Efficient Grouping Methods for the Annotation and Sorting of Single Cells

Markus Lux

Bielefeld University

Faculty of Technology

Ph.D. Thesis

Efficient Grouping Methods for the Annotation and Sorting of Single Cells

submitted by

Markus Lux

for the degree of *Dr. rer. nat.*

Referees

Prof. Dr. Barbara Hammer
CITEC centre of excellence
Bielefeld University

Prof. Dr. Cedric Chauve
Dept. of Mathematics
Simon Fraser University, Vancouver

Prof. Dr. Gyan Bhanot
Dept. of Molecular Biology and Biochemistry & Physics
Rutgers University, New Jersey

Disputation on April 24, 2018

Gedruckt auf alterungsbeständigem Papier nach DIN ISO 9706.
Printed on permanent, non-aging paper according to DIN ISO 9706.

Markus Lux
Computational Methods for the Analysis of the Diversity and Dynamics of Genomes
German-Canadian DFG International Research Training Group (1906/1)

Bielefeld University – Faculty of Technology
P.O. Box 10 01 31
D-33501 Bielefeld, Germany
mlux@techfak.uni-bielefeld.de

Zusammenfassung

Computergestützte Methoden, welche verborgene Datenstrukturen und Muster zuverlässig und effizient erkennen können, sind nötig um biologische Forschung voranzutreiben. Um die damit verbundenen, komplexen Probleme lösen zu können, ist es oft notwendig, homogene Gruppen in Daten aufzudecken. Damit verbunden sind Clustering- und Klassifikationsmethoden, welche in allen Wissenschaftszweigen angewendet werden. Oft erschweren dabei Störfaktoren die Datenanalyse und eine sorgfältige Wahl von Methoden und Parametern ist unabdingbar. Diese Arbeit beschäftigt sich mit Methoden zur Analyse von Einzelzellen – Dazu habe ich für drei verschiedene Technologien Clustering- und Klassifikationsmethoden entwickelt, evaluiert, verglichen und angepasst:

1. Ein vorherrschendes Problem in metagenomischen Proben ist die Detektion von Clustern, welche die darin befindlichen Spezies repräsentieren ("Binning"). Obwohl Methoden zur Erkennung von bekannten Taxa existieren, ist de novo Binning ungelöst. In diesem Kontext habe ich eine optimale Wahl von Methoden und Parameterisierungen für solche Daten analysiert. Daraus resultiert, basierend auf der Integration von Dimensionsreduktion und Clustering, eine automatisierte Binning-Methodik.

2. Kontamination mit Fremdgenomen ist bei der Sequenzierung von Einzelzellen nach wie vor eines der Hauptprobleme. Aus der Sicht der Informatik können sowohl in der Metagenomik als auch in der Einzelzellanalyse Genome als Cluster dargestellt werden. Jedoch ist die Analyse der Cluster für Einzelzellen eine grundlegend andere. In diesem Kontext habe ich eine Anwendung entwickelt, die es ermöglicht, Kontamination in Einzelzellen automatisch zu detektieren und Konfidenzen zu berechnen.

3. Eine weitere Herausforderung besteht in der Erkennung von Zellpopulationen in der Durchflusszytometrie. Dieses Problem wird oft durch manuelle und mühsame Zellannotation gelöst. Automatisierte Methoden existieren, benötigen jedoch schwierige Feinjustierung von Hyperparametern. Um diese Beschränkung aufzuheben, habe ich eine halb-überwachte Methodik zur Identifikation von Zellpopulationen entwickelt. Diese besitzt nur wenige, sehr robuste Parameter, ist präzise, schnell und gleichzeitig interpretierbar.

Abstract

Insights into large-scale biological data require computational methods which reliably and efficiently recognize latent structures and patterns. In many cases, it is necessary to find homogeneous subgroups of the data in order to solve complex problems and to enable the discovery of novel knowledge. Here, clustering and classification techniques are commonly employed in all fields of research. Confounding factors often complicate data analysis and require a thorough choice of methods and parameters. This thesis is focused on methods around single-cell research – I developed, evaluated, compared and adapted grouping methods for open problems from three different technologies:

First, metagenomics is typically confronted with the problem of detecting clusters representing involved species in a given sample (binning). Albeit powerful technologies exist for the identification of known taxa, de novo binning is still in its infancy. In this context, I evaluated optimal choices of techniques and parameters regarding the integration of modern machine learning methods, such as dimensionality reduction and clustering, resulting in an automated binning pipeline.

Second, in single-cell sequencing, a major problem is given by sample contamination with foreign genomic material. From a computational point of view, in both metagenomics and single-cell genome assemblies, genomes can be represented as clusters. Contrary to metagenomics, the clustering task for single cells is a fundamentally different one. Here, I developed a methodology to automatically detect contamination and estimate confidences in single-cell genome assemblies.

A third challenge can be seen in the field of flow cytometry. Here, the precise identification of cell populations in a sample is crucial and requires manual, tedious, and possibly biased cell annotation. Automated methods exist, however they require difficult fine-tuning of hyper-parameters to obtain the best results. To overcome this limitation, I developed a semi-supervised tool for cell population identification, with few very robust parameters, being fast, accurate and interpretable at the same time.

Acknowledgements

Doing research and publishing papers is rewarding and frustrating at the same time. It is often hard to make a cut between personal and professional life, and a large number of individuals helped me to keep going forward in both. Having had magnificent 3+ years with many of you and knowing that I won't meet some of you ever again, it is hard to avoid getting sentimental. In no particular order I want to thank:

Alex S., Andi, Irena and Boas who always took the time and answered my stupid questions even though, often, I did not know myself what I wanted. Also, my colleagues and friends from my time in Cambridge, namely Madis, Alex K., Camilla, Kasia, Karolina and Joey for making the best out of the after-work hours. Next, Tanja, Bill and others from the JGI who supported me in the development of acdc. In Vancouver, it was much fun to work together with you, Justin, Mehrnoush, Albina and Sibyl. But let's not forget my close working environment in Bielefeld: Many thanks to Alex S., Bassam, Benjamin, Lydia, Babak, the Göpferts and all others of the Machine Learning group for having much fun together on numerous occasions! Possibly the most time I spent with all the people on the U10 floor: Many thanks go to Tina, Georges, Bruce Li and Linda for having an awesome time, to Guillaume for not taking his ping pong defeats too serious, to Pina always being up for a good discussion, to Lukas for being the perfect office companion and to all the other DiDies. I deeply believe that having so much fun with you was a big stress relief. Huge thanks also go to Heike, Roland, Gisela and Angelika for keeping everything run smoothly.

Thank you, Lea, for always standing by my side, listening to my problems and caring when I needed you. Thanks, Jan, for being a great friend at all times. Thanks to my family for the continuous belief in me.

Last, but not least, I want to thank my advisors for making all of this possible. Thanks, Ryan Brinkman and Cedric Chauve for helping me with my project, and letting me know that I am doing good work. And thank you, Barbara Hammer, for being the global optimum in a large space of PhD advisors. I still cannot comprehend how you balance so much work, answer emails within a few minutes, take so much time for your students, and still be positive about everyone and everything. It deserves much respect.

Contents

1 Introduction

1.1 Motivation

Rapid advances in biotechnology and the wide-spread use of high throughput devices enable the gathering of seemingly unlimited amounts of information. Next-Generation-Sequencing (NGS, Goodwin et al., 2016) machines can reliably read the DNA of a human genome in under three days at low cost, and Flow Cytometry (FCM, Shapiro, 2005) devices can accurately characterize thousands of individual cells per minute. The availability of such novel technologies results in unprecedented opportunities, and a large number of applications from different research fields find great benefit in them. Particular interest can be seen in the context of single cells and the interaction thereof. An example is given by the metagenomic analysis of hot springs. Here, looking at individual cells or organisms of microbial communities generates new insights into "microbial dark matter", possibly revealing undiscovered life forms on planet earth (Rinke et al., 2013; Bremges, 2016). Moreover, by analyzing the interaction of biogas-producing bacteria (Bremges et al., 2015; Maus et al., 2016), research for renewable energies greatly benefits from metagenomics as well. While metagenomics deals with the analysis of a large set of related microbes or cells, single-cell genome analysis focuses on the individual characteristics of cells. Using single-cell sequencing (SCS), it is possible to study the heterogeneity of tumors, ultimately leading to more efficient, precise, and successful cancer therapies (Zhang et al., 2016). Furthermore, SCS has shown to be a powerful tool in neurobiology, where single-cell neuronal diversity has important implications for neural circuit function and neurological diseases (Harbom et al., 2016). A large set of other applications include stem cells, developmental biology, drug discovery, reproductive health, and immunology (Illumina, 2016). Those fields, especially immunology also utilize FCM technology to untangle complex interactions of different cell types in the human immune system, making it an important tool, for example, for the detection of tumor markers (Sukhdeo et al., 2013), diagnosing acute myeloid leukemia (Kaleem et al., 2003), or discovering new immunological phenotypes (3i, 2017).

The list of applications could easily be expanded and emphasizes the importance of NGS and FCM technologies for modern research. At least equally important is the extraction of meaningful information from the raw data that is generated. It frequently has to undergo intensive processing in a number of complex transformations. Depending on the task, confounding factors complicate data analysis and computational methods need to reliably and efficiently recognize latent structures and patterns. In many cases, in order to ease analysis, it is necessary to find homogeneous subgroups of the data, i.e. grouping similar data elements together, while at the same time separating them from more dissimilar elements.

In general, such grouping methods can be differentiated into unsupervised and supervised ones. The former type of methods is frequently denoted as "clustering" and does not make use of auxiliary knowledge, i.e. uses only information inherent to the given data. In contrast, the latter, often synonymous to "classification", uses auxiliary knowledge, for example data from which the grouping of interest is already known. An example of unsupervised grouping methods is given by the clustering of DNA sequences. Clustering highly similar sequences together while putting dissimilar sequences far apart from each other is a common pre-processing step for solving many problems. For example, it can be used for error correction in NGS reads (Nikolenko et al., 2013) and the identification of functionally related genes (Yi et al., 2007).

In this thesis, clustering is used, first for finding metagenomic sequences of the same species, and second for the detection and removal of contamination in single cell genomes. When there is prior knowledge about the sequences in questions, these problems can also be solved using classification methods (Wood and Salzberg, 2014). Another classification task is highlighted as a third problem in this thesis: A crucial step in FCM analysis is the identification of cell populations using prior biological knowledge. Sorting single cells based on their characteristics can enable disease diagnostics and aid in immunophenotyping.

The application of grouping methods can be found in nearly all areas of the social and natural sciences, including economics, psychology, sociology, chemistry, physics, and many more. At the same time, there is a large number of clustering and classification methods in existence. In many cases, the question of what method is most appropriate for the application to a

given data set is difficult to decide. Most algorithms are suitable for a certain kind of data only, and many factors and variables complicate the decision of which algorithm works best. Often, it is necessary to evaluate and analyze a good amount of techniques, before finding the right one. Additionally, the majority of methods require the setting of hyper-parameters, i.e. parameters controlling the result of an automated method, typically set by practitioners in a way that the outcome of the method is satisfactory. In some cases, it is also crucial to pre-process the data such that a given method can take advantage of it. This further complicates the evaluation process. Consequently, while the majority of grouping algorithms are surprisingly simple, for an optimal result, the application thereof is complex and requires considerable effort.

This thesis puts a particular focus on the development of computational methods for novel biological data. It spotlights efficient grouping methods for the annotation and sorting of single cells, spanning three areas of research. Specifically, for each of metagenomics, single-cell genomics and flow cytometry, I will present one particular methodology that addresses a predominant problem in the respective field. I evaluated the suitability of clustering and classification algorithms, developed and adapted methods for the automatic determination of crucial parameters, included essential pre-processing steps, integrated auxiliary data, and tested the methodologies on a large number of complex data sets. In the remainder of this chapter, I will briefly introduce the topics and problems of each of these three subprojects, and my contributions to solve them. It is followed by an in-depth presentation of the methodologies in the subsequent chapters.

1.2 Metagenomics

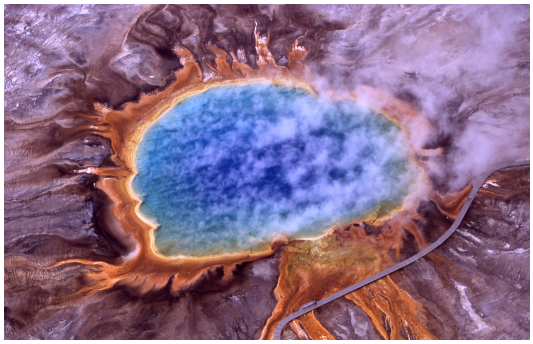

Fig. 1.1.: Thermophile archaea and bacteria produce bright colors in a hot spring. Photo: Public domain (Peaco, 2001).

The vast majority of organisms in most environments on earth is represented by microbial communities and the analysis of the diversity and dynamics of such is highly important (Handelsman, 2004; Forbes et al., 2017). In contrast to classical genomics where the focus is on the genome of one specific organism, in metagenomics all species contained in such samples are of interest. Two examples are given by methanogenic micro-organisms living in meromitic lakes (Gies et al., 2014) or the gut microbiome (Gill et al., 2006), which are of tremendous interest for renewable energies and health, respectively. Our ability to predict the response of such microbial communities to perturbation will improve in accord with the depth of understanding. The standard is higher yet for efforts to engineer the function of microbial systems (Blainey, 2013).

A common problem in metagenomic analysis is "binning": Using shotgun sequencing and sophisticated assembly methods, contigs and scaffolds from hundreds of organisms have to be assigned to their originating genome. Because of missing reference knowledge, in the case of de novo assembly, this is a non-trivial task. To solve it, several promising attempts to partially automate this process have been proposed. Quite a few recent approaches rely on machine learning techniques, in particular clustering. However, so far, there does not exist a fully automated process, nor a thorough evaluation of its accuracy and robustness with respect to parameterization. Given that, in this thesis, I address a particular type of binning, namely taxonomy-independent, sequence-composition based methods, particularly:

1. The integration of modern dimensionality reduction and clustering techniques suitable for high-dimensional data, and an automated selection of the number of clusters.

2. A formal quantitative evaluation of the pipeline in benchmarks.

3. An evaluation of an optimal parameter choice, resulting in an automation of the process.

1.3 Single-cell genomics

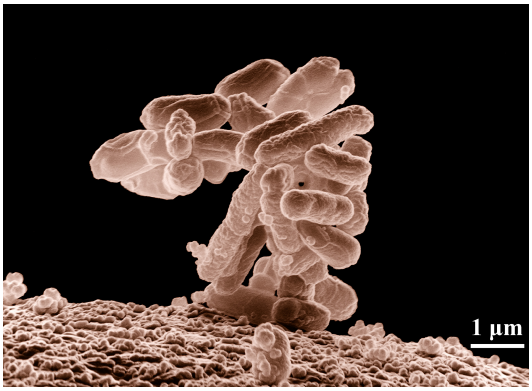

Fig. 1.2.: Electron micrograph of a cluster of *E. coli* bacteria. Photo: Public domain (Erbe, 2005).

Strongly related to metagenomics is the analysis of single cells. Composite genomes assembled from metagenomic data do not presently distinguish between genes that are tightly coupled within the context of the same organism and genes that are coupled across different organisms (Blainey, 2013). Hence, in order to study their individual genetic composition, single cells are often physically isolated from metagenomes. Traditionally, in order to obtain the large amounts of DNA material needed for sequencing, such cells are cultivated in axenic cultures in vitro. But the large majority of microbial organisms, often denoted as "microbial dark matter", cannot be grown in laboratories. Here, single-cell sequencing provides novel techniques, allowing the study of these organisms at the most fundamental biological unit (Hedlund et al., 2014).

Unfortunately, even though sophisticated isolation and amplification methods exist, single-cell research is still confronted with a predominant problem which is contamination (Blainey, 2013; Bowers et al., 2017). To guarantee clean genome assemblies and to prevent the introduction of contamination into public databases, considerable quality control efforts are put into post-sequencing analysis. Contamination screening generally relies on reference-based methods such as database alignment or marker gene search, which limits the set of detectable contaminants to organisms with closely related reference species. As genomic coverage in the tree of life is highly fragmented, and microbial dark matter makes up a large part of it, there is an urgent need for a reference-free methodology for contaminant identification in sequence data.

In this thesis, I developed a methodology specifically to aid the quality control process of genomic sequence data. Combining supervised and unsupervised methods, it reliably detects both known and de novo contaminants. For that task, it builds on the foundations established in Chapter 2. Similar to metagenomic binning, reference-free inspection is enabled by the use of state-of-the-art machine learning techniques that include fast, non-linear dimensionality reduction, and subsequent clustering algorithms that automatically estimate the number of clusters. In my contribution, I address the following open problems:

1. The integration of cluster validity indices to accurately and reliably detect contamination in de novo single-cell sequencing.

2. Provision of interpretable confidence measures.

3. The ability to both interactively and automatically clean a given sample, eliminating the costly necessity of re-sequencing in the case of contamination.

1.4 Flow cytometry

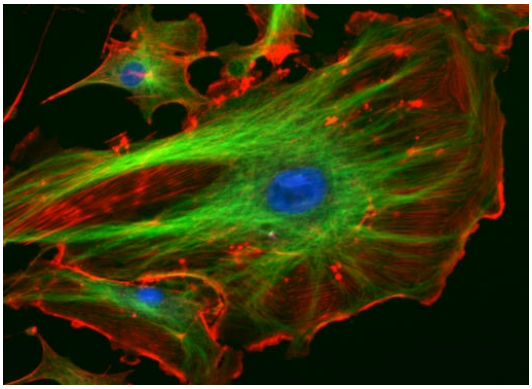

Fig. 1.3.: Fluorophore stained endothelial cells under the microscope. Photo: Public domain (ImageJ, 2005).

One possibility to isolate single cells from metagenomes is targeted enrichment. Given a metagenomic sample, it is possible to sort cells of interest based on their shape, density, and spectral characteristic of fluorescent labels (Blainey, 2013). The latter "fluorophores" can be attached to different parts of the cell, for example to certain characterizing genomic regions such as conserved genes or to characteristic receptors on the cell membrane. Using fluorescence-activated cell sorting (FACS) devices, cells pass a laser beam one-by-one at high rate. As each fluorophore, when excited, emits light of a certain wave length, it is possible to sort cells, enabling the sequencing and assembly of a single genome. Flow cytometry technology is not only used for cell isolation in single-cell sequencing. It is also widely used in medicine, such as for the diagnosis of cancer and other heterogeneous immunodeficiencies. It is also widely used in immunophenotyping which holds great promise for assessing the immune status of patient populations.

A critical part of flow cytometry analysis is the identification of cell populations, which lays the groundwork for both clinical diagnostics and research discovery. One classical technique is manual analysis which consists of the inspection of bivariate cell plots and drawing shapes around populations of interest, commonly denoted as "gating". This current paradigm is time consuming and subjective. Although a large number of automated methods exist and supervised tools are on par with manual gating (Aghaeepour et al., 2013), they require fine-scale parameterization. In order to obtain the best results, a careful choice of hyper-parameters is essential. Consequently, there

is a strong need for methods that are fast to setup, accurate and interpretable at the same time.

To overcome the drawbacks present in current gating standards, in this thesis I report a novel semi-supervised approach for population identification. Specifically, within my methodology, the following problems are addressed:

1. The integration of density alignments for fast prediction of diverse cell populations, being more accurate than existing techniques.

