells with subsequent detection of correct barcodes, and a prediction of reads belonging to real cells instead of being ambient background RNA. Also, some downstream analysis steps were carried out as well, providing a first insight into gene expression throughout the data set. However, results mostly focused on providing a simple overview of the data set with a variety of text-based information exhibiting read quality metrics such as number of reads, valid barcodes, mapping and feature rates and number of identified cells. Further, PCA and t-SNE plots were provided to check for basic expression information of a specific gene throughout cells in the data set. However, a variety of crucial information is missing, for example the detection of cellular clusters enabling prediction of cell types and consequently, the detection of marker genes or

differential gene expression. Also, downstream analysis is performed without any further information about used threshold values or quality cutoffs and thus does not allow any customization of parameters to account for data set specific properties. Thus, the SureCell pipeline and the BaseSpace platform in general do not fulfill the principles of FAIR data mentioned in section 1.6. While more information about used software and parameters was provided for pre-processing of the data, a user-based customization of these steps was also not possible. Furthermore, each step of the SureCell pipeline is controlled by Illumina, which means data is for example stored and processed on Amazon Web Services (AWS), which can be an issue for sensitive data and also results in users being charged for each analysis run as BaseSpace is a commercial platform.

To overcome the mentioned issues, a custom analysis had to be performed. For this, the main focus was set on generating a tailored but also reusable analysis pipeline to be compatible with future data sets. The analysis was separated into two major parts - pre-processing as described in section 1.5.1 and downstream analysis as described in section 1.5.2. Generally, pre-processing was expected to take up a significantly higher amount of compute resources and thus ideally needed to be performed only once or a few times, while downstream analysis was expected to be performed multiple times with changing parameters for *e.g.* clustering or expression analysis for a changing list of genes. Naturally, pre-processing of the raw data is a required process generating the necessary files for the following downstream analysis providing more biological insights. Thus, in a first step a pre-processing workflow was established focusing on the following steps:

- Sequencing quality control
- Extraction of valid barcodes and demultiplexing
- Mapping to mm10 *Mus musculus* reference genome
- Feature extraction

- UMI counting
- Removal of false-positive barcodes

As each of the steps requires a different tool, a variety of tools have been evaluated. For all steps applicable, software which was shown to be suitable for single cell purposes was selected. Mapping was performed using STAR, feature extraction was carried out using featureCounts and for UMI counting the software UMI-tools was applied. Another benefit for result comparison to BaseSpace was the usage of STAR and featureCounts in the SureCell workflow in BaseSpace as well. Demultiplexing and removal of erroneous barcodes however required a tailored solution, as previously available solutions mostly supported the 10x barcode and UMI scheme, but not the fragmented scheme used with the ddSEQ protocol. This led to the challenge of identifying the ddSEQ scheme as well as generating an algorithm following the manufacturers description of correcting small errors within the sequence and removal of sequences exceeding the expected error rate. The sequence schemata are already shown and explained in more detail in section 3.1.1 and figure 16. Following a request, Illumina and BioRad provided a technical note (Pub. No. 1070-2016-015-A) which contained a whitelist of correct barcodes as well as a more detailed description on barcode decoding and allowed edit distances for per base error correction of each sequence fragment. To address these limitations, a Python script was developed which reads in sequences from the FASTQ read file, processes each read, corrects errors and filters out sequences which do not match the described schemata. Furthermore, the script also implements demultiplexing based on the detected barcode, resulting in a separate FASTQ file for each barcode comprising all associated sequences. Thus, a FASTQ file was generated for each proposed cell, which could then be used for further processing.

However, this approach revealed a major disadvantage regarding the processing time resulting in two to three days computation time for a full pre-processing approach of the experimental data set comprising around 233 million reads. Especially for testing and development of a pre-processing pipeline, this led to unnecessary long waiting times. Additionally, KFO309 researchers would have to wait days until further biolog-

ical analysis could be performed. Furthermore, a larger data set or a scaled up experiment implies an even longer waiting time for users. And finally, this approach still required a large amount of compute resources to fulfill the analysis within the mentioned time, as each barcode needed to be processed individually resulting in hundreds of thousands of analysis runs performing mapping, feature extraction and UMI counting. In order to reduce the computational burden and improve analysis time, the current approach was evaluated for possible bottlenecks. A major issue in this regard was the separation of reads based on the barcodes as a first step, resulting in hundreds of thousands of FASTQ files. As a result, each following analysis step needed to be carried out hundreds of thousands of times as well, causing a huge number of input and output operations which had to be managed by the CPU. This results in a great overhead in input and output operations leading to an extended processing time. This became clear when the previous workflow was compared to a direct mapping and feature extraction of the non-separated FASTQ file from the data set. When mapping and feature extraction were performed on the non-separated FASTQ file, the whole processing time was reduced to about four hours. In case of mapping, each performed step requires generating a genome index for *Mus musculus* which is a time-intensive and also memory-intensive process taking up about 30 GB of RAM, limiting the number of parallel processes. Feature extraction, on the other hand, is less memory-intensive, but running featureCounts with the non-separated input file takes less than 30 minutes, which indicates a massive input/output bound delay when processing hundreds of thousands of files.

In addition, extraction, correction or removal of barcodes has been separated from the demultiplexing process itself to enable a parallel and thus faster processing. For this, the BAM file obtained after mapping and feature extraction was split into chunks of 1,000,000 reads which were then validated using a Python script. Based on the description from Illumina and BioRad, each read was checked for a valid barcode and UMI scheme, base errors within the edit distance limitations were corrected and verified using the official whitelist. Reads exhibiting verified barcodes were then written

to a new BAM file with barcode and UMI sequence added to the QNAME entry using underscores for separation and finally all generated chunk BAM files merged together. The newly merged file was then processed with UMI-tools to generate UMI counts per barcode and gene. Due to the merging, UMI counting did also prevent a higher input/output bound delay which would appear when being run for each detected barcode individually. In the next step, the counted UMIs were demultiplexed, resulting in a TSV file for each barcode. The final step was then to generate a gene expression matrix containing only barcodes which are likely associated with a real cell. For this, a custom R script was generated which plotted UMIs for each data set in decreasing number of UMI per barcode as so-called knee plot (section 1.5.2, figure 12) and calculated the first inflection point of the curve as proposed cutoff value. Therefore, the improved order of analysis steps is as follows:

- Sequencing quality control
- Mapping to mm10 *Mus musculus* reference genome
- Feature extraction
- Extraction of valid barcodes
- UMI counting
- Demultiplexing
- Removal of false-positive barcodes

During the experimental analysis, a variety of steps were carried out separately, especially scripts that were under development. Following the successful processing of the single cell data, all steps except false-positive barcode removal have been transformed into a Snakemake script to ensure an easy, scalable and reproducible analysis. Following the experimental BALO analysis, this Snakemake script was then implemented into the WASP software to perform the pre-processing of single cell data.

Table 2: **Read metrics of ddSEQ-based single cell sequencing of mouse lung organoids.** The data set consists of three samples 'ddSeq5', 'ddSeq6', 'ddSeq7' which are all results of single cell RNA sequencing of mouse BALOs. BS = BaseSpace, Kp = Knee plot

Sample	Reads total	Reads valid	Barcodes	Predicted cells (BS)	Predicted cells (Kp)
ddSeq5	172,518,245	125,868,400	343,404	321	346
ddSeq6	60,596,840	46,333,757	244,494	424	399
ddSeq7	49,832,257	36,263,149	227,730	228	191

The last step of barcode validation was further developed for WASP as an interactive R Shiny web application and complemented with various visualizations of other read-based quality metrics to provide an easy and direct overview about the data set quality.

Results of the pre-processed BALO data set are shown in table 2. Although data sets were based on a similar input of mouse lung organoids, a difference in data set size was observed with ddSeq5 containing the largest amount of reads, ddSeq6 as second largest sample including slightly less than two thirds of the reads of ddSeq5 while ddSeq7 is the smallest data set containing less than one third of the reads of ddSeq5. Following the removal of reads without a valid barcode, 72.96% reads of ddSeq5, 76.46% reads of ddSeq6, and 72.22% reads of ddSeq7 remained. While ddSeq5 and ddSeq7 are separated by more than 100,000 detected barcodes, the high numbers still underlined the need for removal of ambient RNA-based barcodes. Due to information provided by the manufacturer and previous experiences, we expected up to 400 cells in the best case. As table 2 shows, ddSeq5 and ddSeq6 were close to the expected value, while ddSeq7 reached only about half of the 400 cells. Also, a difference in cell calling was observed when comparing the custom knee plot approach with the BaseSpace results. While cell calling of ddSeq5 resulted in 25 cells more in the custom analysis compared to BaseSpace, ddSeq6 resulted in 25 cells less and ddSeq7 in 37 cells less compared to the BaseSpace prediction, respectively. For each data set, cells predicted using the knee plot approach were then combined into a gene expression matrix, which was then used for the following downstream analysis.

Similar to the pre-processing, a variety of tools were evaluated, although the downstream analysis often requires software more tailored to single cell application compared to multi-purpose tools for pre-processing steps such as mapping. An overview from 2018 about single cell analysis tools found that the most commonly used platform for these tools is the statistical programming language R, used for around 60% of available software [198]. These packages were typically available via the repositories Bioconductor and CRAN. The second largest platform used is the programming language Python, used for around 25% of available software [198]. This was also reflected in the fact that two of the most popular and comprehensive single cell tools - Seurat and Scanpy - are R or Python-based. However, the wide range of R-based tools enables an easy extension of an analysis workflow as stored and processed data can easily be exchanged between different packages due to compatible data classes. The compatibility was a great benefit during the workflow development, as this required frequent changing of analysis tools and methods. Furthermore, extension of an established pipeline can also be performed with less complexity. Finally, the Shiny package provided the opportunity for a comprehensive integration of an R-based analysis workflow into a web interface with GUI. Therefore, the decision was made to focus on an R-based workflow and the analysis was performed mostly using the Seurat R package.

As a first step, quality control was performed which led to removal of genes expressed in less than three cells and cells expressing less than 200 genes. In the next step, UMI values for each cell were normalized by dividing per gene counts for each cell by the cell's total counts, multiplying the result with a scale factor of 10,000, subsequently adding a count of 1 and performing natural-log transformation (log1p). Following normalization, highly variable genes were identified in the data set by calculating average expression and dispersion for each gene, placing genes into bins and calculating a z-score for dispersion within each bin. The data set was then scaled and centered by regressing values based on the number of UMIs per cell. Based on previously identified highly variable genes, a PCA was performed for dimensionality reduction and significant PCs selected based on an elbow plot visualization. The resulting data was

then used to perform clustering of cells with Seurat's Louvain algorithm combining graph construction based on euclidean distances in PCA space and modularity optimization techniques. Calculated clusters were then visualized using t-SNE, up- and downregulated genes or marker genes with differential expression were identified for each cluster and visualized in form of violin plots and heat maps (Fig. 32).

The clustering resulted in a detection of four distinct clusters referred as C1, C2, C3 and C4 (Fig. 32A). Evaluation of the marker genes that were upregulated in each cluster enabled a biological function assignment of the cells. C1 and C2 represented lung epithelial subpopulations expressing airway and alveoli-associated genes whereas C3 and C4 represented lung mesenchymal subpopulations expressing fibroblast-associated genes. C1 was further characterized as an airway cell cluster with expression of cellular markers for ciliated cells (Itgb4), basal cells (Trp63 and Krt7) and respiratory epithelial cells (Sox2) (Fig. 32C) [199], [200], [201]. C2 was further characterized as an alveolar cell cluster with expression of AEC type II markers (Cxcl15, Lyz and Sftpc) and AEC type I markers (Hopx) (Fig. 32D) [199], [200], [201]. C3 was further characterized as a myofibroblast cell cluster with expression of according markers such as PDGFRA, Tagln, Acta, Eln and Axin2 (Fig. 32E) [202], [203], [204]. C4 was further characterized as a lipofibroblast cell cluster with expression of according markers such as Fgf10, Apoe, Serpina3n, Gsn and Gas6 (Fig. 32F) [202], [203], [204].

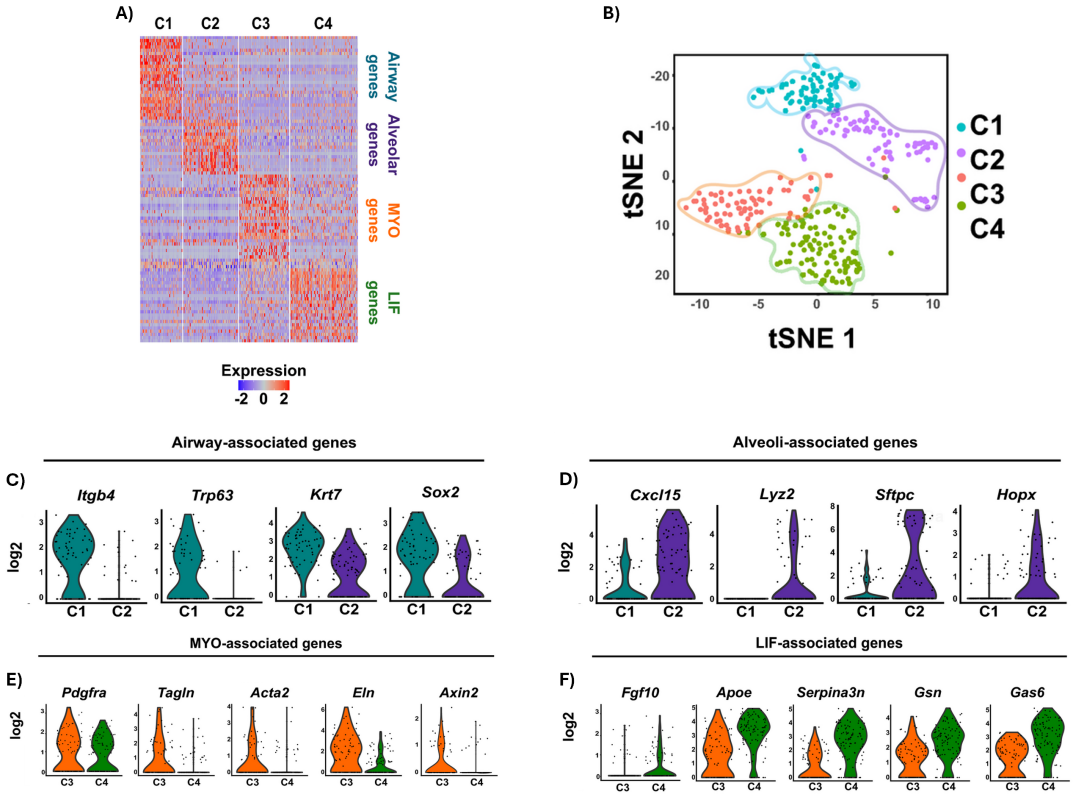


Figure 31: **Downstream analysis results of BALO single cell data set sample ddSeq5.** A) Heat map providing an overview of the detected clusters C1, C2, C3 and C4 with log2fold change expression values of cell type specific marker genes. B) t-SNE plot with the four distinct clusters. C) Violin plots with log2fold change values of airway-associated genes. D) Violin plots with log2fold change values of alveolar-associated genes. E) Violin plots with log2fold change values of myofibroblast-associated genes. F) Violin plots with log2fold change values of lipofibroblast-associated genes. All figures were published in Vazquez-Armendariz *et al.* 2020 [195]

Apart from the *in silico* characterization, BALOs were also analyzed using FACS and electron microscopy. In line with the computational analysis, FACS sorting of day 0 and day 21 BALO cultures revealed an Epcam^{+} epithelial and an Epcam^{-} mesenchymal fraction.

The Epcam^{+} epithelial fraction exhibited expression of typical markers of differentiated airway and alveolar epithelial cells and consisted of small airway bronchial epithelial cells as well as type I and type II AECs. Quantitative polymerase chain re-

action (qPCR) showed an upregulation of adult lung differentiated cell type markers over the BALO development time including Hopx and Sftpc for AEC type I and II, respectively and further Foxj1, Muc5ac and p63 for ciliated, goblet and basal cells, respectively, confirming the computational analysis results. Electron microscopy further showed that a single layer of epithelial cells lined the BALO's alveoli. The cells also exhibited various characteristics of mature type II AECs such as interconnection via tight junctions, lamellar bodies and abundant mitochondria. BALO regions resembling bronchioalveolar duct junctions contained bronchial and intermediate cell types leading into alveolar-like regions that contain AEC type I cells. Regions resembling airway tubes, on the other hand, contained different phenotypes, ranging from undifferentiated epithelial cells to ciliated airway cells that were also interconnected by junctional complexes at their apical surfaces. Using LysoTracker and LipidTOX, which accumulates in lamellar bodies and phospholipids, respectively, surfactant production within the BALO was confirmed with lamellar bodies and phospholipids appearing in the alveolar-like regions as expected. Surfactant production has additionally been verified using Western Blot analysis. A more detailed analysis of the distal alveoli structures revealed a decrease in the mean diameter by 23% between days 15 and 21 and a further decrease by 13% between days 21 and 30. On the other hand, the mean number of alveolar-like regions in the BALO showed a threefold increase between days 15 and 30. These results both resemble a previous analysis of murine lung alveoli development [205]. After 40 days of development, BALOs presented an even further differentiated state with alveolar-like structures filled with lamellar surfactant including thin, elongated type I AEC that integrate cuboidal type II AEC joined through tight junctions. The airway-like structures on the other hand, formed a pseudostratified epithelium which includes basal cells, differentiated secretory cells containing secretory granules and ciliated cells including mature cilia and basal bodies.

After the epithelial compartment has been characterized, the mesenchymal compartment comprising the $Epcam^-$ cell fraction was analyzed as well. As previously mentioned, $Epcam^-Sca\text{-}1^+$ rMC are important for BALO formation. Furthermore, a previous

analysis identified rMC as a heterogenous population which included progenitors of myofibroblast and lipofibroblast cells. Microscopic analysis of BALO cultures identified at least two distinct fibroblast cell types, with one population containing lipofibroblast characteristic lipid bodies, identified using LipidTOX, and the other population contaning myofibroblast characteristic spindle-shaped cells that were alpha-smooth muscle actin-positive and exhibited elongated cellular extensions. Furthermore, differential expression of PDGFRA and alpha-smooth muscle actin was previously described as characteristics of myofibroblast and lipofibroblast cells [202]. These morphological findings were in line with myofibroblast (C3) and lipofibroblast (C4) cell clusters identified in the computational analysis of the BALO single cell data. Genes identified upregulated within these clusters such as Fgf10, PDGFRA, Tagln, Acta2 and Eln were previously associatead with myofibroblast and lipofibroblast phenotypes. Furthermore, expression lipofibroblast-associated genes such as Apoe, Serpina3n, Gsn and Gas6 and the myofibroblast-associated gene Axin2 was identified. Also, the observed higher expression of PDGFRA in myofibroblast compared to lipofibroblast matched previous findings of BALO-based mesenchymal cell subsets [202]. Similar to the epithelial BALO cell development, the mesenchymal compartment showed a structure matching murine lung development. Using LipidTOX, lipid-droplet containing lipofibroblast cells were detected to be located around developing BALOs which matches their role of promoting epithelial growth and AEC type II differentiation as well as their location in close proximity to alveolar epithelium in murine lung development [206]. Myofibroblasts were observed to be located at branching sites in the BALO center, which is in line with their role of generating an extracellular matrix in the neonatal lung to provide a scaffold for alveolar development [207].

In general, the observations confirmed the presence of myofibroblast and lipofibroblast subsets with different levels of PDGFRA expression and spatial localization around alveolar-like structures. Together with the identified phenotype of epithelial BALO compartments, the definitions for organoids were matched [208]. This included composition of multiple organ-specific cell types, exhibition of organ-specific features, *e.g.*

secretion of pulmonary surfactant by AEC type II-like cells and finally a spatially restricted cellular organization leading to airway- and alveolar-like compartments including a proximo-distal distribution.

Even though the previous analyses gave proof that BALOs comprise epithelial and mesenchymal lung cell types and resembles lung structures, immune cells - tissue-resident yolk sac-derived alveolar macrophages (TR-Mac) - were still missing. However, these are crucial to study cellular processes in lung development, homeostasis, disease and following tissue regeneration [209]. In order to overcome this limitation, TR-Mac were isolated from murine bronchioalveolar lung fluid samples collected from td-Tomato-expressing adult mice. Collected cells were checked for surface antigen signature using FACS and were subsequently microinjected into central regions of 14 day old BALOs. Analysis 10 days post injection detected more than 80% of inserted TR-Mac with a viability of 87%. Further, the TR-Mac phenotype was detected 14 days post injection in the alveolar niche by positive staining of alveolar macrophage surface markers CD206 and Siglec-F and microscopy analysis 28 days post injection still showed a successful engraftment of TR-Mac into the alveolar-like regions. In addition to the engraftment, a direct interaction between TR-Mac and AEC based direct filopodia contact was observed using electron microscopy. Furthermore, staining of TR-Mac enriched BALOs revealed expression of the tight junction molecule connexin Cx43 within alveolar-like regions which is in line with a previous study showing the role of Cx43 in TR-Mac AEC interaction [210]. This result was further underlined by staining TR-Mac mono-cultures without BALO which showed no signs of Cx43 expression.

In order to evaluate a possible effect of TR-Mac addition on composition and differentiation of epithelial BALO cells, single cell RNA-seq of 23 days old BALOs 9 days post infection was performed. For this, a total of six BALO samples were generated with three samples representing TR-Mac enriched BALO cultures 'CM1', 'CM2', 'CM3' and three samples representing BALO culture without macrophage enhancement C1, C2, C3, respectively. The samples were processed in a similar way as the previous BALO samples using the droplet-based ddSEQ single cell isolator system followed by

Table 3: **Read metrics of ddSEQ-based single cell sequencing of mouse lung organoids with and without TR-Mac enrichment.** The data set consists of six total samples with three samples 'C1', 'C2' and 'C3' representing BALO cultures without added TR-Mac and three samples 'CM1', 'CM2' and 'CM3' representing TR-Mac enriched BALO cultures. Kp = Knee plot.

Sample	Reads total	Reads valid	Barcodes	Predicted cells (Kp)
C1	45,058,963	36,399,437	275,807	192
C2	51,008,862	35,245,120	303,402	143
C3	70,465,453	45,118,236	312,667	100
CM1	60,209,430	37,250,287	213,393	117
CM2	39,059,038	26,842,413	222,094	135
CM3	56,048,146	31,567,714	238,934	194

sequencing with the Illumina NextSeq 500. Also pre-processing of the data was similar as described above with results shown in table 3.

Due to the low yield and in order to better compare the two conditions, samples were combined for TR-Mac enriched BALO and BALO mono culture. Similar to the first data set, downstream analysis was mostly based on using the Seurat R package. However, the analysis had to be modified and extended in order to integrate both conditions into the analysis. This was important as compared to bulk sequencing, the data needed to be clustered based on similar subpopulations first to enable detection of gene expression within the same cell type between conditions. This required a different approach, as typical bulk batch correction methods assume uniform effects on all cells based on confounding variables in different conditions. Another issue could have been differences in cellular density across conditions. Therefore, a so-called CCA implemented in Seurat was used to integrate both conditions into one data set. Essentially CCA aims to identify linear combinations across different data sets such as multiple covariance matrices that provide the highest correlation, thus detecting shared correlation structures between data sets.

Table 4: Identified marker genes and according cell types of BALO data sets with and without TR-Mac enrichment

Cluster	Marker genes	Assigned cell type
C1	Scgb3a2, Muc5b, Bpifa1	Club/Secretory airway
C2	Krt5, Krt14, Aqp3, Trp63	Basal airway
C3	Col1a2, Igfbp4, Apoe	rMC
C4	Sftpc, Cxcl15, Sftpb	AEC type II
C5	Hopx, Ager, Cldn18	AEC type I
C6	Foxj1, Tppp3, Lrrc23	Ciliatead airway

In this case, each data set is treated as measurement of a gene to gene covariance structure with the aim to identify patterns common across the data sets.

For this, genes with high variation in at least one data set are selected and used to identify basis vectors to project cells from each data set into a low-dimensional space with the aim to achieve a maximized correlation of variation among the vectors between data sets. These basis vectors can be seen as sort of metagene, representing a weighted expression average based on the expression of the top genes that showed a robust correlation with the basis vector. Each pair of vectors is then linearly transformed to account for global shifts in feature scale, and subsequently non-linearly transformed to account for shifts in subpopulation density. As a result, a single and aligned low dimensional space is defined which represents all data sets and can be used for further downstream analysis such as clustering. For this, canonical marker genes - marker genes conserved across conditions - are identified based on differential gene expression testing for each dataset and combination of the p-values.

Analysis of the data set resulted in a total of six cellular clusters named 'C1' to 'C6. Based on identified marker genes, clusters were assigned to be club/secretory cells (C1), rMC (C2), basal airway cells (C3), AEC type II (C4), AEC type I and ciliated airway cells (C6) (Table 4, Fig. 32A and B). Following the identification of genes conserved across the conditions, the was further analyzed for genes showing a differential

expression between conditions to identify possible effects of TR-Mac addition to the BALOs. This revealed that engraftment of TR-Mac correlated with downregulation of genes related to cell proliferation such as Fos, Fosb, Areg and Klf4 (Fig. 32 C) as well as genes associated with inflammatory processes and cellular stress such as Erg1 and Atf3 (Fig. 32C). On the other hand, enriched BALOs cultures showed an upregulation of genes associated with cell differentiation such as Neat1 (Fig. 32C) and genes related to club/secretory cell maturation such as Cyp2f2 and Ces1d (Fig. 32 C). This result was in line with the observed significantly higher percentage of terminally differentiated epithelial cells, including type I AEC and ciliated airway cells. Thus, presence of TR-Mac in BALO seemed to increase epithelial differentiation and simultaneously decrease cell proliferation and stress signaling, which ultimately resulted in an accelerated BALO maturation exhibiting functions relevant for lung homeostasis.