2. Minimal manual annotation and biological interpretability of results.

3. Evaluation and quality-checking of existing gating sets.

1.5 Structural overview

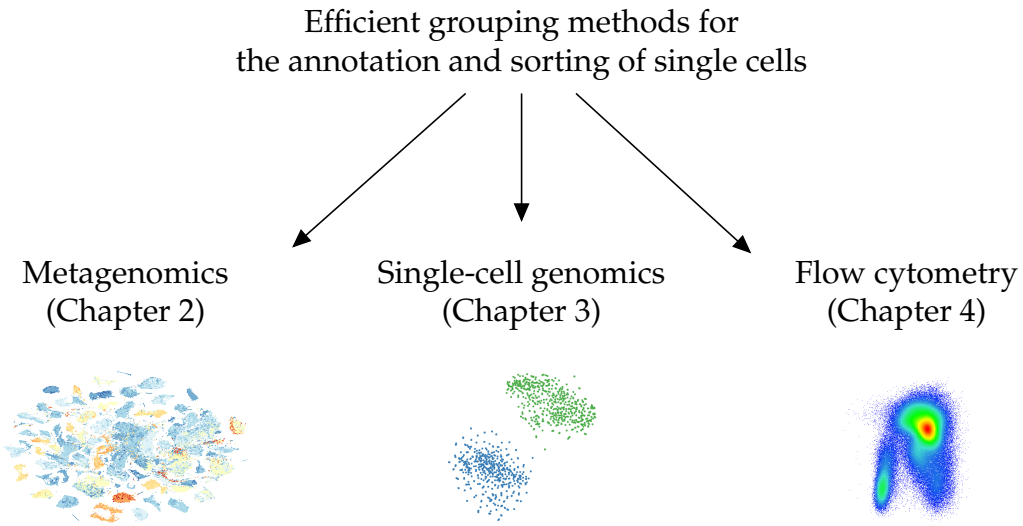

Fig. 1.4.: This work investigates the use of efficient grouping methods for open problems from three different fields related to single-cell research.

Given the three sub-domains of my thesis, this work is organized into three chapters, respectively (Figure 1.4). Each chapter provides a more in-depth introduction to the topic, before theoretical and methodological foundations are established. Based on this, the methodology in question is described

in detail, results are presented and discussed. Each chapter is summarized with an individual conclusion and outlook.

Complete list of publications

I had the opportunity to present parts of the work in the thesis to an international audience within the following publications.

Main contributions

- Markus Lux, Ryan Remy Brinkman, Cedric Chauve, Adam Laing, Anna Lorenc, Lucie Abeler-Dörner, Barbara Hammer. **flowLearn: Fast and precise identification and quality checking of cell populations in flow cytometry**. Bioinformatics, 2018.
  DOI: 10.1093/bioinformatics/bty082

- Markus Lux, Jan Krüger, Christian Rinke, Irena Maus, Andreas Schlüter, Tanja Woyke, Alexander Sczyrba, Barbara Hammer. **acdc – Automated Contamination Detection and Confidence estimation for single-cell genome data**. BMC Bioinformatics, 2016.
  DOI: 10.1186/s12859-016-1397-7

- Markus Lux, Alexander Sczyrba, Barbara Hammer. **Automatic discovery of metagenomic structure**. 2015 International Joint Conference on Neural Networks (IJCNN).
  DOI: 10.1109/ijcnn.2015.7280500

Additional contributions

- Markus Lux, Barbara Hammer, Alexander Sczyrba. **Automated Contamination Detection in Single-Cell Sequencing**. bioRxiv, 2015.
  DOI: 10.1101/020859

- Bassam Mokbel, Sebastian Gross, Markus Lux, Niels Pinkwart, Barbara Hammer. **How to Quantitatively Compare Data Dissimilarities for Unsupervised Machine Learning?** Artificial Neural Networks in Pattern Recognition, 2012.
  DOI: 10.1007/978-3-642-33212-8_1

2 Automatic discovery of metagenomic structure

Parts of this chapter are based on

Markus Lux, Alexander Sczyrba, Barbara Hammer
Automatic discovery of metagenomic structure
2015 International Joint Conference on Neural Networks (IJCNN)
DOI: 10.1109/ijcnn.2015.7280500

2.1 Background

Microorganisms make up the major part of all life on earth and can be found in every part of the ecosphere. Being highly diverse, they are able to adapt to the most extreme conditions, such as the deep sea, hot springs, permafrost, and high altitude. Their study is highly important for modern research and contributes to many technological advancements. In that context, their genetic composition and function is particularly relevant and the field of metagenomics provides methods for their analysis.

In a typical metagenomic workflow (Figure 2.1), samples are taken from environmental sites in order to inspect their microbial diversity. Examples include the analysis of microbes in the gastrointestinal system (Frank and Pace, 2008) or methane-producing archaea (methanogens) out of samples from marine sites (Gies et al., 2014). Furthermore, it has been shown (Rinke et al., 2013) that the metagenomic analysis of microbial dark matter can yield new insights into phylogeny and coding potential of such organisms, discovering new branches in the tree of life (Parks et al., 2017). Applications in other fields, including agriculture (Charles and Marco, 2010), renewable energies, biofuel, biomass, (Charles and Marco, 2010; Hess et al., 2011), health (Qin et al., 2010), and ecology (Raes et al., 2011) are plenty.

Given an environmental sample, the analysis of a metagenome is done in a number of steps, depicted in Figure 2.1. The sample has to be prepared in the laboratory in order to separate and extract the included DNA. It is then

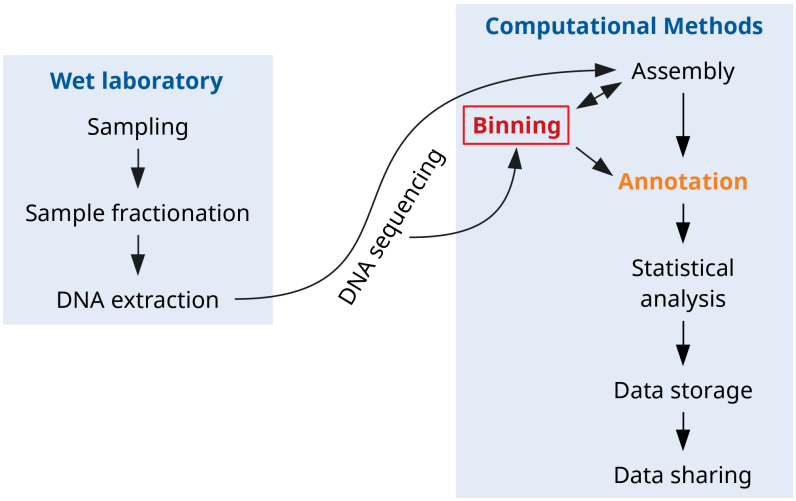

Fig. 2.1.: Metagenomics workflow (Thomas et al., 2012): A major part of it consists of computational methods. An important step for annotating the data is the binning of sequences.

sequenced and the resulting data, depending on the sequencing technology, consists of a large number of small sequences with a length between 100 bp and 300 bp (base pairs), so called "reads". To extract meaningful information from these data, a large part of the analysis process is made of computational methods. First, the sequences have to be assembled into larger, contiguous sequences ("contigs"). Depending on the sequence read coverage, contig lengths vary, and the largest contigs are typically 1×10^4 bp to 1×10^5 bp long. Once assembled, contigs are annotated, functionally (gene annotation) or taxonomically. It is followed by statistical analysis and data storage and sharing.

The taxonomic annotation of metagenome assemblies is a crucial step in the pipeline. As the large majority of microbial species is still unknown (Rinke et al., 2013), the identification of such mostly cannot depend on existing data (i.e. known taxa from reference databases). Hence, an unsupervised taxonomy-free analysis is required (Mande et al., 2012). The analysis of microbial diversity heavily depends on proper tools that allow for selection of individual genomes out of a given metagenome. Here, "binning" plays an important role: it refers to the process of sorting DNA sequences into groups that might represent an individual genome or genomes from closely related organisms (Thomas et al., 2012). In general, taxonomy-independent binning methods can be divided into two types: sequence composition based and abundance based (Sedlar et al., 2017). The former is based on the assumption that taxons are defined by unique genomic signatures. A widely used

signature is the distribution of oligonucleotides as a characteristic feature. A common strategy is to compare oligonucleotide frequencies between different sequences and bin those based on the resulting composition similarity or dissimilarity. In contrast, abundance based approaches build a statistical model over the oligonucleotide abundance, which is the frequency with which these occur in the sample, and not in just one sequence. Lately, hybrid approaches, utilizing both compositional and abundance features gained attention. A detailed review of existing techniques has been provided by Sedlar et al., 2017.

This thesis focuses specifically on the analysis of taxonomy-independent, sequence composition based approaches. In general, this type of binning can be performed in three steps:

1. Transform sequence data into vectorial data and calculate compositional, genomic signatures.

2. Reduce dimensionality of vectorial data, remove noise.

3. Clustering and actual binning.

Sequence data is transformed into frequencies of oligonucleotides. Depending on their size, the resulting vectorial representation can be high dimensional. Hence, from the perspective of computational intelligence, metagenome binning using oligonucleotide frequencies corresponds to the problem to reliably detect clusters in a high dimensional data space. Quite a few challenges arise in this context.

To circumvent negative side effects in such high dimensional spaces and to enable human expert inspection, it is crucial to use appropriate subspace embeddings to transform the data into an easily visualizable representation, i.e. two or three dimensions. A few dimensionality reduction techniques have been tested in this context, prominent examples being the self-organizing map or modern nonlinear dimensionality reduction techniques (Wang et al., 2014; Albertsen et al., 2013; Laczny et al., 2014). Another challenge consists in the automatic determination of the number of clusters and its cluster validity, a deep and crucial question in the context of clustering (Vendramin et al., 2010; Jain, 2010). Interestingly, methods proposed in the context of metagenomic binning determine this number either in an interactive, not

fully automatic process (Laczny et al., 2014), or they rely on heuristics and lack a thorough integration and comparison of standard techniques (Wang et al., 2014; Albertsen et al., 2013).

Summarizing, metagenomic binning relies on crucial parameters that heavily influence the final binning result, but there does not exist a thorough evaluation of their robustness and influence on accuracy. For each of the three steps shown above, I will compare state-of-the-art techniques to automate the process of cluster detection as regards cluster shape and its number, and evaluate its robustness in several benchmarks. For this task, the remainder of this chapter will continue with building the methodological foundation for each of the three steps. It is followed by proposing a more specific, automated binning pipeline on the basis of which different choices of included techniques and parameters are going to be evaluated.

2.2 Methodology

2.2.1 Data representation

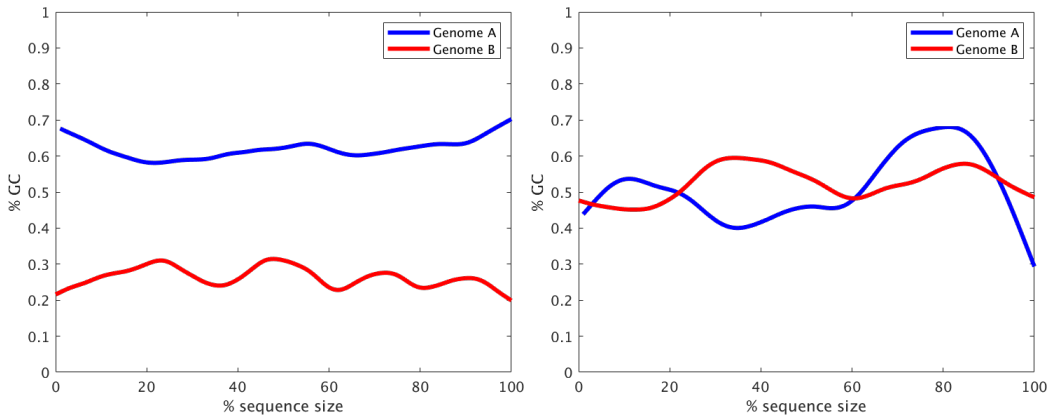

Fig. 2.2.: Using the GC-content to separate genomes. Left two genomes are linearly separable just by their GC-content. Right: Two more related genomes are not anymore linearly separable by their GC-content.

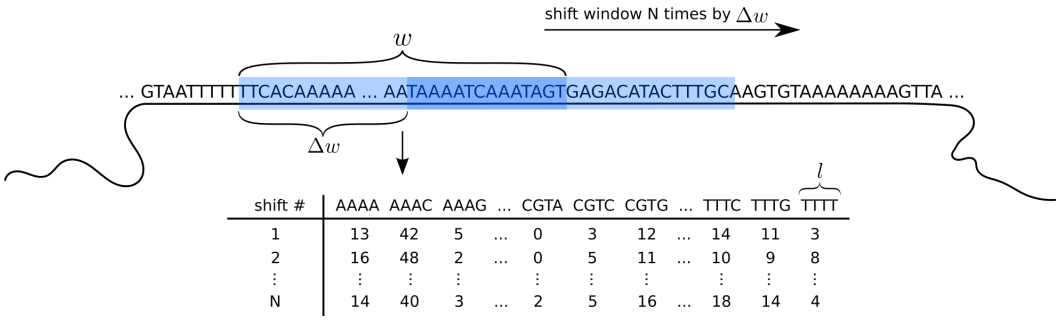

Fig. 2.3.: Sliding window algorithm: A window of width w is shifted over the input sequence N times. On each shift, each l-mer within the window is counted and tabulated, obtaining a vectorial representation with N data points of dimensionality 4^l.

As the majority of techniques from machine learning work on vectorial data only, it is necessary to convert the input contigs to vectors that represent signatures. One signature that can separate genomes for binning can be seen in the GC-content of a sequence. Genomes from two distinct species, for example, may be distinguished just by the means of their GC content while more related genomes are not easily separable (Figure 2.2). Hence, a more sophisticated representation is necessary. Here, it is common practice to look at signatures over small sub-sequences of DNA (Teeling et al., 2004). Given an input contig sequence, and any sub-sequence $s \in S$, where S are all DNA sequences of length w, it is the task to find a mapping $g : S \to \mathbb{R}^d$ such that the resulting d-dimensional representation sufficiently captures the genomic characteristics inherent to the original sequence.

To achieve this, a window of fixed width w is subsequently shifted over the input contig sequence with step Δw (Figure 2.3). For each shift and underlying sequence s, $g(s) = (a_1, \ldots, a_d)$, where $a_i = f_i$, with f_i being the frequency of the i-th l-mer (Gori et al., 2011). Here, l-mers are all sub-sequences of length l that are counted and normalized, resulting in N data points with dimension $d = 4^l$ (corresponding to the four nucleotide bases) using N shifts. In the case of tetramers ($l = 4$), this results in $d = 256$ dimensions. Since the underlying sequencing technologies can sequence both DNA strands, it is also beneficial to account for reverse complements in a symmetrized signature. Given the reverse complement of the i-th l-mer, d can be reduced: frequencies of l-mers and their reverse complement are added to the same vectorial feature: $a_i = f_i + f_i^C$ if the i-th l-mer is not equal to its reverse complement, and $a_i = f_i$ otherwise. Exemplary, for $l = 4$, the

reduced dimensionality is $d = 136$, because 16 tetramers coincide with their reverse complement and 240 do not (Gori et al., 2011).

Furthermore, it is worth to note that by taking $\Delta w < w$, windows overlap, and with decreasing Δw, neighboring data points share increasingly similar signatures. Consequently, data points from the same genomic origin are also located near each other in the resulting vectorial space. This observation is important: An optimal choice of the two connected parameters w and Δw is crucial for the clustering step in a binning pipeline. The window size and step determine the type of the captured genetic signature. Choosing a smaller w will capture more of the local characteristics of a genome such as individual genes, while a larger w will capture information that describes more of the global characteristics. Choosing the parameters too small will result in neighboring data points that are nearly identical, and choosing them too large will result in the loss of identifying global characteristics.

The parameter l controls how fine-grained the signature is. Taking $l = 1$ corresponds to the special case of capturing the GC-content of the underlying sequence, hence l-mer frequencies are a generalization of the GC-content, a widely used compositional feature for metagenomic assessment (Sedlar et al., 2017; Land et al., 2015). Greater l will result in a finer resolution of the signature, however choosing a large l is computationally more expensive as the number of dimensions d grows exponentially, and increasing l decreases the probability that two neighboring windows share l-mers.

2.2.2 Dimensionality reduction

Problems of high-dimensional spaces

Genomic data in general and l-mer frequencies in particular are usually high-dimensional. Such data has a number of disadvantageous properties that complicate their analysis. Those drawbacks are often summarized under the umbrella term "Curse of Dimensionality" (Hastie et al., 2009) and can be best described by looking at the geometric properties of such spaces. Considering a regular d-dimensional grid with 10 equally spaced cells along each dimension, it is worth to note that the number of cells $N = 10^d$ within the whole grid is growing exponentially with increasing d (Bellman et al.,

1961). Therefore, already with only nine dimensions, such a grid contains 10^9 cells, making such spaces inherently sparsely populated by data. Another phenomenon is the volume of a d-dimensional hypersphere. When looking only at a thin ϵ-shell of the hypersphere, the relative volume contained in this shell is $V_{rel} = 1 - (1 - \epsilon)^d$ (Lee and Verleysen, 2007). Thus, with increasing dimension, nearly all the sphere's volume is contained only in this small shell, confirming that most of the space in a high-dimensional hypersphere is in fact empty and data accumulates at the boundary. Connected to the volume of a hypersphere is a third, and for the purpose of this thesis most important point of high-dimensional data analysis: distances. Most clustering methods depend on the notion of a distance or similarity, i.e. they have to assess whether a given pair of points is close to or far away from each other, being in the same or a different cluster, respectively. Unfortunately, distances tend to become relatively uniform with increasing dimensionality (Beyer et al., 1999). This is, given an independently selected data point, the relative distances D_{min} and D_{max} of its closest and furthest data point, respectively, approaches zero:

$$\lim_{d \to \infty} \frac{D_{max} - D_{min}}{D_{min}} = 0 \tag{2.1}$$

Again, consider a hypersphere centered at one data point, with a radius $r = D_{min}$. Because the relative difference between D_{min} and D_{max} is very small, increasing the radius only slightly will include many more data points (Steinbach et al., 2004). This behavior of distances makes the notion of a nearest neighbor become meaningless in such high-dimensional spaces. But as clustering algorithms heavily depend on this notion, such effects have serious consequences and have to be taken care of, for example by the use dimensionality reduction techniques.

Dimensionality reduction is the task of embedding high dimensional data points $x_i \in \mathbb{R}^d$ in a lower dimensional subspace as points $y_i \in \mathbb{R}^{d'}$ where $d' \ll d$ such that as many properties as possible (e.g. distances or similarities) are preserved. This avoids negative effects in high dimensional spaces as discussed earlier and allows for direct visualization when $d \in \{2, 3\}$. In general, dimensionality reduction methods can be divided into linear and nonlinear ones. Using an orthogonal linear map, the most prominent linear technique is Principal Component Analysis (PCA). Its main objective is the preservation of variance and it finds a transformed coordinate system, such that the new axis directions explain as much of the data's variance as

possible. As PCA considers linear relationships between the input features only, it cannot reliably detect structure in nonlinear data. As real-world data is seldom linear, in order to discover cluster structures in l-mer frequencies of metagenomic sequences, nonlinear methods have to be applied.

t-SNE

There are many methods for nonlinear dimensionality reduction in existence, but recently one particular technique, named "t-distributed stochastic neighborhood embedding" (t-SNE, Maaten and Hinton, 2008) gained immense popularity in many fields of research. In the following I will introduce t-SNE, and additional improvements concerning runtime and memory complexity (Van Der Maaten, 2014).