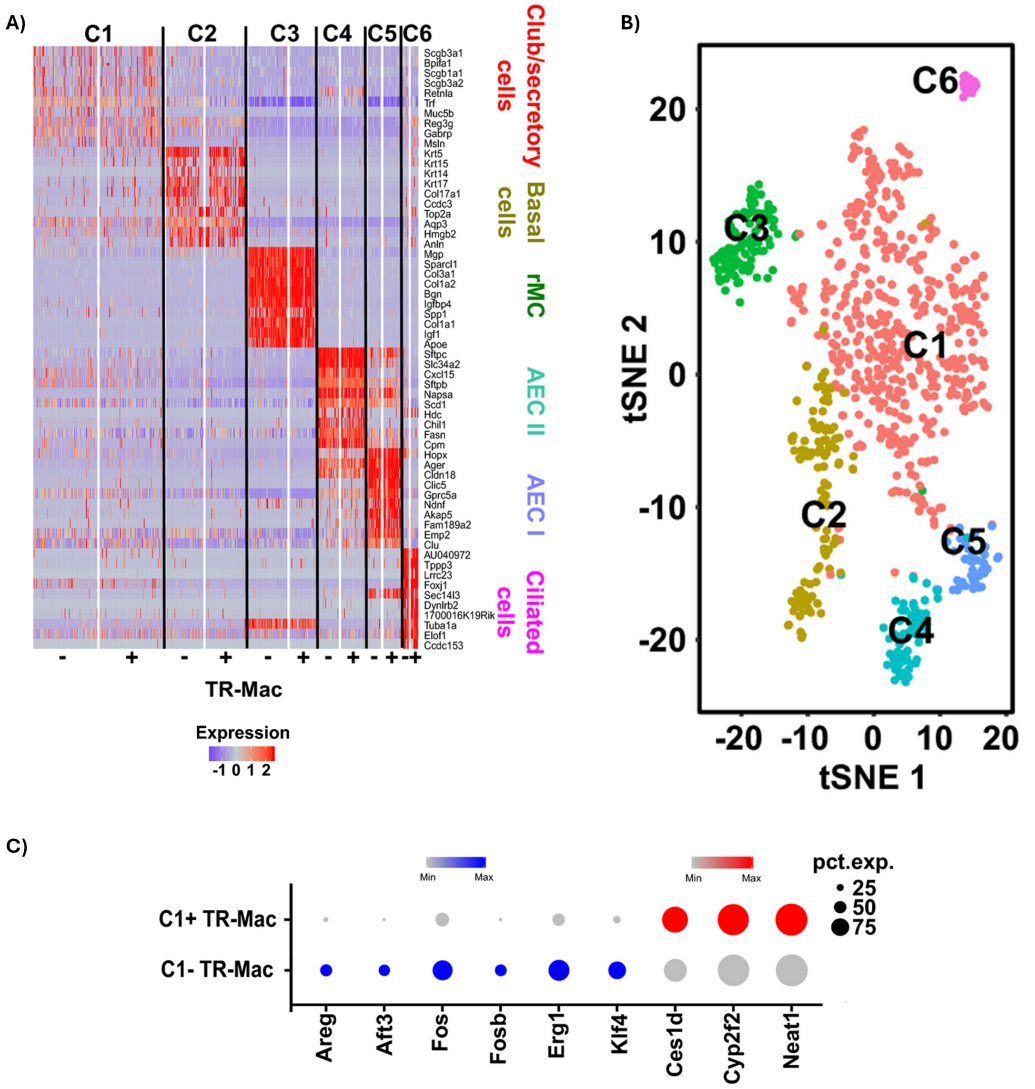


Figure 32: **Downstream analysis results of TR-Mac enriched and non-enriched BALO single cell data sets.** Enriched samples CM1, CM2 and CM3 were combined into on sample as well as C1, C2 and C3. Both conditions were integrated using Seurat's CCA approach. A) Heat map providing an overview of the detected clusters, conserved marker genes and assigned cell clusters. Combined TR-Mac enriched BALO sample marked as '+' and combined non-enriched sample marked as '-'. B) t-SNE plot with the six distinct clusters. C) Dot plot of genes differentially expressed in Cluster 'C1' between conditions TR-Mac enriched '+' and non-enriched '-'. Dot size shows the percentage of cells within the cluster expressing the gene, expression level is indicated by color. All figures were published in Vazquez-Armendariz *et al.* 2020 [195]

Following the detailed and *in silico* single cell-aided characterization of the cellular and structural composition, BALOs were further analyzed in two additional biological experiments. The first experiment was aimed at demonstrating the usability of BALO for genetic manipulation experiments. Therefore, the knockdown of miRNA 142-3p was targeted, which controls WNT-dependent mesenchymal progenitor cell proliferation during murine lung development [211]. Knockdown was performed by repetitively applying a miRNA 142-3p-specific morpholino antisense oligonucleotide (mo142-3p) to six day old BALO cultures for five days. Using qPCR, knockdown of miRNA 142-3p by mo142-3p in $Epcam^+$ epithelial and $Epcam^-Sca1^+$ mesenchymal cells was confirmed. While cell viability was not affected compared to scrambled morpholino control samples, organoid growth showed a significant decrease in knockdown samples without effect on colony-forming. Furthermore, knockdown samples revealed an upregulated gene expression of Apc which is known to be regulated by miRNA 142-3p. Using β-galactosidase-sensitive reporter BASCs and rMC, WNT signaling was observed in both mesenchymal and epithelial lung cells during BALO growth. Ultimately, mo142-3p-treated organoids were observed to be significantly smaller, showed impaired secondary branching and suffered a significant reduction of type II AEC and club cell numbers. These results were in line with previously observed *in vivo* loss-of-function experiments targeting miRNA 142-3p [211], thus demonstrating a usability of BALO as a possible platform to study effects of genetic manipulation on developmental pathways and their role in morphogenesis and regeneration following injury.

The second experiment aimed at demonstrating the usability of BALO for disease modeling. For this, H1N1 and H7H7 IAV reporter viruses [212] were injected into BALO central airway-like structures to model the typical *in vivo* proximal-to-distal epithelial infection. This resulted in a viral infection spreading towards the distal alveolar-like regions observed within 12 hours post infection (pi) as well as release of infectious virions detected 48 hours pi by plaque assay. Infection was further confirmed in approximately 8% of $Epcam^+$ BALO epithelial cells by detection of H1N1 viral nucleoprotein expression using qPCR. Combination of reporter virus strains and reporter mice

cells-derived BALOs enabled live cell imaging which revealed viral infection spreading to adjacent cells between 10 and 26 hours pi and a significant cell death in alveolar-like regions 25 hours pi following their infected 14 hours pi. Finally, host response was evaluated using qPCR showing a significant upregulation of interferon-beta expression in IAV-infected epithelial BALO cells. This was further evaluted by TR-Mac enrichment of BALOs 48 hours pi modelling *in vivo* macrophage-epithelial interaction. As a result, an increased release of pro-inflammatory cytokines such as TNF-α, IL-6 and IL-1β compared to non-enriched BALOs was observed. In summary, these results demonstrated successful IAV infection, spread and antiviral response of BALO.

4.1.2 Comparison of bioinformatic analysis of *Mus musculus* bronchioalveolar lung organoids and WASP

Analysis of BALO single cell data required a variety of different processing steps, separated generally into pre-processing and downstream analysis. In the beginning, both analysis parts were separated into multiple steps as tools needed to be evaluated first, data needed to be transformed to be compatible with different tools or analysis steps, and required resources changed. Also mentioned before, this resulted in a number of optimization iterations such as moving the demultiplexing step to the end to speed up the analysis. Following the establishment of a tailored workflow for the data set, the focus was placed on simplified execution and easy reproducibility. Therefore, the pre-processing was turned into a Snakemake workflow to reduce the complexity of running data analysis and resource allocation. Downstream analysis was turned into an R script combining a variety of packages and analysis steps. Furthermore, the workflow was generalized for compatibility with different platforms, *e.g.* by adding support for different barcode and UMI schemata from different manufacturers. Finally, the workflow was combined with R Shiny as the software WASP to provide a low entrance barrier and to enable single cell analysis for researchers without bioinformatic knowledge. Thus, the analysis of BALO simultaneously led to the development of WASP, described extensively in chapter 3.1.

The analysis results generated with WASP were then compared with the published and reviewed BALO data [195] described in chapter 4.1. For this, data from 'ddSeq5' was exemplary processed with WASP's pre-processing Snakemake workflow, barcodes selected and downstream analysis performed using WASP's Shiny web application. The analysis was performed using the automatic mode of WASP in which the software autonomously selects cutoff and parameter values. While providing a variety of additional visualization possibilities (Chapter 3.1.3), for comparison similar plots as in chapter 4.1 were generated with WASP. Similar to the manual BALO analysis before (Fig. 33A, C), WASP also detected four distinct clusters (Fig. 33B, D). Analysis of marker genes and cell type specific genes revealed the same cellular identities with expression patterns matching airway epithelial cells (Cluster 1 WASP, C1 Vazquez-Armendariz *et al.* 2020 [195]), alveolar epithelial cells (Cluster 3 WASP, C2 Vazquez-Armendariz *et al.* 2020 [195]), myofibroblasts (Cluster 2 WASP, C3 Vazquez-Armendariz *et al.* 2020 [195]) and lipofibroblasts (Cluster 4 WASP, C4 Vazquez-Armendariz *et al.* 2020 [195]) (Fig. 33).

In addition to the summarized results, each of the four detected clusters was evaluated with cell type-associated markers used in the previous analysis of 'ddSeq5'. Results for all four assigned cellular identities: airway (Fig. 34), alveolar (Fig. 35), lipofibroblast (Fig. 36) and myofibroblaste (Fig. 37) showed a similar pattern in cluster-specific gene expression including similar log2fold change values. Expression of the lipofibroblast-associated gene Apoe appeared in WASP also in cluster 3, which seemed noticeable compared to the analysis in Vazquez-Armendariz *et al.* 2020 [195]. However, the manual analysis results were limited to the two fibroblast-like cellular clusters for sake of an easier interpretation, while the WASP analysis showed expression across all clusters. Furthermore, expression of Apoe is generally known to also appear in type I and type II AECs [213]. In summary, the WASP analysis is in line with results experimentally previously generated and validated in Vazquez-Armendariz *et al.* 2020 [195], confirming applicability of WASP as an analysis platform for single cell RNA-seq data.

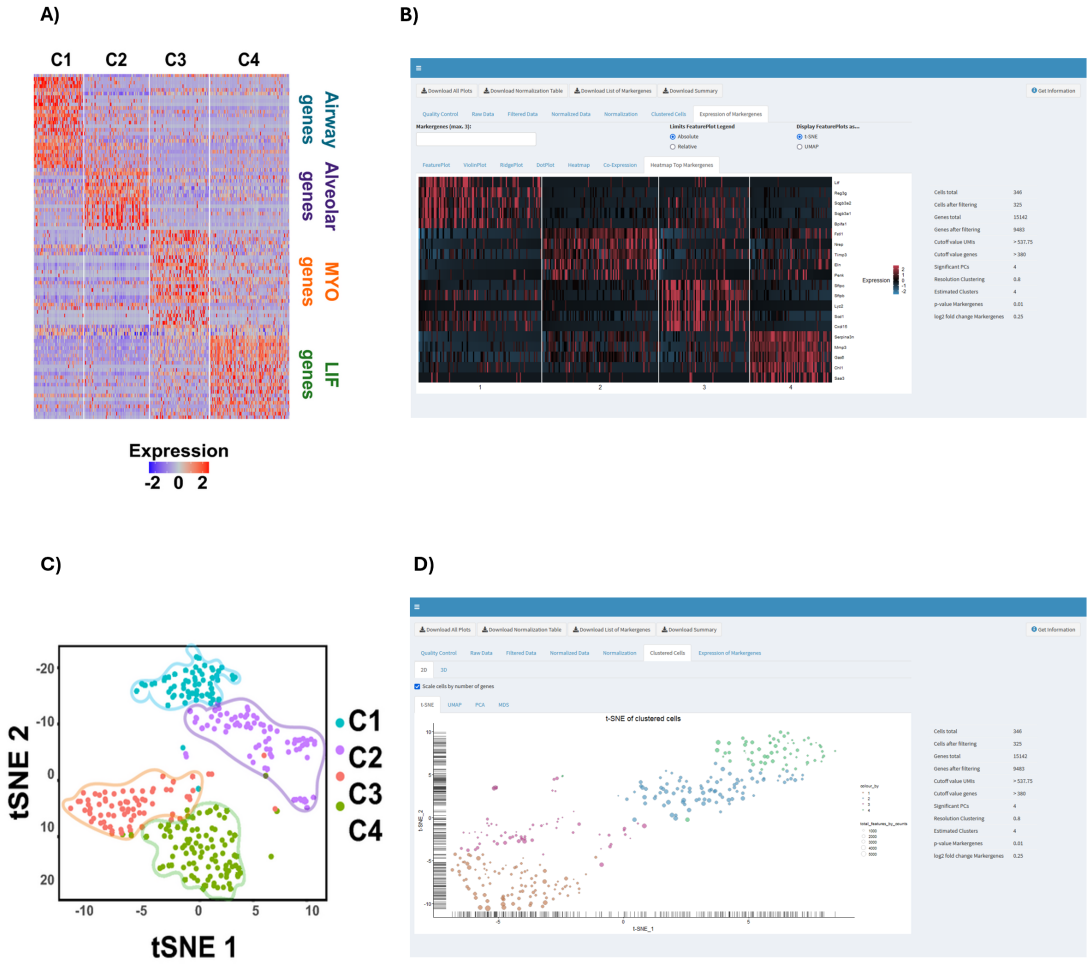


Figure 33: **Comparison of manual and WASP analysis of single cell BALO data set 'ddSeq5'.** A) Manual analysis heat map providing an overview of the detected clusters C1, C2, C3 and C4 with log2fold change expression values of cell type specific marker genes. Selected parameters and cutoff values are shown on the right side. B) WASP-generated heat map providing an overview of the detected clusters 1, 2, 3 and 4 with log2fold change expression values of cluster-specific upregulated genes. C) Manual analysis t-SNE plot with four distinct clusters. D) WASP-generated t-SNE plot with four distinct clusters. Selected parameters and cutoff values are shown on the right side. A) and C) were published in Vazquez-Armendariz *et al.* 2020 [195]

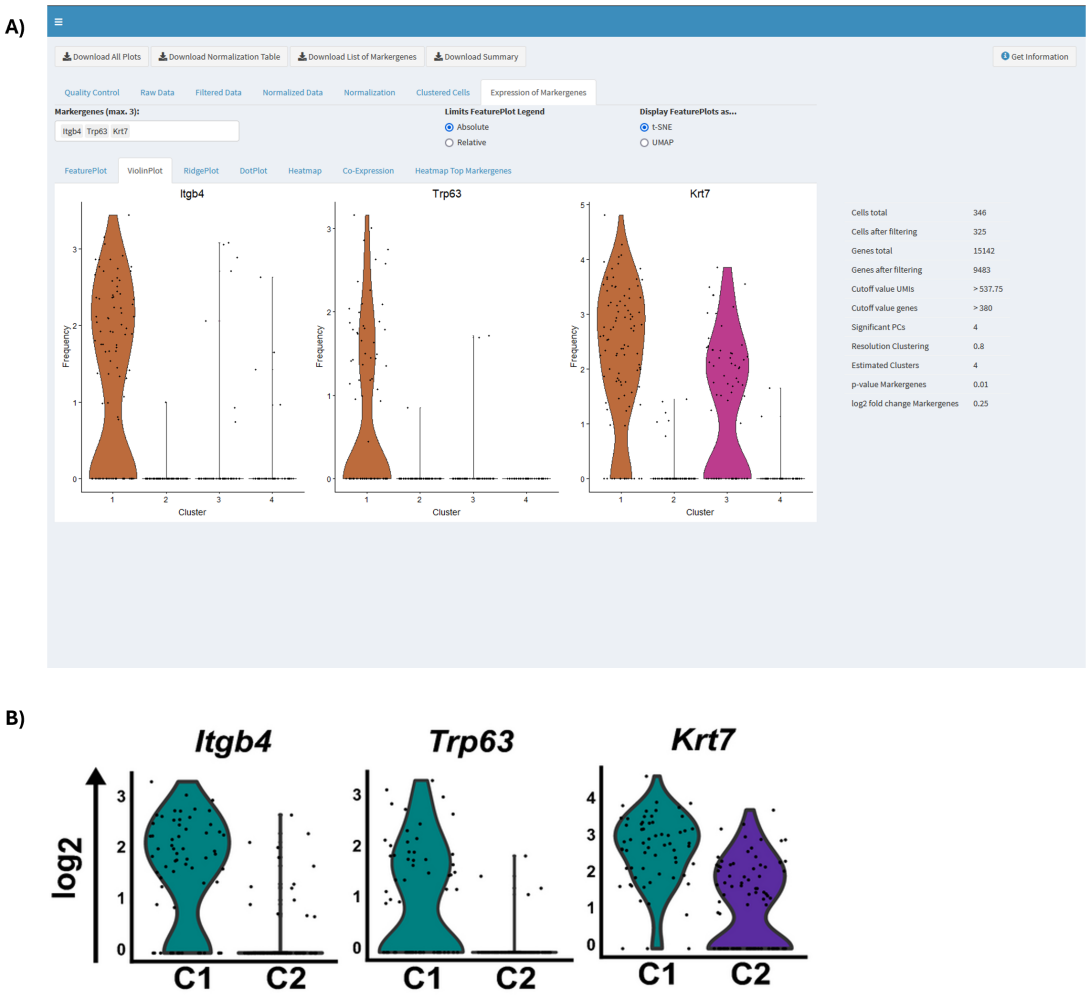


Figure 34: **Comparison of manual and WASP analysis of airway-like cluster from single cell BALO data set 'ddSeq5'.** A) WASP-generated violin plots with log2fold change values showing expression of airway-associated genes. Selected parameters and cutoff values are shown on the right side. B) Manual analysis violin plots with log2fold change values showing expression of the same airway-associated genes. B) was published in Vazquez-Armendariz *et al.* 2020 [195]

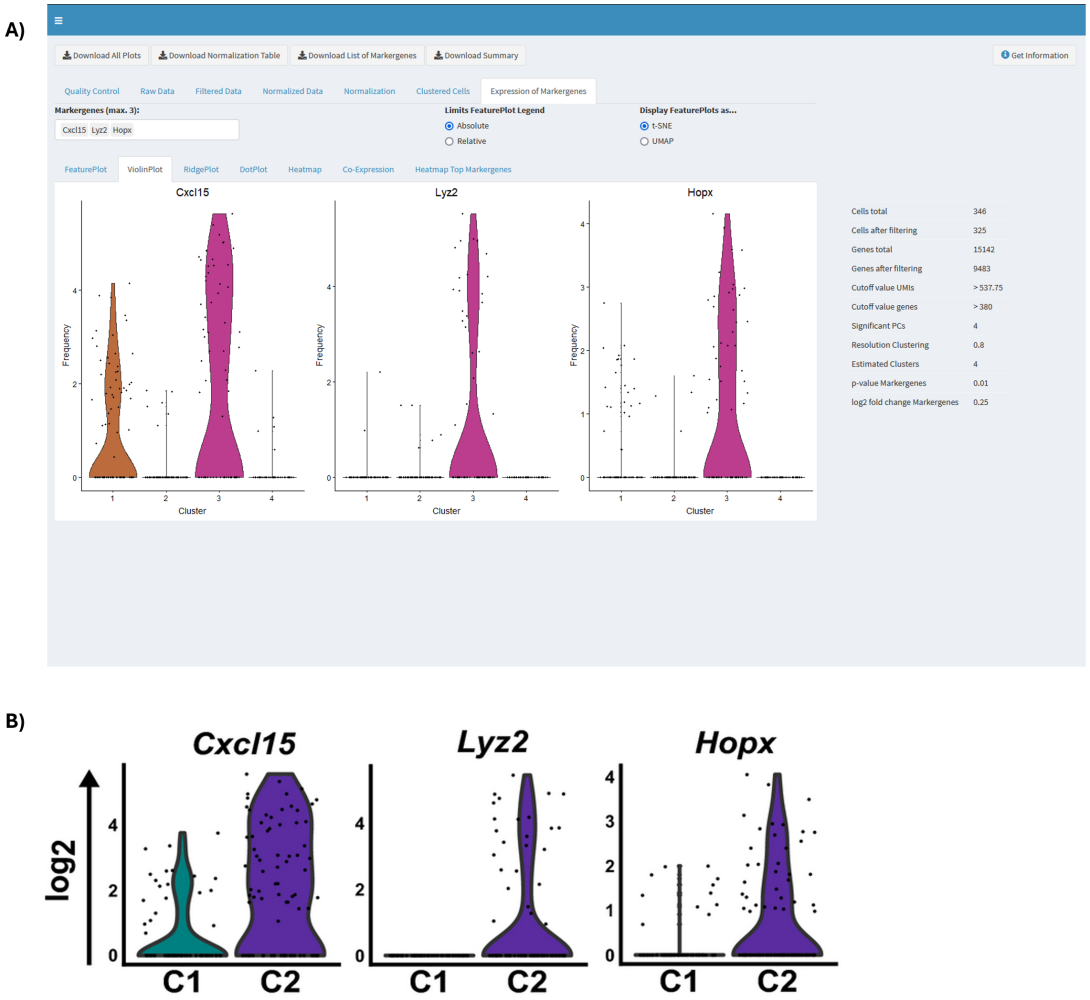


Figure 35: **Comparison of manual and WASP analysis of alveolar-like cluster from single cell BALO data set 'ddSeq5'.** A) WASP-generated violin plots with log2fold change values showing expression of alveolar-associated genes. Selected parameters and cutoff values are shown on the right side. B) Manual analysis violin plots with log2fold change values showing expression of the same alveolar-associated genes. B) was published in Vazquez-Armendariz *et al.* 2020 [195]

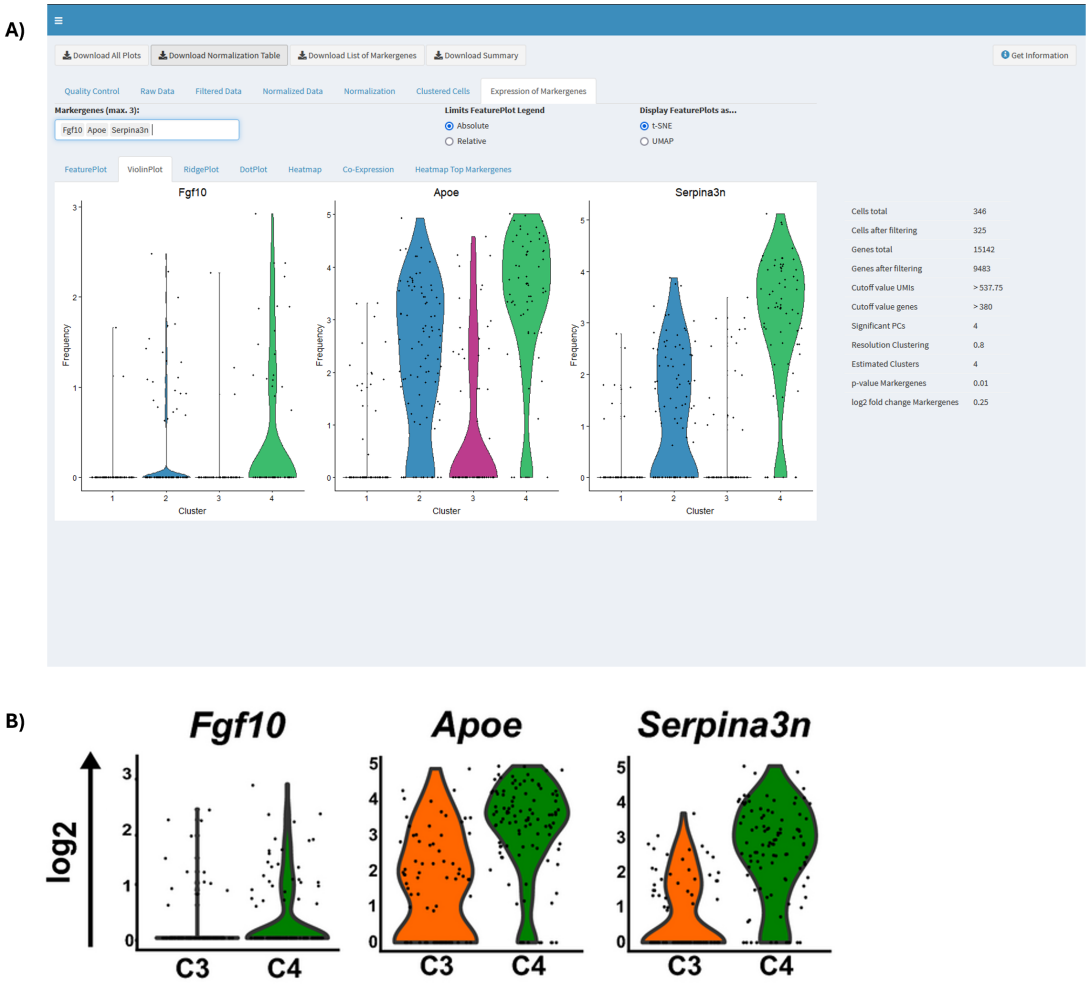


Figure 36: **Comparison of manual and WASP analysis of lipofibroblast-like cluster from single cell BALO data set 'ddSeq5'.** A) WASP-generated violin plots with log2fold change values showing expression of lipofibroblast-associated genes. B) Manual analysis violin plots with log2fold change values showing expression of the same lipofibroblast-associated genes. Selected parameters and cutoff values are shown on the right side. B) was published in Vazquez-Armendariz *et al.* 2020 [195]