The original t-SNE algorithm aims to minimize the difference between two distributions of pairwise probabilities in the high and lower dimensional space. Considering N data points, high dimensional probabilities are defined as $p_{ij} = (p_{i|j} + p_{j|i}) \ / \ (2N)$ where

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2/(2\sigma_i^2)\right)}{\sum_{l \neq i} \exp\left(-||x_i - x_l||^2/(2\sigma_i^2)\right)} \tag{2.2}$$

can be interpreted as the probability that x_i would pick x_j as its neighbor under the assumption that it was picked from a Gaussian distribution centered at x_i. The parameter σ_i for each data point is automatically determined using a hyper-parameter called *perplexity* that is usually insensitive. This parameter is an integer and specifies the effective number of neighbors. In general, a larger data set requires a higher perplexity. Probabilities in $\mathbb{R}^{d'}$ are modeled by

$$q_{ij} = \frac{\left(1 + ||y_i - y_j||^2\right)^{-1}}{\sum_{m \neq l} \left(1 + ||y_m - y_l||^2\right)^{-1}} \tag{2.3}$$

Using the long-tailed student-t distribution instead of the Gaussian has the advantage that it allows to avoid the "crowding problem" in low dimensional spaces, leaving more space for distant pairs of points. The Kullback-Leibler divergence $C = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$ is used to minimize the difference between both probability distributions by numerical optimization. Because the t-SNE cost function is not convex, the resulting representation is likely to originate from a local optimum, although it is usually satisfying.

The sum over all pairs of data points in the denominator of both p_{ij} and q_{ij} requires a quadratic computational runtime and memory complexity, making it unsuitable for larger data sets such as metagenomes. To overcome this limitation, an approximative method named "Accelerated t-SNE" (Van Der Maaten, 2014) approximates the similarities between input points, effectively reducing its runtime complexity to linearithmic time while using only linear memory. This is achieved by using sparse approximations of the probabilities p_{ij}. Observing that p_{ij} for distant pairs of points is nearly infinitesimal, these can be neglected without substantial adverse consequences. This boils down to finding only a subset of neighbors which is efficiently done using vantage point trees. Accelerated t-SNE employs a robust parameter $0 \leq \theta \leq 1$ that controls the trade-off between speed and accuracy, where $\theta = 0$ corresponds to the original t-SNE method and $\theta > 0$ controls the amount of approximation. Because Accelerated t-SNE heavily relies on approximations, the resulting representation can contain artifacts such as points that belong one cluster appearing wrongly in another cluster. Such artifacts can also result from the original t-SNE, for example when stuck in a low quality local optimum. However, this can be remedied by running t-SNE multiple times and taking the result with the lowest cost function value.

Even though t-SNE is widely applied in many fields, the algorithm's success has not been fully understood, yet. A particular aspect is the suitability of the method for clustered data. Given that the perplexity parameter is chosen correctly, t-SNE is capable of preserving cluster structures very well. To support this observation, Shaham and Steinerberger, 2017 proved that under certain weak assumptions, and if the input data is already well separated, an optimal t-SNE embedding into any dimension preserves cluster structure. Interestingly, this fact does not depend on the number of clusters. Therefore, it makes t-SNE well suited for metagenomic binning.

2.2.3 Cluster Analysis

The segmentation of a set of objects into groups such that similar objects are placed in the same group while dissimilar objects are placed in different groups, is called clustering. Any such group is called a cluster of objects. Clusters occur naturally in many types of structured data. A simple example is given by Figure 2.4, which shows exemplars of the *Iris* flower, defined

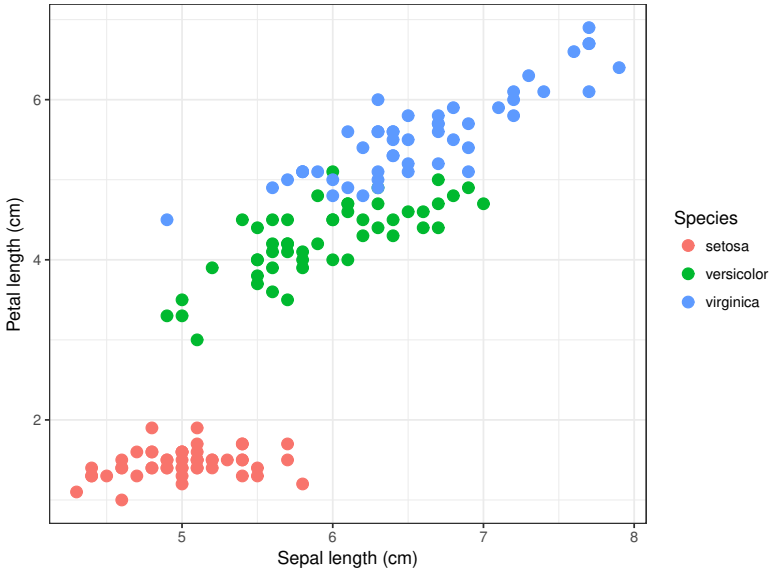

Fig. 2.4.: Example of clusters that represent three different species of the *Iris* flower. While *I. setosa* has a clearly smaller petal length, *I. versicolor* and *I. virginica* are more similar.

by sepal and petal lengths. From the coloring by species, three clusters are visible, with two species being more similar to each other than a third one.

In the absence of cluster labels, it is the task of a clustering algorithm to group objects, which are data points defined by a number of features, into their respective cluster. As the notion of a cluster is not well-defined, this is an ill-posed task: the definition of a cluster is driven by human perception, and given one data set, people will give different answers on the question of how many clusters are visible (Jain and Dubes, 1988). Exemplary in Figure 2.4, if there was no colored labeling by species, some people would guess the number of clusters to be two instead of three. Therefore, there cannot be one single correct definition of what constitutes a cluster, and many algorithms for different types of data and clustering objectives exist. As the goal of this chapter is to evaluate cluster analysis techniques in the context for metagenomic binning, I applied a number of algorithms, each with different objectives, and methods for estimating cluster validity. In the following, I will briefly introduce those techniques.

Clustering algorithms

A clustering algorithm can be seen as a mapping $C : \mathbb{R}^d \to \{1, \ldots, k\}$, where each data point $x_i \in \mathbb{R}^d$ is assigned one integer $C(x_i)$, representing one out of k unique clusters. There exist many categorizations of clustering algorithms (Estivill-Castro, 2002). A general distinction can be made using five different types, being based on:

- *centroids*: clusters are defined by a single data point that can be an average of its cluster members, or itself be a cluster member.
- *distributions*: clusters are based on the underlying assumption that they were generated by a certain data distribution.
- *connectivity*: clusters are defined based on having loose or tight connections, often in a hierarchical way.
- *density*: data points in regions with high density constitute a cluster.
- *graphs*: clusters are given by properties of a graph that is constructed from the data.

Next, one or more examples of algorithms for each of these types are described. In the remainder of this thesis, the number of clusters will be referred to as k and the correct (optimal) number of clusters is k_{opt}. Since the algorithms are not the main focus of the thesis, they are introduced only briefly.

K-Means Perhaps the most recognized clustering algorithm is k-Means (KM, described in detail in Hastie et al., 2009). Being an exemplar of centroid-based methods, it optimizes

$$\min \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - \mu_j\|^2 \tag{2.4}$$

where μ_j denotes the mean of cluster j. Hence, it simply minimizes the within-cluster variance, and because of that is restricted to convex clusters only. Because the search for an optimal solution is NP-hard, different ap-

proximation algorithms exist. For example, a solution can be found based on expectation-maximization (Bottou and Bengio, 1995) by randomly initializing centers μ_j, and iteratively assigning each data point x_i to its nearest center, followed by a re-calculation of means. Unfortunately, the quality of the found solution heavily depends on the initialization of means. Another approach named k-Means++ (KM++, Arthur and Vassilvitskii, 2007) tries to improve this point by spreading the initial cluster centers according to the data and already chosen centers.

Neural Gas Another centroid-based algorithm is given by Neural Gas (NG, Martinetz et al., 1993). Similar to self-organizing maps (SOM), the cluster centers are given by SOM weight neurons. In contrast to SOMs however, NG does not use a fixed lattice. As in k-Means, it minimizes within-cluster sum-of-squares distances, resulting in a Voronoi-tesselation, which also restricts NG to convex clusters. The cluster centers are found by iteratively and randomly selecting a data point, and instead of only updating one winning center weight, all weights are updated by a soft-max rule, taking into account a neighborhood ranking of the weights.

Gaussian Mixture Model As an example for distribution-based methods, Gaussian Mixture Models (GMM, described in detail in Hastie et al., 2009) were included. As GMM models are also based on optimization by expectation-maximization, they are much related to k-Means. However, they assume a Gaussian distribution as a data-generating model. Besides fitting the cluster center, this requires the fitting of covariances, as well. Effectively, this makes GMMs a soft version of k-Means, where data points are not a hard member of one particular cluster, but are rather defined based on membership probabilities for each cluster. The convergence of GMMs is much slower than the original k-Means algorithm (Bishop, 2006). Therefore, it is useful to initialize the Gaussian parameters with estimations from clusters found by k-Means or k-Means++. In the following, the latter is referred to as GMM++.

Hierarchical Clustering The most commonly used connectivity-based clustering method is Hierarchical Clustering (HC). It works either in a top-down or bottom-up fashion. In the former, starting with one cluster for all data points, the cluster is recursively split until each data point is in its own cluster. Conversely, in bottom-up, starting with one cluster per data point, they are merged until everything is in one cluster. This results in a cluster

hierarchy, where on each tree level two clusters are split or merged. The key step lies in the determination of which clusters to link. Here, a number of linkage criteria exist, based on the distances between members of each cluster. A common criterion is average linkage, where distance between two clusters A and B is given as the mean distance between all points in both:

$$d(A, B) = \frac{1}{|A| \cdot |B|} \sum_{x_a \in A} \sum_{x_b \in B} d(x_a, x_b) \tag{2.5}$$

The two clusters with minimum distance are then linked together. Finally, the resulting linkage tree is cut below level k to extract an assignment with k clusters.

DBSCAN An example of density-based clustering is given by the popular DBSCAN algorithm (DBS, Ester et al., 1996). It defines density by the notion of different types of points. First, core points are ones that have $minPts$ points within a radius of ϵ. Both values have to be provided by the user. Second, density-reachable points are defined by being connected to at least one core point over a path of other core points. This type of point can be thought to appear at cluster borders. Third, outlier points are neither core points, nor density-reachable. Given a good understanding of the data, choosing $minPts$ and ϵ is straightforward for a user. Two advantages of the algorithm include its direct estimation of the number clusters and the applicability to arbitrarily shaped clusters. It is difficult to apply to data which have large differences in density, because the supplied parameters fit for a certain density distribution only.

Spectral Clustering Last, a prominent graph-based algorithm is Spectral Clustering (SC, Von Luxburg, 2007). Here, a graph is constructed on top of the data. For example, in a nearest-neighbor graph, every data point is connected by an edge to its k_{nn} nearest neighbors. Given the adjacency matrix A and degree matrix D, the normalized Laplacian matrix is given by $L = D^{-1/2} A D^{-1/2}$. Decomposing L, Spectral Clustering takes the eigenvectors for its k (number of clusters) largest eigenvalues and clusters them row-wise using a partitional algorithm such as k-Means++. SC basically calculates a normalized cut (Shi and Malik, 2000) and can find arbitrarily shaped clusters. Additionally, the algebraic multiplicity of the eigenvalue zero defines the number of connected components (CC), which is a simple way of determining the number of clusters. However, a computationally

less expensive way to count the number of connected components is given by Tarjan's algorithm (Tarjan, 1972).

Clustering evaluation

The term clustering evaluation summarizes techniques for the determination of the number of clusters k. Often, the structure of a given data set is unknown. As most clustering algorithms require k to be specified as a parameter, it has to be estimated first. Because the notion of a cluster is not well-defined (section 2.2.3), a variety of methods to estimate k exist. The term "clustering evaluation" stems from the fact that, in order to estimate k_{opt}, usually a number of clusterings for different k have to be evaluated with respect to certain criteria of validity. In general, these criteria can be divided into external and internal cluster validity indices. External indices take into account information other than the data points themselves, mostly an existing reference clustering. The result of such indices can then be used to evaluate the clustering quality of an algorithm on known data, in order for it to perform well on unknown data as well. In contrast, internal validity measures only use information inherent to the data itself. In the evaluation of metagenomic binning, one external and five internal cluster validity indices are used. I will discuss them in the following.

Given a reference clustering, the evaluation of another clustering is as simple as comparing wrongly and correctly clustered data points. For that task, the Jaccard index (Levandowsky and Winter, 1971) is commonly used. The index itself measures the similarity between two sets A and B: $J = |A \cap B| / |A \cup B|$. Hence, if the two sets are equal, their intersection equals their union and $J = 1$. Now, given two clusterings, their similarity can be calculated by comparing the two respective sets of pairs of points that are clustered together in both. More formally (Ben-Hur et al., 2001), a labeling for a given clustering can be represented by a matrix C with components $C_{ij} = 1$, if x_i and x_j belong to the same cluster and $i \neq j$, and $C_{ij} = 0$, otherwise. In order to compare two labelings $C^{(1)}$ and $C^{(2)}$, let N_{ij} for $i, j \in \{0, 1\}$ be the number of entries on which $C^{(1)}$ and $C^{(2)}$ have values i and j, respectively. Then, the Jaccard index is defined as

$$J(C^{(1)}, C^{(2)}) = \frac{N_{11}}{N_{01} + N_{10} + N_{11}} \tag{2.6}$$

If, in both clusterings, all pairs of points appear in the same clusters, $N_{11} = 1$, $N_{01} = N_{10} = 0$, and $J = 1$. The number N_{01} counts false positives, and N_{10} are false negatives. Besides the Jaccard index, there exist a number of other external validity indices, for example the Adjusted Rand index (Rand, 1971). Such measures mainly differ in their way of counting entries in the contingency table.

Next, different methods for the assessment of internal cluster validity are discussed. In some sense, all of these methods define what constitutes a cluster differently.

Dunn, Davies-Bouldin and Xie-Beni index A common notion is based on intra- and inter-cluster distances that measure the compactness and separation of clusters: in a good clustering, pairwise intra-cluster distances are small (making the cluster compact), and inter-cluster distances are large (setting a good separation between clusters). A detailed analysis of measures that take these distances into account is given in Ben-Hur et al., 2001. From there, three selected examples are given:

- The Dunn index (DUNN) uses the minimum pairwise distance between data points in different clusters to define separation, while the maximum diameter among all clusters defines compactness.

- The Xie-Beni index (XB) simply defines separation as the minimum square distance between cluster centers, and compactness as the within-cluster mean-square distance to the center.

- For each cluster C, the Davies-Bouldin index (DB) calculates the maximum similarity between C and all other clusters and averages these maxima to obtain a validity index for the whole data set.

Given a set of clusterings, with different partitionings and different number of clusters, an optimal clustering is then given by the minimum or maximum of such indices.

Gap statistic Another very popular method to assess the number of clusters is given by the Gap Statistic (GAP, Tibshirani et al., 2001). In contrast to the measures discussed before, where the index is directly given from a

certain clustering, the Gap Statistic compares its compactness to an expected reference null distribution. The compactness is simply measured by the pooled within-cluster sum of squares around the cluster means (dispersion). The key idea behind is that clustering a reference uniform distribution will produce less compact clusters than the optimal clustering with $k = k_{opt}$, and any sub-optimal clustering with $k \neq k_{opt}$ is nearer to the reference distribution. A simplified formalization is to maximize

$$Gap_n(k) = \mathbb{E}_n \left(\log W_k\right) - \log W_k \tag{2.7}$$

$$W_k = \sum_{r=1}^{k} \sum_{x_i, x_j \in C_r} d(x_i, x_j) \tag{2.8}$$

where W_k denotes the within-cluster dispersion for a k-clustering, and its expectation is calculated by generating a reference distribution with same size and dimensionality B times, clustering each one. This step makes the Gap Statistic computationally expensive, because in order to obtain a reliable expected dispersion, $B \geq 100$ is usually large. The estimated number of clusters is chosen as $k =$ smallest k' such that $Gap(k) \geq Gap(k+1) - s_{k+1}$, where s_k is the observed standard deviation of the dispersion. This rule is also known as "elbow" rule, because k_{opt} is located at a perceived elbow in the distribution of gap values.

Validity based on stability A last method to estimate k, considered in this thesis, can be seen in measuring the clustering stability with respect to random sub-sampling (SS, Ben-Hur et al., 2001; Von Luxburg, 2010). For different values of k, a fraction $f = 0.8$ of the data is sub-sampled two times and each subset is clustered. Both clusterings can be compared by taking the intersection of both sets and evaluating them using an external validity measure, such as the Jaccard index. While for an optimal clustering, each cluster is labelled uniquely, for $k \neq k_{opt}$, either multiple clusters are assigned to one label ($k < k_{opt}$) or multiple labels are assigned to one cluster ($k > k_{opt}$). Because most clustering algorithms are not deterministic with respect to different sub-samples, a sub-optimal clustering is unstable because it is likely that assignments change for each sub-sample. In contrast, given an optimal clustering, an algorithm is assumed to be stable. However, in practice, even for $k < k_{opt}$ clusterings can be stable. For that reason, k_{opt} is determined as the largest k for which the clustering similarity is above a certain threshold.

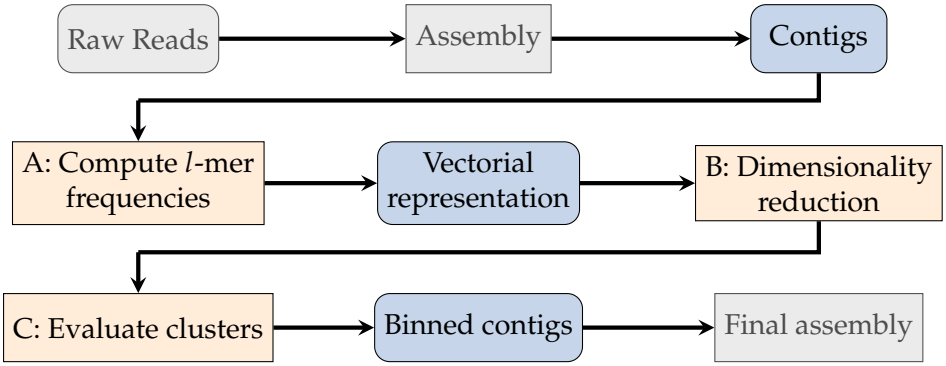

Fig. 2.5.: Metagenomic binning pipeline. Grey boxes represent steps with which the data are processed either before or after the actual binning. Short reads from the sequencing process are first assembled into contigs. Based on tetramer frequencies, a vectorial representation is obtained, and subsequently reduced to two dimensions. On this basis, the number of clusters is estimated and the actual binning takes place. The resulting assignments can be used to refine the metagenome assembly.

2.2.4 Binning pipeline

To perform metagenomic binning, I propose a pipeline as outlined in Figure 2.5. Input to the pipeline are contigs which resulted from the assembly of a given metagenome. To convert sequences into vectorial data, a sliding window approach (section 2.2.1) is used to calculate oligonucleotide frequencies (step A). As this data has high dimensionality, it has to be mapped (section 2.2.2) to a lower-dimensional manifold (step B). The resulting representation is suited for clustering and it is the task to estimate the number of clusters (step C, Equation 2.2.3) and finally bin the contigs.