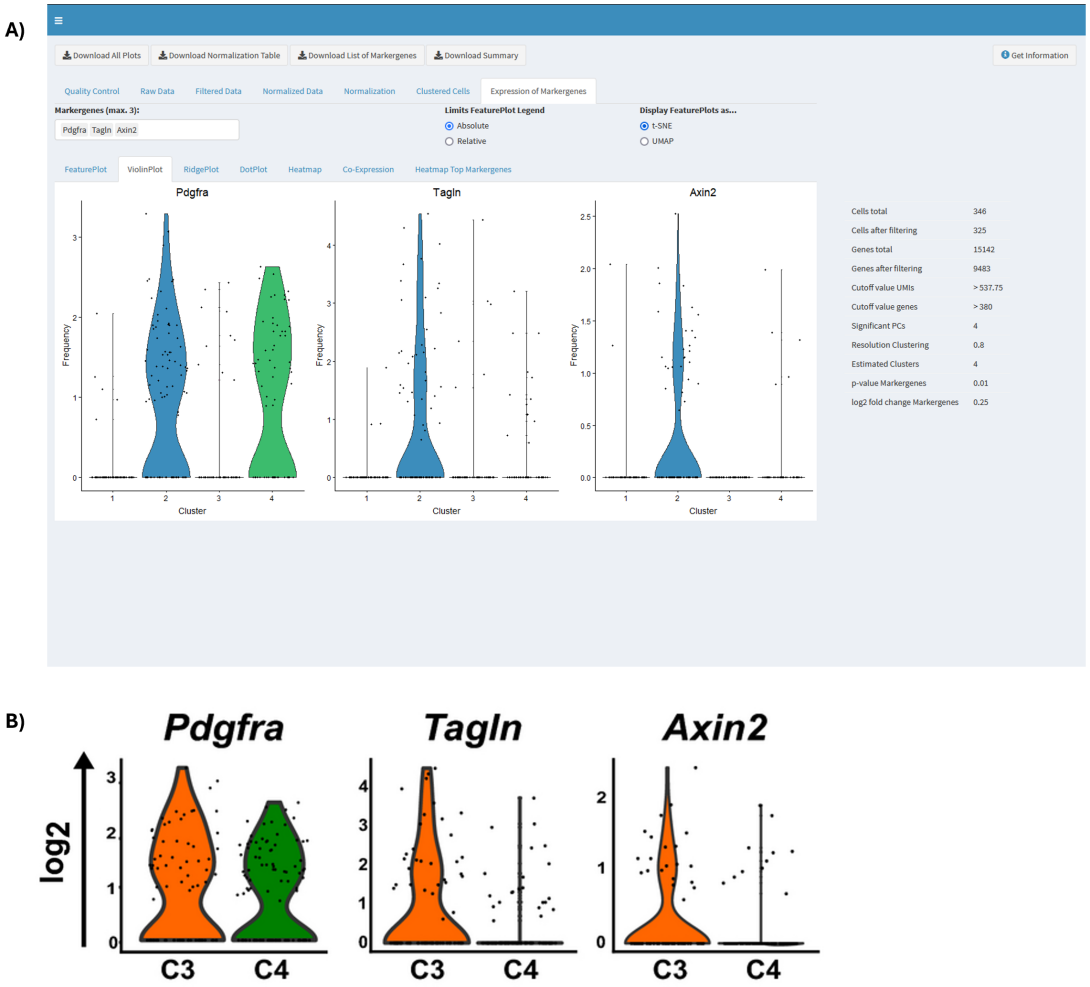


Figure 37: **Comparison of manual and WASP analysis of myofibroblast-like cluster from single cell BALO data set 'ddSeq5'.** A) WASP generated violin plots with log2fold change values showing expression of myofibroblast-associated genes. B) Manual analysis violin plots with log2fold change values showing expression of the same myofibroblast-associated genes. Selected parameters and cutoff values are shown on the right side. B) was published in Vazquez-Armendariz *et al.* 2020 [195]

4.2 Single cell analysis of *Mycobacterium tuberculosis*-infected *Galleria melonella* larvae

Pulmonary diseases prove to be a drastic burden for global health, highlighting the need for research in this field. While the COVID-19 pandemic has drastically underlined the significance of this topic and sent the world into a pandemic state, other diseases, especially bacterial-borne, targeting the respiratory tract have moved into the background. However tuberculosis, - an airborne communicable disease caused by *Mycobacterium tuberculosis* - is among the leading global causes of death [214]. Until the COVID-19 pandemic, tuberculosis was the leading cause of death induced by a single infectious agent, causing twice as many deaths as the human immunodeficiency virus (HIV) and the resulting acquired immunodeficiency disease (AIDS) [214]. First identification of *M. tuberculosis* was performed by Robert Koch in 1884 and together with its related disease, remains to be a global research concern until today [215]. Estimations suggest that around a quarter of the global population has been infected [216] with *M. tuberculosis* with approximately 5% of patients developing an active disease within the first two years post infection [217]. However, some persons may develop an active infection even years later, suggesting that some infected patients are at a lifetime risk of developing the disease [218]. Although first drug treatments have been available since the 1950s, especially less developed countries still struggle with diagnosis and availability of treatment options [214] while also suffering from the highest incidence rates [214] (Fig. 38). However, untreated smear-positive tuberculosis patients, showing visible bacteria in a stained sputum sample, indicating a high bacterial load, have a death rate of approximately 70% and also smear-negative, but culture-positive patients still exhibit a death rate of around 20% [219]. Adding to that challenge is that successful treatment requires patients taking a combination of multiple antibiotic drugs over a period of typically at least six months following a strict schedule [220]. Furthermore, appearance of multi-resistance tuberculosis strains increases the global health risk while also increasing the economic burden of the disease with the cost corresponding to 0.52%

of the global gross national product and resulting in over 500 million Euros per year in the European Union [221] [222] [223].

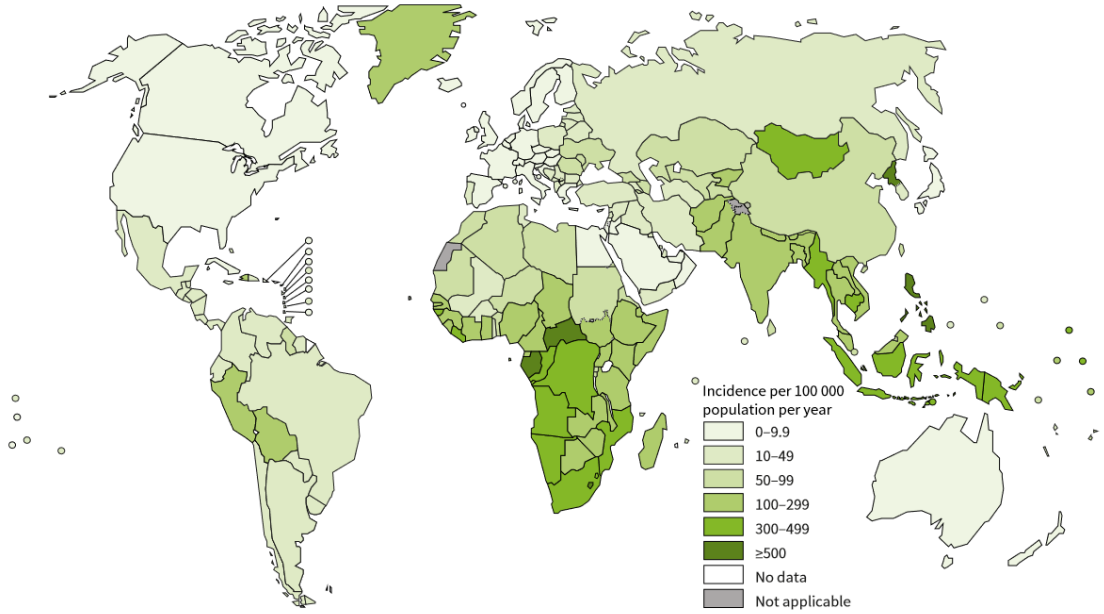


Figure 38: **Estimated global tuberculosis incidence rate according to the World Health Organization (WHO).** This figure was originally published in the Global tuberculosis report 2023 by the WHO [214] and is licensed under CC BY-NC- SA 3.0 IG.

Hence, there is a great need for research to achieve a deeper understanding of the tuberculosis disease and to develop novel approaches in vaccinations, diagnosis and treatments. This is also crucial to meet the aims of the WHO-initiated end tuberculosis strategy, targeting a 95% reduction of tuberculosis deaths and 90% reduction of tuberculosis incidence rate compared to 2015, respectively [214]. Efficient research, however, does require suitable infection models which currently come with a variety of limitations regarding biological mimicking of the disease, such as non-occurrence of granuloma and lung cavitation in *Mus musculus*, which is a characteristic for tuberculosis transmission in humans. Additional restrictions may include cost, space, and regulations for acquisition and housing of animals. Although lung organoids are emerging as a possible future infection model, replication of *Mycobacterium tuberculosis* provides a

hurdle, as this is located in the alveolar part of the lung, and human-based lung organoids are currently limited by absence of pneumocytes [224]. Furthermore, successful introduction of macrophages to human organoids, similar to the TR-Mac enrichment of murine organoids described in section 4.1 [195], would be necessary for a more complete tuberculosis infection modeling [224]. In order to overcome this bottleneck, additional organisms have been researched regarding their potential to serve as model organism with the larvae of the greater wax moth *Galleria melonella* turning out to be a very promising alternative. Previously, *Galleria* has already been used as an infection model for over 65 different strains of bacteria and fungi [225], [226]. The recently started use of *G. melonella* as infection model is based on a variety of advantages:

1. Complex immune system with haemocytes (phagocytic cells) resembling the function of mammalian neutrophils and macrophages
2. Incubation at 37 °C
3. Simple housing without need of specialized maintenance or equipment
4. Infection via injection enables precise dosing
5. No ethical approval required as for mammalian organisms such as mouse or non-human primate animals
6. Fast experimental data generation due to short lifespan

In summary, *G. melonella* represents a cost-effective and easy to care organism with potential as a model system. A bottleneck in this regard, however is the currently limited availability of immunological and molecular methods for analysis of infection-response. To overcome these limitations, Masanori Asai *et al.* established a variety of *Galleria* infection models for the *Myocbacterium tuberculosis* complex, with the first one employing a luminescent reporter vaccine strain of *Mycobacterium bovis* Bacillus Calmette-Guérin (BCG) as surrogate for *Mycobacterium tuberculosis* [227]. Further approaches focused on applying the widely used virulent *Mycobacterium tuberculosis* H37rv

reference strain, first as attenuated double-auxotroph ΔleuDΔpanCD mutant [228], being unable to replicate without supplementation of panthothenate and leucine [228], and subsequently the wild type H37rv strain [229]. *M. bovis* BCG exhibits a genetic similarity of more than 99.9 % with *M. tuberculosis* [230] and is in compliance with biosafety level 2 standards, enabling research without these specialized facilities [230]. Reduced biosafety level requirements also applied to the double-attenuated H37rv strain, while H37rv wild type is limited to research in biosafety level 3 and above facilities [228], [230], thus complicating research efforts. Further studies of the models provided promising results, such as development of Granuloma-like structures during active tuberculosis infection, which resembles a crucial aspect of the disease, virulence differentiation between different *Mycobacterium* species, and the opportunity of using *G. melonella* for antimicrobial drug screening [229]. However, to classify results in relation to other model organisms and further validate the models, a deeper understanding of *G. melonella*'s immune response is necessary. This includes detection of expression of inflammatory marker genes, necrosis-related marker genes and phagocytic receptor genes on hemocytes, which are involved in developing Granuloma-like structures [229]. In order to further characterize the immune response of *Galleria*, a single cell RNA-seq experiment, involving the double-auxotroph ΔleuDΔpanCD *M. tuberculosis* H37rv variant, was conducted by M. Asai *et al.*, Imperial College London. Following the current analysis, results of the data generated using WASP and R are presented. Furthermore, a manuscript including all results is currently in preparation.

The experiment was performed with three treatment conditions of *Galleria melonella* - uninfected, 2 days pi and 7 days pi. Larvae from each condition were processed using the Scipio Asteria single cell protocol (section 3.1.1, Fig. 19, [32]) and sequenced using an Illumina NextSeq 2000 with samples for each condition distributed over two lanes. The raw sequencing FASTQ files were then pre-processed using WASP with results shown in Table 5.

Using the WASP pre-processing Shiny application, a gene expression matrix was generated for each lane. Based on capture rates provided by Scipio, the number of barcodes

Table 5: **Read metrics of *Galleria melonella* single cell data set over three different conditions.** The data set consists of six total samples separated into three conditions 'Uninfected' and two different *Mycobacterium tuberculosis*-infected time points '2 days pi' and '7 days pi'. For each condition, the sample is distributed over two lanes due to sequencing limitations. As the Scipio Asteria protocol does not provide a barcode whitelist, no removal of invalid barcodes was performed. Instead each read was processed according to the barcode and UMI scheme shown in Fig. 19 and reads sharing the same barcode sequence were combined.

Sample	Reads total	Barcodes total
Uninfected L1	223,131,9443	3,305,411
Uninfected L2	199,725,363	2,977,764
2 days pi L1	214,004,026	3,256,109
2 days pi L2	192,197,288	2,909,947
7 days pi L1	98,667,688	2,972,186
7 days pi L2	90,762,880	2,661,328

was selected accordingly, resulting in 3,750 cells per condition and gene expression matrices for each condition merged into a single matrix over all samples. Additionally, a meta data CSV file was generated, combining barcode sequence and corresponding conditions. The resulting matrix and the meta data sheet was then uploaded to the WASP downstream Shiny application and processed using the automatic analysis mode; chosen parameters are shown in Fig. 39. As a result, 13 clusters have been detected across all conditions. As was clearly observed within the UMAP plot, clusters belonging to the same time point seem to be located closer to each other (Fig. 39).

Figure 39: **WASP clustering of *Galleria melonella* single cell data set.** The visualization shows a UMAP plot of the clustering analysis of all conditions ('Uninfected', '2 days pi and 7 days pi) of the *Galleria* data set. Conditions are further visualized with different symbols: 'Uninfected' = Square, '2 days pi' = dot, '7 days pi = Triangle. Therefore, diversity in the data seemed to be mostly driven by condition. The table on the right provides parameters selected in WASPs automatic mode. Hoek A. & Asai M., *et al.*, in preparation.

Hence, different conditions seemed to be the largest driver for difference in the data. Although this might not seem surprising, this was a problem in regard to *e.g.* identification of possible cell types, their according biological meaning, and effects occurring within this cell type across conditions. Therefore, integration of different conditions was necessary to enable further analysis which was performed using the R package Harmony. While the general aim is similar to the previously used Seurat CCA, Harmony, which was published a few years later than Seurat, provides overall comparable results while showing a more efficient analysis for larger data sets in regards to CPU and RAM usage. Furthermore, a previous study analyzing a similar composed single cell data set of *Drosophila melanogaster* described a successful integration based on Harmony. Integration with Harmony begins by calculating a low-dimensional PCA-based embedding, which is used to group cells into multi-data set clusters. Following, so-called soft or fuzzy clusters are defined, enabling assignment of a cell to multiple potential clusters. In the next step, a Harmony-specific modified 'soft' k-means algorithm is applied, which favors clusters exhibiting a composition of cells belonging to multiple

data sets. Additionally, clusters comprising a disproportionate composition are penalized. Following the clustering, cluster-specific centroids are defined for each data set, that are then used to calculate cluster-specific linear correction factors. These correction factors are then applied as cluster-weighted average, resulting in a unique correction factor for each cell moving the cell to one or multiple clusters. Similar to the 'normal' k-means, this process is iterated multiple times until a stable cluster assignment has been achieved [74].

As shown in Fig. 40, integration of the *Galleria* data set was performed successfully using Harmony. Composition of the data set now exhibits a more equal distribution across different conditions, enabling a characterization of cluster identities.

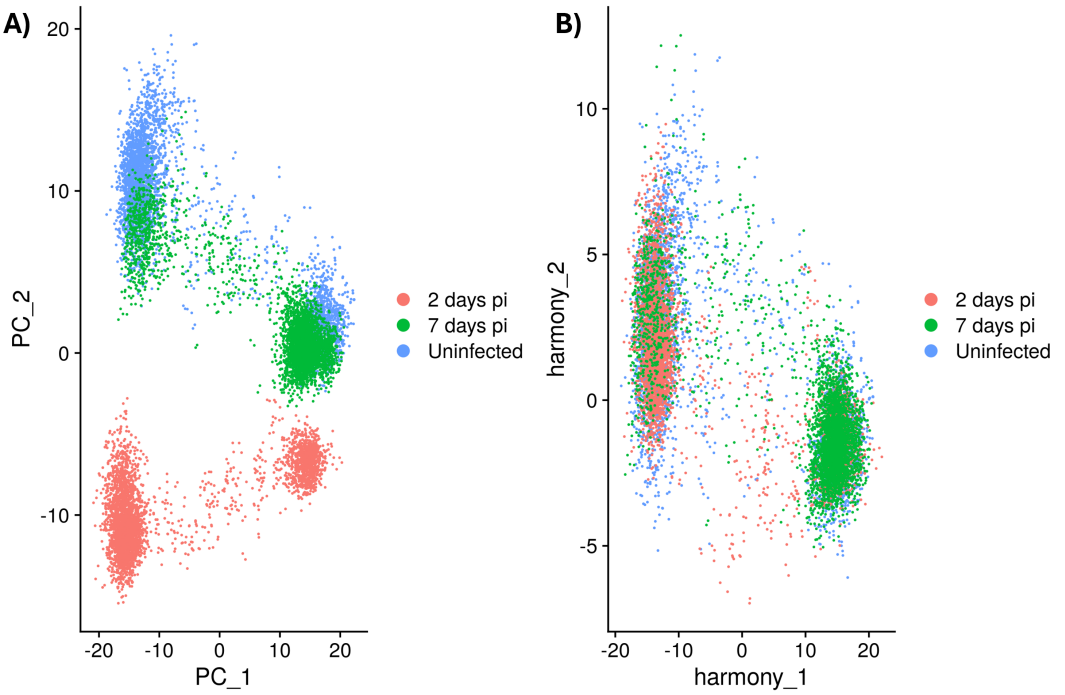


Figure 40: **PCA plot showing *Galleria melonella* data sets after integration of different conditions using Harmony.** A) Before Harmony integration. B) After Harmony integration. Manuscript in preparation.

As an advantage, Harmony can directly be integrated within a Seurat analysis, enabling a subsequent usage of Harmony-corrected Seurat objects in R without the need of further modification or transition of the data. Therefore, and similar to the WASP workflow, the corrected data was clustered with a resolution value of 0.25 resulting in a total of 11 clusters and enabled detection of associated up- and downregulated marker genes. As a first result, two genes of interest were identified, LOC113516725 (Hdd11) and LOC113513798 (Croquemort-like), genes associated with a defense protein involved in formation of Granuloma-like structures [230] and a hemocyte receptor recognizing apoptotic cells [231], respectively. Using the Seurat-based co-expression visualization in WASP, an overall distinct expression of the two genes was observed, which was in line with expectations that cells do not co-express both genes simultaneously. Based on these results, probes for a validation via RNA-Fluorescence *in situ* hybridization (FISH) were designed. RNA-FISH describes usage of a fluorescently tagged single-stranded RNA fragment consisting of a short sequence complementary to a target RNA [232]. Thus, using the bound FISH probe, localization of target *e.g.* mRNAs can be performed using fluorescent microscopy. Analysis of the experiment can be seen in Fig. 41 and was consistent with the results from WASP's co-expression analysis (Fig. 42).

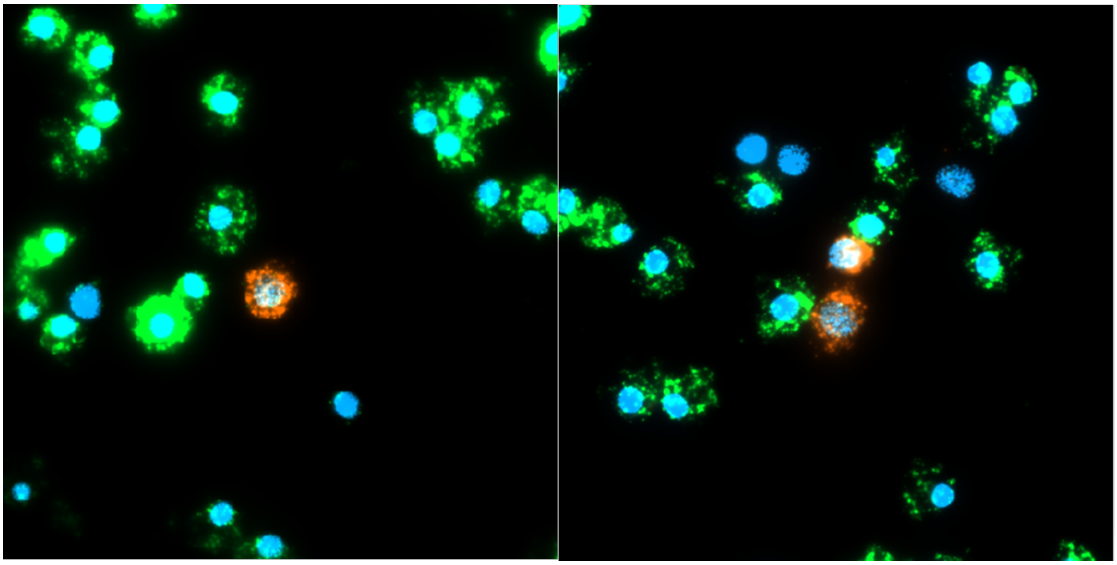


Figure 41: **RNA-FISH analysis of *Galleria melonella*.** Fluorescence probes marking LOC113516725 (Hdd11) in orange and LOC113513798 (Croquemort-like) in green. Cells were stained using 4',6-Diamidin-2-phenylindol (DAPI, blue). In line with *in silico* results, cells express exclusively Hdd11 or Croquemort. Image generated and provided by Masanori Asai. Manuscript in preparation.

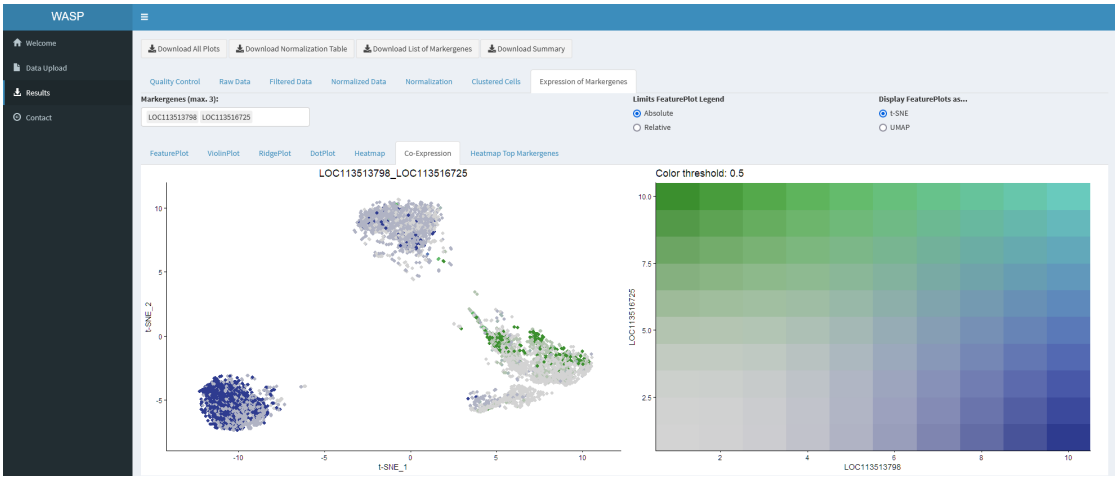


Figure 42: **Co-Expression plot from WASP showing with LOC113516725 (Hdd11) and LOC113513798 (Croquemort-like).** The plot on the left side shows a UMAP representation of all conditions from the *Galleria* data set. Expression of LOC113516725 (Hdd11) is shown shown in green, expression of LOC113513798 (Croquemort-like) is shown in blue and expressing of both genes within one cell is marked in turquoise. Manuscript in preparation.

However, a challenge in regard of further cluster identification is the less extensive genomic annotation of *Galleria melonella* compared to more commonly used model organisms such as *Mus musculus* or *Drosophila melanogaster*.

Therefore, an additional Gene Ontology (GO) term enrichment analysis will be performed using the extracted marker genes from each detected cluster. This will enable to combine identified genes to characterize biological functions for each cluster and also allows to include homologous genes with similar function in additional organisms. Also, identification of which genes are the largest drivers of the proposed biological functions will be helpful as a starting point for future investigations utilizing, for example, *G. melonella* knockout mutants.

4.3 Demonstration of the WASP integration within openBIS

In order to facilitate data storage with simple and FAIR data analysis, the WASP platform was planned to be integrated into the openBIS data management system with the KFO309 as an exemplary beneficiary. The actual implementation is comprehensively described in section 3.2.1. While the successful analysis of BioRad-based data has already been shown in section 4.1.2, WASP has also been extended for additional protocols including the popular 10x platform, which was also increasingly used within the KFO. Therefore, a 10x-based data set 'PMBC 1k v3' was selected to be used for demonstration of WASPs openBIS integration, to also show a successful pre-processing of this specific barcode UMI scheme. The data set itself is used in various single cell workflow examples, is publicly provided by 10x and contains human peripheral blood mononuclear cells. Cells were analyzed with the 10x v3 protocol resulting in two FASTQ files including 167,543,760 and 172,740,011 reads, respectively.

For the purpose of visualizing the interaction with openBIS, this chapter shows images of web browser screenshots that have been cropped for a better visualization of the respective page. openBIS was accessed via the Liferay frontend using a web browser, (section 1.6.1, Fig. 43). At first, a new project was created within the KFO openBIS instance, requiring further information such as a project name, involved investigators

and a project description (Fig. 44). This then enabled the creation of a new experiment, which is used to store experimental data such as the sample FASTQ reads. Project and experiment creation were guided by the 'Project Wizard' and 'Experiment Wizard' portlets described in section 1.6.1 (Fig. 44, 45). Upon entering all necessary data set information, the experiment and described samples were successfully registered within openBIS (Fig. 45). Following the registration, the project was now accessible via the 'Project Manager' portlet (Fig. 46). Now, the raw FASTQ files were uploaded under 'Samples' tab (Fig. 47), which provides a detailed overview of uploaded data in combination with information previously entered during experiment creation(Fig. 48). Furthermore, a corresponding reference genome and annotation were uploaded as 'Project Related Data' (Fig. 49). As all required data was now uploaded to the experiment, the WASP pre-processing analysis was ready to be executed via the 'Workflows' tab. For the analysis, all uploaded data was selected, the 10x protocol selected and a job name provided, enabling monitoring the analysis progress (Fig. 50). After the analysis was performed, information about the workflow and resulting files were accessed via the 'Steps' tab (Fig. 51). With the successful workflow execution, the Snakemake-based WASP analysis pipeline was executed, thus reads were quality checked, mapped to the reference genome, feature checked, demultiplexed based on barcodes and UMIs counted. Therefore, openBIS provided a GUI access to the previously CLI-restricted workflow.

Figure 43: **openBIS KFO landing page.** The starting page of the openBIS KFO site based on the Liferay frontend. The red square marks menu points including the 'Project Wizard' (red arrow) and the 'Project Manager' portlets. Selecting 'Create Project' initiates creation of a new project with the user being guided through the 'Project Wizard'.

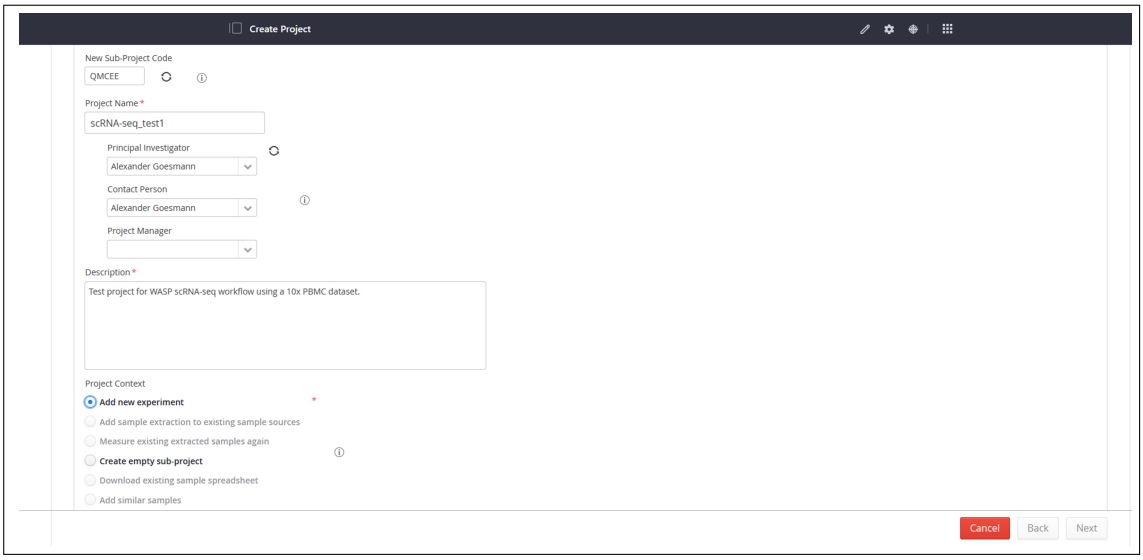


Figure 44: **openBIS Project Wizard.** Example image of the 'Project Wizard'-guided project creation. Mandatory inputs are marked with a red asterisk. Upon project creation, the 'Experiment Wizard' is started.

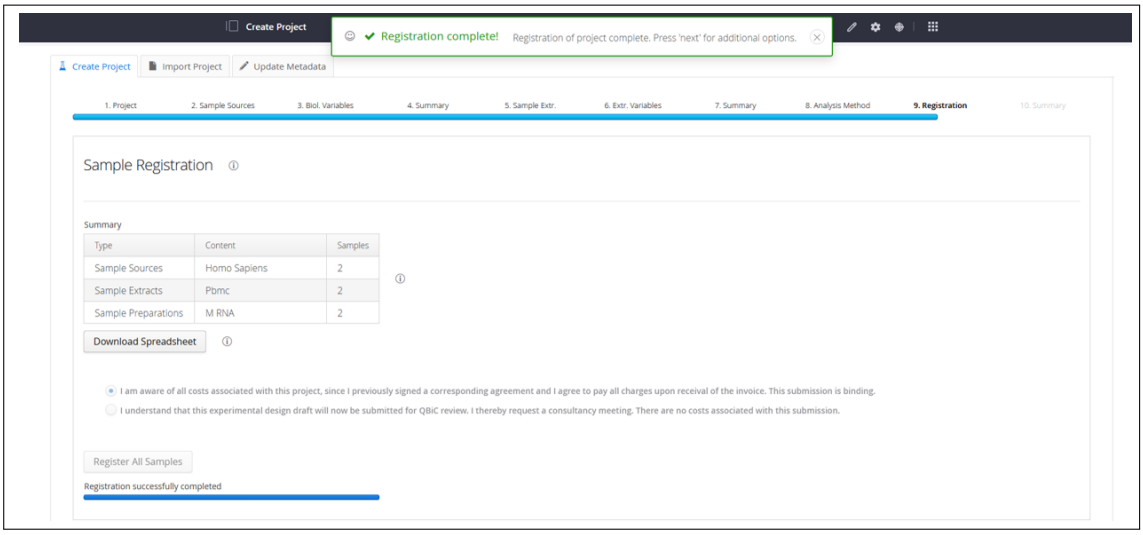


Figure 45: **openBIS Experiment Wizard.** Example image of the 'Experiment Wizard'-guided experiment creation. Upon entering all necessary information, the experiment is registered within openBIS, enabling a subsequent data upload.

Figure 46: **openBIS KFO landing page.** The starting page of the openBIS KFO site based on the Liferay frontend. The red square marks menu points including the 'Project Wizard' and the 'Project Manager' (red arrow) portlets. Selecting 'Project Manager' enables navigation through previously created projects for data upload, workflow execution, and analysis result visualization.

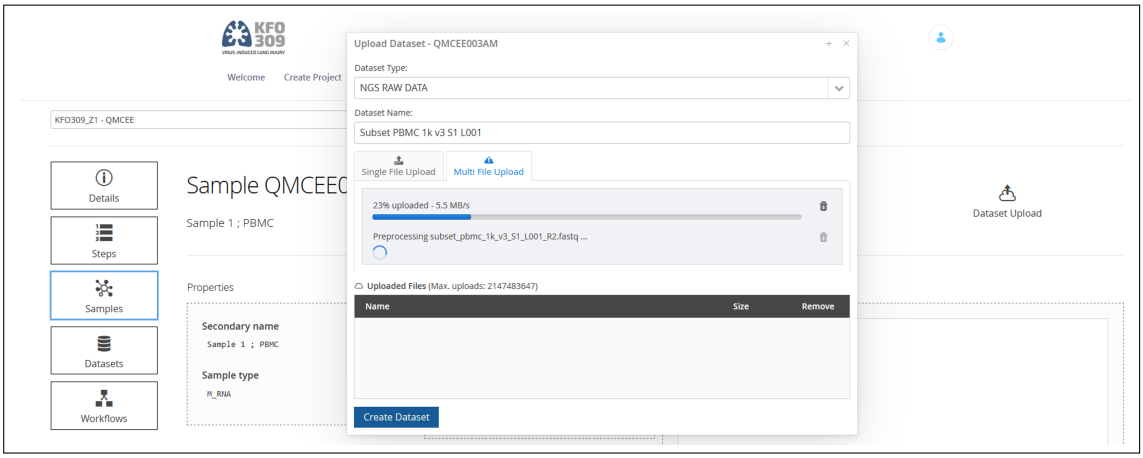


Figure 47: **openBIS KFO experiment sample page during upload.** The 'Sample' menu guides the user to an overview of all samples uploaded within the experiment. Furthermore, openBIS enables a simple data upload by drag and drop of files into the web browser triggering an overlay window showing the current upload progress.

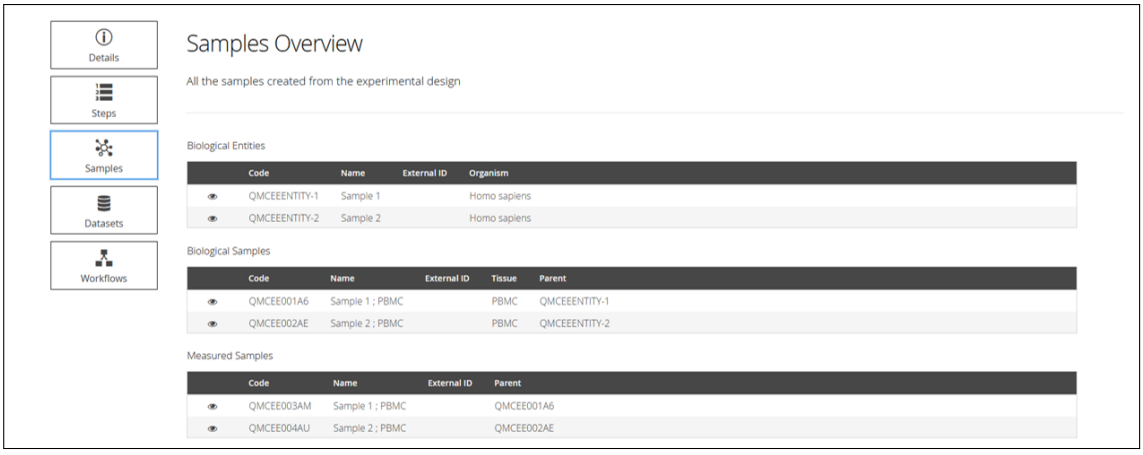


Figure 48: **openBIS KFO experiment sample page summarizing uploaded data.** Upon successful uploading of data (Fig. 47), the 'Sample' tab provides a detailed overview of uploaded data, based on information entered during experiment creation (Fig. 45).

Figure 49: **openBIS KFO experiment upload of project-related data.** In addition to the sample data, users can further upload project-related data such as reference genomes and corresponding annotations.

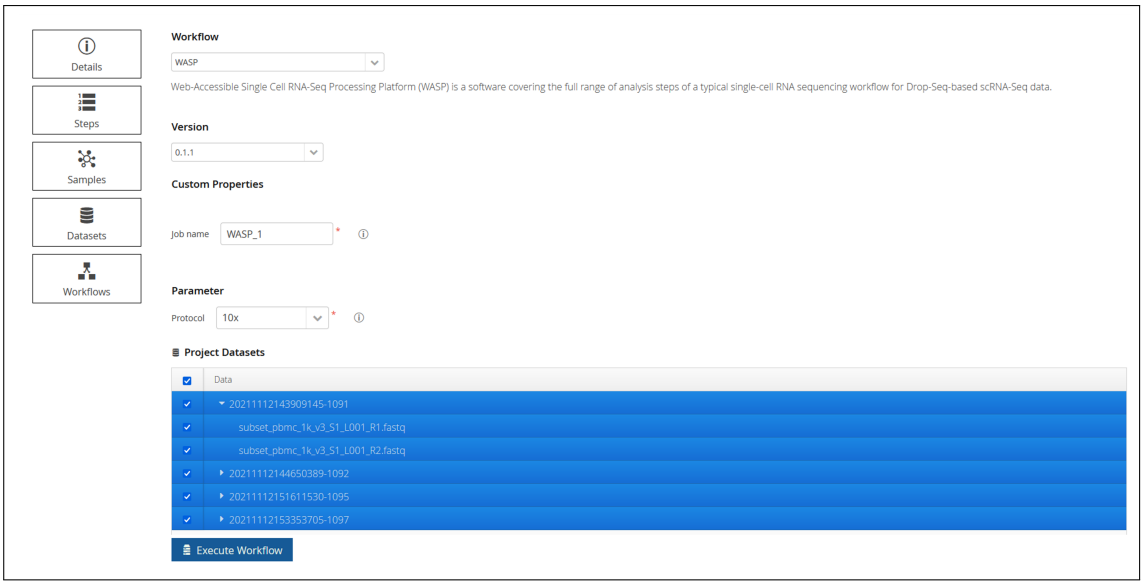


Figure 50: **openBIS KFO experiment workflow selection.** Using the 'Workflow' tab, users are able to select workflows to be run with their uploaded data. Depending on the selected workflow, additional parameters might be required, such as the used single cell protocol in case of WASP. Further, different versions of the workflow can be selected, *e.g.* for reproducibility. Also, users can select which data should be included for analysis by selecting the check mark for uploaded data in the bottom menu, enabling to perform analysis also with subsets.

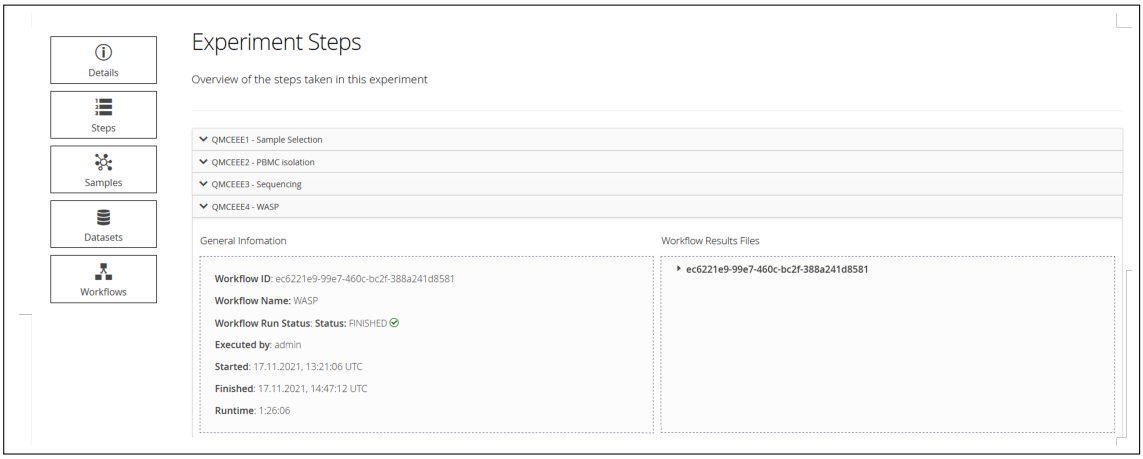


Figure 51: **openBIS KFO WASP analysis summary.** Using the 'Steps' tab, users are provided with an overview of previously performed steps within this experiment. The WASP subsection provides information such as runtime and workflow status for the WASP pre-processing Snakemake pipeline on the left side. Analysis result files can be accessed on the right side enabling to further run an interactive WASP session outside of openBIS.

However, as mentioned in section 3.2.1, further analysis steps require step-by-step interaction with generated data, which is provided via WASP's Shiny applications. As openBIS currently does not support Shiny applications, this is achieved by uploading the data with a unique ID at the end of the pre-processing workflow to the de.NBI S3 storage and providing interactive WASP sessions. For this, a WASP Docker container is executed in a VM which automatically accesses uploaded data using a custom server, described extensively in chapter 3.2.1. In order to keep this process as simple as possible for the user, the pre-processing workflow generates an HTML file which contains the corresponding data set ID and re-directs the user inside the web browser to an interactive WASP session. This HTML document was accessed in the 'Workflow Results Files' leading to a loading animation. Due to network latencies and data set size, the redirect failed in some cases which was solved by adding a waiting time of 15 seconds to provide enough time for the container to start and retrieve the data set (Fig. 53). Finally, the interactive WASP Shiny GUI was presented, providing a detailed overview

of the data set and enabling further analysis of the data set in the web browser (Fig. 53), similar as described in section (3.1.2).

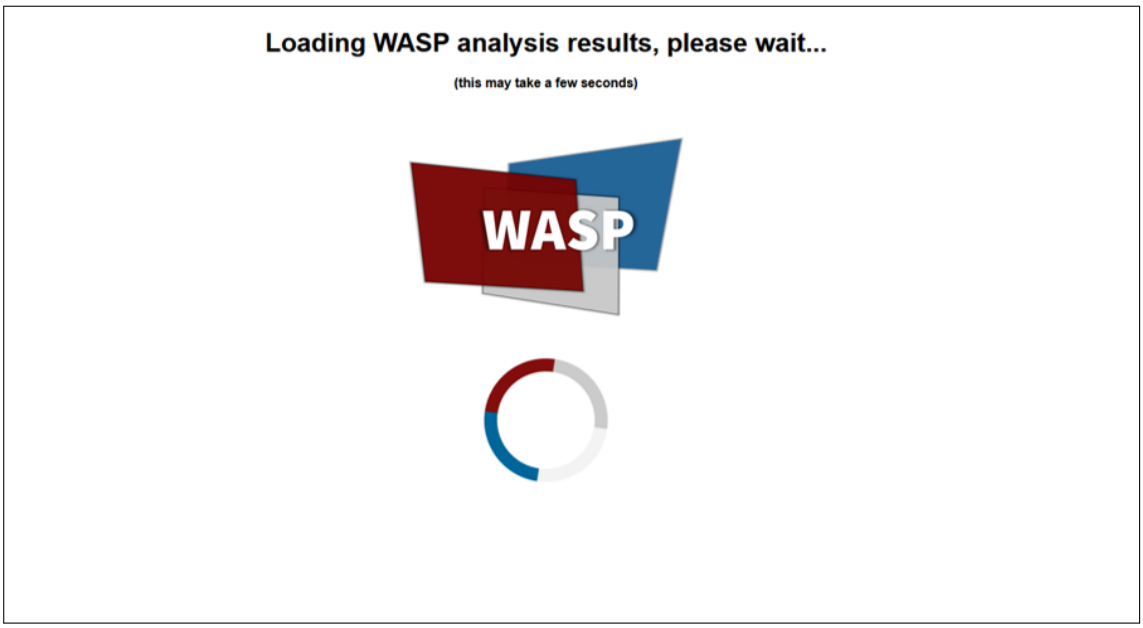


Figure 52: **WASP session loading screen.** Upon selecting WASP results HTML file within openBIS, the user is redirected inside the browser window to an interactive WASP session running on a VM in the de.NBI cloud. Users are presented with this loading screen first to provide enough time and account for network latencies for the WASP session to be initiated.

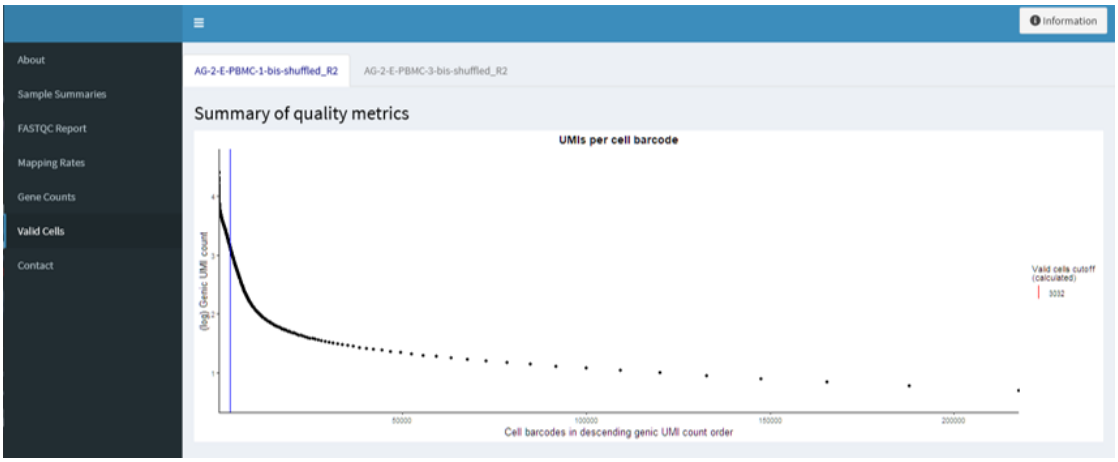


Figure 53: **Results of redirected WASP interactive session.** Following the loading screen, the user is presented with a WASP session, enabling further data processing as described in section (3.1.2).

5 Discussion and outlook

The first aim of this thesis was the development of a single cell RNA-seq analysis platform with easy usability, covering pre- and post-processing steps without limitation to a single manufacturer protocol. This goal was realized in the form of the software WASP, presented extensively in Chapter 3.1. In order to achieve an efficient fulfillment of this aim, the software was implemented in form of three modules:

- Pre-processing
- Pre-processing visualization
- Downstream analysis

Pre-processing was originally developed to perform analysis of a KFO-based murine BALO data set described in section 4.1. Following an evaluation of the best-suited tools, pre-processing has been realized as a Snakemake workflow, which enables an efficient and simple execution in combination with reproducibility in regards to FAIR criteria. Functionality of the workflow was firstly demonstrated by analyzing multiple BALO data sets described in section 4.1. However, at this time, the workflow was tailored to data generated with the BioRad/Illumina ddSEQ protocol, and hence was subsequently extended to overcome the mentioned restriction to a single manufacturer. This was important as popularity of the 10x-based single cell protocol, for example, has massively increased in the past years. Also, other devices and protocols such as Nadia Dolomite and Scipio Asteria were added to increase flexibility and enable an analysis of experimental single cell data regardless of the laboratories choice of a manufacturer. By implementing pre-processing as a separate module, easier distribution and use of the other software modules was also achieved, as pre-processing is limited to a variety of established Linux-based tools and, due to the enormous size of the raw data, requires execution on a more powerful system such as a workstation or even an HPC in order to achieve timely analysis. Still, for a most simple distribution, a Conda envi-

ronment covering necessary dependencies was described and the pre-processing was also implemented as a Docker container.

Pre-processing visualization was implemented using the R Shiny framework, enabling a straightforward combination of R-based code with a GUI, accessed directly in the web browser. This enables a visual representation of the Linux-based analysis steps, that, with exception of FastQC, do not provide graphical output. Usage of Shiny also enabled a simple implementation of interactive visualizations, aiding users in data selection, such as the knee plot (section 1.5.2). Furthermore, the separation also enabled a distribution independent of the user's OS, as only the R interpreter is required, which is available for all major OS. For Windows, even an installation-free distribution was achieved using a portable R version. Consistent with the pre-processing pipeline, however, the Shiny application is also offered as a Docker container. Separation of the pre-processing and the pre-processing visualization also enables a more efficient analysis workflow, as usually quality control assessment of raw reads and the generation of the gene expression matrix is performed once, while downstream analysis is typically performed in various iterations to establish optimal parameters. Additionally, as the gene expression matrix marks the end of this module, data generated by WASP provides users with the freedom of choice in regards to which tool they would like to employ for downstream analysis.

Downstream analysis was implemented in a similar way as the previous module, using R Shiny. This adds the same benefits as described above but is even more useful in this case, as many of the performed analysis steps are based on established R packages (Chapter 4.1, [198]). Thanks to R Shiny, interactive visualizations in combination with user-friendly parameter modifications were implemented, providing researchers with direct feedback. Also, this module only requires a gene expression matrix as input, therefore enabling usage of WASP with data generated with other pre-processing pipelines or analysis of data sets that are not available as raw data. While the downstream analysis steps were originally performed separately using R scripts, the resulting combination of various R packages in form of the WASP Shiny application enables a

very simple usage of complex tools with a GUI significantly lowering the entry barrier for scientists to analyze their own data.

Functionality of all WASP modules were compared by re-analyzing the manually analyzed *Mus musculus* BALO data set shown in section 4.1.2. The intermediate steps developed and the insights gained during this analysis formed the basis of the WASP project. However, as the software marked a new approach and included a variety of novel developments, it was important to verify consistency. All three WASP modules were then applied for analysis of the raw BALO reads. Obtained results were consistent with originally generated and published results, which were further validated by additional wet-lab experiments including FACS, qPCR and fluorescent antibody staining with microscopy (section 4.1).

WASP was also used for the analysis of a single cell RNA-seq data set based on *Galleria melonella* larvae infected with *Mycobacterium tuberculosis*. Again, all three WASP modules were used for data analysis. Results of all performed analysis steps are shown in section 4.2. As this data set was generated based on the Asteria single cell protocol, the successful processing again underlines WASPs overcoming of a limitation a single manufacturer protocol. To this day, WASP represents the sole freely available application able to analyze Asteria data; the Cytonaout application by Scipio Bioscience represents a commercial application offered on a pay-per-sample basis and requires prior deposition of the data on cloud-based systems. Furthermore, the generation and application of RNA-FISH probes based on the *in silico* results further validated results proposed by WASP.

In summary, the implementation of WASP successfully addresses the above mentioned aim and its functionality was validated by successful application of the software to multiple single cell RNA-seq data sets.

The second goal of this thesis was the integration of the developed software platform into the openBIS workflow repository, enabling scientists to perform single cell RNA-seq analysis of data stored inside the openBIS system.'

This aim was realized by integrating the WASP software into the openBIS system available at Giessen University. The openBIS installation itself was extensively described in section 1.6.1. A separate KFO309 openBIS partition was provided by Jannis Hochmuth and Frank Förster. This was then used for the upload of raw FASTQ read files of a 10x-based data set. Subsequently, the WASP pre-processing Snakemake workflow (Chapter 3.1.1) was then tested with the data set utilizing the workflow registry of Sven Griep (Chapter 3.2.1). Following, a successful processing within a few hours was observed, further exhibiting the compatibility of WASP to 10x-based data. The openBIS integration lowered the entry barrier of WASP's pre-processing workflow, as this part is restricted to Linux-based software, requiring a user to perform at least the software's execution on a CLI. Due to openBIS and the workflow registry, a GUI for this WASP module was provided. Additionally, the responsibility for resource management was taken away from the user as well which further simplified usage. Despite successful processing, result visualization was very limited as openBIS only supports visualization of static HTML and image files, contradicting the essential concept of WASP for interactive visualization.

Thus, the pre-processing workflow was extended with mechanisms for the upload of the processed data set to the de.NBI cloud. The uploaded data can then be accessed from within openBIS. Using a container-based WASP version running on VMs in the de.NBI cloud, the data set is automatically retrieved and subsequently enables its interactively visualization and further processing in the user's web browser. While this implementation required some more complex processes performed in the background, it provides a very simple solution to the openBIS users as they are completely shielded from any complex tasks. As before, users only need to select the WASP workflow, retaining the above mentioned entry barrier simplification and finally select the result HTML file upon workflow completion. The newly developed solution allows to circumvent current restrictions of the openBIS system and is able to connect workflows executed within openBIS to dynamic output visualizations. This approach is further suitable for all web-based applications and not restricted to R Shiny. For an even more

sophisticated resource management, the server could be modified to only connect users and their associated data to a container executed and managed in an environment such as Kubernetes. All in all, the described solution meets the target of integrating the WASP software into openBIS and enabling users to perform analysis of single cell RNA-seq data.

In general, WASP provides a useful addition to the landscape of single cell RNA-seq analysis solutions. By supporting the most commonly used manufacturer-based protocols, the software does not restrict researchers in their choice of technology and can be applied to the majority of Dropseq-based single cell data sets. This stands in contrast to various other software solutions such as ddSeeker, Illumina BaseSpace, 10x CellRanger or Scipio Cytonaut, which are specifically limited to a certain protocol or manufacturer.