2.3 Evaluation

The influence of a variety of method choices and hyper-parameters involved in a metagenomic binning pipeline is assessed, ultimately to give guidance for future research on using dimensionality reduction and clustering for metagenomic binning. In the following, used data sets are introduced and described (section 2.3.1). Next, steps A–C as shown in Figure 2.5 are evaluated thoroughly. For a clustering algorithm to correctly identify metagenomic clusters, the correct representation thereof is essential. It is shaped by both oligonucleotide frequency calculation and dimensionality reduction. Therefore, parameters of steps A&B are evaluated (section 2.3.2, 2.3.3) before

suitable clustering algorithms are going to be discussed (section 2.3.4). Suitable choice for all involved methods and parameters are then applied to low, medium and high complexity metagenomes, closely resembling real-world assemblies (section 2.3.5).

2.3.1 Data

In order to evaluate the performance of the ingredients of a binning pipeline, a total of 16 metagenomic samples from three different sources are used, including theoretical and simulated data. The genomic composition of real-world metagenomes is often unknown, and hence it is not possible to define a gold standard binning to compare to. Even though existing binnings for such metagenomes may be used, only the use of theoretical and simulated data enables full and 100% correct control over evaluation.

`THEO-k` In order to verify the correct functionality of the used algorithms, six theoretical data sets with $k \in \{5, 10, 20, 30, 40, 50\}$ clusters were generated. Each cluster contains a random number $r \in \{25, \ldots, 75\}$ of data points $x \in \mathbb{R}^2$ which are sampled uniformly from an ellipse with center μ_k and random principal axis lengths $a, b \in [0, 1]$. This procedure generates convex clusters of different sizes and shapes. To ensure non-overlapping clusters, the centers μ_k are randomly chosen from an equally spaced lattice set of data points using a probabilistic seeding rule (Arthur and Vassilvitskii, 2007). It gives centers that are located near already selected centers a lower selection probability than centers that are more distant.

`NCBI-k` Seven metagenomic samples containing $k \in \{5, 9, 10, 20, 30, 40, 50\}$ genomes were simulated. The data was generated by randomly selecting k complete microbial genomes from the NCBI database (for $k \neq 9$) (Geer et al., 2009) and simply concatenating the finished assemblies. This way, the evaluation process can rely on correct ground truth data. A special case is given by `NCBI-9`, for which genomes were selected manually according to low cophenetic similarity based on a dendrogram constructed using 16S sequence identity, resulting in nine related archaea, including three closely related methanogens. A list of all included genomes is given in Table A.1. In the following section, the `NCBI-9` metagenome is used to assess suitable parameter values which concern sequence signature generation and dimensionality reduction. All other `NCBI` samples are then used to

verify those choices, while simultaneously assessing suitable choices for of clustering algorithms.

`CAMI-k` The "Critical Assessment of Metagenome Interpretation" initiative (CAMI, Sczyrba et al., 2017) provides benchmark metagenomes that were generated from a large number of newly sequenced microorganisms and novel viruses and plasmids, including genomes with varying degrees of relatedness to each other and to publicly available ones, representing common experimental setups. All samples were simulated by generating 150 bp reads with an Illumina HighSeq error profile, providing a gold standard assembly and binning. Included species have partial strain-level diversity, viruses and plasmids, providing a near-realistic simulation. Metagenomes with low, medium and high complexity ($k \in \{30, 93, 306\}$, respectively) were used to assess the performance of evaluated parameters on these assemblies. The numbers of genomes contained in the original data is higher (Table 2.2) than the numbers of clusters k_{opt} in this data. This is due to the setting of a minimum contig length as described in section 2.3.5. The `CAMI-k` data set is used to apply the method and parameter choices assessed earlier on data that closely resemble real-world assemblies, including a significant amount of strains.

2.3.2 Dimensionality reduction

A suitable data representation optimally produces compact and separated clusters, in order for a clustering algorithm to pick up the structure without any problems. For that, the calculation of oligonucleotide frequencies and dimensionality reduction go hand in hand. Even though, in the binning pipeline the former is followed by the latter, for reasons of understandability, they are discussed in reverse order.

To quantitatively evaluate different dimension reduction techniques, possible combinations of oligonucleotide frequency window size and step parameters on the `NCBI-9` data set were evaluated. As good compactness and large separation of clusters will have a positive impact on the quality of a subsequent clustering, the Davies-Bouldin index (DB) as an evaluation measure is used. As it is shown later in this chapter, this index is a good choice for estimating the number of clusters, too. Additionally, using the true class label information, the 9-nearest-neighbor classification error (9-

NN) is evaluated. For a given data point, it is defined as the proportion of its 9 nearest neighbors that are not assigned to the same cluster. The value is averaged over the full data set, normalized by the number of data points that result from the given window parameters. Because this methodology takes into account a wide range of window parameters and because it only relies on characteristics specific to the embedding, it rules out eventual influences of both the oligonucleotide frequency calculation and the clustering step.

Three choices for dimensionality reductions (DR) are reviewed: no DR, linear DR using PCA, and nonlinear DR using t-SNE. For that task, the distributions of DB and 9-NN values for different window-sizes were compared using a one sided Mann-Whitney U test (null hypothesis: tested distributions are equal). The columns of Figure 2.6 depict the DR techniques in question, while the rows correspond the test measure. As it is visible, using t-SNE as a dimension reduction method yields the best values: it has a significantly better (smaller) DB index both compared to no dimensionality reduction and linear PCA (both $p < 10^{-24}$), indicating clusters that are more compact and separated. The same holds for the 9-NN error (compared to no dimensionality reduction: $p = 0.0033$; PCA: $p = 10^{-33}$). This confirms earlier preliminary results as reported by Gisbrecht et al., 2013 and Laczny et al., 2014. Visually inspecting the difference of exemplary 2D embeddings as shown in Figure 2.7 also supports these findings. It is clearly visible that t-SNE (right column) delivers embeddings with clusters that are much more compact and separated when compared to PCA (left column) where clusters are partially fuzzy and overlapping. It is worth to note that, since a non-linear method resolves clusters well, there has to be some non-linearity in the data. In the same way, the one-dimensional GC-content may result in linear inseparability for more related species (Figure 2.2), similar effect may be expected in the more general, high-dimensional oligonucleotide feature set.

Parameters of t-SNE are mostly robust to small changes. However, it was observed that some parameter values deliver better results in terms of the above evaluation measures. Initially, t-SNE performs a PCA to remove noise from the data. Here, the number of dimensions was set to 50 as it was found to work well in all settings. This is supported by the fact that the kept variance for this value is high ($> 99\%$). Next, the t-SNE perplexity was chosen as $\text{perp}(n) = \lfloor \log(n)^2 \rfloor$, with n being the number of data points. This way, a small number of data points receives a small

perplexity whereas with growing n the perplexity saturates. Consequently, given the assumption that with fewer data points, clusters contain fewer data points, they can still be resolved. At the same time, with larger data, the perplexity values do not grow as much, as suggested in Van Der Maaten, 2014. Finally, the parameter θ controls the trade-off between speed and accuracy of the resulting embedding. While still delivering reasonably good results, the default value of $\theta = 0.5$ was found to be too inaccurate, resulting in less compact clusters. Here, a value of $\theta = 0.2$ was chosen, taking the consequence of a higher computational complexity.

2.3.3 Data representation by oligonucleotide frequencies

As pointed out in section 2.2.1, the vectorial representation, and with it, the resulting clustering heavily depends on the choice of window parameters. Different combinations of these parameters were evaluated. First, different lengths of oligonucleotides (l-mers) are reviewed. In general, the Davies-Bouldin index decreases with increasing l, with improvements for $l > 4\,\text{bp}$ being negligible (sometimes even worse) such that tetramers ($l = 4\,\text{bp}$) are used in all further settings, avoiding a significantly higher computational complexity for larger choices of l.

Figure 2.6 depicts the effects of different window sizes w and steps Δw on the compactness and separation as well as on the 9-nearest-neighbor error exemplary on the `NCBI-9` data set (similar results were obtained on all other data sets). Window size values ranged from $w = 2000\,\text{bp}$ to $w = 0.1 \cdot \max_g \text{length}(s_g)$ where s_g denotes the sequence of a contained genome g. Window step values $\Delta w = w / w_{div}$ in the range $w_{div} \in [1, 2]$ were tested, resulting in half to no overlapping windows. Looking at only the t-SNE results on the right, it is visible that starting from $w = 2000\,\text{bp}$ there is an improvement both in terms of DB and 9-NN. The same effect can be seen in Figure 2.7 where the embeddings of smaller window sizes are noisier than embeddings with a larger window size. As a smaller window size captures more of the local characteristics, this manifests itself in a higher variance, resulting in more noise. In opposite, a larger window size will cancel out such effects. Another advantage of the latter lies in the fact that it generates fewer data, keeping computational complexity low. This raises the question what window size is large enough?

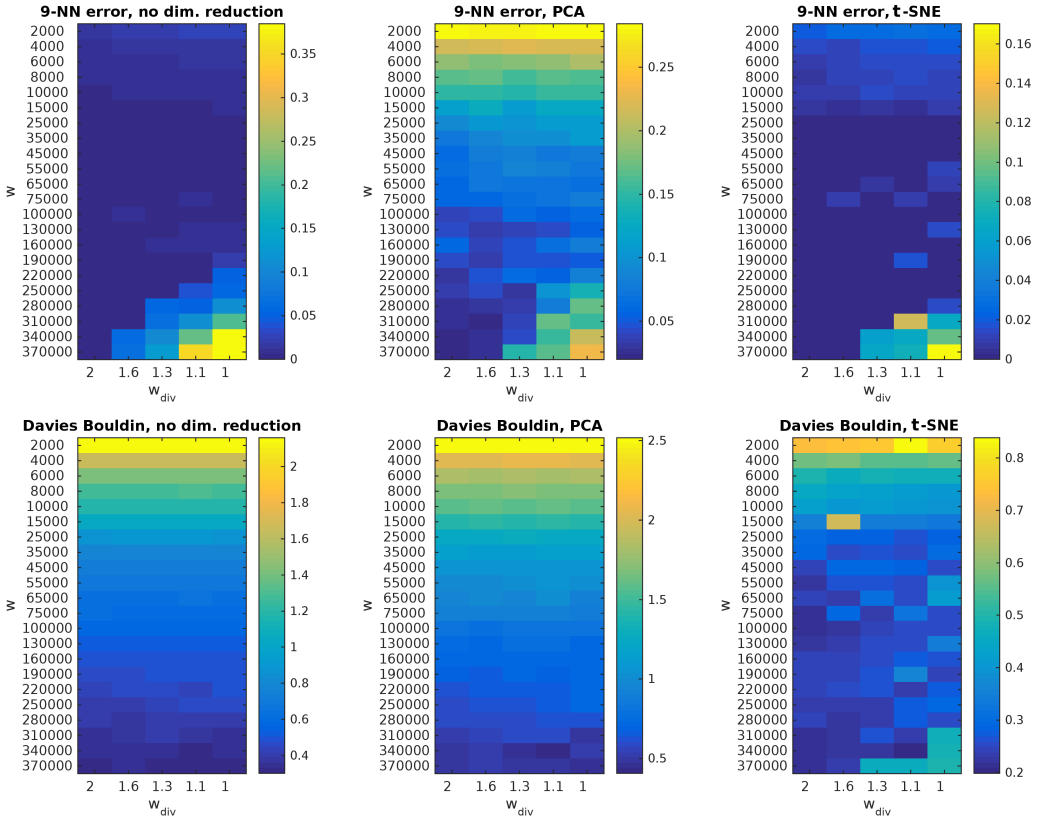

Fig. 2.6.: Illustration of the effects of different oligonucleotide window sizes and steps on the 9-nearest-neighbor error (top) and Davies-Bouldin index (bottom) using no dimension reduction (left), PCA (center) and t-SNE (right) on the `NCBI-9` data set. Tetramer frequencies were used ($l = 4\,\text{bp}$). The window step is indirectly given by $\Delta w = \frac{w}{w_{div}}$, indicating half to no overlapping windows.

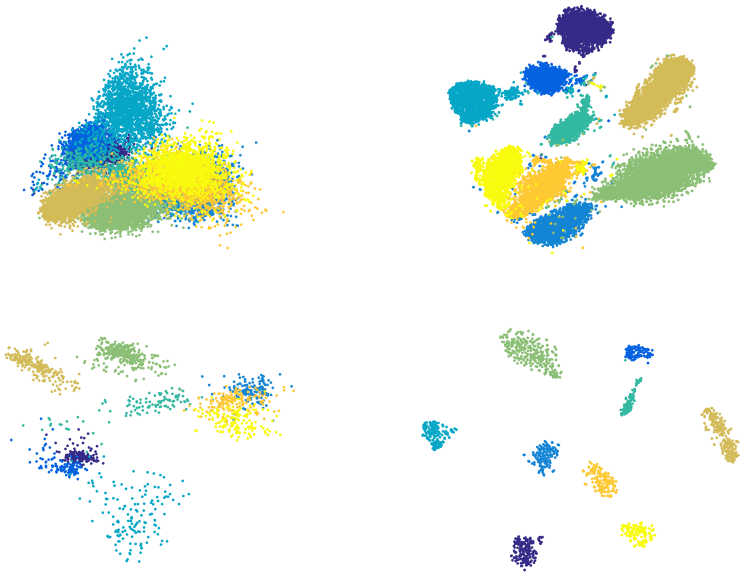

Fig. 2.7.: Different projections of the NCBI-9 data set with PCA (left) and t-SNE (right) using tetramer frequencies ($l = 4\,\text{bp}$) computed with different window sizes (top: $w = 2000\,\text{bp}, \Delta w = 1000\,\text{bp}$; bottom: $w = 25\,000\,\text{bp}, \Delta w = 12\,500\,\text{bp}$).

Further investigation of the t-SNE plots in Figure 2.6 shows that 9-NN stays optimal on a large range, only getting worse for extremal w. DB does not show such a stable behavior in this area, being smaller towards areas with larger window sizes and steps. This would indicate that choosing w and Δw as large as possible is optimal. However, a very large window size will also result in only very few data points for each cluster. This is even more pronounced if there is only small to no overlap between, which can also be seen in Figure 2.6 where DB and 9-NN increase for such values. A certain number of data points is also needed for clustering algorithms. As these utilize data characteristics such as centroids, distributions or densities, a small number of data points will result in inaccurate estimations of such. Additionally, window sizes in the range of $w = 1 \times 10^5\,\text{bp}$ might be larger than most contigs, making them inapplicable. Therefore, it can be argued that choosing window sizes in the range of $w \in [25\,000\,\text{bp}, 50\,000\,\text{bp}]$ and $\Delta w = \frac{w}{2}$ is a good default value, which is also consistent with the observations on other data sets and with respect to clustering.

However, for large metagenomes, choosing window sizes in that range can be computationally intractable, depending on the available hardware. For example, already for a sample such as `CAMI-306`, containing genomes with

a total size of $n_{bp} = 2.8\,\text{Gbp}$, and choosing $\Delta w = 25\,\text{kbp}$, around 112000 data points are generated. As hierarchical clustering requires quadratic memory complexity, on a 64 bit system, storing the matrix would require around 93 GB of main memory. Therefore, it might be necessary to increase the window size in order to accommodate the data in memory. Given a target number of data points n, one can estimate the according parameters $\Delta w = \lceil \frac{n_{bp}}{n} \rceil$ and $w = 2 \cdot \Delta w$.

2.3.4 Clustering algorithms and cluster validation

Having determined an optimal range of parameter values both for oligonucleotide frequency calculation and dimension reduction, it is the task to cluster the data. Here, a particularly difficult step lies in finding the correct number k_{opt} of genomes/clusters which is unknown for real world data. A quantitative comparison of different combinations of state-of-the-art clustering algorithms and procedures for estimating k_{opt} was performed, in order to find combinations which work well when applied to metagenomic data. For each data set `THEO-k` and `NCBI-k`, the estimated number k of clusters was evaluated. To account for the possibility that k was estimated correctly but the actual partitioning is wrong, the Jaccard-index J for each respective optimal clustering was calculated. The results are depicted in Table 2.1. Here, missing values either indicate undefined values (e.g. caused by empty clusters) or tasks that were skipped because of a too long computation time. The Jaccard index of SS is missing because in its stability calculation, there is no specific optimal clustering.

Looking at the theoretical data, most combinations of algorithms and k-estimation methods deliver good to optimal results. For larger k preferences, differences arise. Specifically, KM++ and GMM++ deliver better results than their randomly initialized counterparts KM and GMM. In most cases, GAP fails to find a correct partitioning. Still, for all theoretical data sets, the corresponding gap curves show a clear elbow close to k_{opt}. However, for $k > k_{opt}$, the gap value approximately stays the same, in some cases even slightly increases. In the presence of a non-negligible standard error, this makes it difficult to automatically estimate the correct elbow location. As an exception, for $k_{opt} = 40$, GAP gives a perfect estimation for some clustering algorithms. Here, for $k > k_{opt}$, the gap value decreases, enabling a correct elbow location. Furthermore, when applied using an appropriate

THEO	DB		DUNN		XB		GAP		SS
	K	J	K	J	K	J	K	J	K
	$k_{opt} = 5$								
KM	**5**	**1.00**	**5**	**1.00**	3	0.56	55	0.10	**5**
KM++	**5**	**1.00**	**5**	**1.00**	**5**	**1.00**	64	0.09	6
NG	**5**	**1.00**	**5**	**1.00**	**5**	**1.00**	**5**	**1.00**	**5**
HC	**5**	**1.00**	**5**	**1.00**	**5**	**1.00**	65	0.12	6
GMM	**5**	**1.00**	**5**	**1.00**	-	-	7	0.80	**5**
GMM++	**5**	**1.00**	**5**	**1.00**	-	-	12	0.47	7
SC	**5**	**1.00**	**5**	**1.00**	**5**	**1.00**	-	-	20
DBS	**K = 5, J = 1.00**								
CC	**K = 5, J = 1.00**								
	$k_{opt} = 10$								
KM	12	0.90	9	0.92	-	-	29	0.36	2
KM++	**10**	**1.00**	**10**	**1.00**	**10**	**1.00**	32	0.34	**10**
NG	9	0.93	**10**	**1.00**	11	**1.00**	9	0.93	2
HC	**10**	**1.00**	**10**	**1.00**	**10**	**1.00**	26	0.46	14
GMM	12	0.95	**10**	**1.00**	-	-	13	0.78	1
GMM++	**10**	**1.00**	**10**	**1.00**	-	-	15	0.71	11
SC	**10**	**1.00**	**10**	**1.00**	**10**	**1.00**	-	-	13
DBS	**K = 10, J = 1.00**								
CC	**K = 10, J = 1.00**								
	$k_{opt} = 20$								
KM	23	0.79	8	0.36	-	-	51	0.42	2
KM++	**20**	**1.00**	**20**	**1.00**	**20**	**1.00**	56	0.44	25
NG	**20**	0.89	15	0.74	18	0.86	18	0.80	5
HC	**20**	**1.00**	**20**	**1.00**	**20**	**1.00**	56	0.54	39
GMM	28	0.81	5	0.22	-	-	13	0.55	1
GMM++	**20**	**1.00**	**20**	**1.00**	**20**	**1.00**	40	0.58	23
SC	33	0.55	**20**	**1.00**	7	0.11	-	-	21
DBS	**K = 20, J = 1.00**								
CC	**K = 20**, J = 0.88								
	$k_{opt} = 30$								
KM	27	0.80	19	0.63	-	-	67	0.54	1
KM++	**30**	**1.00**	**30**	**1.00**	**30**	**1.00**	68	0.50	39
NG	26	0.86	34	0.94	32	0.94	24	0.83	3
HC	**30**	**1.00**	**30**	**1.00**	**30**	**1.00**	49	0.79	62
GMM	25	0.73	23	0.59	-	-	9	0.24	1
GMM++	**30**	**1.00**	**30**	**1.00**	**30**	**1.00**	44	0.80	35
SC	33	0.93	31	0.97	31	0.97	-	-	31
DBS	**K = 30, J = 1.00**								
CC	K = 31, J = 0.97								
	$k_{opt} = 40$								
KM	39	0.66	21	0.49	-	-	68	0.65	1
KM++	39	0.96	35	0.77	35	0.77	**40**	**1.00**	43
NG	34	0.82	29	0.64	45	0.96	26	0.58	3
HC	39	0.96	35	0.77	35	0.77	**40**	**1.00**	61
GMM	44	0.62	9	0.21	-	-	6	0.15	3
GMM++	39	0.96	35	0.77	35	0.77	**40**	**1.00**	44
SC	35	0.56	3	0.03	2	0.28	-	-	41
DBS	K = 35, J = 0.77								
CC	K = **40**, J = 0.88								
	$k_{opt} = 50$								
KM	29	0.52	16	0.27	-	-	63	0.63	3
KM++	**50**	**1.00**	**50**	**1.00**	**50**	**1.00**	**50**	**1.00**	54
NG	35	0.63	51	0.83	43	0.74	24	0.45	11
HC	**50**	**1.00**	**50**	**1.00**	**50**	**1.00**	**50**	**1.00**	70
GMM	49	0.59	9	0.15	-	-	3	0.06	2
GMM++	**50**	**1.00**	**50**	**1.00**	**50**	**1.00**	**50**	**1.00**	55
SC	58	0.67	2	0.02	4	0.02	-	-	13
DBS	K = 36, J = 0.48								
CC	K = 51, J = 0.65								

NCBI	DB		DUNN		XB		GAP		SS
	K	J	K	J	K	J	K	J	K
	$k_{opt} = 5$								
KM	4	0.87	4	0.87	4	0.87	12	0.38	**5**
KM++	**5**	**1.00**	4	0.87	4	0.87	11	0.41	7
NG	**5**	**1.00**	4	0.87	4	0.87	10	0.45	8
HC	**5**	**1.00**	4	0.87	4	0.87	6	0.80	6
GMM	**5**	0.98	4	0.87	-	-	7	0.69	**5**
GMM++	**5**	0.98	4	0.87	4	0.87	-	-	8
SC	4	0.73	4	0.73	4	0.73	-	-	10
DBS	K = 4, J = 0.87								
CC	K = **5**, J = 0.67								
	$k_{opt} = 10$								
KM	**10**	**0.98**	6	0.63	9	0.94	68	0.13	4
KM++	**10**	**0.98**	**10**	**0.98**	**10**	**0.98**	53	0.16	11
NG	**10**	**0.98**	6	0.63	8	0.89	17	0.50	12
HC	**10**	**0.98**	**10**	**0.98**	**10**	**0.98**	18	0.50	11
GMM	9	0.79	8	0.71	-	-	13	0.89	9
GMM++	**10**	**0.98**	**10**	**0.98**	**10**	**0.98**	50	0.20	11
SC	2	0.13	2	0.13	2	0.13	-	-	12
DBS	**K = 10, J = 0.98**								
CC	K = 12 J = 0.79								
	$k_{opt} = 20$								
KM	22	0.85	5	0.32	-	-	61	0.26	3
KM++	22	0.73	16	0.91	16	0.90	63	0.24	**20**
NG	**20**	0.74	4	0.27	13	0.76	30	0.53	18
HC	19	**0.97**	17	0.90	18	0.96	39	0.44	19
GMM	18	0.91	16	0.85	-	-	16	0.88	1
GMM++	16	0.87	17	0.94	17	0.92	34	0.48	**20**
SC	14	0.61	12	0.46	11	0.44	-	-	21
DBS	K = 18, J = 0.96								
CC	K = 24, J = 0.58								
	$k_{opt} = 30$								
KM	23	0.78	10	0.40	-	-	60	0.45	4
KM++	**30**	**0.93**	12	0.49	17	0.63	51	0.46	**30**
NG	27	0.77	18	0.71	26	0.92	28	0.80	16
HC	27	0.91	13	0.53	26	0.89	65	0.47	36
GMM	23	0.75	52	0.67	-	-	17	0.65	1
GMM++	31	0.84	19	0.70	21	0.76	55	0.49	**30**
SC	2	0.44	9	0.14	2	0.04	-	-	14
DBS	K = 28, J = 0.91								
CC	K = 39, J = 0.22								
	$k_{opt} = 40$								
KM	**39**	0.75	17	0.53	-	-	70	0.53	4
KM++	46	0.75	25	0.70	24	0.66	68	0.49	4
NG	37	0.81	17	0.53	16	0.51	31	0.78	14
HC	**41**	0.92	11	0.35	36	**0.95**	53	0.69	48
GMM	29	0.78	17	0.49	4	0.12	17	0.45	1
GMM++	43	0.81	29	0.80	4	0.12	-	-	1
SC	31	0.6	4	0.04	7	0.06	-	-	-
DBS	K = 36, J = 0.91								
CC	K = 49, J = 0.25								
	$k_{opt} = 50$								
KM	40	0.76	34	0.71	-	-	66	0.70	1
KM++	**50**	0.81	68	0.64	31	0.68	69	0.63	4
NG	42	0.80	37	0.80	34	0.76	33	0.73	1
HC	47	**0.89**	15	0.33	44	0.88	46	**0.89**	66
GMM	65	0.82	65	0.79	-	-	6	0.12	1
GMM++	49	0.83	2	0.05	28	0.62	-	-	1
SC	2	0.03	2	0.03	2	0.03	-	-	10
DBS	K = 45, J = 0.82								
CC	K = 68, J = 0.18								

Tab. 2.1.: Clustering results for both the `THEO` and `NCBI` data. Bold values depict the best result in a given data set.

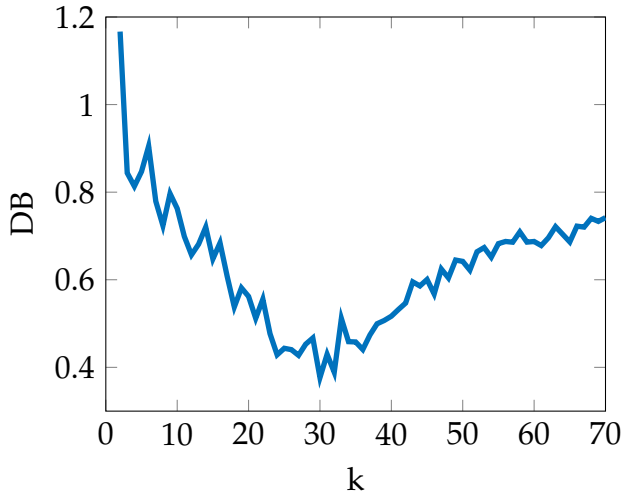

Fig. 2.8.: The Davies-Bouldin index for $k = \{1..70\}$ on the exemplary NCBI-30 data set. It is continuously decreasing before and increasing after the correct number $k = 30$ of clusters.

clustering algorithm, SS tends to pick k slightly larger than k_{opt}. Looking at the stability curve, there are pronounced maxima at the correct number of clusters for all theoretical data sets. However, for $k > k_{opt}$, the curves are decreasing only slowly such that these values of k also lie above the threshold which is hard to choose. Last, while delivering very good results when estimating k using only the eigenspectrum (connected components, CC), SC delivers less promising results when used as a clustering algorithm itself. The curves of internal validation indices and SS show a high variance, which can be attributed to the arbitrary nature of the corresponding graph cut for different k. In general, results on the theoretical data show, that with nearly all combinations of methods, it is possible to accurately estimate the number of clusters even for larger data sets. Here, the clusters are optimally distributed and noise-free, and results on the `NCBI` data can be expected to differ.

For the oligonucleotide frequency computation of the `NCBI` data, tetramers and window parameters of $w = 25\,000$ bp, $\Delta w = 12\,500$ bp were used. Here, GAP curves show a pronounced elbow only for $k_{opt} < 20$, making it even more difficult to locate the correct number of clusters. SS stability curves still yield a maximum for $k \approx k_{opt}$. However, its value drops below the threshold for larger metagenomes, adding to the problem of finding an optimal threshold. Additionally, because of outliers and a heterogeneous cluster structure, CC can over-estimate the number of clusters and displays low values of the Jaccard index. Still, estimations of other combinations are often

near-optimal and very accurate. In particular, the Davies-Bouldin index shows very good performance for most clustering algorithms. Additionally, exemplarily shown for the NCBI-30 data set in Figure 2.8, DB displays a nearly monotonic decrease before and increases after the optimum number of clusters $k_{opt} = 30$. Thus, provided a suitable conversion, it could also be related to a confidence measure to facilitate a post hoc human expert analysis. Results on metagenomic data sets do not perform as well as on the optimal theoretical data. As in the latter clusters are well pronounced while in the former they are often noisy, this behavior is expected. Still, some methods, in particular the Davies-Bouldin index, have shown to be able to work well in this setting, too.

2.3.5 Application to complex metagenomes

Given the results on the THEO and NCBI data, optimal window parameters and method choices were taken to assess performance on metagenome data that closely resembles the real world. For this task, the CAMI data sets are well suited. Due to the size of the samples and computational limitations, it was not possible to test all possible combinations of methods and parameters. Therefore, evaluation was limited to the use of hierarchical clustering in combination with the Davies-Bouldin index, in addition to DBSCAN and connected component clustering (CC). Results are shown in Table 2.2.

Data representation For the low complexity metagenome, a large and fixed window size of 30 kbp was used. For the more complex data sets, because of limited memory the number of data points n was restricted, resulting in much larger window sizes, even exceeding 100 kbp for the most complex one. Looking at the resulting cluster representations in Figure 2.9, and Figures A.1, A.2, there is a clear structure visible, with compact and separated clusters that match their originating genomes very well. However, in some cases a cluster contains contigs from multiple species, exemplary seen in the low complexity metagenome, in which only $k = 22$ instead of $k_{opt} = 30$ clusters are visible. This behavior can be explained by both high similarity between different species (e.g. strain level) or t-SNE artifacts.

Clustering For DBSCAN, it was difficult to find good choices for its $minPts$ and ϵ parameters. The resulting number of clusters varied heavily with small changes of those parameters and for many combinations of these

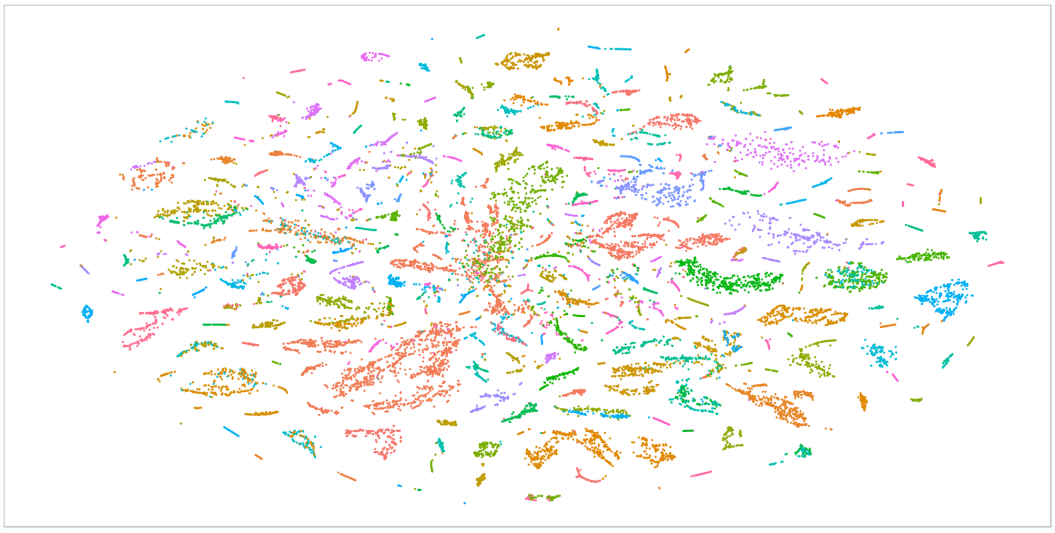

Fig. 2.9.: Data representation using t-SNE of the CAMI-306 high complexity metagenome, colored by gold standard binning.

	CAMI-30	CAMI-93	CAMI-306
genomes (strains)	40 (22)	132 (100)	596 (399)
min. contig length	1000 bp	5000 bp	10 000 bp
kept base pairs	96.7 %	90.2 %	97.9 %
size	155 Mbp	507 Mbp	2.75 Gbp
w	30 kbp	33.8 kbp	137 kbp
Δw	$\frac{w}{2}$	$\frac{w}{2}$	$\frac{w}{2}$
n	14463	27717	36592
complexity	low	medium	high
k_{opt}	30	93	306
k_{DB}	20	91	400
k_{CC}	30	81	302
J_{opt}	0.36	0.41	0.20
J_{DB}	0.53	0.42	0.17
J_{CC}	0.27	0.32	0.30

Tab. 2.2.: Parameters and results on the CAMI data. The number of clusters k and the Jaccard index J were evaluated for a hierarchical clustering using the $k = k_{opt}$, and using the number of clusters as given by the Davies-Bouldin index k_{DB}, as well as the number for connected components clustering k_{CC}.

values, the algorithm identified a large number of points as outliers. As the results were unusable, they were not included in the results. Next, because of the assemblies containing numerous contigs that are much smaller than the window size, this resulted in one data point per contig. Therefore, it was necessary to ignore all contigs smaller than a certain minimum length. However, the kept percentage of total base pairs was large ($> 90\%$). As regards the estimated number of clusters, using the DB index, for the low and medium complexity data sets, k was close to the correct number of clusters. Interestingly, for the medium complexity data set, k was estimated better than for the low complexity data. This is likely due to highly similar species as mentioned earlier. Similar to the `NCBI` data, for the low and medium complexity data, the DB index shows a clear minimum near the optimal number of clusters (Figure A.3). In contrast, for the high complexity data, the DB index fails to correctly identify k_{opt}. The number of clusters as given by CC apparently gives a much better estimation than DB, especially for the high complexity data where it nearly perfectly estimates the $k_{opt} = 306$ clusters. Interestingly, the according values of the Jaccard index are not as good and much lower than the ones obtained on the `NCBI` data. This indicates the limitations of the evaluated approach. Here, the high strain-level diversity could not be matched, i.e. individual strains of the same species that are highly similar could not be resolved into individual clusters.

2.4 Summary

I showed that a taxonomy-independent, sequence composition based pipeline as in Figure 2.5 can assign metagenomic fragments accurately, given the involved methods and parameters are correctly chosen. In particular, the combination of the highly flexible dimensionality reduction method t-SNE, followed by clustering, can reveal correct results for artificial data for cluster numbers as large as 50, as well as excellent results ($J \geq 0.9$) for small, simulated metagenomic data represented in terms of tetramer frequencies for almost the same number of clusters. Additionally, this simple approach can detect the number of genomes in data sets from the CAMI challenge for low, medium and high complexity metagenomes. However, possibly due to a high strain-level diversity, it could not find a good partitioning of the data. Here, incorporating additional post-processing steps that account for

t-SNE artifacts, and possibly clustering multiple times may remedy the low performance and is subject to future investigation.

Moreover, it was shown, that a modern dimensionality reduction such as t-SNE is crucial to deal with the high data dimensionality, while several popular clustering paradigms reveal reasonable values for the clusters. On average, cluster validity indices, in particular the Davies-Bouldin index, seem more reliable than alternatives such as direct methods or statistical counterparts. Interestingly, classical clustering techniques such as hierarchical clustering or K-means++ perform better for metagenomic data than computationally complex alternatives. Clustering by connected components is suited as well, particularly for highly complex metagenomes. These results are robust to the choice of meta-parameters, and default values which allow a complete automation of the process for a reasonable first binning to be derived. Referring to the three ingredients for metagenomic binning (section 2.1), optimal default parameters can be summarized:

1. Transform sequence data into vectorial data by calculating tetramer frequencies within a sliding window with window width $w \in [25\,000\,\text{bp}, 50\,000\,\text{bp}]$ and window step $\Delta w = \frac{w}{2}$. For larger data sets, computational limitations can be an issue and larger window sizes can be calculated from a given maximum number of points.

2. Reduce dimensionality of vectorial data using t-SNE to two dimensions.

3. On the resulting representation, estimate the number of clusters using either the Davies-Bouldin index and a simple clustering method such as k-Means++ or hierarchical clustering, or using connected component clustering.

In consequence, this contribution not only quantitatively evaluates the performance of these machine learning techniques for metagenomic binning and their stability for the first time, but it also shows the potential of this pipeline for a fully automated process in the de novo setting. These findings open the way towards further progress and more advanced technology: often, auxiliary information besides the mere DNA fragments is available such as coverage information (for a better characterization of statistically relevant frequencies for the nucleotides), 16S sequences (for an initial guess

about reasonable ranges for the number of clusters), or paired-end read information. This information could be integrated into the pipeline for even more robust results. In addition, binning is rarely a single loop process, rather, partial knowledge for known species can be integrated by relying on databases, and existing assembly tools can be applied again since they greatly benefit from grouping information to arrive at a better assembly of the given data. The latter observation opens the possibility of a loop which can be either automatic, interleaving assembly technology and machine learning based approaches, or involving human experts, integrating expert knowledge or database queries on demand. Such a loop can even open the possibility to overcome the problem of a suitable initial data representation based on tetramer frequencies for short fragments. Since short fragments only allow short windows and a potentially noisy data representation, a start from shorter windows together with loops seems a particularly promising approach to automatically arrive at a high quality binning.

To conclude, taxonomy-independent, sequence composition based data provides an essential source of information for binning.Taken alone, it showed good performance on small metagenomes, but showed its limitations on more complex data. Therefore, in the future, it is of high interest to see whether the presented findings can be incorporated into existing tools, adding to their quality and leveraging other information, too.

3 Single-cell genome contamination detection

Parts of this chapter are based on

Markus Lux, Jan Krüger, Christian Rinke, Irena Maus, Andreas Schlueter, Tanja Woyke, Alexander Sczyrba, Barbara Hammer

acdc – Automated Contamination Detection and Confidence estimation for single-cell genome data

BMC Bioinformatics, 2016

DOI: 10.1186/s12859-016-1397-7

Markus Lux, Barbara Hammer, Alexander Sczyrba

Automated Contamination Detection in Single-Cell Sequencing

bioRxiv, 2015

DOI: 10.1101/020859

3.1 Background

Single-cell sequencing In the previous chapter, focus was directed to the analysis of genomes from cell communities, i.e. metagenomes. Here, it is not uncommon to physically isolate a single cell from a metagenome using modern techniques by the means of single-cell sequencing (SCS). Named "Method of the year 2013" (Nature, 2013), SCS plays a continuously important role in many domains. Notable areas of research include medicine and the analysis of disease pathways (Eberwine et al., 2014), especially in cancer biology (Navin, 2015) and the development of targeted treatments (personalized medicine, Speicher, 2013). Additionally, SCS has proven a valuable and very powerful tool in evolutionary and environmental microbiology, for example by assessing intra- and inter-phylum relationships of bacteria and archaea (Rinke et al., 2013) and providing insights into key metabolic functions of uncultivated clades within their ecosystems (Swan et al., 2011).