Also, the modularization of WASP increases the reusability of the software, as an externally generated gene expression matrix can also be used for WASPs downstream analysis. The general concept of WASP to provide automated analysis workflows in combination with the Shiny-based GUI further reduces the entry barrier to perform single cell analysis. Thus, WASP enables a complete typical RNA-seq workflow beginning with raw reads or optional later entry points ending with a large variety of biological insights and corresponding visualizations. The software also enables FAIR research by exhibiting parameters used for each module. Furthermore, the philosophy of WASP to provide all parts to be downloaded and run locally on-premise enables analysis of restricted data sets, ensuring privacy of the data. Due to the possibility to run the Shiny modules on a normal laptop or desktop computer, this further enables analysis for researchers without access to large computational infrastructures. As a versatile and complete application fully addressing all steps required for RNA-seq analysis, the repeated exchange of information between lab researchers and bioinformaticians in the course of a manual analysis is avoided. This issue is solved by providing researchers with user friendly interactive elements enabling, *e.g.* direct look up for expression of a specific gene within the analyzed data set. These features are complemented by the

integration of WASP into openBIS, as this adds a GUI for the pre-processing module, reducing the complexity of analysis even more. Also, the FAIR concept benefits noticeable from this integration, as researchers are now able to deposit meta data along with the data set itself and the corresponding WASP analysis results. An important cornerstone to enable efficient usage of AI-driven data collecting and processing, which is essential to ensure efficient analysis of the ever-increasing amounts of generated data in the future. Many software solutions, in comparison, aim to provide either pre-processing (ddSeeker, STARSolo) or downstream analysis (Single Cell Explorer, Granatum, ASAP). Therefore, researchers have to select at least two or more tools and possibly transform a tool's output to be suitable as another tool's input in order perform a complete analysis, thus increasing the entry-barrier. On the other hand, solutions providing a full workflow analysis such as Illumina BaseSpace, 10x CellRanger or Scipio Cytonaut are, as described above, usually limited in regards to protocol or manufacturer compatibility.

As a perspective, future developments in the field of single cell RNA-seq analysis will continously be added to WASP, thereby extending its applicability. One interesting task for this is data set or batch integration. This was previously externally performed using Seurat's CCA (section 4.1) and Harmony (section 4.2). While both packages are already compatible with hardware requirements for some of the analysis are going to become an issue. For example, Seurat's CCA correction requires more than 16 gigabyte (GB) of RAM for data sets of more than approximately 30,000 cells. With WASP being conceptualized for interactive use, runtime considerations have to kept in mind, as well; some analysis types require more than 30 minutes to compute, and therefore results will not immediately available to the user. According to their own publication, Harmony performs a lot faster making it more suitable for this task. However, complexity of the methods needs to be evaluated as this might prevent a simplified usage of WASP in case users need to select a variety of unknown parameters for a successful integration process. This is further associated with the risk of masking biological variation when trying to remove technical effects during data set integration. Thus, a

possible solution needs to perform a balancing act between usability and application of complex algorithms for adequate data processing.

Another interesting additional processing step is the integration of a pseudotime analysis module. This increasingly popular method enables to analyze dynamic changes in the cellular population based on gene expression and a subsequent ordering of cells along the calculated progression trajectories (section 1.5.3). A variety of tools is available for this task with varying performance depending on the trajectory type, *e.g.* circular or linear progression, with or without branching [97]. In this regard, a previous study evaluated the tool 'Slingshot' [99] as best overall performing for different trajectory structures [97]. Another advantage of Slingshot is its implementation in R and compatibility with internal data objects used within WASP. Slingshot itself is designed to perform Pseudotime analysis following pre-processing, dimensionality reduction and clustering. Thereby an integration following the current WASP workflow could be a feasible solution (Fig. 54). First internal evaluations using a few thousand *Mus musculus* cells, however, indicate high memory demands exceeding 40 GB of RAM and multiple hours of processing time. Similar to the data set integration processing described above, this contradicts the philosophy of WASP of performing quick and interactive analysis, which can easily be repeated with different parameters. Also, this requires WASP to be run on either a HPC system or a workstation instead of a laptop or standard desktop computer, typically available in laboratories. To overcome this challenge, a possible solution to this problem is a conceptual change of the software to asynchronous calculation instead, separating analysis initiation and result visualization. A concept that could also be suitable for accommodating the potentially growing large data sets in the rapidly evolving field of single cell transcriptomics.

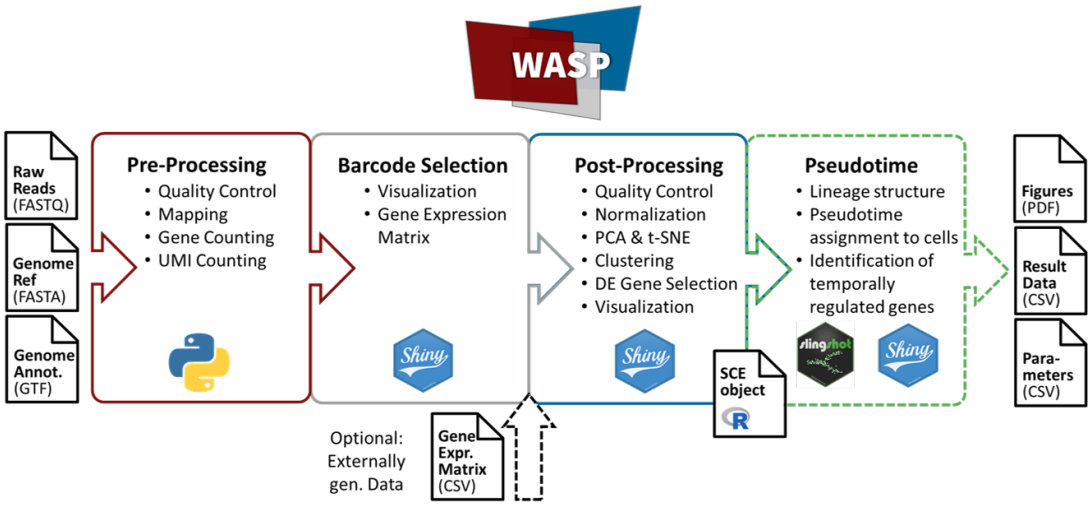


Figure 54: **Former concept of pseudotime analysis integration into WASP.** Pseudotime analysis would be performed using the R package Slingshot shown in the dashed green box.

Less of a specific analysis step, but also an exciting technological addition might be integration of AI-based components. While the topic of AI is on the rise for quite a time, some major breakthroughs have been observed especially in the last few years. This can currently be observed even in mainstream media coverage for topics such as 'ChatGPT' and similar AI assistant systems [233]. Apart from these everyday focused use cases, a variety of AI-based applications in bioinformatics and more specific also in the single cell field emerged. Most approaches focus on optimizing identification and assignment of cell types using machine learning in combination with known cell type markers obtained from publicly available repositories such as the human cell atlas to generate prediction models [234], [235], [236]. Additional studies aim at providing an improved pseudotime analysis workflow identifying more genes than traditional solutions [237] or aggregating and interpreting data obtained from various laboratory processing steps such as microscopy and FACS [238]. While a general use of AI-supported methods is still somewhat in its infancy in the single cell field, the publications to date already indicate a promising future and should be closely monitored with regard to possible integration opportunities. Although, as already mentioned hardware requirements might

also play an important role here, as many AI tools require the use of graphic cards or specified chips with sufficient memory.

Beyond implementation of other analysis steps, integration of interoperable file formats is an interesting addition. For example, in the genomics field standardized formats such as GTF or genbank are widely used and supported. However, single cell transcriptomics has unique requirements which information need to be stored along the gene expression values, *e.g.* dimension reductions, and cluster identity of cells. R-based software, and thereby also WASP, uses the single cell experiment format and Seurat objects to represent this information. A similar promising format is the ANNDATA format [239], implemented as Python package. This format is used in software such as Scanpy or the recently published CellxGene exploration tool by the Chan Zuckerberg foundation [240]. Enabling the export of ANNDATA files from WASP would therefore further expand the already comprehensive integrated concept of FAIR data analysis and sharing. Also, the integration of further standard formats in exchange with the community is envisaged.

Additionally, the adaptation of the ANNDATA format could provide new opportunities for enhancing WASP's performance with very large data sets. While R Shiny has proven to be a powerful framework for interactive data visualization, WASP has demonstrated excellent performance on typical single cell RNA-seq data sets ranging from hundreds to thousands of cells (e.g. chapters 4.1.1 and 4.2), scaling to data sets with 100,000 cells or more introduces novel challenges which may result in slow performances or significant hardware requirements [241], [242]. Recent studies have highlighted that the Python-based tools Scanpy and CellxGene, internally relying on the ANNDATA format, offer superior scalability for such large data sets compared to R and Shiny. Thus, integrating these tools into WASP could enhance its ability to visualize and analyze large scale data sets while maintaining compatibility with its existing workflows.

However, implementing these changes would require a fundamental shift in WASP's design philosophy. This paradigm shift, also discussed in the context of pseudotime

analysis integration above, would allow WASP to handle increasingly large and complex data sets such as cell atlas-scale data. However, this may come at the cost of reduced usability or accessibility, as CellxGene requires computationally intensive analysis steps, such as PCA, t-SNE or UMAP, to be pre-calculated which likely depends on more powerful workstations, HPCs or cloud systems rather than standard laptop or desktop computers, as typically found in laboratories. While this adjustment might challenge WASP's focus on dynamic and interactive analyses, its modular design provides an advantage as individual components could be adapted or replaced without abandoning the entire software. For example, the Snakemake-based pre-processing module could be retained while replacing the Shiny-based modules by Python-based tools such as Scanpy and CellxGene. Furthermore, it might even enable maintaining multiple versions tailored to different use cases - one version optimized for smaller data sets and dynamic workflows, and another version designed for large-scale data exploration.

6 Summary

Since its first application in 2009, single cell RNA-seq technology has provided an unprecedented resolution in understanding biological processes on a cellular level. Within the last decade, the technology was streamlined in various applications leading to a drastic improvement in regards to throughput and specificity in combination with cost reduction. This enabled numerous novel insights such as understanding of organ development, detection of cellular heterogeneity of tissues including detection of previously unknown rare cell types, analysis of cellular communications and many more. As a result of this vast range of application possibilities, single cell RNA-seq is used almost ubiquitously in life sciences, especially in human and mammalian cell-based systems. In this regard, the single cell-level resolution massively improved analysis and characterization of tumors, their development, involved cellular communication processes and revealed possible treatment targets. Also, a more complete understanding of various disease mechanisms was achieved including, *e.g.* COVID-19. Furthermore, multiple modeling processes were optimized using single cell RNA-seq such as development and characterization of organoids and application of model organisms, ultimately leading to generation of extensive data bases such as the human cell atlas providing additional resources for future studies.

While the single cell-based benefit for biological and medical research is undisputed, it is accompanied by increasing complexity. Apart from additional laboratory tasks, the bioinformatic analysis is affected in this process. Although the landscape of bioinformatic tools has massively grown in the last decades with the advent of NGS, not all tools are already suitable for analysis of single cell data. Instead a variety of novel challenges requires tailored solutions to cover processes such as barcode and UMI detection and correction, identification of real cells against ambient RNA-based artifacts or normalization of zero-inflated count matrices. Each manufacturer tends to develop an own protocol, hindering bioinformatic analyses by lack of standardization. This resulted in an ongoing development of new tools and software solutions for over a decade tackling these issues. However, many open source tools are either complex to use and

require a deeper knowledge in using a CLI or even programming skills in R or Python. Commercial tools are often limited to a specific manufacturer, subject to a charge or paid subscription and represent a black box in relation to analysis steps performed. Finally, a typical analysis process of a single cell RNA-seq data set requires application of multiple specific tools increasing the entry barrier even further for researchers without bioinformatic knowledge. Thus, analysis is often delayed and inefficient as even small and repetitive processing steps have to be requested by a researcher and then carried out by a bioinformatician. This is even more challenging due to an ever growing amount of data making making it difficult to keep track of analysis steps performed, identify useful data sets and comply with FAIR.

In order to overcome these challenges, the single cell RNA-seq analysis software WASP was created and published as part of this thesis. WASP has been designed as a complete solution to process single cell RNA-seq data, covering all aspects from initial QC to final interpretation. WASP supports all major scRNA-seq protocols and is continously extended. The software covers pre-processing and following downstream analysis finishing with clustering of cellular populations and detection of according up- and downregulated marker genes. By applying automated analysis workflows and GUI combined with interactive visualizations, WASP reduces the entry barrier for researchers to perform data analysis themselves. Additionally, multiple popular manufacturer-based protocols are supported giving researches freedom of choice in regards to their preferred or available laboratory single cell platform. Furthermore, the software is open source, publicly available and can be run on-premise to process even sensitive data not approved for data upload. In addition to analysis visualizations, WASP further exhibits parameters used during analysis in order to facilitate FAIR research. This has been further addressed by integrating WASP into an instance of the openBIS data management platform hosted at the Justus Liebig University. openBIS enables a browser-based data upload providing a GUI supporting upload, storage and searchability of experimental data while also providing detailed access control. Integration of WASP using the BCF's workflow registry, and utilization of the de.NBI

cloud S3 storage, and VMs finally provides direct analysis of single cell data from within openBIS in the web browser while still retaining WASP's interactive features. Thus, further improving easy access to single cell data analysis for researchers without bioinformatic knowledge and support of FAIR data storage and processing. Single cell RNA-seq is an emerging field lacking in standardization, but WASP, and especially data exchange between its individual components represents a major step towards successful analysis and reproducible interpretation of scRNA-seq data.

> *The important thing is to never stop questioning. Never lose a holy curiosity.*
>
> **— Albert Einstein**
> (Theoretical physicist)

7 List of Figures

8 List of Tables

9 References

[1] Soroush Tahmasebi, Nahum Sonenberg, John W.B. Hershey, and Michael B. Mathews. Protein Synthesis and Translational Control: A Historical Perspective. *Cold Spring Harbor Perspectives in Biology*, 11(9), sep 2019.

[2] Amelia Casamassimi, Antonio Federico, Monica Rienzo, Sabrina Esposito, and Alfredo Ciccodicola. Transcriptome profiling in human diseases: New advances and perspectives. *International Journal of Molecular Sciences*, 18(8):1652, aug 2017.

[3] Monica Sager, Nai Chien Yeat, Stefan Pajaro-Van Der Stadt, Charlotte Lin, Qiuyin Ren, and Jimmy Lin. Transcriptomics in cancer diagnostics: Developments in technology, clinical research and commercialization. *Expert Review of Molecular Diagnostics*, 15(12):1589–1603, dec 2015.

[4] Yasuo Nagafuchi, Haruyuki Yanaoka, and Keishi Fujio. Lessons From Transcriptome Analysis of Autoimmune Diseases. *Frontiers in Immunology*, 13:2303, may 2022.

[5] Andrea R. Daamen, Prathyusha Bachali, Katherine A. Owen, Kathryn M. Kingsmore, Erika L. Hubbard, Adam C. Labonte, Robert Robl, Sneha Shrotri, Amrie C. Grammer, and Peter E. Lipsky. Comprehensive transcriptomic analysis of COVID-19 blood, lung, and airway. *Scientific Reports*, 11(1):1–19, dec 2021.

[6] Mark D. Adams, Jenny M. Kelley, Jeannine D. Gocayne, M. A.K. Dubnick, Mihael H. Polymeropoulos, Hong Xiao, Cal R. Merril, Andrew Wu, Bjorn Olde, Ruben F. Moreno, Anthony R. Kerlavage, W. Richard McCombie, and J. Craig Venter. Complementary DNA sequencing: Expressed sequence tags and human genome project. *Science*, 252(5013):1651–1656, 1991.

[7] John Parkinson and Mark Blaxter. Expressed sequence tags: An overview. *Methods in Molecular Biology*, 533:1–12, 2009.

[8] Rohan Lowe, Neil Shirley, Mark Bleackley, Stephen Dolan, and Thomas Shafee. Transcriptomics technologies. *PLoS Computational Biology*, 13(5), may 2017.

[9] Zhong Wang, Mark Gerstein, and Michael Snyder. RNA-Seq: A revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, jan 2009.

[10] Medline Trend: Automated yearly statistics of PubMed results for any query. http://dan.corlan.net/medline-trend.html, 2023.

[11] Shawn E. Levy and Braden E. Boone. Next-generation sequencing strategies. *Cold Spring Harbor Perspectives in Medicine*, 9(7), jul 2019.

[12] Illumina and Gene Sequencing Technology - SBIR Success Story - NCI https://web.archive.org/web/20230311071702/https://sbir.cancer.gov/portfolio/success-stories/illumina, 2022.

[13] Karl V Voelkerding, Shale A Dames, and Jacob D Durtschi. Next-Generation Sequencing: From Basic Research to Diagnostics. *Clinical Chemistry*, 55(4), 2009.

[14] David R. Bentley, Shankar Balasubramanian, Harold P. Swerdlow, Geoffrey P. Smith, John Milton, Clive G. Brown, Kevin P. Hall, Dirk J. Evers, Colin L. Barnes, Helen R. Bignell, Jonathan M. Boutell, Jason Bryant, Richard J. Carter, R. Keira Cheetham, Anthony J. Cox, Darren J. Ellis, Michael R. Flatbush, Niall A. Gormley, Sean J. Humphray, Leslie J. Irving, Mirian S. Karbelashvili, Scott M. Kirk, Heng Li, Xiaohai Liu, Klaus S. Maisinger, Lisa J. Murray, Bojan Obradovic, Tobias Ost, Michael L. Parkinson, Mark R. Pratt, Isabelle M. J. Rasolonjatovo, Mark T. Reed, Roberto Rigatti, Chiara Rodighiero, Mark T. Ross, Andrea Sabot, Subramanian V. Sankar, Aylwyn Scally, Gary P. Schroth, Mark E. Smith, Vincent P. Smith, Anastassia Spiridou, Peta E. Torrance, Svilen S. Tzonev, Eric H. Vermaas, Klaudia Walter, Xiaolin Wu, Lu Zhang, Mohammed D. Alam, Carole Anastasi, Ify C. Aniebo, David M. D. Bailey, Iain R. Bancarz, Saibal Banerjee, Selena G. Barbour, Primo A. Baybayan, Vincent A. Benoit, Kevin F. Benson, Claire Bevis, Phillip J. Black, Asha Boodhun, Joe S. Brennan, John A. Bridgham, Rob C. Brown, Andrew A. Brown,

Dale H. Buermann, Abass A. Bundu, James C. Burrows, Nigel P. Carter, Nestor Castillo, Maria Chiara E. Catenazzi, Simon Chang, R. Neil Cooley, Natasha R. Crake, Olubunmi O. Dada, Konstantinos D. Diakoumakos, Belen Dominguez-Fernandez, David J. Earnshaw, Ugonna C. Egbujor, David W. Elmore, Sergey S. Etchin, Mark R. Ewan, Milan Fedurco, Louise J. Fraser, Karin V. Fuentes Fajardo, W. Scott Furey, David George, Kimberley J. Gietzen, Colin P. Goddard, George S. Golda, Philip A. Granieri, David E. Green, David L. Gustafson, Nancy F. Hansen, Kevin Harnish, Christian D. Haudenschild, Narinder I. Heyer, Matthew M. Hims, Johnny T. Ho, Adrian M. Horgan, Katya Hoschler, Steve Hurwitz, Denis V. Ivanov, Maria Q. Johnson, Terena James, T. A. Huw Jones, Gyoung-Dong Kang, Tzvetana H. Kerelska, Alan D. Kersey, Irina Khrebtukova, Alex P. Kindwall, Zoya Kingsbury, Paula I. Kokko-Gonzales, Anil Kumar, Marc A. Laurent, Cynthia T. Lawley, Sarah E. Lee, Xavier Lee, Arnold K. Liao, Jennifer A. Loch, Mitch Lok, Shujun Luo, Radhika M. Mammen, John W. Martin, Patrick G. McCauley, Paul McNitt, Parul Mehta, Keith W. Moon, Joe W. Mullens, Taksina Newington, Zemin Ning, Bee Ling Ng, Sonia M. Novo, Michael J. O'Neill, Mark A. Osborne, Andrew Osnowski, Omead Ostadan, Lambros L. Paraschos, Lea Pickering, Andrew C. Pike, Alger C. Pike, D. Chris Pinkard, Daniel P. Pliskin, Joe Podhasky, Victor J. Quijano, Come Raczy, Vicki H. Rae, Stephen R. Rawlings, Ana Chiva Rodriguez, Phyllida M. Roe, John Rogers, Maria C. Rogert Bacigalupo, Nikolai Romanov, Anthony Romieu, Rithy K. Roth, Natalie J. Rourke, Silke T. Ruediger, Eli Rusman, Raquel M. Sanches-Kuiper, Martin R. Schenker, Josefina M. Seoane, Richard J. Shaw, Mitch K. Shiver, Steven W. Short, Ning L. Sizto, Johannes P. Sluis, Melanie A. Smith, Jean Ernest Sohna Sohna, Eric J. Spence, Kim Stevens, Neil Sutton, Lukasz Szajkowski, Carolyn L. Tregidgo, Gerardo Turcatti, Stephanie Vandevondele, Yuli Verhovsky, Selene M. Virk, Suzanne Wakelin, Gregory C. Walcott, Jingwen Wang, Graham J. Worsley, Juying Yan, Ling Yau, Mike Zuerlein, Jane Rogers, James C. Mullikin, Matthew E. Hurles, Nick J. McCooke, John S. West, Frank L. Oaks, Peter L. Lundberg, David Klenerman, Richard Durbin, and An-

thony J. Smith. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, 456(7218):53–59, 2008.

[15] James M Heather and Benjamin Chain. The sequence of sequencers: The history of sequencing DNA. *Genomics*, 107(1):1–8, 2016.

[16] Scott J. Emrich, W. Brad Barbazuk, Li Li, and Patrick S. Schnable. Gene discovery and annotation using LCM-454 transcriptome sequencing. *Genome Research*, 17(1):69–73, jan 2007.

[17] Kimberly R. Kukurba and Stephen B. Montgomery. RNA sequencing and analysis. *Cold Spring Harbor Protocols*, 2015(11):951–969, nov 2015.

[18] Andreas Schroeder, Odilo Mueller, Susanne Stocker, Ruediger Salowsky, Michael Leiber, Marcus Gassmann, Samar Lightfoot, Wolfram Menzel, Martin Granzow, and Thomas Ragg. The RIN: An RNA integrity number for assigning integrity values to RNA measurements. *BMC Molecular Biology*, 7(1):3, jan 2006.

[19] Jana-Charlotte Hegenbarth, Giuliana Lezzoche, Leon J. De Windt, and Monika Stoll. Perspectives on Bulk-Tissue RNA Sequencing and Single-Cell RNA Sequencing for Cardiac Transcriptomics. *Frontiers in Molecular Medicine*, 2:2, feb 2022.

[20] Adam H. Freedman, John M. Gaspar, and Timothy B. Sackton. Short paired-end reads trump long single-end reads for expression analysis. *BMC Bioinformatics*, 21(1):149, apr 2020.

[21] Miten Jain, Robin Abu-Shumays, Hugh E. Olsen, and Mark Akeson. Advances in nanopore direct RNA sequencing. *Nature Methods*, 19(10):1160–1164, oct 2022.

[22] Sui Huang. Non-genetic heterogeneity of cells in development: More than just noise. *Development*, 136(23):3853–3862, sep 2009.

[23] Nicholas E. Navin. The first five years of single-cell cancer genomics and beyond. *Genome Research*, 25(10):1499–1507, oct 2015.

[24] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B. Tuch, Asim Siddiqui, Kaiqin Lao, and M. Azim Surani. mRNA-Seq whole-transcriptome analysis of a single cell. *Nature Methods*, 6(5):377–382, 2009.

[25] Jian Wang and Yuanlin Song. Single cell sequencing: a distinct new field. *Clinical and Translational Medicine*, 6(1):10, dec 2017.

[26] Aimaiti Yasen, Abudusalamu Aini, Hui Wang, Wending Li, Chuanshan Zhang, Bo Ran, Tuerhongjiang Tuxun, Yusufukadier Maimaitinijiati, Yingmei Shao, Tuerganaili Aji, and Hao Wen. Progress and applications of single-cell sequencing techniques. *Infection, Genetics and Evolution*, 80:104198, jun 2020.

[27] Suzan Yilmaz and Anup K. Singh. Singe Cell Genome Sequencing. *Current Opinion in Biotechnology*, 23(3):437, jun 2012.

[28] Katherine M. McKinnon. Flow Cytometry: An Overview. *Current protocols in immunology*, 120:5.1.1, feb 2018.

[29] Gabor Tajti, Tibor Gabor Szanto, Agota Csoti, Greta Racz, César Evaristo, Peter Hajdu, and Gyorgy Panyi. Immunomagnetic separation is a suitable method for electrophysiology and ion channel pharmacology studies on T cells. *Channels*, 15(1):53, 2021.

[30] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. Single-cell RNA sequencing technologies and bioinformatics pipelines. *Experimental Molecular Medicine 2018 50:8*, 50(8):1–14, aug 2018.

[31] Evan Z. Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R. Bialas, Nolan Kamitaki, Emily M. Martersteck, John J. Trombetta, David A. Weitz, Joshua R. Sanes, Alex K. Shalek, Aviv Regev, and Steven A. McCarroll. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, may 2015.

[32] Jun Komatsu, Alba Cico, Raya Poncin, Maël Le Bohec, Jörg Morf, Stanislav Lipin, Antoine Graindorge, Hélène Eckert, Azadeh Saffarian, Léa Cathaly, Frédéric Guérin, Sara Majello, Damien Ulveling, Anaïs Vayaboury, Nicolas Fernandez, Dilyana Dimitrova, Xavier Bussell, Yannick Fourne, Pierre Chaumat, Barbara André, Elodie Baldivia, Ulysse Godet, Mathieu Guinin, Vivien Moretto, Joy Ismail, Olivier Caille, Natacha Roblot, Carine Beaupère, Alexandrine Liboz, Ghislaine Guillemain, Bertrand Blondeau, Pierre Walrafen, and Stuart Edelstein. RevGel-seq: instrument-free single-cell RNA sequencing using a reversible hydrogel for cell-specific barcoding. *Scientific Reports 2023 13:1*, 13(1):1–11, mar 2023.

[33] Na Liu, Lin Liu, and Xinghua Pan. Single-cell analysis of the transcriptome and its application in the characterization of stem cells and early embryos. *Cellular and Molecular Life Sciences*, 71(14):2707–2715, jul 2014.

[34] R. N. Van Gelder, M. E. Von Zastrow, A. Yool, W. C. Dement, J. D. Barchas, and J. H. Eberwine. Amplified RNA synthesized from limited quantities of heterogeneous cDNA. *Proceedings of the National Academy of Sciences of the United States of America*, 87(5):1663–1667, 1990.

[35] Randall K. Saiki, Stephen Scharf, Fred Faloona, Kary B. Mullis, Glenn T. Horn, Henry A. Erlich, and Norman Arnheim. Enzymatic amplification of beta-globin genomic sequences and restriction site analysis for diagnosis of sickle cell anemia. *Science (New York, N.Y.)*, 230(4732):1350–1354, 1985.

[36] Lilit Garibyan and Nidhi Avashia. Research Techniques Made Simple: Polymerase Chain Reaction (PCR). *The Journal of investigative dermatology*, 133(3):e6, 2013.

[37] Fuchou Tang, Kaiqin Lao, and M Azim Surani. Development and applications of single cell transcriptome analysis Europe PMC Funders Group. *Nat Methods*, 8:6–11, 2011.

[38] Ashraful Haque, Jessica Engel, Sarah A. Teichmann, and Tapio Lönnberg. A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications. *Genome Medicine*, 9(1):1–12, aug 2017.

[39] Claude Thermes. Ten years of next-generation sequencing technology. *Trends in genetics : TIG*, 30(9):418–426, sep 2014.

[40] Alex A. Pollen, Tomasz J. Nowakowski, Joe Shuga, Xiaohui Wang, Anne A. Leyrat, Jan H. Lui, Nianzhen Li, Lukasz Szpankowski, Brian Fowler, Peilin Chen, Naveen Ramalingam, Gang Sun, Myo Thu, Michael Norris, Ronald Lebofsky, Dominique Toppani, Darnell W. Kemp, Michael Wong, Barry Clerkson, Brittnee N. Jones, Shiquan Wu, Lawrence Knutsson, Beatriz Alvarado, Jing Wang, Lesley S. Weaver, Andrew P. May, Robert C. Jones, Marc A. Unger, Arnold R. Kriegstein, and Jay A.A. West. Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. *Nature Biotechnology 2014 32:10*, 32(10):1053–1058, aug 2014.

[41] Kevin Lebrigand, Virginie Magnone, Pascal Barbry, and Rainer Waldmann. High throughput error corrected Nanopore single cell transcriptome sequencing. *Nature Communications 2020 11:1*, 11(1):1–8, aug 2020.

[42] Roger Volden and Christopher Vollmers. Single-cell isoform analysis in human immune cells. *Genome Biology*, 23(1):1–21, dec 2022.

[43] Peter J.A. Cock, Christopher J. Fields, Naohisa Goto, Michael L. Heuer, and Peter M. Rice. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Research*, 38(6):1767, dec 2010.

[44] Brent Ewing, La Deana Hillier, Michael C. Wendl, and Phil Green. Base-Calling of Automated Sequencer Traces UsingPhred. I. Accuracy Assessment. *Genome Research*, 8(3):175–185, mar 1998.

[45] Brent Ewing and Phil Green. Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. *Genome Research*, 8(3):186–194, mar 1998.

[46] Marcel Martin. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal*, 17(1):10, may 2011.

[47] Anthony M. Bolger, Marc Lohse, and Bjoern Usadel. Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics*, 30(15):2114–2120, aug 2014.

[48] Richard A. Moore, René L. Warren, J. Douglas Freeman, Julia A. Gustavsen, Caroline Chénard, Jan M. Friedman, Curtis A. Suttle, Yongjun Zhao, and Robert A. Holt. The sensitivity of massively parallel sequencing for detecting candidate infectious agents associated with human tissue. *PLoS ONE*, 6(5), 2011.

[49] Heng Li and Richard Durbin. Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics*, 26(5):589–595, mar 2010.

[50] Alexander Dobin, Carrie A. Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R. Gingeras. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21, jan 2013.

[51] Nuala A. O'Leary, Mathew W. Wright, J. Rodney Brister, Stacy Ciufo, Diana Haddad, Rich McVeigh, Bhanu Rajput, Barbara Robbertse, Brian Smith-White, Danso Ako-Adjei, Alexander Astashyn, Azat Badretdin, Yiming Bao, Olga Blinkova, Vyacheslav Brover, Vyacheslav Chetvernin, Jinna Choi, Eric Cox, Olga Ermolaeva, Catherine M. Farrell, Tamara Goldfarb, Tripti Gupta, Daniel Haft, Eneida Hatcher, Wratko Hlavina, Vinita S. Joardar, Vamsi K. Kodali, Wenjun Li, Donna Maglott, Patrick Masterson, Kelly M. McGarvey, Michael R. Murphy, Kathleen O'Neill, Shashikant Pujar, Sanjida H. Rangwala, Daniel Rausch, Lillian D. Riddick, Conrad Schoch, Andrei Shkeda, Susan S. Storz, Hanzhen Sun, Francoise Thibaud-Nissen, Igor Tolstoy, Raymond E. Tully, Anjana R. Vatsan, Craig Wallin, David Webb, Wendy Wu, Melissa J. Landrum, Avi Kimchi, Tatiana Tatusova, Michael DiCuccio, Paul Kitts, Terence D. Murphy, and Kim D. Pruitt. Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Research*, 44(Database issue):D733, jan 2016.

[52] Luis R Nassar, Galt P Barber, Anna Benet-Pagès, Jonathan Casper, Hiram Clawson, Mark Diekhans, Clay Fischer, Jairo Navarro Gonzalez, Angie S Hinrichs, Brian T Lee, Christopher M Lee, Pranav Muthuraman, Beagan Nguy, Tiana Pereira, Parisa Nejad, Gerardo Perez, Brian J Raney, Daniel Schmelter, Matthew L Speir, Brittney D Wick, Ann S Zweig, David Haussler, Robert M Kuhn, Maximilian Haeussler, and W James Kent. The UCSC Genome Browser database: 2023 update. *Nucleic Acids Research*, 51(D1):D1188–D1195, jan 2023.

[53] Ka Ming Nip, Readman Chiu, Chen Yang, Justin Chu, Hamid Mohamadi, René L. Warren, and Inanc Birol. RNA-Bloom enables reference-free and reference guided sequence assembly for single-cell transcriptomes. *Genome Research*, 30(8):1191–1200, aug 2020.

[54] Yang Liao, Gordon K. Smyth, and Wei Shi. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics*, 30(7):923–930, apr 2014.

[55] Simon Anders, Paul Theodor Pyl, and Wolfgang Huber. HTSeq - a Python framework to work with high-throughput sequencing data. *Bioinformatics*, 31(2):166, jan 2015.

[56] Tom Smith, Andreas Heger, and Ian Sudbery. UMI-tools: Modelling sequencing errors in Unique Molecular Identifiers to improve quantification accuracy. *Genome Research*, 27(3):gr.209601.116, jan 2017.

[57] Swati Parekh, Christoph Ziegenhain, Beate Vieth, Wolfgang Enard, and Ines Hellmann. zUMIs - A fast and flexible pipeline to process RNA sequencing data with UMIs. *GigaScience*, 7(6):1–9, jun 2018.

[58] Tomislav Ilicic, Jong Kyoung Kim, Aleksandra A. Kolodziejczyk, Frederik Otzen Bagger, Davis James McCarthy, John C. Marioni, and Sarah A. Teichmann. Classification of low quality cells from single-cell RNA-seq data. *Genome Biology*, 17(1):29, feb 2016.

[59] Malte D Luecken and Fabian J Theis. Current best practices in single-cell RNA-seq analysis: a tutorial. *Molecular Systems Biology*, 15(6):e8746, jun 2019.

[60] Christopher S. McGinnis, Lyndsay M. Murrow, and Zev J. Gartner. DoubletFinder: Doublet Detection in Single-Cell RNA Sequencing Data Using Artificial Nearest Neighbors. *Cell Systems*, 8(4):329–337.e4, apr 2019.

[61] Caleb Weinreb, Samuel Wolock, and Allon M Klein. SPRING: a kinetic interface for visualizing high dimensional single-cell expression data. *Bioinformatics*, 34(7):1246–1248, apr 2018.

[62] Aaron T.L. Lun, Karsten Bach, and John C. Marioni. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biology*, 17(1):75, apr 2016.

[63] Michael B. Cole, Davide Risso, Allon Wagner, David DeTomaso, John Ngai, Elizabeth Purdom, Sandrine Dudoit, and Nir Yosef. Performance Assessment and Selection of Normalization Procedures for Single-Cell RNA-Seq. *Cell Systems*, 8(4):315–328.e8, apr 2019.

[64] Valentine Svensson, Kedar Nath Natarajan, Lam Ha Ly, Ricardo J. Miragaia, Charlotte Labalette, Iain C. Macaulay, Ana Cvejic, and Sarah A. Teichmann. Power analysis of single-cell rnA-sequencing experiments. *Nature Methods*, 14(4):381–387, mar 2017.

[65] Christian Mayer, Christoph Hafemeister, Rachel C. Bandler, Robert Machold, Renata Batista Brito, Xavier Jaglin, Kathryn Allaway, Andrew Butler, Gord Fishell, and Rahul Satija. Developmental diversification of cortical inhibitory interneurons. *Nature*, 555(7697):457–462, mar 2018.

[66] Andrew Butler, Paul Hoffman, Peter Smibert, Efthymia Papalexi, and Rahul Satija. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature Biotechnology*, 36(5):411–420, jun 2018.

[67] Philip Brennecke, Simon Anders, Jong Kyoung Kim, Aleksandra A. Kołodziejczyk, Xiuwei Zhang, Valentina Proserpio, Bianka Baying, Vladimir Benes, Sarah A. Teichmann, John C. Marioni, and Marcus G. Heisler. Accounting for technical noise in single-cell RNA-seq experiments. *Nature Methods*, 10(11):1093–1098, 2013.

[68] F. Alexander Wolf, Philipp Angerer, and Fabian J. Theis. SCANPY: Large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):15, feb 2018.

[69] Florian Buettner, Kedar N. Natarajan, F. Paolo Casale, Valentina Proserpio, Antonio Scialdone, Fabian J. Theis, Sarah A. Teichmann, John C. Marioni, and Oliver Stegle. Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. *Nature Biotechnology*, 33(2):155–160, jan 2015.

[70] Florian Buettner, Naruemon Pratanwanich, Davis J. McCarthy, John C. Marioni, and Oliver Stegle. f-scLVM: Scalable and versatile factor analysis for single-cell RNA-seq. *Genome Biology*, 18(1):212, nov 2017.

[71] Jase Gehring, Jong Hwee Park, Sisi Chen, Matthew Thomson, and Lior Pachter. Highly multiplexed single-cell RNA-seq by DNA oligonucleotide tagging of cellular proteins. *Nature Biotechnology*, 38(1):35–38, jan 2020.

[72] W. Evan Johnson, Cheng Li, and Ariel Rabinovic. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1):118–127, jan 2007.

[73] Laleh Haghverdi, Aaron T.L. Lun, Michael D. Morgan, and John C. Marioni. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nature Biotechnology*, 36(5):421–427, jun 2018.

[74] Ilya Korsunsky, Nghia Millard, Jean Fan, Kamil Slowikowski, Fan Zhang, Kevin Wei, Yuriy Baglaenko, Michael Brenner, Po ru Loh, and Soumya Raychaudhuri.

Fast, sensitive and accurate integration of single-cell data with Harmony. *Nature Methods*, 16(12):1289–1296, dec 2019.

[75] Yeonjae Ryu, Geun Hee Han, Eunsoo Jung, and Daehee Hwang. Integration of Single-Cell RNA-Seq Datasets: A Review of Computational Methods. *Molecules and Cells*, 46(2):106–119, feb 2023.

[76] Paulo Amaral, Silvia Carbonell-Sala, Francisco M. De La Vega, Tiago Faial, Adam Frankish, Thomas Gingeras, Roderic Guigo, Jennifer L. Harrow, Artemis G. Hatzigeorgiou, Rory Johnson, Terence D. Murphy, Mihaela Pertea, Kim D. Pruitt, Shashikant Pujar, Hazuki Takahashi, Igor Ulitsky, Ales Varabyou, Christine A. Wells, Mark Yandell, Piero Carninci, and Steven L. Salzberg. The status of the human gene catalogue. *Nature*, 622(7981):41–47, oct 2023.

[77] Graham Heimberg, Rajat Bhatnagar, Hana El-Samad, and Matt Thomson. Low-dimensionality in gene expression data enables the accurate extraction of transcriptional programs from shallow sequencing. *Cell systems*, 2(4):239, apr 2016.

[78] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space . *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, nov 1901.

[79] Neo Christopher Chung and John D. Storey. Statistical significance of variables driving systematic variation in high-dimensional data. *Bioinformatics (Oxford, England)*, 31(4):545–554, feb 2015.

[80] Laurens Van Der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[81] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction | https://arxiv.org/abs/1802.03426v2. *arXiv [PREPRINT]*, feb 2018.

[82] Etienne Becht, Leland McInnes, John Healy, Charles Antoine Dutertre, Immanuel W.H. Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W. Newell. Dimensionality reduction for visualizing single-cell data using UMAP. *Nature Biotechnology 2018 37:1*, 37(1):38–44, dec 2018.

[83] J. MacQueen. Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297, 1967.

[84] E. Fix and J.L. Hodges. Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *USAF School of Aviation Medicine, Randolph Field, Texas*, 1951.

[85] Vincent D. Blondel, Jean Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.

[86] Aviv Regev, Sarah A. Teichmann, Eric S. Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, Peter Campbell, Piero Carninci, Menna Clatworthy, Hans Clevers, Bart Deplancke, Ian Dunham, James Eberwine, Roland Eils, Wolfgang Enard, Andrew Farmer, Lars Fugger, Berthold Göttgens, Nir Hacohen, Muzlifah Haniffa, Martin Hemberg, Seung Kim, Paul Klenerman, Arnold Kriegstein, Ed Lein, Sten Linnarsson, Emma Lundberg, Joakim Lundeberg, Partha Majumder, John C. Marioni, Miriam Merad, Musa Mhlanga, Martijn Nawijn, Mihai Netea, Garry Nolan, Dana Pe'er, Anthony Phillipakis, Chris P. Ponting, Stephen Quake, Wolf Reik, Orit Rozenblatt-Rosen, Joshua Sanes, Rahul Satija, Ton N. Schumacher, Alex Shalek, Ehud Shapiro, Padmanee Sharma, Jay W. Shin, Oliver Stegle, Michael Stratton, Michael J.T. Stubbington, Fabian J. Theis, Matthias Uhlen, Alexander Van Oudenaarden, Allon Wagner, Fiona Watt, Jonathan Weissman, Barbara Wold, Ramnik Xavier, and Nir Yosef. The human cell atlas. *eLife*, 6, dec 2017.

[87] Nicole Almanzar, Jane Antony, Ankit S. Baghel, Isaac Bakerman, Ishita Bansal, Ben A. Barres, Philip A. Beachy, Daniela Berdnik, Biter Bilen, Douglas Brownfield, Corey Cain, Charles K.F. Chan, Michelle B. Chen, Michael F. Clarke, Stephanie D. Conley, Spyros Darmanis, Aaron Demers, Kubilay Demir, Antoine de Morree, Tessa Divita, Haley du Bois, Hamid Ebadi, F. Hernán Espinoza, Matt Fish, Qiang Gan, Benson M. George, Astrid Gillich, Rafael Gòmez-Sjöberg, Foad Green, Geraldine Genetiano, Xueying Gu, Gunsagar S. Gulati, Oliver Hahn, Michael Seamus Haney, Yan Hang, Lincoln Harris, Mu He, Shayan Hosseinzadeh, Albin Huang, Kerwyn Casey Huang, Tal Iram, Taichi Isobe, Feather Ives, Robert C C. Jones, Kevin S. Kao, Jim Karkanias, Guruswamy Karnam, Andreas Keller, Aaron M. Kershner, Nathalie Khoury, Seung K. Kim, Bernhard M. Kiss, William Kong, Mark A. Krasnow, Maya E. Kumar, Christin S. Kuo, Jonathan Lam, Davis P. Lee, Song E. Lee, Benoit Lehallier, Olivia Leventhal, Guang Li, Qingyun Li, Ling Liu, Annie Lo, Wan Jin Lu, Maria F. Lugo-Fagundo, Anoop Manjunath, Andrew P. May, Ashley Maynard, Aaron McGeever, Marina McKay, M. Windy McNerney, Bryan Merrill, Ross J. Metzger, Marco Mignardi, Dullei Min, Ahmad N. Nabhan, Norma F. Neff, Katharine M. Ng, Patricia K. Nguyen, Joseph Noh, Roel Nusse, Róbert Pálovics, Rasika Patkar, Weng Chuan Peng, Lolita Penland, Angela Oliveira Pisco, Katherine Pollard, Robert Puccinelli, Zhen Qi, Stephen R. Quake, Thomas A. Rando, Eric J. Rulifson, Nicholas Schaum, Joe M. Segal, Shaheen S. Sikandar, Rahul Sinha, Rene V. Sit, Justin Sonnenburg, Daniel Staehli, Krzysztof Szade, Michelle Tan, Weilun Tan, Cristina Tato, Krissie Tellez, Laughing Bear Torrez Dulgeroff, Kyle J. Travaglini, Carolina Tropini, Margaret Tsui, Lucas Waldburger, Bruce M. Wang, Linda J. van Weele, Kenneth Weinberg, Irving L. Weissman, Michael N. Wosczyna, Sean M. Wu, Tony Wyss-Coray, Jinyi Xiang, Soso Xue, Kevin A. Yamauchi, Andrew C. Yang, Lakshmi P. Yerra, Justin Youngyunpipatkul, Brian Yu, Fabio Zanini, Macy E. Zardeneta, Alexander Zee, Chunyu Zhao, Fan Zhang, Hui Zhang, Martin Jinye Zhang, Lu Zhou, and James Zou. A single-cell transcriptomic atlas characterizes ageing tissues in the mouse.

Nature, 583(7817):590–595, jul 2020.

[88] Vladimir Yu Kiselev, Andrew Yiu, and Martin Hemberg. scmap: projection of single-cell RNA-seq data across data sets. *Nature Methods*, 15(5):359–362, apr 2018.

[89] Hannah A. Pliner, Jay Shendure, and Cole Trapnell. Supervised classification enables rapid annotation of cell atlases. *Nature Methods*, 16(10):983–986, sep 2019.

[90] Koen Van den Berge, Fanny Perraudeau, Charlotte Soneson, Michael I. Love, Davide Risso, Jean Philippe Vert, Mark D. Robinson, Sandrine Dudoit, and Lieven Clement. Observation weights unlock bulk RNA-seq tools for zero inflation and single-cell applications. *Genome Biology*, 19(1):24, feb 2018.

[91] Michael I. Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12), dec 2014.

[92] Mark D. Robinson, Davis J. McCarthy, and Gordon K. Smyth. edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, nov 2009.

[93] Greg Finak, Andrew McDavid, Masanao Yajima, Jingyuan Deng, Vivian Gersuk, Alex K. Shalek, Chloe K. Slichter, Hannah W. Miller, M. Juliana McElrath, Martin Prlic, Peter S. Linsley, and Raphael Gottardo. MAST: A flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. *Genome Biology*, 16(1), dec 2015.

[94] Charity W. Law, Yunshun Chen, Wei Shi, and Gordon K. Smyth. Voom: Precision weights unlock linear model analysis tools for RNA-seq read counts. *Genome Biology*, 15(2):R29, feb 2014.

[95] Matthew E. Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W. Law, Wei Shi, and Gordon K. Smyth. Limma powers differential expression analyses for

RNA-sequencing and microarray studies. *Nucleic Acids Research*, 43(7):e47, jan 2015.

[96] Cole Trapnell, Davide Cacchiarelli, Jonna Grimsby, Prapti Pokharel, Shuqiang Li, Michael Morse, Niall J. Lennon, Kenneth J. Livak, Tarjei S. Mikkelsen, and John L. Rinn. Pseudo-temporal ordering of individual cells reveals dynamics and regulators of cell fate decisions. *Nature biotechnology*, 32(4):381, 2014.

[97] Wouter Saelens, Robrecht Cannoodt, Helena Todorov, and Yvan Saeys. A comparison of single-cell trajectory inference methods. *Nature Biotechnology*, 37(5):547–554, apr 2019.

[98] Sean C. Bendall, Kara L. Davis, El Ad David Amir, Michelle D. Tadmor, Erin F. Simonds, Tiffany J. Chen, Daniel K. Shenfeld, Garry P. Nolan, and Dana Pe'Er. Single-Cell Trajectory Detection Uncovers Progression and Regulatory Coordination in Human B cell Development. *Cell*, 157(3):714, apr 2014.

[99] Kelly Street, Davide Risso, Russell B. Fletcher, Diya Das, John Ngai, Nir Yosef, Elizabeth Purdom, and Sandrine Dudoit. Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. *BMC Genomics*, 19(1):477, jun 2018.

[100] F. Alexander Wolf, Fiona K. Hamey, Mireya Plass, Jordi Solana, Joakim S. Dahlin, Berthold Göttgens, Nikolaus Rajewsky, Lukas Simon, and Fabian J. Theis. PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biology*, 20(1):1–9, mar 2019.

[101] Gioele La Manno, Ruslan Soldatov, Amit Zeisel, Emelie Braun, Hannah Hochgerner, Viktor Petukhov, Katja Lidschreiber, Maria E. Kastriti, Peter Lönnerberg, Alessandro Furlan, Jean Fan, Lars E. Borm, Zehua Liu, David van Bruggen, Jimin Guo, Xiaoling He, Roger Barker, Erik Sundström, Gonçalo Castelo-Branco, Patrick Cramer, Igor Adameyko, Sten Linnarsson, and Peter V. Kharchenko. RNA velocity of single cells. *Nature*, 560(7719):494, aug 2018.

[102] Volker Bergen, Marius Lange, Stefan Peidli, F. Alexander Wolf, and Fabian J. Theis. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nature Biotechnology*, 38(12):1408–1414, aug 2020.

[103] Volker Bergen, Ruslan A Soldatov, Peter V Kharchenko, and Fabian J Theis. RNA velocity - current challenges and future perspectives. *Molecular Systems Biology*, 17(8):e10282, aug 2021.

[104] S. Carbon, H. Dietze, S. E. Lewis, C. J. Mungall, M. C. Munoz-Torres, S. Basu, R. L. Chisholm, R. J. Dodson, P. Fey, Paul D. Thomas, H. Mi, A. Muruganujan, X. Huang, S. Poudel, J. C. Hu, S. A. Aleksander, B. K. McIntosh, D. P. Renfro, D. A. Siegele, G. Antonazzo, H. Attrill, N. H. Brown, S. J. Marygold, P. Mc-Quilton, L. Ponting, G. H. Millburn, A. J. Rey, R. Stefancsik, S. Tweedie, K. Falls, A. J. Schroeder, M. Courtot, D. Osumi-Sutherland, H. Parkinson, P. Roncaglia, R. C. Lovering, R. E. Foulger, R. P. Huntley, P. Denny, N. H. Campbell, B. Kramarz, S. Patel, J. L. Buxton, Z. Umrao, A. T. Deng, H. Alrohaif, K. Mitchell, F. Ratnaraj, W. Omer, M. Rodríguez-López, M. C. Chibucos, M. Giglio, S. Nadendla, M. J. Duesbury, M. Koch, B. H.M. Meldal, A. Melidoni, P. Porras, S. Orchard, A. Shrivastava, H. Y. Chang, R. D. Finn, M. Fraser, A. L. Mitchell, G. Nuka, S. Potter, N. D. Rawlings, L. Richardson, A. Sangrador-Vegas, S. Y. Young, J. A. Blake, K. R. Christie, M. E. Dolan, H. J. Drabkin, D. P. Hill, L. Ni, D. Sitnikov, M. A. Harris, J. Hayles, S. G. Oliver, K. Rutherford, V. Wood, J. Bahler, A. Lock, J. De Pons, M. Dwinell, M. Shimoyama, S. Laulederkind, G. T. Hayman, M. Tutaj, S. J. Wang, P. D'Eustachio, L. Matthews, J. P. Balhoff, R. Balakrishnan, G. Binkley, J. M. Cherry, M. C. Costanzo, S. R. Engel, S. R. Miyasato, R. S. Nash, M. Simison, M. S. Skrzypek, S. Weng, E. D. Wong, M. Feuermann, P. Gaudet, T. Z. Berardini, D. Li, B. Muller, L. Reiser, E. Huala, J. Argasinska, C. Arighi, A. Auchincloss, K. Axelsen, G. Argoud-Puy, A. Bateman, B. Bely, M. C. Blatter, C. Bonilla, L. Bougueleret, E. Boutet, L. Breuza, A. Bridge, R. Britto, H. Hye- A-Bye, C. Casals, E. Cibrian-Uhalte, E. Coudert, I. Cusin, P. Duek-Roggli, A. Es-

treicher, L. Famiglietti, P. Gane, P. Garmiri, G. Georghiou, A. Gos, N. Gruaz-Gumowski, E. Hatton-Ellis, U. Hinz, A. Holmes, C. Hulo, F. Jungo, G. Keller, K. Laiho, P. Lemercier, D. Lieberherr, A. Mac- Dougall, M. Magrane, M. J. Martin, P. Masson, D. A. Natale, C. O'Donovan, I. Pedruzzi, K. Pichler, D. Poggioli, S. Poux, C. Rivoire, B. Roechert, T. Sawford, M. Schneider, E. Speretta, A. Shypitsyna, A. Stutz, S. Sundaram, M. Tognolli, C. Wu, I. Xenarios, L. S. Yeh, J. Chan, S. Gao, K. Howe, R. Kishore, R. Lee, Y. Li, J. Lomax, H. M. Muller, D. Raciti, K. Van Auken, M. Berriman, L. Stein, Paul Kersey, P. W. Sternberg, D. Howe, and M. Westerfield. Expansion of the Gene Ontology knowledgebase and resources. *Nucleic Acids Research*, 45(Database issue):D331, jan 2017.