To obtain the genome of a given organism, modern sequencing technologies rely on large amounts of DNA material. Up until a few years ago, the only way to obtain the needed mass of DNA molecules was cultivating the organism of interest in vitro. Unfortunately, the large majority of microbial species is still unknown (Rinke et al., 2013), and therefore in many cases the optimal cultivation conditions for novel organisms are unknown. Hence, the majority of organisms could not be grown in the laboratory and their genomes could not be sequenced by the means of conventional technologies. This situation was remedied by the rise of SCS, providing new techniques such as multiple displacement amplification that allows for the million-fold amplification of one single DNA molecule (Gawad et al., 2016). This enables the sequencing of one single amplified genome (SAG) and at the same time makes it possible to analyze its individual characteristics with high resolution. During the rise of SCS, another method to obtain single genomes got popular. With metagenomic binning methods becoming increasingly powerful, it is possible to extract metagenome-assembled genomes (MAG) from the individual bins of a metagenome. However, in contrast to SAGs, this method reconstructs composite genomes made of many cells from the same organism contained in a metagenome.

The problem of contamination A primary technological challenge in the analysis of SAG data is the potential presence of contamination and the detection and removal thereof (Bowers et al., 2017; Blainey, 2013). Foreign DNA which does not belong to the target genome of a given single cell, might be introduced into a sample in different ways. Sources of contamination can include unclean lysis or whole genome amplification reagents, in addition to sample introduced non-target DNA (Woyke et al., 2011; Salter et al., 2014). While much effort has been invested into engineering devices and methods for cell isolation and amplification steps that minimize contamination caused by the surrounding sequencing setup (Woyke et al., 2011; Blainey, 2013; Gawad et al., 2016), careful quality control is vital to prevent the propagation of misleading results in public databases. The same holds for MAG data – as binning cannot yet produce 100% clean bins (section 2), their quality and purity has to be assessed as well. Recently, Bowers et al., 2017 proposed new quality standards for the reporting of bacterial and archaeal genome sequences from SAGs and MAGs. They conclude that contamination rates should be smaller than 10% for low-quality and smaller than 5% for high-quality draft genomes. This emphasizes the importance of tools for the accurate contamination detection and removal. Figure 3.1 gives an overview

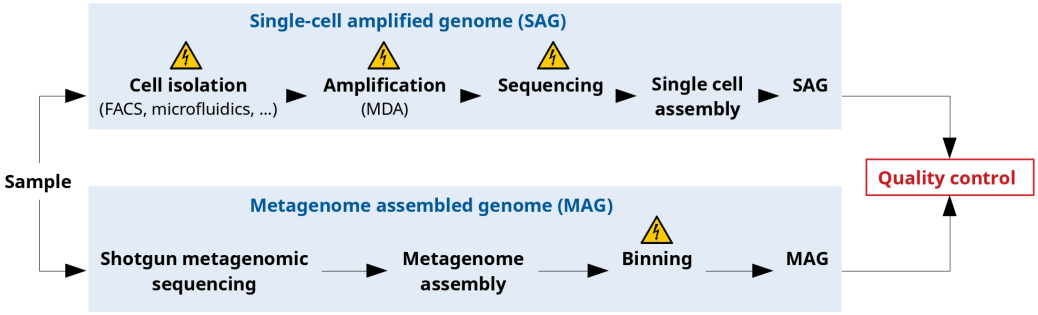

Fig. 3.1.: Single-cell workflow with possible sources of contamination (Gawad et al., 2016): For both SAGs and MAGs, laboratory procedures and sequencing are main sources of contamination. For MAGs, metagenome binning may result in impure bins, too.

of the single-cell workflow for both SAGs and MAGs, including possible sources of contamination.

Given those obstacles, ProDeGe, an automated Protocol for the Decontamination of Genomes was recently developed (Tennessen et al., 2015). ProDeGe combines the BLAST algorithm (Camacho et al., 2009) as a popular choice for database sequence alignment with reference-free PCA-reduced oligonucleotide profiling to enhance classification accuracy. Another method, CheckM (Parks et al., 2015), solely relies on the presence of multiple single-copy marker genes as an indication for contamination in a given sample, not operating reference-free. More recent classification methods (Ander et al., 2013; Naeem et al., 2013), most notably Kraken (Wood and Salzberg, 2014), are as accurate as BLAST but much faster, thus can speed up supervised detection. All these techniques heavily rely on references, hence they require existing knowledge about the characteristics of possible contaminants, making them less applicable either in the case of contaminants not being contained in databases or marker genes not being present in the sample (i.e. contamination is small or incomplete). Since the majority of species is unknown (Rinke et al., 2013), they are difficult to detect by such methods and unsupervised, taxonomy-free analysis is required (Mande et al., 2012).

Contamination and clustering Complementary to reference-based methods, clustering of oligonucleotide signatures is a promising approach that already found early application in metagenomic binning (chapter 2). From the perspective of computational intelligence, contamination detection as a clustering problem is very similar to metagenomic binning. Both metagenomic and SCS samples can be represented as a set of high-dimensional data

points. Binning and also contamination detection then address the same challenge of reliably detecting clusters in a high-dimensional data space. As already shown in chapter 2, quite a few challenges arise: To circumvent negative side effects in such high-dimensional spaces and to enable human expert inspection, it is crucial to use appropriate subspace embeddings to transform the data into an easily visualizable representation, i.e. two or three dimensions. Modern, non-linear dimensionality reduction methods, in particular t-SNE (Van Der Maaten, 2014) have proven useful (Laczny et al., 2014) in that context.

The automatic determination of the number of clusters and its cluster validity, a deep and crucial question in the context of clustering (Vendramin et al., 2010; Jain, 2010), poses yet another challenge. In contrast to metagenomic binning where the aim is to accurately bin sequences in a larger number of clusters, contamination detection in SCS is a fundamentally different task. It requires the discrimination between one or more clusters (genomes). This complication heavily reduces the set of applicable clustering algorithms: The majority of methods for estimating the number of clusters rely on cluster-specific measures such as internal validity indices (Liu et al., 2010). Since these are not defined for only one cluster, a distinctive null model for unimodal data is required, i.e. the techniques are usually not suited to distinguish one versus more than one cluster, hence cannot reliably identify non-contaminated samples.

Last, machine learning methods such as dimensionality reduction and clustering are based on statistics of the data and introduce certain amounts of variance. To overcome this limitation and to provide accurate and interpretable results, it is useful to integrate confidence measures. For this task, bootstrapping (Hastie et al., 2009) is a popular choice.

In this chapter, I present a novel software tool called "acdc" (Automated Contamination Detection and Confidence estimation for single-cell genome data, availability: section A.1.1). It seamlessly integrates reference-based with reference-free methods. Being based on both, very fast exact database alignments and modern techniques from unsupervised machine learning, acdc is able to accurately identify and remove contamination in single-cell sequencing data. To my knowledge, integrating entirely reference-free methodologies is a novelty, and complements existing high performing database-driven approaches such as ProDeGe. The use of appropriate clustering algorithms

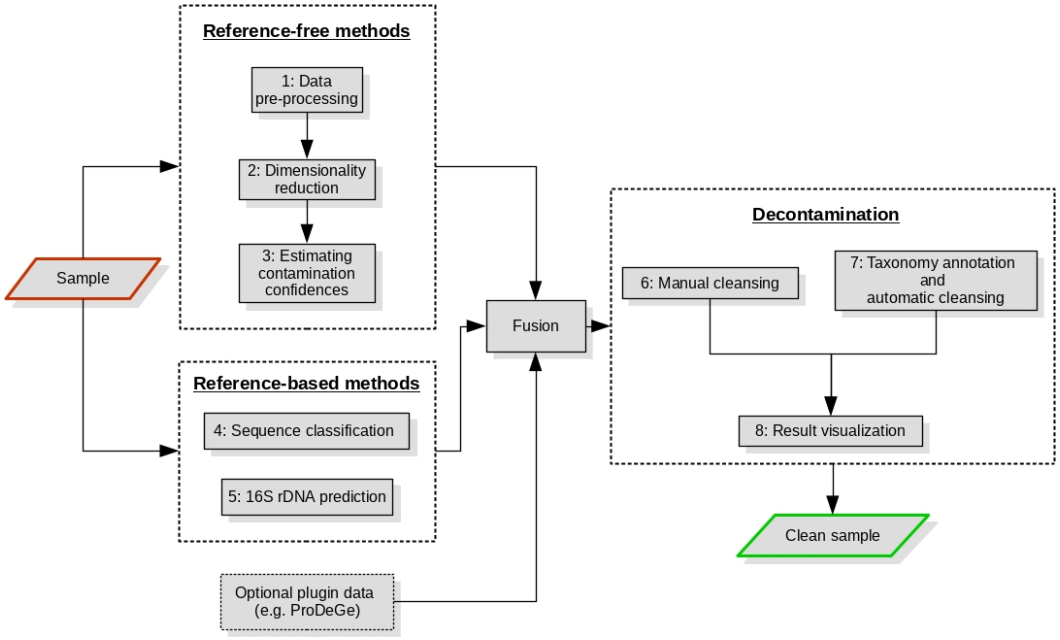

Fig. 3.2.: Acdc contamination detection pipeline: Results from both reference-free and reference-based techniques are fusioned and post-processed to end up with a clean sample.

allows the removal of foreign sequences to yield clean single-cell genome assemblies. Additionally, the integration of confidence values support expert interpretation. As it is expected that single-cell genomes further and rapidly populate public databases, acdc will be a resource-effective tool in the quality assurance of single-cell draft genomes, especially for users who do not have the background to use the included techniques directly.

The remainder of this chapter will continue by first introducing a contamination detection pipeline in a number of different steps. For each step, necessary computational techniques are established. Based on that, the software acdc is presented and evaluated using a large number of annotated SCS assemblies containing clean and contaminated samples. Using metagenomic bins from the previous chapter, acdc's capability to work on MAGs is discussed, as well.

3.2 Methodology

Acdc detects contamination in a series of steps which are depicted in Figure 3.2. Starting with contigs from a given single-cell genome assembly, both reference-free and reference-based methods are employed. In the former,

tetramer frequencies are calculated first (1), resulting in a high-dimensional vectorial representation which makes it possible to apply suitable machine learning algorithms. As its high dimensionality would introduce a number of adverse side effects in further processing, it is crucial to reduce dimensionality (2). This enables the accurate estimation of contamination confidences on the basis of clustering (3). External tools are then used to both classify sequences using ultrafast exact alignment (4) and to predict 16S rRNA genes (5). In the case of detected contamination, further clustering algorithms are employed to enable decontamination and export of clean samples. This is done either manually (6) or automatically (7), with prior taxonomic annotation. Results are then interactively visualized using a flexible web interface (8). Most of these steps include a number of hyper-parameters crucial in machine learning, for which acdc provides an auto-selection mode with well-tested default values. A summary of parameters can be found in section A.4. In the following, the steps depicted in Figure 3.2 are explained in detail.

3.2.1 Reference-free detection

Steps 1&2: Data pre-processing and dimensionality reduction

In order to apply machine learning techniques, it is necessary to transform contigs, represented as sequences, into a vectorial representation. Here, it is common practice to use oligonucleotide signatures (Teeling et al., 2004). This methodology has already been established in Chapter 2.2.1 and does not need further introduction. As shown before, in metagenomics due to a large range of possible sample sizes, selecting appropriate window parameters can be difficult. In contrast, for prokaryote single-cell data, the range is much more narrow. Typical bacterial or archaeal genome sizes average at around 5 Mbp (Koonin, 2011). Given this size, for both enabling accurate clustering estimations and to generate window sizes that are large enough to capture genomic signatures, it was found that a target number of $n_{target} = 1000$ data points delivers the best results. Choosing a smaller number may result in too few data points per cluster to enable accurate clustering while choosing

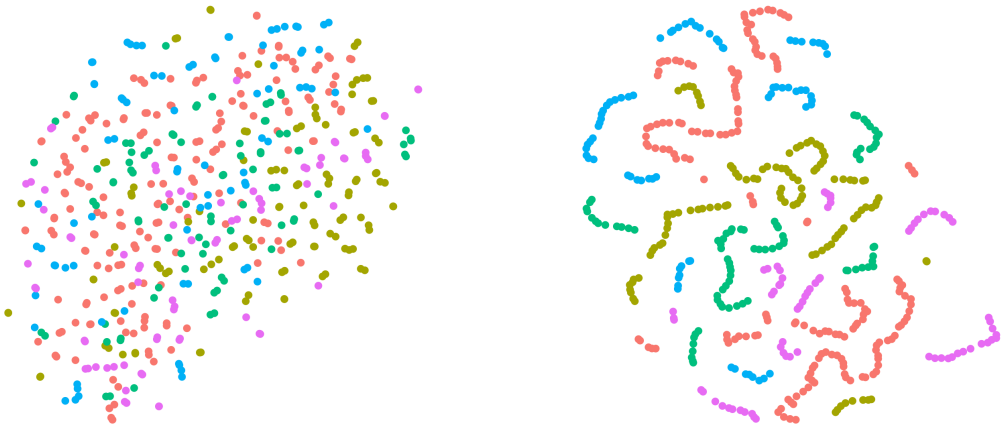

Fig. 3.3.: Difference between small and large window overlaps: two t-SNE representations of the same sample with different window overlaps, colored by contig identity. Left: $w = 1000$, $\Delta w = 2000$. Right: $w = 8000$, $\Delta w = 1000$.

a larger one will be computationally more expensive. Hence, window parameters are estimated using the total number of base pairs n_{bp}:

$$\Delta w = \lceil \frac{n_{bp}}{n_{target}} \rceil, w = 2 \cdot \Delta w \tag{3.1}$$

Because of contigs smaller than w, the real number of data points n usually differs from the given target by a small margin.

Furthermore, besides using l-mers as a characteristic genomic signature, the use of coverage information was considered, as well. However, due to the coverage bias in multiple displacement amplification (Woyke et al., 2011), using this data for single genomes is problematic.

Similar to the application in metagenomic binning (section 2.2.2), the resulting high dimensional representation is reduced to two dimensions using t-SNE (Van Der Maaten, 2014). An example is given by Figure 3.3. On the left, it shows the representation of an assembly that resulted from using half-overlapping windows, while on the right the same assembly was processed using a much larger window overlap. Using this parameterization, neighboring data points share more information and let t-SNE resolve individual contigs as small "worms". Even though the right representation more closely resembles the assembly, some clustering algorithms may cluster contigs as individual clusters which has to be prevented. Hence, it was found that half-overlapping windows are suited better for clustering genomes.

Step 3: Estimating contamination confidences

An integral part of acdc is the confidence and decision of whether a sample is contaminated or not. This problem can be seen as a clustering task. Optimally, one operational taxonomic unit (OTU, the set of genomic sequences from one single cell) is represented as one cluster. This is the core underlying assumption of acdc, implying that the presence of more than one cluster indicates contamination. Thus, the task is to estimate the number of clusters k.

This requires careful selection of methods and parameters (Estivill-Castro, 2002). In contrast to other applications such as metagenomic binning (section 2), one is not primarily interested in the actual number of clusters, rather in the distinction between $k = 1$ (no structure, clean sample) and $k > 1$ (clusters, contaminated sample). As the notion of a cluster is ill-posed, this is an inherently difficult task: Most techniques for estimating k operate on cluster-specific characteristics, defined for $k > 1$ only, making them inapplicable in the presented approach. The case $k = 1$ requires an appropriate null model to which the data is compared to in order to be able to detect no structure. For example, the Davies-Bouldin index established in the previous chapter can be used to estimate k. To achieve that, it uses cluster compactness and separation which are not defined in the presence of only one cluster. Techniques for this task were reviewed and two approaches are particularly promising. They are described in the following

Testing for multimodality The dip-statistic test (DIP, Hartigan and Hartigan, 1985) can detect multimodality of pair-wise distances of data points in the t-SNE representation. Given one cluster, this distribution can be assumed to be unimodal, whereas a significant multimodal distribution indicates $k > 1$. More specific, the dip statistic is based on the idea that a function is unimodal if its cumulative density function (cdf) F is convex in $[-\infty, t_L]$, of constant slope in $[t_L, t_U]$, and concave in $[t_U, \infty]$ (Kalogeratos and Likas, 2012), where t_L and t_U denote lower and upper bounds of the cdf, respectively. To test whether F belongs to the class of unimodal distributions U, the dip statistic is defined as

$$dip(F) = \min_{G \in U} \max_{t} |F(t) - G(t)| \tag{3.2}$$

where G are cdfs of functions in U. It measures the maximal deviation of F from the nearest unimodal distribution. Hence, the smaller the dip values,

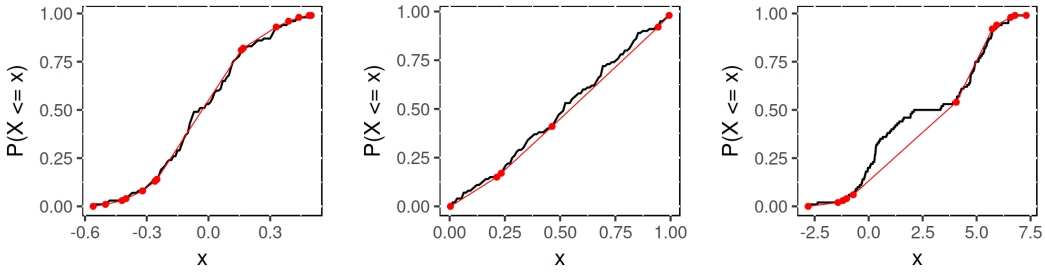

Fig. 3.4.: Construction of the nearest unimodal distribution (red) from a given empirical distribution function (black). Support points of the greatest convex minorant and least concave majorant are shown as dots. Three different distributions are fitted. Left: normal, center: uniform, right: bimodal.

the more likely F is unimodal. In practice, given an empirical cdf F with n observations, the dip is calculated by constructing the nearest unimodal distribution using the properties described above. In linear time, the greatest convex minorant (gcm) in $[t_{min}, t_L]$ and least concave majorant (lcm) in $[t_U, t_{max}]$ are calculated using an algorithm described in Hartigan and Hartigan, 1985. Then, the maximal deviation from this curve defines the dip. An example is given in Figure 3.4. It can be seen that the bimodal distribution (right) has a much larger deviation (dip) from the fitted unimodal distribution than both the normal and uniform distributions (left and center, respectively). Based on this, the dip statistic calculates a p-value based on R dip calculations on a reference distribution. It is defined as

$$p_{dip} = \frac{1}{R} \sum_{r=1}^{R} \mathbb{1}_{dip(F) < dip(U_n^r)} \tag{3.3}$$

where U_n^r consists of n elements sampled from a uniform distribution and $\mathbb{1}$ denotes the indicator function. The choice of the uniform distribution is motivated by the observation that its dip value is asymptotically larger than for any other unimodal distribution. The null hypothesis that F is unimodal is rejected in favor of multimodality if $p_{dip} < \alpha_{dip}$, a significance level.