[105] Minoru Kanehisa, Miho Furumichi, Mao Tanabe, Yoko Sato, and Kanae Morishima. KEGG: new perspectives on genomes, pathways, diseases and drugs. *Nucleic Acids Research*, 45(Database issue):D353, jan 2017.

[106] Antonio Fabregat, Steven Jupe, Lisa Matthews, Konstantinos Sidiropoulos, Marc Gillespie, Phani Garapati, Robin Haw, Bijay Jassal, Florian Korninger, Bruce May, Marija Milacic, Corina Duenas Roca, Karen Rothfels, Cristoffer Sevilla, Veronica Shamovsky, Solomon Shorser, Thawfeek Varusai, Guilherme Viteri, Joel Weiser, Guanming Wu, Lincoln Stein, Henning Hermjakob, and Peter D'Eustachio. The Reactome Pathway Knowledgebase. *Nucleic Acids Research*, 46(Database issue):D649, jan 2018.

[107] Roser Vento-Tormo, Mirjana Efremova, Rachel A. Botting, Margherita Y. Turco, Miquel Vento-Tormo, Kerstin B. Meyer, Jong Eun Park, Emily Stephenson, Krzysztof Polański, Angela Goncalves, Lucy Gardner, Staffan Holmqvist, Johan Henriksson, Angela Zou, Andrew M. Sharkey, Ben Millar, Barbara Innes, Laura Wood, Anna Wilbrey-Clark, Rebecca P. Payne, Martin A. Ivarsson, Steve Lisgo, Andrew Filby, David H. Rowitch, Judith N. Bulmer, Gavin J. Wright, Michael J.T. Stubbington, Muzlifah Haniffa, Ashley Moffett, and Sarah A. Teich-

mann. Single-cell reconstruction of the early maternal-fetal interface in humans. *Nature*, 563(7731):347–353, nov 2018.

[108] Method of the Year 2013. *Nature Methods*, 11(1):1–1, dec 2013.

[109] Mireya Plass, Jordi Solana, F. Alexander Wolf, Salah Ayoub, Aristotelis Misios, Petar Glažar, Benedikt Obermayer, Fabian J. Theis, Christine Kocks, and Nikolaus Rajewsky. Cell type atlas and lineage tree of a whole complex animal by single-cell transcriptomics. *Science*, 360(6391), may 2018.

[110] Daniel E. Wagner, Caleb Weinreb, Zach M. Collins, James A. Briggs, Sean G. Megason, and Allon M. Klein. Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. *Science*, 360(6392):981, jun 2018.

[111] James A. Briggs, Caleb Weinreb, Daniel E. Wagner, Sean Megason, Leonid Peshkin, Marc W. Kirschner, and Allon M. Klein. The dynamics of gene expression in vertebrate embryogenesis at single-cell resolution. *Science*, 360(6392), jun 2018.

[112] Fengying Wu, Jue Fan, Yayi He, Anwen Xiong, Jia Yu, Yixin Li, Yan Zhang, Wencheng Zhao, Fei Zhou, Wei Li, Jie Zhang, Xiaosheng Zhang, Meng Qiao, Guanghui Gao, Shanhao Chen, Xiaoxia Chen, Xuefei Li, Likun Hou, Chunyan Wu, Chunxia Su, Shengxiang Ren, Margarete Odenthal, Reinhard Buettner, Nan Fang, and Caicun Zhou. Single-cell profiling of tumor heterogeneity and the microenvironment in advanced non-small cell lung cancer. *Nature Communications*, 12(1):1–11, may 2021.

[113] Bram Van de Sande, Joon Sang Lee, Euphemia Mutasa-Gottgens, Bart Naughton, Wendi Bacon, Jonathan Manning, Yong Wang, Jack Pollard, Melissa Mendez, Jon Hill, Namit Kumar, Xiaohong Cao, Xiao Chen, Mugdha Khaladkar, Ji Wen, Andrew Leach, and Edgardo Ferran. Applications of single-cell RNA sequencing in drug discovery and development. *Nature Reviews Drug Discovery*, 22(6):496–520, apr 2023.

[114] Kyungtae Lim, Alex P.A. Donovan, Walfred Tang, Dawei Sun, Peng He, J. Patrick Pett, Sarah A. Teichmann, John C. Marioni, Kerstin B. Meyer, Andrea H. Brand, and Emma L. Rawlins. Organoid modeling of human fetal lung alveolar development reveals mechanisms of cell fate patterning and neonatal respiratory disease. *Cell Stem Cell*, 30(1):20–37.e9, jan 2023.

[115] Valentine Svensson, Eduardo da Veiga Beltrame, and Lior Pachter. A curated database reveals trends in single-cell transcriptomics. *Database*, 2020, 2020.

[116] Dario Romagnoli, Giulia Boccalini, Martina Bonechi, Chiara Biagioni, Paola Fassan, Roberto Bertorelli, Veronica De Sanctis, Angelo Di Leo, Ilenia Migliaccio, Luca Malorni, and Matteo Benelli. DdSeeker: A tool for processing Bio-Rad ddSEQ single cell RNA-seq data. *BMC Genomics*, 19(1):960, dec 2018.

[117] Benjamin Kaminow, Dinar Yunusov, and Alexander Dobin. STARsolo: accurate, fast and versatile mapping/quantification of single-cell and single-nucleus RNA-seq data. *bioRxiv*, page 2021.05.05.442755, may 2021.

[118] Di Feng, Charles E. Whitehurst, Dechao Shan, Jon D. Hill, and Yong G. Yue. Single Cell Explorer, collaboration-driven tools to leverage large-scale single cell RNA-seq data. *BMC Genomics*, 20(1):676, aug 2019.

[119] Xun Zhu, Thomas K. Wolfgruber, Austin Tasato, Cédric Arisdakessian, David G. Garmire, and Lana X. Garmire. Granatum: A graphical single-cell RNA-Seq analysis pipeline for genomics scientists. *Genome Medicine*, 9(1):108, dec 2017.

[120] Fabrice P.A. David, Maria Litovchenko, Bart Deplancke, and Vincent Gardeux. ASAP 2020 update: An open, scalable and interactive web-based portal for (single-cell) omics analyses. *Nucleic Acids Research*, 48(W1):W403–W414, jul 2021.

[121] Grace X.Y. Zheng, Jessica M. Terry, Phillip Belgrader, Paul Ryvkin, Zachary W. Bent, Ryan Wilson, Solongo B. Ziraldo, Tobias D. Wheeler, Geoff P. McDermott, Junjie Zhu, Mark T. Gregory, Joe Shuga, Luz Montesclaros, Jason G. Underwood,

Donald A. Masquelier, Stefanie Y. Nishimura, Michael Schnall-Levin, Paul W. Wyatt, Christopher M. Hindson, Rajiv Bharadwaj, Alexander Wong, Kevin D. Ness, Lan W. Beppu, H. Joachim Deeg, Christopher McFarland, Keith R. Loeb, William J. Valente, Nolan G. Ericson, Emily A. Stevens, Jerald P. Radich, Tarjei S. Mikkelsen, Benjamin J. Hindson, and Jason H. Bielas. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8(1):1–12, jan 2017.

[122] Zachary D. Stephens, Skylar Y. Lee, Faraz Faghri, Roy H. Campbell, Chengxiang Zhai, Miles J. Efron, Ravishankar Iyer, Michael C. Schatz, Saurabh Sinha, and Gene E. Robinson. Big Data: Astronomical or Genomical? *PLoS Biology*, 13(7), 2015.

[123] Vishal H. Oza, Jordan H. Whitlock, Elizabeth J. Wilk, Angelina Uno-Antonison, Brandon Wilk, Manavalan Gajapathy, Timothy C. Howton, Austyn Trull, Lara Ianov, Elizabeth A. Worthey, and Brittany N. Lasseigne. Ten simple rules for using public biological data for your research. *PLOS Computational Biology*, 19(1):e1010749, jan 2023.

[124] Dennis A. Benson, Mark Cavanaugh, Karen Clark, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and Eric W. Sayers. GenBank. *Nucleic Acids Research*, 41(D1):D36–D42, jan 2013.

[125] Tanya Barrett, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Michelle Holko, Andrey Yefanov, Hyeseung Lee, Naigong Zhang, Cynthia L. Robertson, Nadezhda Serova, Sean Davis, and Alexandra Soboleva. NCBI GEO: archive for functional genomics data sets - update. *Nucleic Acids Research*, 41(Database issue):D991, jan 2013.

[126] Juan Antonio Vizcaíno, Florian Reisinger, Richard Côté, and Lennart Martens. PRIDE: Data submission and analysis. *Current protocols in protein science*, Chapter

25, apr 2010.

[127] Ethan Cerami, Jianjiong Gao, Ugur Dogrusoz, Benjamin E. Gross, Selcuk Onur Sumer, Bülent Arman Aksoy, Anders Jacobsen, Caitlin J. Byrne, Michael L. Heuer, Erik Larsson, Yevgeniy Antipin, Boris Reva, Arthur P. Goldberg, Chris Sander, and Nikolaus Schultz. The cBio Cancer Genomics Portal: An Open Platform for Exploring Multidimensional Cancer Genomics Data. *Cancer Discovery*, 2(5):401–404, may 2012.

[128] Saumyendra N. Basu, Ravi Kollu, and Sharmila Banerjee-Basu. AutDB: a gene reference resource for autism research. *Nucleic Acids Research*, 37(Database issue), 2009.

[129] Gerda Cristal Villalba and Ursula Matte. Fantastic databases and where to find them: Web applications for researchers in a rush. *Genetics and Molecular Biology*, 44(2):20200203, 2021.

[130] Zenodo - https://zenodo.org/, 2023.

[131] FigShare https://figshare.com/, 2023.

[132] Damien Lecarpentier, Peter Wittenburg, Willem Elbers, Alberto Michelini, Riam Kanso, Peter Coveney, and Rob Baxter. EUDAT: A New Cross-Disciplinary Data Infrastructure for Science. *International Journal of Digital Curation*, 8(1):279–287, jun 2013.

[133] Jon Ison, Kristoffer Rapacki, Hervé Ménager, Matúš Kalaš, Emil Rydza, Piotr Chmura, Christian Anthon, Niall Beard, Karel Berka, Dan Bolser, Tim Booth, Anthony Bretaudeau, Jan Brezovsky, Rita Casadio, Gianni Cesareni, Frederik Coppens, Michael Cornell, Gianmauro Cuccuru, Kristian Davidsen, Gianluca Della Vedova, Tunca Dogan, Olivia Doppelt-Azeroual, Laura Emery, Elisabeth Gasteiger, Thomas Gatter, Tatyana Goldberg, Marie Grosjean, Björn Gruüing, Manuela Helmer-Citterich, Hans Ienasescu, Vassilios Ioannidis, Martin Closter

Jespersen, Rafael Jimenez, Nick Juty, Peter Juvan, Maximilian Koch, Camille Laibe, Jing Woei Li, Luana Licata, Fabien Mareuil, Ivan Mičetić, Rune Møllegaard Friborg, Sebastien Moretti, Chris Morris, Steffen Möller, Aleksandra Nenadic, Hedi Peterson, Giuseppe Profiti, Peter Rice, Paolo Romano, Paola Roncaglia, Rabie Saidi, Andrea Schafferhans, Veit Schwämmle, Callum Smith, Maria Maddalena Sperotto, Heinz Stockinger, Radka Svobodová Varěková, Silvio C.E. Tosatto, Victor De La Torre, Paolo Uva, Allegra Via, Guy Yachdav, Federico Zambelli, Gert Vriend, Burkhard Rost, Helen Parkinson, Peter Løngreen, and Søren Brunak. Tools and data services registry: A community effort to document bioinformatics resources. *Nucleic Acids Research*, 44(D1):D38–D47, jan 2016.

[134] Mihai Glont, Tung V.N. Nguyen, Martin Graesslin, Robert Hälke, Raza Ali, Jochen Schramm, Sarala M. Wimalaratne, Varun B. Kothamachu, Nicolas Rodriguez, Maclej J. Swat, Jurgen Eils, Roland Eils, Camille Laibe, Rahuman S. Malik-Sheriff, Vijayalakshmi Chelliah, Nicolas Le Novère, and Henning Hermjakob. BioModels: Expanding horizons to include more modelling approaches and formats. *Nucleic Acids Research*, 46(D1):D1248–D1253, jan 2018.

[135] Philip A. Ewels, Alexander Peltzer, Sven Fillinger, Harshil Patel, Johannes Alneberg, Andreas Wilm, Maxime Ulysse Garcia, Paolo Di Tommaso, and Sven Nahnsen. The nf-core framework for community-curated bioinformatics pipelines. *Nature Biotechnology*, 38(3):276–278, mar 2020.

[136] Matthias Scheffler, Martin Aeschlimann, Martin Albrecht, Tristan Bereau, Hans Joachim Bungartz, Claudia Felser, Mark Greiner, Axel Groß, Christoph T. Koch, Kurt Kremer, Wolfgang E. Nagel, Markus Scheidgen, Christof Wöll, and Claudia Draxl. FAIR data enabling new horizons for materials research. *Nature*, 604(7907):635–642, apr 2022.

[137] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J.

Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C. t Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan Van Der Lei, Erik Van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. Comment: The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3(1):1–9, mar 2016.

[138] Katherine Wolstencroft, Stuart Owen, Olga Krebs, Quyen Nguyen, Natalie J. Stanford, Martin Golebiewski, Andreas Weidemann, Meik Bittkowski, Lihua An, David Shockley, Jacky L. Snoep, Wolfgang Mueller, and Carole Goble. SEEK: A systems biology data and model management platform. *BMC Systems Biology*, 9(1):33, jul 2015.

[139] Ludwig Lautenbacher, Patroklos Samaras, Julian Muller, Andreas Grafberger, Marwin Shraideh, Johannes Rank, Simon T. Fuchs, Tobias K. Schmidt, Matthew The, Christian Dallago, Holger Wittges, Burkhard Rost, Helmut Krcmar, Bernhard Kuster, and Mathias Wilhelm. ProteomicsDB: Toward a FAIR open-source resource for life-science research. *Nucleic Acids Research*, 50(D1):D1541–D1552, jan 2022.

[140] Mathieu Servillat, Catherine Boisson, Matthias Fuessling, and Bruno Khelifi. FAIR high level data for Cherenkov astronomy. *arXiv [PREPRINT]*, jan 2022.

[141] Philipp Pugliese, Christian Knell, and Jan Christoph. Exchange of Clinical and Omics Data According to FAIR Principles: A Review of Open Source Solutions. *Methods of Information in Medicine*, 59(S 01):e13–e20, jun 2020.

[142] Belinda Giardine, Cathy Riemer, Ross C. Hardison, Richard Burhans, Laura Elnitski, Prachi Shah, Yi Zhang, Daniel Blankenberg, Istvan Albert, James Taylor, Webb Miller, W. James Kent, and Anton Nekrutenko. Galaxy: A platform for interactive large-scale genome analysis. *Genome Research*, 15(10):1451–1455, oct 2005.

[143] Enis Afgan, Anton Nekrutenko, Bjórn A. Grüning, Daniel Blankenberg, Jeremy Goecks, Michael C. Schatz, Alexander E. Ostrovsky, Alexandru Mahmoud, Andrew J. Lonie, Anna Syme, Anne Fouilloux, Anthony Bretaudeau, Anton Nekrutenko, Anup Kumar, Arthur C. Eschenlauer, Assunta D. Desanto, Aysam Guerler, Beatriz Serrano-Solano, Bérénice Batut, Björn A. Grüning, Bradley W. Langhorst, Bridget Carr, Bryan A. Raubenolt, Cameron J. Hyde, Catherine J. Bromhead, Christopher B. Barnett, Coline Royaux, Cristóbal Gallardo, Daniel Blankenberg, Daniel J. Fornika, Dannon Baker, Dave Bouvier, Dave Clements, David A. De Lima Morais, D. L. Tabernero, Delphine Lariviere, Engy Nasr, Enis Afgan, Federico Zambelli, Florian Heyl, Fotis Psomopoulos, Frederik Coppens, Gareth R. Price, Gianmauro Cuccuru, Gildas Le Corguillé, Greg Von Kuster, Gulsum Gudukbay Akbulut, Helena Rasche, Hotz Hans-Rudolf, Ignacio Eguinoa, Igor Makunin, Isuru J. Ranawaka, James P. Taylor, Jayadev Joshi, Jennifer Hillman-Jackson, John M. Chilton, Kaivan Kamali, Keith Suderman, Krzysztof Poterlowicz, Le Bras Yvan, Lucille Lopez-Delisle, Luke Sargent, Madeline E. Bassetti, Marco Antonio Tangaro, Marius Van Den Beek, Martin Cech, Matthias Bernt, Matthias Fahrner, Mehmet Tekman, Melanie C. Föll, Michael C. Schatz, Michael R. Crusoe, Miguel Roncoroni, Natalie Kucher, Nate Coraor, Nicholas Stoler, Nick Rhodes, Nicola Soranzo, Niko Pinter, Nuwan A. Goonasekera, Pablo A. Moreno, Pavankumar Videm, Petera Melanie, Pietro Mandreoli, Pratik D. Jagtap, Qiang Gu, Ralf J.M. Weber, Ross Lazarus, Ruben H.P. Vorderman, Saskia Hiltemann, Sergey Golitsynskiy, Shilpa Garg, Simon A. Bray, Simon L. Gladman, Simone Leo, Subina P. Mehta, Timothy J. Griffin, Vahid Jalili, Vandenbrouck Yves, Victor Wen, Vijay K. Nagampalli, Wendi A. Bacon, Willem

De Koning, Wolfgang Maier, and Peter J. Briggs. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2022 update. *Nucleic Acids Research*, 50(W1):W345–W351, jul 2022.

[144] Adam Rauch, Matthew Bellew, Jimmy Eng, Matthew Fitzgibbon, Ted Holzman, Peter Hussey, Mark Igra, Brendan Maclean, Chen Wei Lin, Andrea Detter, Ruihua Fang, Vitor Faca, Phil Gafken, Heidi Zhang, Jeffrey Whitaker, David States, Sam Hanash, Amanda Paulovich, and Martin W. McIntosh. Computational proteomics analysis system (CPAS): An extensible, open-source analytic system for evaluating and publishing proteomic data and high throughput biological experiments. *Journal of Proteome Research*, 5(1):112–121, jan 2006.

[145] Elizabeth K. Nelson, Britt Piehler, Josh Eckels, Adam Rauch, Matthew Bellew, Peter Hussey, Sarah Ramsay, Cory Nathe, Karl Lum, Kevin Krouse, David Stearns, Brian Connolly, Tom Skillman, and Mark Igra. LabKey Server: An open source platform for scientific data integration, analysis and collaboration. *BMC Bioinformatics*, 12(1):71, mar 2011.

[146] Angela Bauch, Izabela Adamczyk, Piotr Buczek, Franz Josef Elmer, Kaloyan Enimanev, Pawel Glyzewski, Manuel Kohler, Tomasz Pylak, Andreas Quandt, Chandrasekhar Ramakrishnan, Christian Beisel, Lars Malmström, Ruedi Aebersold, and Bernd Rinn. OpenBIS: A flexible framework for managing and analyzing complex data in biology research. *BMC Bioinformatics*, 12(1):468, dec 2011.

[147] Caterina Barillari, Diana S. M. Ottoz, Juan Mariano Fuentes-Serna, Chandrasekhar Ramakrishnan, Bernd Rinn, and Fabian Rudolf. openBIS ELN-LIMS: an open-source database for academic laboratories. *Bioinformatics*, 32(4):638–640, feb 2016.

[148] Jelle Scholtalbers, Jasmin Rößler, Patrick Sorn, Jos de Graaf, Valesca Boisguérin, John Castle, and Ugur Sahin. Galaxy LIMS for next-generation sequencing. *Bioinformatics*, 29(9):1233–1234, 2013.

[149] Alexander Senf, Robert Davies, Frédéric Haziza, John Marshall, Juan Troncoso-Pastoriza, Oliver Hofmann, and Thomas M. Keane. Crypt4GH: a file format standard enabling native access to encrypted data. *Bioinformatics*, 37(17):2753–2754, sep 2021.

[150] Fabienne Thelen, Jannis Hochmuth, Sven Griep, Benedikt Schwab, Alexander Goesmann, and Frank Förster. Crypt4GH-JS: securely storing sensitive data online with client-side encryption. *Bioinformatics*, 41(1), dec 2024.

[151] Andreas Hoek, Katharina Maibach, Ebru Özmen, Ana Ivonne Vazquez-Armendariz, Jan Philipp Mengel, Torsten Hain, Susanne Herold, and Alexander Goesmann. WASP: a versatile, web-accessible single cell RNA-Seq processing platform. *BMC Genomics*, 22(1):1–11, dec 2021.

[152] Bjørn Fjukstad and Lars Ailo Bongo. A Review of Scalable Bioinformatics Pipelines. *Data Science and Engineering*, 2(3):245–251, sep 2017.

[153] Azza E. Ahmed, Joshua M. Allen, Tajesvi Bhat, Prakruthi Burra, Christina E. Fliege, Steven N. Hart, Jacob R. Heldenbrand, Matthew E. Hudson, Dave Deandre Istanto, Michael T. Kalmbach, Gregory D. Kapraun, Katherine I. Kendig, Matthew Charles Kendzior, Eric W. Klee, Nate Mattson, Christian A. Ross, Sami M. Sharif, Ramshankar Venkatakrishnan, Faisal M. Fadlelmola, and Liudmila S. Mainzer. Design considerations for workflow management systems use in production genomics research and the clinic. *Scientific Reports*, 11(1):1–18, nov 2021.

[154] Paolo Di Tommaso, Maria Chatzou, Evan W. Floden, Pablo Prieto Barja, Emilio Palumbo, and Cedric Notredame. Nextflow enables reproducible computational workflows. *Nature Biotechnology*, 35(4):316–319, apr 2017.

[155] Johannes Köster and Sven Rahmann. Snakemake - a scalable bioinformatics workflow engine. *Bioinformatics*, 28(19):2520–2522, oct 2012.

[156] Wolfgang Gentzsch. Sun Grid Engine: Towards creating a compute power grid. In *Proceedings - 1st IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGrid 2001*, pages 35–36, 2001.

[157] Andy B. Yoo, Morris A. Jette, and Mark Grondona. SLURM: Simple Linux Utility for Resource Management. In *Lecture Notes in Computer Science*, volume 2862, pages 44–60. Springer Verlag, 2003.

[158] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, and Richard Durbin. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078, aug 2009.

[159] Robert A. Amezquita, Aaron T.L. Lun, Etienne Becht, Vince J. Carey, Lindsay N. Carpp, Ludwig Geistlinger, Federico Marini, Kevin Rue-Albrecht, Davide Risso, Charlotte Soneson, Levi Waldron, Hervé Pagès, Mike L. Smith, Wolfgang Huber, Martin Morgan, Raphael Gottardo, and Stephanie C. Hicks. Orchestrating single-cell analysis with Bioconductor. *Nature Methods*, 17(2):137–145, feb 2020.

[160] Wes Felter, Alexandre Ferreira, Ram Rajamony, and Juan Rubio. An updated performance comparison of virtual machines and Linux containers. In *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 171–172. IEEE, mar 2015.

[161] Jonas Kupschus, Stefan Janssen, Andreas Hoek, Jan Kuska, Jonathan Rathjens, Carsten Sonntag, Katja Ickstadt, Lisa Budzinski, Hyun Dong Chang, Andrea Rossi, Charlotte Esser, and Katrin Hochrath. Rapid detection and online analysis of microbial changes through flow cytometry. *Cytometry Part A*, 103(5):419–428, may 2023.

[162] Ilker Polatoğlu, Tulay Oncu-Oner, Irem Dalman, and Senanur Ozdogan. COVID-19 in early 2023: Structure, replication mechanism, variants of SARS-CoV-2, diagnostic tests, and vaccine drug development studies. *MedComm*, 4(2), apr 2023.

[163] Ali M. Zaki, Sander van Boheemen, Theo M. Bestebroer, Albert D.M.E. Osterhaus, and Ron A.M. Fouchier. Isolation of a Novel Coronavirus from a Man with Pneumonia in Saudi Arabia. *New England Journal of Medicine*, 367(19):1814–1820, nov 2012.

[164] Ron Eccles. Understanding the symptoms of the common cold and influenza. *The Lancet. Infectious Diseases*, 5(11):718, nov 2005.

[165] David M. Morens, Jeffery K. Taubenberger, and Anthony S. Fauci. Predominant role of bacterial pneumonia as a cause of death in pandemic influenza: Implications for pandemic influenza preparedness. *Journal of Infectious Diseases*, 198(7):962–970, oct 2008.

[166] Laura Dwyer-Lindgren, Amelia Bertozzi-Villa, Rebecca W. Stubbs, Chloe Morozoff, Shreya Shirude, Mohsen Naghavi, Ali H. Mokdad, and Christopher J.L. Murray. Trends and patterns of differences in chronic respiratory disease mortality among US counties, 1980-2014. *JAMA - Journal of the American Medical Association*, 318(12):1136–1149, sep 2017.

[167] Jamal S. Rana, Sadiya S. Khan, Donald M. Lloyd-Jones, and Stephen Sidney. Changes in Mortality in Top 10 Causes of Death from 2011 to 2018. *Journal of General Internal Medicine*, 36(8):2517–2518, aug 2021.

[168] Veronika Bosáková, Marco De Zuani, Lucie Sládková, Zuzana Garlíková, Shyam Sushama Jose, Teresa Zelante, Marcela Hortová Kohoutková, and Jan Frič. Lung Organoids - The Ultimate Tool to Dissect Pulmonary Diseases? *Frontiers in Cell and Developmental Biology*, 10:899368, jul 2022.

[169] Brian Cunniff, Joseph E. Druso, and Jos L. van der Velden. Lung organoids: advances in generation and 3D-visualization. *Histochemistry and Cell Biology*, 155(2):301–308, feb 2021.

[170] Georg A. Busslinger, Bas L.A. Weusten, Auke Bogte, Harry Begthel, Lodewijk A.A. Brosens, and Hans Clevers. Human gastrointestinal epithelia of

the esophagus, stomach, and duodenum resolved at single-cell resolution. *Cell Reports*, 34(10), mar 2021.