Furthermore, to distinguish between $k = 1$ and $k > 1$ clusters, a method by Kalogeratos and Likas, 2012 is adapted. Given a data set containing n points, for each "viewer" the distribution of distances to all other points is tested for multimodality using the dip statistic. The percentage of viewers $q = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{p_{dip} < \alpha_{dip}}$ indicating significant multimodality is observed and as soon as q is larger than a small threshold q_{thresh}, the full data set is considered multimodal. The default parameter $q_{thresh} = 0.01\%$ is robust, as evaluated

by Kalogeratos and Likas, 2012. Hence, the estimated number of clusters using the dip statistic is given by

$$k_{dip} \begin{cases} = 1 & , \; q < q_{thresh} \\ > 1 & , \; q \geq q_{thresh} \end{cases} \tag{3.4}$$

The runtime complexity of detecting multimodality with the dip statistic is $\mathcal{O}(n_v \cdot R \cdot n)$: For each of the considered n_v viewer data points, a p-value is obtained by $R + 1$ times calculating the dip value. The computation of one dip value can be performed in $\mathcal{O}(n)$ (Kalogeratos and Likas, 2012). In the case of significant multimodality, the computation of p-values can be stopped early, after considering $n_v = q_{thresh} \cdot n \ll n$ data points only.

Connected component clustering A second method to distinguish between $k = 1$ and $k > 1$ is counting the number of connected components (CC) in a m-nearest-neighbor graph G_{cc}. It is constructed by connecting each data point to its m (symmetric or mutual) nearest neighbors. In contrast to DIP, where clusters have to have a reasonable size in order for multimodality to be significant, the cluster size in CC can be arbitrarily small. CC clustering is related to spectral clustering, in which the number of connected components is given by the algebraic multiplicity of the eigenvalue zero of the Laplacian matrix of a given neighborhood graph (Von Luxburg, 2007). Although, the expensive computations of spectral clustering are not needed and k can be determined much more efficiently. Using Tarjan's algorithm (Tarjan, 1972), the result can be found in linear time. For undirected graphs, the algorithm is even simpler and only has to perform a depth-first search to visit all data points in one cluster.

To accurately resolve contaminant clusters using CC clustering, the connectivity of G_{cc}, in particular the choice of which neighbors are connected and the setting of the parameter m, have to be chosen carefully. Maier et al., 2007 argue that for the discovery of connected components as clusters, it does not matter whether symmetric or mutual connectivity is used. In the former, two data points x_i and x_j are connected by an edge if x_i is a nearest neighbor of x_j and vice versa. In the latter case, the points are connected only if both are mutual nearest neighbors. Empirical observations showed mutual connectivity having the advantage that very small clusters with a

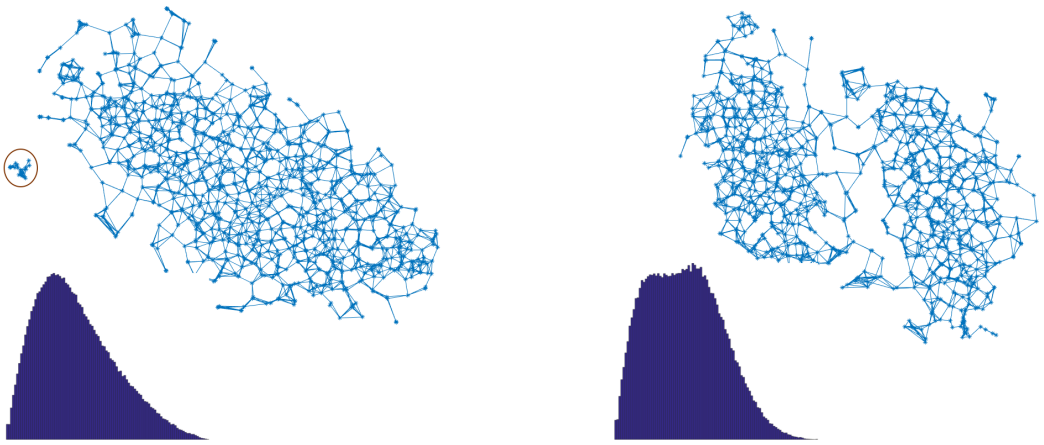

Fig. 3.5.: Illustration of the complementary detection capabilities of DIP and CC using two different contaminated samples. Left: Using a mutual 9-nearest-neighbor graph, CC identifies two clusters (very small contamination) while DIP isn't able to detect multimodality as seen in the distribution of pairwise distances below. Right: Two overlapping clusters prevent CC from detecting two components while DIP detects significant multimodality in the distribution of pairwise distances.

number of data points $< m$ are still disconnected from other clusters, even though the overall within-cluster connectivity decreases. Moreover, to identify the clusters as the connected components of the mutual m-NN graph, m should be chosen to optimize the trade-off between having high probability of being connected within clusters and high probability of having no edges between the different clusters (Maier et al., 2007). Further empirical analyses have shown that, in contamination detection in acdc, a default value of $m = 9$ is robust. Here, the value of m was determined by taking a set of 201 clean assemblies (`mdm`, section 3.3.2) that are supposed to produce a single connected component. For each assembly, from a range of different values of $m' \in \{1, \ldots, 20\}$, the largest value m' for which G_{cc} does not dissolve into individual connected components anymore was recorded. From the distribution of m' (Figure A.4), the optimal value for $m = 9$ was chosen to be the 90%-quantile. In the following, the estimated number connected components (clusters) is given by k_{cc}.

The time complexity of CC clustering is given by the depth/breadth-first search performed in $\mathcal{O}(n + E)$ where E is the number of edges in G_{cc}. Because m is small, in this nearest-neighbor graph the number of edges is small as well ($E \ll n^2$). Hence, the dominating factor in the complexity is n, keeping this computation in linear time.

Contamination detection example Contamination may occur in a variety of different cluster shapes and sizes. Both DIP and CC have been chosen to be employed in acdc to detect those in an antagonistic fashion. While the former is able to detect large and possibly overlapping clusters, the latter is able to detect small and outlier clusters. An example is given in Figure 3.5 where two different, contaminated samples are shown, left and right, respectively. The 9-nearest-neighbor graph is shown, together with the respective distribution of pairwise distances as used by the dip statistic. While, in the left sample, the latter does not indicate significant multimodality, a small trace of contamination can still be detected using CC. In contrast, on the right-hand side sample, there are two large clusters that form only one connected component, but now contamination is detected by significant bimodality. Consequently, a given genome assembly is marked as contaminated if $k_{dip} + k_{cc} > 2$. It is worth to note that, while this approach can detect both small and large contaminants as individual clusters, it is difficult to detect small clusters overlapping large ones, i.e. small traces of contaminants highly related to the target genome.

Cluster re-assignments and outliers Furthermore, noisy data, e.g. from very short contigs or from the inherent structure of some species might form separate clusters even in the presence of only one OTU. There may be t-SNE artifacts as discussed in section 2.2.2. To prevent false positive contaminant identification from wrongly formed clusters, acdc modifies cluster assignments in two steps:

1. Disregarding the possibility of chimeric contigs, a contig is expected to appear in only one OTU. Thus, data points that occur in different clusters, but belong to the same contig, indicate a wrong clustering. All points of such a contig are reassigned to the cluster which has the most points of the contig assigned.

2. An *aggressive threshold* is included that determines the minimum number of base pairs that is allowed to form a separate cluster. Smaller clusters are considered as outliers and are neither included into the calculation of contamination confidences nor into cleansing. The default threshold of 5000 bp works well throughout all tested data sets. A lower threshold provides more sensitive results towards very low levels of contamination and can be adapted by the user easily. This gives

a more fine-grained control than for example setting m (the number of nearest neighbors for CC clustering).

Confidence estimation using bootstrapping Last, the machine learning techniques used in acdc, namely dimensionality reduction and clustering, depending on data statistics, are not necessarily deterministic, hence introduce certain amounts of variance over different runs. In the case of clear contamination, i.e. well separated and compact clusters, these techniques agree with high probability. The same holds true for the case of a clean sample and one well-shaped cluster. However, in the case of an unclear contamination state such as strongly overlapping clusters, results may vary. Hence, in order to ease interpretation of results by domain experts, it is desirable to provide confidence values gathered over different runs. For this task, acdc employs bootstrapping (Hastie et al., 2009) with which it is possible to calculate interpretable confidence measures. Bootstraps are generated by randomly sub-sampling 75 percent of the original high-dimensional tetramer data B-fold. Each fold is processed by applying dimensionality reduction with t-SNE and subsequent testing using DIP/CC. A contamination confidence value is obtained by counting the percentage of folds which detected contamination:

$$\nu_{dip|cc} = \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}_{k_{dip}+k_{cc}>2} \tag{3.5}$$

3.2.2 Reference-based detection

Step 4: Sequence classification

Kraken (Wood and Salzberg, 2014) is employed as a fast alternative to the popular BLAST method (Camacho et al., 2009). Based on a pre-built database, Kraken assigns taxonomic labels to each contig from a sample. Through the use of exact alignments of l-mers, it achieves classification accuracy comparable to BLAST while being much faster. In acdc, Kraken classifies contigs on a species level and assigns a taxonomy label to each data point, depending on its originating contig. In case of an unclassified species, a contig remains unknown.

Acdc primarily focuses on de novo analysis without existing knowledge from databases and it tackles the challenge to reliably answer whether a given sample is contaminated or not. Reference-based cleansing is restricted to the fast Kraken method while unsupervised detection of potentially non-linear data clusters is performed by acdc. This distinguishes acdc from ProDeGe which relies on both BLASTing predicted genes and a supervised linear separation of contaminants, primarily aiming for an aggressive cleansing with high precision.

Step 5: 16S rRNA gene prediction

Acdc utilizes RNAmmer (Lagesen et al., 2007) to predict the location of highly conserved 16S rRNA gene sequences. Even if data could not be classified by Kraken, this enables researchers to identify the higher-level taxonomy of novel species quickly. Additionally, the location of the 16S rRNA gene sequence can be seen as a marker: It is representative for the whole cluster it is located in, and by exporting a cluster (cleansing), the taxonomy for a full OTU can be obtained.

3.2.3 Decontamination

Step 6: Manual cleansing

If contamination is detected, acdc finds a clustering which allows for the export of contigs from individual clusters, i.e. from the OTU of interest. As this is a process of cleaning the sequence data from unwanted contaminant data, it is referred to as cleansing or decontamination.

For this task, an optimal clustering has to be estimated. While CC provides an optimal assignment by itself, for DIP the number of clusters k has to be estimated. In contrast to detecting contamination where the task is to determine either $k = 1$ or $k > 1$, the cleansing step is slightly different. Similar to metagenomic binning, it is known that $k > 1$, which makes it possible to apply methods that estimate the number of clusters using cluster-specific characteristics, only defined for that case. Many clustering and k-estimation techniques are available for this task. In the previous chapter, it is suggested

that the combination of hierarchical clustering and the Davies-Bouldin index as a cluster validity measure works well for binning metagenomic tetramer profiles. In acdc, the same approach is used. This can also be motivated by the fact that this combination estimates the number of clusters accurately even in the presence of imbalanced clusters, which was found to be the case in contaminated SCS samples. Therefore, an optimal cluster assignment is determined by finding the minimal (optimal) Davies-Bouldin index for a given range of $k \in \{2, 3, 4, k_{max}\}$-clusterings using hierarchical clustering. Because it was found that given sample assemblies rarely contain more than five large clusters, indicating large amounts of contamination, the number of maximal clusters, set to $k_{max} = 5$, is a robust choice. In the following, the number of clusters for the dip statistic, as estimated by the Davies-Bouldin index is denoted as k_{dip}.

Based on this clustering, it is possible to clean an assembly by exporting the given cluster as an individual fasta file. This can be achieved either manually using the interactive web interface (Figure 3.7) or automatically as explained in the following section.

Step 7: Taxonomy annotation and automatic cleansing

In the case of zero external information, the manual cleansing of an assembly cannot be avoided. However, in many cases, at least a small percentage of the contained sequences can be taxonomically annotated. This information can be used to annotate more sequences by the means of the discovered cluster structure. Here, the underlying assumption is that if a taxonomy is found in a given cluster, the whole cluster can be labelled by it. However, it has to be accounted for the occurrence of multiple taxonomies.

For this to work, first an optimal clustering C_{opt} has to be selected. The one with the highest contamination confidence ν_{dip} or ν_{cc} is taken with corresponding number of clusters k_{opt}. Next, for each cluster, taxonomies annotated by Kraken, rRNA genes, or other tools are evaluated. Taxonomy confidences for each cluster are calculated with respect to the number of

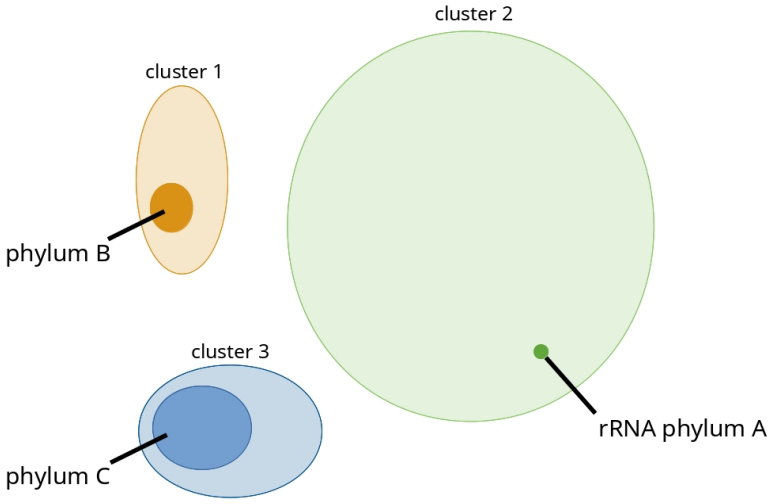

Fig. 3.6.: Illustration of labelling contaminants and clean contigs using taxonomy inference. The target taxonomy is phylum A. Assuming the rRNA of phylum A is not present, cluster 2 can still be assumed to be clean, because all other clusters can be labelled as contaminants. Assuming, clusters 1 and 3 are unlabelled, the taxonomy given by rRNA in cluster 2, lets conclude that clusters 1 and 3 are contaminants.

base pairs annotated with a given taxonomy in one cluster versus in all other clusters:

$$
\begin{aligned}
f_{t,i} &= \frac{n_t}{n_i} \\
\xi_{t,i} &= \frac{1}{2}\left(f_{t,i} + \frac{f_{t,i}}{\sum_{j=1}^{k_{opt}} f_{t,j}}\right)
\end{aligned}
\tag{3.6}
$$

First, the within-cluster frequency $f_{t,i}$ for taxonomy t and cluster i are calculated, with n_t, n_i being the number of base pairs annotated with t and the total number of base pairs in cluster i, respectively. The confidence $\xi_{t,i}$ of how well cluster i is annotated with t is then given by the weighted sum of the within-cluster frequency $f_{t,i}$ and its proportion in the whole sample. Consequently, the taxonomy confidence is maximal if a cluster is annotated with only one taxonomy and the same taxonomy does not appear in any other cluster.

Next, given a cluster t that is partially annotated with multiple taxonomies i, all unknown contigs are assigned the taxonomy with the highest confidence $\xi_{t,i}$. This way, all contigs in a given assembly can be assigned a taxonomy as long as in each cluster at least one contig can be annotated using external knowledge. Furthermore, in many cases, analysts know to some extent what

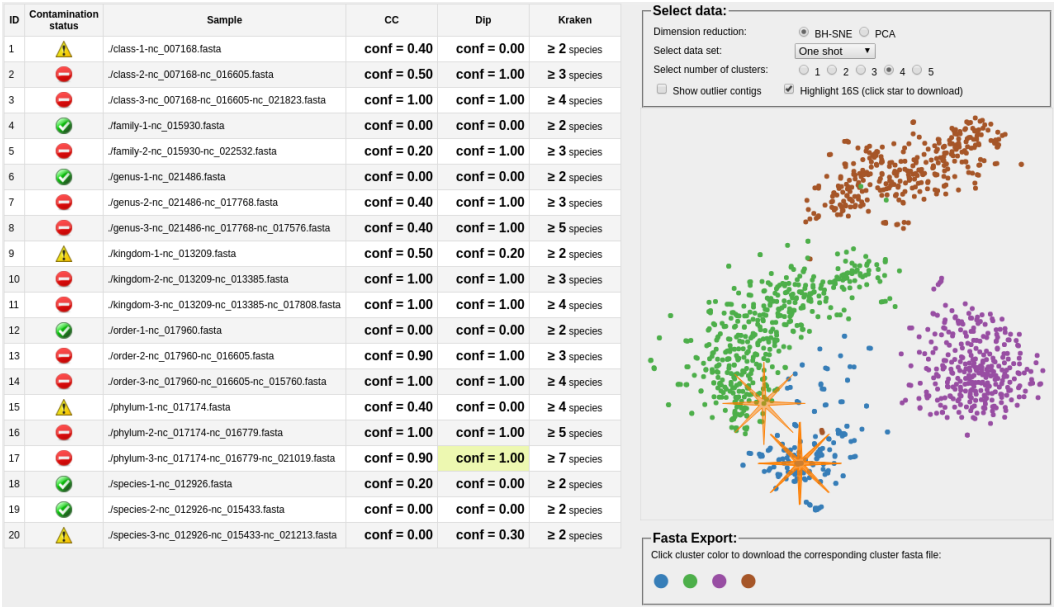

Fig. 3.7.: Acdc interactive result interface. For each sample shown in the left-hand side table, visualizations are shown on the right-hand side. Individual clusters can be exported in fasta format by clicking on the respective cluster color on the bottom right.

taxonomy they are searching for. This information can be used to annotate contaminants and clean contigs. A target taxonomy can be given by any taxonomic rank. For example, if the target is known to be a specific taxonomy (for example based on prior 16S rRNA analysis), all clusters annotated with a different taxonomy can be assumed to be contaminants. An example is given by Figure 3.6. Given a known target phylum A, clean and contaminant contigs can be identified in two ways. First, assuming that cluster 1 and 3 have no taxonomy assigned at all, and one contig in cluster 2 was annotated with rRNA indicating the target phylum, then based on the core assumption of acdc (one cluster equals one OTU), clusters 1 and 3 can be assumed to be contaminants. Second, if cluster 2 is assumed to be not annotated at all, but clusters 1 and 3 are found to be of phyla different from A, both are labelled as contaminants and the only remaining cluster 2 is assumed to be clean.

Step 8: Result visualization

Besides a fully automated mode, acdc provides contamination screening results as interactive web pages. An exemplary result of twenty simulated SCS samples is shown in Figure 3.7. For each sample on the left hand side, represented as rows, confidences ν_{cc} and ν_{dip} are shown. A sample is

marked clean when both $v_{cc} < 0.25$ and $v_{dip} < 0.25$. If either DIP or CC found more than 75 percent of all folds to be contaminated, the sample is marked appropriately. In case of no clear result, a sample is marked with a warning status symbol. A third column with the number of species reported by Kraken is shown. The user is able to inspect each sample for CC, DIP and Kraken. On the right hand side, the sample is visualized using t-SNE by default. In the unlikely event of a wrong cluster assignment, the number of clusters k can be selected manually, with the most likely k being selected by default. For Kraken, the assignments are fixed and can be inspected by hovering on each data point. The interface allows for manual cleansing. For that task, contigs from each cluster can be exported by clicking on the corresponding color in the panel below. Locations of predicted 16S rRNA gene sequences as reported by RNAmmer are indicated by an orange star. A click on it will show the corresponding sequence. All information shown in the interactive interface are also exported as a file in YAML format. This allows for the easy integration in existing large-scale sequencing and quality assurance pipelines.