[171] Prakash Ramachandran, Kylie P. Matchett, Ross Dobie, John R. Wilson-Kanamori, and Neil C. Henderson. Single-cell technologies in hepatology: new insights into liver biology and disease pathogenesis. *Nature Reviews Gastroenterology and Hepatology*, 17(8):457–472, aug 2020.

[172] Kai Sen Tan, Anand Kumar Andiappan, Bernett Lee, Yan Yan, Jing Liu, See Aik Tang, Josephine Lum, Ting Ting He, Yew Kwang Ong, Mark Thong, Hui Fang Lim, Hyung Won Choi, Olaf Rotzschke, Vincent T Chow, and De Yun Wang. RNA Sequencing of H3N2 Influenza Virus-Infected Human Nasal Epithelial Cells from Multiple Subjects Reveals Molecular Pathways Associated with Tissue Injury and Complications. *Cells*, 8(9):986, aug 2019.

[173] Na Zhu, Dingyu Zhang, Wenling Wang, Xingwang Li, Bo Yang, Jingdong Song, Xiang Zhao, Baoying Huang, Weifeng Shi, Roujian Lu, Peihua Niu, Faxian Zhan, Xuejun Ma, Dayan Wang, Wenbo Xu, Guizhen Wu, George F. Gao, and Wenjie Tan. A Novel Coronavirus from Patients with Pneumonia in China, 2019. *New England Journal of Medicine*, 382(8):727–733, feb 2020.

[174] Moumita Ghosh, York E. Miller, Ichiro Nakachi, Jennifer B. Kwon, Anna E. Barón, Alexandra E. Brantley, Daniel T. Merrick, Wilbur A. Franklin, Robert L. Keith, and R. William Vandivier. Exhaustion of airway basal progenitor cells in early and established chronic obstructive pulmonary disease. *American Journal of Respiratory and Critical Care Medicine*, 197(7):885–896, apr 2018.

[175] Benjamin David, Mona Bafadhel, Leo Koenderman, and Antony De Soyza. Eosinophilic inflammation in COPD: From an inflammatory marker to a treatable trait. *Thorax*, 76(2):188–195, feb 2021.

[176] Chiel van Geffen, Astrid Deißler, Markus Quante, Harald Renz, Dominik Hartl, and Saeed Kolahian. Regulatory Immune Cells in Idiopathic Pulmonary Fibrosis: Friends or Foes? *Frontiers in Immunology*, 12, apr 2021.

[177] Menno Tamminga, Thijo Jeroen N Hiltermann, Ed Schuuring, Wim Timens, Rudolf SN Fehrmann, and Harry JM Groen. Immune microenvironment composition in non-small cell lung cancer and its association with survival. *Clinical Translational Immunology*, 9(6):e1142, jan 2020.

[178] Meritxell Huch and Bon Kyoung Koo. Modeling mouse and human development using organoid cultures. *Development*, 142(18):3113–3125, sep 2015.

[179] Toshiro Sato, Daniel E. Stange, Marc Ferrante, Robert G.J. Vries, Johan H. Van Es, Stieneke Van Den Brink, Winan J. Van Houdt, Apollo Pronk, Joost Van Gorp, Peter D. Siersema, and Hans Clevers. Long-term expansion of epithelial organoids from human colon, adenoma, adenocarcinoma, and Barrett's epithelium. *Gastroenterology*, 141(5):1762–1772, nov 2011.

[180] Sina Bartfeld, Tülay Bayram, Marc Van De Wetering, Meritxell Huch, Harry Begthel, Pekka Kujala, Robert Vries, Peter J. Peters, and Hans Clevers. In vitro expansion of human gastric epithelial stem cells and their responses to bacterial infection. *Gastroenterology*, 148(1):126–136.e6, jan 2015.

[181] Laura Broutier, Amanda Andersson-Rolf, Christopher J. Hindley, Sylvia F. Boj, Hans Clevers, Bon Kyoung Koo, and Meritxell Huch. Culture and establishment of self-renewing human and mouse adult liver and pancreas 3D organoids and their genetic manipulation. *Nature Protocols*, 11(9):1724–1743, sep 2016.

[182] Norman Sachs, Angelos Papaspyropoulos, Domenique D Zomer-van Ommen, Inha Heo, Lena Böttinger, Dymph Klay, Fleur Weeber, Guizela Huelsz-Prince, Nino Iakobachvili, Gimano D Amatngalim, Joep de Ligt, Arne van Hoeck, Natalie Proost, Marco C Viveen, Anna Lyubimova, Luc Teeven, Sepideh Derakhshan, Jeroen Korving, Harry Begthel, Johanna F Dekkers, Kuldeep Kumawat,

Emilio Ramos, Matthijs FM van Oosterhout, G Johan Offerhaus, Dominique J Wiener, Eduardo P Olimpio, Krijn K Dijkstra, Egbert F Smit, Maarten van der Linden, Sridevi Jaksani, Marieke van de Ven, Jos Jonkers, Anne C Rios, Emile E Voest, Coline HM van Moorsel, Cornelis K van der Ent, Edwin Cuppen, Alexander van Oudenaarden, Frank E Coenjaerts, Linde Meyaard, Louis J Bont, Peter J Peters, Sander J Tans, Jeroen S van Zon, Sylvia F Boj, Robert G Vries, Jeffrey M Beekman, and Hans Clevers. Long-term expanding human airway organoids for disease modeling. *The EMBO Journal*, 38(4):e100300, feb 2019.

[183] Pieter S. Hiemstra and Arnaud Bourdin. Club cells, CC10 and self-control at the epithelial surface. *European Respiratory Journal*, 44(4):831–832, oct 2014.

[184] C. Bezençon, A. Fürholz, F. Raymond, R. Mansourian, S. Métairon, J. Le Coutre, and Sami Damak. Murine intestinal cells expressing Trpm5 are mostly brush cells and express markers of neuronal and inflammatory cells. *Journal of Comparative Neurology*, 509(5):514–525, aug 2008.

[185] Jordan D. Davis and Tomasz P. Wypych. Cellular and functional heterogeneity of the airway epithelium. *Mucosal Immunology*, 14(5):978–990, feb 2021.

[186] Ernest Cutz, Herman Yeger, and Jie Pan. Pulmonary Neuroendocrine Cell System in Pediatric Lung Disease - Recent Advances. *Pediatric and Developmental Pathology*, 10(6):419–435, nov 2007.

[187] Yanjie Wang, Zan Tang, Huanwei Huang, Jiao Li, Zheng Wang, Yuanyuan Yu, Chengwei Zhang, Juan Li, Huaping Dai, Fengchao Wang, Tao Cai, and Nan Tang. Pulmonary alveolar type I cell population consists of two distinct subtypes that differ in cell fate. *Proceedings of the National Academy of Sciences of the United States of America*, 115(10):2407–2412, mar 2018.

[188] Anna Demchenko, Alexander Lavrov, and Svetlana Smirnikhina. Lung organoids: current strategies for generation and transplantation. *Cell and Tissue Research*, 390(3):317–333, dec 2022.

[189] Lei Ye, Cory Swingen, and Jianyi Zhang. Induced Pluripotent Stem Cells and Their Potential for Basic and Clinical Sciences. *Current Cardiology Reviews*, 9(1):63, feb 2013.

[190] Chris S. Hughes, Lynne M. Postovit, and Gilles A. Lajoie. Matrigel: A complex protein mixture required for optimal growth of cell culture. *Proteomics*, 10(9):1886–1890, may 2010.

[191] Jason R. Rock, Mark W. Onaitis, Emma L. Rawlins, Yun Lu, Cheryl P. Clark, Yan Xue, Scott H. Randell, and Brigid L.M. Hogan. Basal cells as stem cells of the mouse trachea and human airway epithelium. *Proceedings of the National Academy of Sciences of the United States of America*, 106(31):12771–12775, aug 2009.

[192] Huaiyong Chen, Keitaro Matsumoto, Brian L. Brockway, Craig R. Rackley, Jiurong Liang, Joo Hyeon Lee, Dianhua Jiang, Paul W. Noble, Scott H. Randell, Carla F. Kim, and Barry R. Stripp. Airway epithelial progenitors are region specific and show differential responses to bleomycin-induced lung injury. *Stem Cells*, 30(9):1948–1960, sep 2012.

[193] Christina E. Barkauskas, Michael J. Cronce, Craig R. Rackley, Emily J. Bowie, Douglas R. Keene, Barry R. Stripp, Scott H. Randell, Paul W. Noble, and Brigid L.M. Hogan. Type 2 alveolar cells are stem cells in adult lung. *Journal of Clinical Investigation*, 123(7):3025–3036, jul 2013.

[194] Joo Hyeon Lee, Dong Ha Bhang, Alexander Beede, Tian Lian Huang, Barry R. Stripp, Kenneth D. Bloch, Amy J. Wagers, Yu Hua Tseng, Sandra Ryeom, and Carla F. Kim. Lung stem cell differentiation in mice directed by endothelial cells via a BMP4-NFATc1-thrombospondin-1 axis. *Cell*, 156(3):440–455, jan 2014.

[195] Ana Ivonne Vazquez-Armendariz, Monika Heiner, Elie El Agha, Isabelle Salwig, Andreas Hoek, Marie Christin Hessler, Irina Shalashova, Amit Shrestha, Gianni Carraro, Jan Philip Mengel, Andreas Günther, Rory Edward Morty, István

Vadász, Martin Schwemmle, Wolfgang Kummer, Torsten Hain, Alexander Goesmann, Saverio Bellusci, Werner Seeger, Thomas Braun, and Susanne Herold. Multilineage murine stem cells generate complex organoids to model distal lung development and disease. *The EMBO Journal*, 39(21), nov 2020.

[196] Isabelle Salwig, Birgit Spitznagel, Ana Ivonne Vazquez-Armendariz, Keynoosh Khalooghi, Stefan Guenther, Susanne Herold, Marten Szibor, and Thomas Braun. Bronchioalveolar stem cells are a main source for regeneration of distal lung epithelia in vivo. *The EMBO Journal*, 38(12), jun 2019.

[197] Jennifer Quantius, Carole Schmoldt, Ana I. Vazquez-Armendariz, Christin Becker, Elie El Agha, Jochen Wilhelm, Rory E. Morty, István Vadász, Konstantin Mayer, Stefan Gattenloehner, Ludger Fink, Mikhail Matrosovich, Xiaokun Li, Werner Seeger, Juergen Lohmeyer, Saverio Bellusci, and Susanne Herold. Influenza Virus Infects Epithelial Stem/Progenitor Cells of the Distal Lung: Impact on Fgfr2b-Driven Epithelial Repair. *PLoS Pathogens*, 12(6), jun 2016.

[198] Luke Zappia, Belinda Phipson, and Alicia Oshlack. Exploring the single-cell RNA-seq analysis landscape with the scRNA-tools database. *PLOS Computational Biology*, 14(6):e1006245, jun 2018.

[199] Barbara Treutlein, Doug G. Brownfield, Angela R. Wu, Norma F. Neff, Gary L. Mantalas, F. Hernan Espinoza, Tushar J. Desai, Mark A. Krasnow, and Stephen R. Quake. Reconstructing lineage hierarchies of the distal lung epithelium using single-cell RNA-seq. *Nature*, 509(7500):371–375, may 2014.

[200] Yina Du, Minzhe Guo, Jeffrey A. Whitsett, and Yan Xu. 'LungGENS': A web-based tool for mapping single-cell gene expression in the developing lung. *Thorax*, 70(11):1092–1094, nov 2015.

[201] Yina Du, Joseph A. Kitzmiller, Anusha Sridharan, Anne K. Perl, James P. Bridges, Ravi S. Misra, Gloria S. Pryhuber, Thomas J. Mariani, Soumyaroop Bhattacharya, Minzhe Guo, S. Steven Potter, Phillip Dexheimer, Bruce Aronow, Alan H. Jobe,

Jeffrey A. Whitsett, and Yan Xu. Lung Gene Expression Analysis (LGEA): An integrative web portal for comprehensive gene expression data analysis in lung development. *Thorax*, 72(5):481–484, jan 2017.

[202] Anne Karina T. Perl and Emily Gale. FGF signaling is required for myofibroblast differentiation during alveolar regeneration. *American Journal of Physiology - Lung Cellular and Molecular Physiology*, 297(2), aug 2009.

[203] Stephen E. McGowan and Diann M. McCoy. Regulation of fibroblast lipid storage and myofibroblast phenotypes during alveolar septation in mice. *American Journal of Physiology-Lung Cellular and Molecular Physiology*, 307(8):L618–L631, oct 2014.

[204] Denise Al Alam, Elie El Agha, Reiko Sakurai, Vahid Kheirollahi, Alena Moiseenko, Soula Danopoulos, Amit Shrestha, Carole Schmoldt, Jennifer Quantius, Susanne Herold, Cho Ming Chao, Caterina Tiozzo, Stijn De Langhe, Maksim V. Plikus, Matthew Thornton, Brendan Grubbs, Parviz Minoo, Virender K. Rehan, and Saverio Bellusci. Evidence for the involvement of fibroblast growth factor 10 in lipofibroblast formation during embryonic lung development. *Development*, 142(23):4139–4150, dec 2015.

[205] Agnieszka Pozarska, José Alberto Rodríguez-Castillo, David E. Surate Solaligue, Aglaia Ntokou, Philipp Rath, Ivana Mižíková, Alicia Madurga, Konstantin Mayer, István Vadász, Susanne Herold, Katrin Ahlbrecht, Werner Seeger, and Rory E. Morty. Stereological monitoring of mouse lung alveolarization from the early postnatal period to adulthood. *American Journal of Physiology - Lung Cellular and Molecular Physiology*, 312(6):L882–L895, 2017.

[206] Elie El Agha, Alena Moiseenko, Vahid Kheirollahi, Stijn De Langhe, Slaven Crnkovic, Grazyna Kwapiszewska, Djuro Kosanovic, Felix Schwind, Ralph T. Schermuly, Ingrid Henneke, Bre Anne MacKenzie, Jennifer Quantius, Susanne Herold, Aglaia Ntokou, Katrin Ahlbrecht, Rory E. Morty, Andreas Günther,

Werner Seeger, and Saverio Bellusci. Two-Way Conversion between Lipogenic and Myogenic Fibroblastic Phenotypes Marks the Progression and Resolution of Lung Fibrosis. *Cell Stem Cell*, 20(2):261–273.e3, feb 2017.

[207] Elie El Agha and Saverio Bellusci. Walking along the Fibroblast Growth Factor 10 Route: A Key Pathway to Understand the Control and Regulation of Epithelial and Mesenchymal Cell-Lineage Formation during Lung Development and Repair after Injury. *Scientifica*, 2014:1–20, 2014.

[208] Madeline A. Lancaster and Juergen A. Knoblich. Organogenesis in a dish: Modeling development and disease using organoid technologies. *Science*, 345(6194):1, jul 2014.

[209] Thomas A. Wynn and Kevin M. Vannella. Macrophages in tissue repair, regeneration, and fibrosis. *Immunity*, 44(3):450, mar 2016.

[210] Kristin Westphalen, Galina A. Gusarova, Mohammad N. Islam, Manikandan Subramanian, Taylor S. Cohen, Alice S. Prince, and Jahar Bhattacharya. Sessile alveolar macrophages modulate immunity through connexin 43-based epithelial communication. *Nature*, 506(7489):503, feb 2014.

[211] Gianni Carraro, Amit Shrestha, Jana Rostkovius, Adriana Contreras, Cho Ming Chao, Elie El Agha, Breanne MacKenzie, Salma Dilai, Diego Guidolin, Makoto Mark Taketo, Andreas Günther, Maya E. Kumar, Werner Seeger, Stijn De Langhe, Guillermo Barreto, and Saverio Bellusci. miR-142-3p balances proliferation and differentiation of mesenchymal cells during lung development. *Development (Cambridge)*, 141(6):1272–1281, mar 2014.

[212] Peter Reuther, Kristina Göpfert, Alexandra H. Dudek, Monika Heiner, Susanne Herold, and Martin Schwemmle. Generation of a variety of stable Influenza A reporter viruses by genetic engineering of the NS gene segment. *Scientific Reports*, 5, jun 2015.

[213] Xianglan Yao, Elizabeth M. Gordon, Debbie M. Figueroa, Amisha V. Barochia, and Stewart J. Levine. Emerging roles of apolipoprotein e and apolipoprotein A-I in the pathogenesis and treatment of lung disease, aug 2016.

[214] World Health Organization (WHO). Global tuberculosis report 2023. *2023*, 2023.

[215] A. Sakula. Robert Koch: centenary of the discovery of the tubercle bacillus, 1882. *Thorax*, 37(4):246, 1982.

[216] Rein M.G.J. Houben and Peter J. Dodd. The Global Burden of Latent Tuberculosis Infection: A Re-estimation Using Mathematical Modelling. *PLOS Medicine*, 13(10):e1002152, oct 2016.

[217] Nicolas A. Menzies, Emory Wolf, David Connors, Meghan Bellerose, Alyssa N. Sbarra, Ted Cohen, Andrew N. Hill, Reza Yaesoubi, Kara Galer, Peter J. White, Ibrahim Abubakar, and Joshua A. Salomon. Progression from latent infection to active disease in dynamic tuberculosis transmission models: a systematic review of the validity of modelling assumptions. *The Lancet. Infectious diseases*, 18(8):e228, aug 2018.

[218] Jon C. Emery, Alexandra S. Richards, Katie D. Dale, C. Finn McQuaid, Richard G. White, Justin T. Denholm, and Rein M.G.J. Houben. Self-clearance of Mycobacterium tuberculosis infection: implications for lifetime risk and population at-risk of tuberculosis disease. *Proceedings of the Royal Society B: Biological Sciences*, 288(1943), jan 2021.

[219] Edine W. Tiemersma, Marieke J. van der Werf, Martien W. Borgdorff, Brian G. Williams, and Nico J.D. Nagelkerke. Natural History of Tuberculosis: Duration and Fatality of Untreated Pulmonary Tuberculosis in HIV Negative Patients: A Systematic Review. *PLoS ONE*, 6(4), 2011.

[220] Véronique A. Dartois and Eric J. Rubin. Anti-tuberculosis treatment strategies and drug development: challenges and priorities. *Nature Reviews Microbiology 2022*, 20(11):685–701, apr 2022.

[221] Roland Diel, Joris Vandeputte, Gerard De Vries, Jonathan Stillo, Maryse Wanlin, and Albert Nienhaus. Costs of tuberculosis disease in the European Union: a systematic analysis and cost calculation. *The European respiratory journal*, 43(2):554–565, feb 2014.

[222] Kaori L. Fonseca, Pedro N.S. Rodrigues, I. Anna S. Olsson, and Margarida Saraiva. Experimental study of tuberculosis: From animal models to complex cell systems and organoids. *PLoS Pathogens*, 13(8), aug 2017.

[223] Temesgen Yihunie Akalu, Archie C.A. Clements, Haileab Fekadu Wolde, and Kefyalew Addis Alene. Economic burden of multidrug-resistant tuberculosis on patients and households: a global systematic review and meta-analysis. *Scientific Reports*, 13(1):1–11, dec 2023.

[224] Nino Iakobachvili, Stephen Adonai Leon-Icaza, Kèvin Knoops, Norman Sachs, Serge Mazères, Roxane Simeone, Antonio Peixoto, Célia Bernard, Marlène Murris-Espin, Julien Mazières, Kaymeuang Cam, Christian Chalut, Christophe Guilhot, Carmen López-Iglesias, Raimond B.G. Ravelli, Olivier Neyrolles, Etienne Meunier, Geanncarlo Lugo-Villarino, Hans Clevers, Céline Cougoule, and Peter J Peters. Mycobacteria-host interactions in human bronchiolar airway organoids. *Molecular Microbiology*, 117(3):682, mar 2022.

[225] Catherine Jia Yun Tsai, Jacelyn Mei San Loh, and Thomas Proft. Galleria mellonella infection models for the study of bacterial diseases and for antimicrobial drug testing, apr 2016.

[226] Ulrike Binder, Elisabeth Maurer, and Cornelia Lass-Flörl. Galleria mellonella: An invertebrate model to study pathogenicity in correctly defined fungal species. *Fungal Biology*, 120(2):288–295, feb 2016.

[227] Yanwen Li, John Spiropoulos, William Cooley, Jasmeet Singh Khara, Camilla A. Gladstone, Masanori Asai, Janine T. Bossé, Brian D. Robertson, Sandra M. New-

ton, and Paul R. Langford. Galleria mellonella - a novel infection model for the Mycobacterium tuberculosis complex. *Virulence*, 9(1):1126–1137, jan 2018.

[228] Masanori Asai, Yanwen Li, John Spiropoulos, William Cooley, David Everest, Brian D. Robertson, Paul R. Langford, and Sandra M. Newton. A novel biosafety level 2 compliant tuberculosis infection model using a Δ leuD Δ panCD double auxotroph of Mycobacterium tuberculosis H37Rv and Galleria mellonella. *Virulence*, 11(1):811–824, jan 2020.

[229] Masanori Asai, Yanwen Li, John Spiropoulos, William Cooley, David J. Everest, Sharon L. Kendall, Carlos Martín, Brian D. Robertson, Paul R. Langford, and Sandra M. Newton. Galleria mellonella as an infection model for the virulent Mycobacterium tuberculosis H37Rv. *Virulence*, 13(1):1543, 2022.

[230] Masanori Asai, Gerard Sheehan, Yanwen Li, Brian D. Robertson, Kevin Kavanagh, Paul R. Langford, and Sandra M. Newton. Innate Immune Responses of Galleria mellonella to Mycobacterium bovis BCG Challenge Identified Using Proteomic and Molecular Approaches. *Frontiers in cellular and infection microbiology*, 11, feb 2021.

[231] Deepak Kumar Mahanta, Tanmaya Kumar Bhoi, J. Komal, Ipsita Samal, R. M. Nikhil, Amit Umesh Paschapur, Gaurav Singh, P. V.Dinesh Kumar, H. R. Desai, Mohammad Abbas Ahmad, P. P. Singh, Prasanta Kumar Majhi, U. Mukherjee, Pushpa Singh, Varun Saini, Shahanaz, N. Srinivasa, and Yogesh Yele. Insect-pathogen crosstalk and the cellular-molecular mechanisms of insect immunity: uncovering the underlying signaling pathways and immune regulatory function of non-coding RNAs. *Frontiers in Immunology*, 14, 2023.

[232] R. H. Singer and D. C. Ward. Actin gene expression visualized in chicken muscle tissue culture by using in situ hybridization with a biotinated nucleotide analog. *Proceedings of the National Academy of Sciences of the United States of America*, 79(23):7331, 1982.

[233] Nicole S. Delellis, Yimin Chen, Sarah E. Cornwell, Dominique Kelly, Alex Mayhew, Sodiq Onaolapo, and Victoria L. Rubin. ChatGPT Media Coverage Metrics; Initial Examination. *Proceedings of the Association for Information Science and Technology*, 60(1):935–937, oct 2023.

[234] Aleksandr Ianevski, Anil K. Giri, and Tero Aittokallio. Fully-automated and ultra-fast cell-type identification using specific marker combinations from single-cell transcriptomic data. *Nature Communications*, 13(1):1–10, mar 2022.

[235] Fan Yang, Wenchuan Wang, Fang Wang, Yuan Fang, Duyu Tang, Junzhou Huang, Hui Lu, and Jianhua Yao. scBERT as a large-scale pretrained deep language model for cell type annotation of single-cell RNA-seq data. *Nature Machine Intelligence*, 4(10):852–866, sep 2022.

[236] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scGPT: toward building a foundation model for single-cell multi-omics using generative AI. *Nature Methods*, pages 1–11, feb 2024.

[237] Will Macnair, Revant Gupta, and Manfred Claassen. psupertime: supervised pseudotime analysis for time-series single-cell RNA-seq data. *Bioinformatics*, 38(Supplement 1):i290–i298, jun 2022.

[238] Zhaolong Gao and Yiwei Li. Enhancing single-cell biology through advanced AI-powered microfluidics. *Biomicrofluidics*, 17(5):51301, sep 2023.

[239] Isaac Virshup, Sergei Rybakov, Fabian J. Theis, Philipp Angerer, and F. Alexander Wolf. anndata: Annotated data. *bioRxiv*, page 2021.12.16.473007, dec 2021.

[240] Czi Cell Science Program, Shibla Abdulla, Brian A Ev ermann, Pedro Assis, Seve Badajoz, Sidney M Bell, Emanuele Bezzi, Batuhan Cakir, Jim Chaffer, Signe Chambers, J Michael Cherry, Tiffany Chi, Jennifer Chien, Leah Dorman, Pablo Gar cia Niet, Nayib Gloria, Mim Hastie, Daniel Heg eman, J ason Hilt on, Timmy Huang, Amanda Infeld, Ana-Maria Istr at, Ivana Jelic, Kuni Katsuya, Yang Joon

Kim, Karen Liang, Mike Lin, Maximilian Lombar do, Baile Marshall, Bruce Mar tin, F an McDade, Colin Megill, Nikhil Patel, Alexander Predeus, Brian Raymor, Behnam Robatmili, Da Rog ers, Erica Rutherf or, Dana Sadg at, Andrew Shin, Corinn Small, Trent Smith, Pr athap Sridhar an, Alexander Tar ashansky, Norber Ta ares, Har ley Thomas, Andrew Tolopko, Meghan Urisko, Jo ce Yan, Gar abet Yeretssian, Jennifer Zamanian, Arathi Mani, Jonah Cool, and Ambrose Carr. CZ CELLxGENE Discover: a single-cell data platform for scalable exploration, analysis and modeling of aggregated data. *Nucleic Acids Research*, 53(D1):D886–D900, jan 2025.

[241] Claire Weber, Marissa B. Hirst, Ben Ernest, Nicholas J. Schaub, Kelli M. Wilson, Ke Wang, Hannah M. Baskir, Pei Hsuan Chu, Carlos A. Tristan, and Ilyas Singeç. SEQUIN is an R/Shiny framework for rapid and reproducible analysis of RNA-seq data. *Cell Reports Methods*, 3(3):100420, mar 2023.

[242] Jeffrey M. Pullin and Davis J. McCarthy. A comparison of marker gene selection methods for single-cell RNA sequencing data. *Genome Biology*, 25(1):1–37, dec 2024.