3.3 Results

3.3.1 Computational performance

Acdc has low computational requirements. The steps as shown in Figure 3.2 can be broken down into individual runtime complexities. The data pre-processing step, i.e. calculating oligonucleotide frequencies is done in $\mathcal{O}\left(n_{input}\right)$, n_{input} being the input assembly size. Accelerated t-SNE as reported in Van Der Maaten, 2014 uses $\mathcal{O}\left(n \cdot \log n\right)$ time complexity, with n being the number of input points generated in the previous step. In the case of contamination (significant multimodality), the dip computations can be stopped early, reducing runtime complexity to $\mathcal{O}\left(R \cdot n\right)$. In the case of no contamination, its runtime is in $\mathcal{O}\left(R \cdot n^2\right)$. Hence, the combined runtime complexity of detecting contamination in acdc is quadratic in the case of a clean sample, and linearithmic in the case of contamination. Acdc's decontamination algorithm has quadratic runtime complexity because of the involved hierarchical clustering.

Practically, input data sizes are in the order of a few megabytes, as the inputs are already assembled contigs, not on the raw data, i.e. reads. Given that, using a quad-core consumer laptop, runtimes ranged from a few seconds to ten minutes per sample, depending on the actual size and contamination state. As stated above, the computationally most expensive step is the calculation of the dip statistic which has quadratic runtime in its worst case and has to be run for all bootstrap folds. This is sped up by parallelization.

3.3.2 Evaluation data sets

The evaluation of acdc can be divided into supervised (database-driven) and unsupervised detection analysis. While the former is restricted to only the method to classify sequences and the size of the underlying database, the latter requires more careful assessment. In order to obtain accurate results, it is necessary to use data with correct ground truth. As the manual assignment of contamination is biased, the simulation of single-cell samples or the analysis of existing samples with references are vital. To cover a broad range of contaminant varieties, acdc was tested on several simulated and real single-cell sequence data sets:

`simulated:` 20 single-cell genomes were simulated with varying amounts of contamination and contaminant relatedness. By manually selecting complete genomes from the NCBI database (Geer et al., 2009), clean and contaminated data sets, each containing up to 3 genomes were generated. Species were chosen such that they are related on different taxonomic ranks, expecting that distantly related species can be better separated than very similar species. For each level, 3 assemblies were generated, containing 1 – 3 species. The simulation of reads was done using ART (Huang et al., 2012) followed by subsequent assembly using SPAdes (Bankevich et al., 2012). A more detailed description of this data set can be found in Table A.3.

`mix:` Nine samples containing 6 draft genomes and 3 single chromosomes were obtained from various sequencing projects (Table A.4). All samples are known to be contaminated, however, an exact quantification of contaminated sequences is missing due to the novelty of the data. A detailed description of these data can be found in the supplementary material.

`benchmark`: Sequence data from 30 single-cell genomes with low levels of contamination were obtained (Clingenpeel et al., 2014a; Clingenpeel et al., 2014b). Containing cross-contamination between 3 species (*Escherichia coli*, *Meiothermus ruber*, *Pedobacter heparinus*), the median per-sample contaminant proportion of 3% is very small (min $= 1\%$, max $= 30\%$).

`mdm`: Furthermore, 201 single-cell samples from the microbial dark matter (MDM) project (Rinke et al., 2013) were taken to test the capability of acdc on non-contaminated data. These data were manually curated.

Acdc was compared to the state-of-the-art contamination detection tool ProDeGe (Tennessen et al., 2015) both in terms of supervised and unsupervised detection capabilities. ProDeGe has been optimized to obtain a high precision in the context of a known taxonomic level and database support. It integrates a linear classification model to extend predicted genes to all k-mers, displaying excellent behavior in aggressively curating according samples. Unsupervised inspection is restricted to linear PCA only. In contrast, acdc has been optimized to provide good F-measures (i.e. precision and recall) in curating, and it addresses database independent de novo detection of contamination, thus providing a tool highly complementary to ProDeGe.

3.3.3 Supervised analysis

Both ProDeGe using the BLAST algorithm and acdc using Kraken with the "MiniKraken DB" were tested on the `simulated` and `benchmark` data sets. These are the only two data sets for which entries for known contaminants existed in both used databases. Both tools showed nearly identical high performance ($F_1 > 0.95$) in identifying contaminant sequences and didn't require any further evaluation.

3.3.4 Unsupervised analysis

The evaluation of unsupervised detection performance was carried out `a)` by testing the ability to detect the correct contamination state of a given sample, and `b)` by measuring the ability to correctly identify clean and contaminant contigs.

Data set	Ident. clean samples	Ident. contaminated samples
simulated	4/7 (3 warnings)	11/13 (1 warning, 1 clean)
mix	0/0	8/9 (1 warning)
benchmark	0/0	22/30 (6 warnings, 2 clean)
mdm	150/201 (39 warnings, 12 contaminated)	0/0

Tab. 3.1.: Acdc evaluation of contamination detection performance. Entries depict the number of correctly identified clean and contaminated samples with additional information about false predictions in parentheses.

a)

Acdc correctly identified the majority of both contaminated and clean genome assemblies throughout all data sets (Table 3.1). This result demonstrates the ability of acdc to single out contaminated versus clean genome assemblies, specifically without any reference to a database in de novo settings. For this part of the evaluation, it was not possible to compare to existing methods because they either do not have the functionality to distinguish clean and contaminated samples (ProDeGe), or operate reference-based only (CheckM). Warnings are sometimes issued for assemblies with unclear contamination state. Here, further inspection often revealed the presence of small outlier clusters throughout a small number of bootstraps. In the rare case of strongly imbalanced and additionally overlapping clusters, acdc is not able to detect contamination because of missing structure in the data. Further, if the contaminant is too related to the target (e.g. different strains from the same species), genomic signatures differ only by a very small percentage of all base pairs, making it impossible for acdc to detect them.

b)

Acdc was compared[1] to ProDeGe in terms of precision/recall performance with respect to the number of correctly identified clean base pairs in each sample. For this task, the functionality to export clean sequences common in both tools was used. Since the evaluation is performed for the setting of limited prior biological information, no taxonomy is provided for ProDeGe, restricting the use of reference sequences from databases. Results in Table 3.2

[1] For the comparison the ProDeGe online version at https://prodege.jgi.doe.gov/ was used.

Data set	*Precision* **ProDeGe**	*Recall* **ProDeGe**	F_1 **ProDeGe**	*Precision* **acdc**	*Recall* **acdc**	F_1 **acdc**
`simulated` (kingdom)	no result	no result	no result	1.00	1.00	**1.00**
`simulated` (phylum)	no result	no result	no result	0.99	0.98	**0.99**
`simulated` (class)	no result	no result	no result	1.00	0.99	**0.99**
`simulated` (order)	no result	no result	no result	0.99	0.98	**0.99**
`simulated` (family)	no result	no result	no result	1.00	1.00	**1.00**
`simulated` (genus)	0.22	0.32	0.22	0.95	0.97	**0.96**
`simulated` (species)	0.50	0.33	0.36	0.38	0.77	**0.46**
`benchmark` (*E.coli*)	1.00	0.88	0.93	0.97	0.99	**0.98**
`benchmark` (*M.ruber*)	1.00	0.73	0.83	0.99	0.99	**0.99**
`benchmark` (*P.heparinus*)	1.00	0.70	0.81	1.00	1.00	**1.00**

Tab. 3.2.: Precision, recall and F_1-scores of predicted clean base pairs for both ProDeGe and acdc on the `simulated` and `benchmark` data sets. Each row contains average values of the given sub data set. Bold values depict the best performing entry. Entries marked as "no result" either produced an empty clean fasta file or did not finish computation.

were averaged over different samples from the `simulated` and `benchmark` data sets. Both ProDeGe and acdc correctly identified clean contigs in the `benchmark` data set with high precision. However, on average acdc was able to recall 22% more clean sequences on the data set, due to the more aggressive design of ProDeGe. Next, ProDeGe was not able to identify the majority of clean sequences in the `simulated` data set without taxonomic information. In those cases, mostly all contigs were marked as contaminants, resulting in an empty clean sequences file. This fact can be attributed both to ProDeGes behavior of selecting contaminants with high specificity Tennessen et al., 2015 and to its missing ability to distinguish between clean and contaminated samples. Results of 4 samples could not be obtained, because computation didn't provide any output. On the same data, acdc was able to correctly identify the majority of clean sequences with high precision and recall. For samples that contain closely related species, it is difficult to split clean and contaminated sequences. For example, in the simulated data, samples from the same genus contain species with an average nucleotide identity (ANI) of 73%. This fact led to a slight drop in performance. Sequences containing strains from the same species (ANI in the simulated samples: 95%) didn't contain enough distinct information to be correctly identified, showing the limits of acdc's reference-free detection capabilities.

3.3.5 Assessing the purity of metagenome bins

As shown in the previous chapter, the production of 100% pure bins still constitutes a problem for state-of-the-art unsupervised binning tools. Consequently, an unsupervised detection of impurities in MAGs is required. For that task, operating in the same way as for SAGs, acdc can be applied to MAGs as well. In order to investigate its ability to detect impure bins, acdc was exemplarily applied to all bins of the `CAMI-93` data set from last chapter, for both the gold standard binning which can be assumed to contain pure bins only, and the suboptimal, "estimated binning" found by the clustering procedure described in that section 2.3.5. The latter should contain both pure and impure bins. In the following, purity is based on contamination confidences, i.e. pure bins should have low confidence while for impure bins confidence should be high.

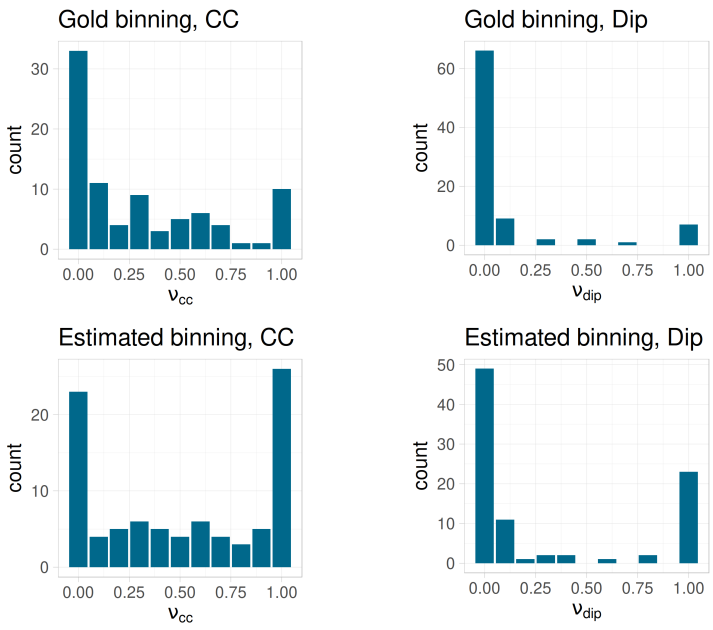

Fig. 3.8.: Distributions of confidences ν_{cc} and ν_{dip} for all bins in both the gold standard and the estimated binning from last chapter for the `CAMI-93` metagenome.

Results are depicted in Figure 3.8 which shows histograms of the contamination confidences ν_{cc} and ν_{dip} for all bins in both the gold standard and the estimated binning. First, looking only at the gold standard binning at the top, it is visible that most bins are indeed clean (contamination confidences concentrated near zero), with few contaminated bins. Contrary, in the estimated binning the number of impure bins outweighs the clean ones. This confirms the above assumption. Interestingly, the gold standard binning could not be detected as fully clean. A closer analysis of the 14 impure bins revealed that contamination in nearly all cases was caused by very small clusters ($<$ 5% of total bin size). A BLAST (Camacho et al., 2009) analysis of the contaminant contigs indicated the presence of DNA from three bacteriophages and five plasmids throughout the impure bins. One bin contained contigs from three different species of the same phylogenetic order. The 37 impure bins from the estimated binning contained contaminants represented by clusters that were more distinct and larger than contaminants in the gold standard binning. This is the expected observation, because contaminants are mostly made of sequences from distinct species.

3.4 Discussion

3.4.1 Influence of assembly size and quality

Depending on the sequencing quality, the assembly supplied to acdc may be of different quality, too. One measure of assembly quality is the N50 value, the median length of all contigs. A low N50 value indicates that a larger number of contigs are small. In some cases, there may even be hundreds of contigs that are smaller than the minimum contig length of acdc (default: 100 bp). If the contamination is present in smaller contigs only, acdc cannot detect it. Furthermore, if the assembly itself is rather large, the estimated window size w is large as well (section 2.2.1). In that case, the assembly may also contain contigs that are smaller than the window size but still larger than the minimum contig length. Such contigs are still included in acdc's detection procedure and result in one data point per contig. Because of the different lengths of these contigs, they capture different characteristics of the genome. Hence, such contigs introduce a certain amount of noise in the calculated representation and, due to the possibly different signature, may be mistaken for contamination. A similar effect could be observed when the assembly is very small. Acdc's set of parameter was chosen to work well for typical sizes of prokaryotic genomes, starting from 2 Mbp. If an assembly is much smaller, w will be small as well. This would result in capturing local genome signatures such as from individual genes. Such local features can differ greatly within one genome, and possibly lead to unwanted effects such as the formation of individual clusters that may be mistaken for contamination. In contrast, the estimated window width for normally sized prokaryote genomes is large enough to smoothen out such effects.

3.4.2 Influence of horizontal gene transfer and repeats

There were few `mdm` samples that have been identified as contaminated. They displayed a quite distinct cluster structure. Further manual investigation on a small subset of these samples revealed the presence of true contamination which was not identified during manual curation. Furthermore, the sequence of a bacteriophage was identified. Other analyzed samples not presented in this work included plasmids, represented as distinct clusters. Hence, the found structures highlight biologically interesting phenomena.

Horizontally transferred genetic material (HGT) such as from bacteriophages or plasmids often have genomic signatures that significantly differ from their host. Along with phylogenetic analysis, this property can be used to detect HGT (Koonin et al., 2001). Consequently, HGT may manifest itself in pronounced clusters that may be mistaken for contamination. Although, in acdc, not all HGT directly leads to the formation of distinct structures, mostly due to the following reasons:

- Given the HGT length is smaller than the used window size, the impact on the resulting signature may be negligible.
- If the formed HGT cluster is smaller than the aggressive threshold, it is discarded as an outlier.
- If the HGT is located in a much larger contig, because of contig re-assignment, the respective data points will be assigned to the cluster of the originating contig.

The same observations can be made for repeating DNA elements which have a signature that strongly differs from the originating genome. Given the assembly in question was generated by a sequencing technology that is able to resolve terminal or tandem repeats, again depending on the above mentioned points, such may form distinct clusters.

3.5 Summary

Building on the previous chapter, the software tool acdc uses oligonucleotide frequencies to detect and remove contamination in single-cell genome data. Operating both in the presence and absence of references from databases, acdc was able to predict the contamination state in the large majority of samples from four unrelated data sets, containing a total of 258 single-cell genome assemblies. Additionally, clean and contaminant sequences were correctly identified with high recall and precision. In the absence of a given target taxonomy which is required by similar methods (i.e. ProDeGe), acdc was still able to correctly predict contamination based on state-of-the-art techniques from unsupervised machine learning. Complementary to other tools, the presented software does neither require the prediction of (marker)

genes nor existing knowledge from databases to detect contaminants and to separate contaminant from clean sequences. Although, supplemental database information will aid identification, for example of closely related species. These findings make acdc an ideal tool to complement state-of-the-art contaminant detection and cleansing methods such as ProDeGe or CheckM in the context of de novo analysis with limited taxonomic information or limited availability of reference sequence information.

Furthermore, early results on the application of acdc for detecting impurities in MAGs, showed promising results and will be pursued in the future. A number of results indicate that acdc may also be used to detect the presence of horizontal gene transfer. Due to the limited ability to detect such sequences in de novo assemblies of unknown species, acdc could be used as a guidance. To that extent, a modification of default parameters could bring improvements. Last, the current approach of using oligonucleotide frequencies as a vectorial representations cannot resolve species-level contamination, even with an optimal choice of window parameters. To overcome this limitation, in the future, alternative representations may be investigated. Here, Local Binary Patterns (Kouchaki et al., 2016) or space-filling Hilbert Curves (Gu et al., 2016) constitute promising candidates.

4 Identification of flow cytometry cell populations

Parts of this chapter are based on

Markus Lux, Ryan Remy Brinkman, Cedric Chauve, Adam Laing, Anna Lorenc, Lucie Abeler-Dörner, Barbara Hammer

flowLearn: Fast and precise identification and quality checking of cell populations in flow cytometry

Bioinformatics, 2018

DOI: 10.1093/bioinformatics/bty082

4.1 Background

Flow cytometry (FCM) is a technology that is commonly used for the rapid characterization of cells of the immune system at the single cell level. Based on antigens presented on the cell surface, cells of interest are targeted by a set of fluorochrome-conjugated antibodies (markers). They pass through a laser beam one-by-one at over 10 000 cells per minute (Shapiro, 2005). Scattered light of different wavelengths for each marker is measured and recorded by sensitive detectors. This subsequently creates a unique intensity profile that allows for differentiation of cell types, a process commonly known as "sorting". FCM can be used to physically sort single cells from a metagenome using targeted enrichment (Blainey, 2013). The most common use of FCM, however, is in a clinical setting, such as for the diagnosis of cancer and heterogeneous immunodeficiencies. It is also widely used in research, for example in immunophenotyping where it holds great promise for assessing the immune status of patient populations. Within a blood or tissue sample, the most common measurement of interest is cell frequency (i.e. the proportion of a cell population), either absolute or relative to a parent population. Variations in cell frequencies can give important information about the immune status, for example they enable easy diagnosis of acute leukemia (Brown and Wittwer, 2000), or allow association of cell types with a biological variable, i.e. phenotype or clinical outcome (Aghaeepour et al., 2016).

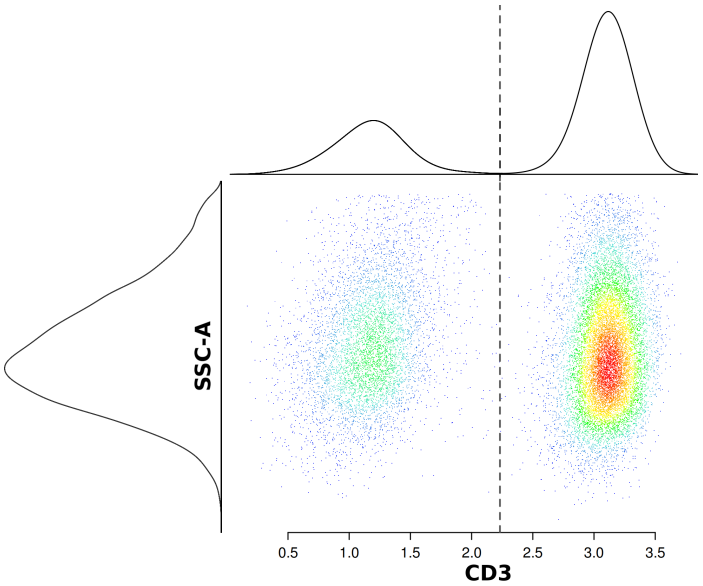

Fig. 4.1.: Density plot for two channels "CD-3" against "SSC-A". Each dot represents a cell and corresponding one-dimensional densities are attached at the top and left. The shown data is an example of gating T-cells from lymphocyte singlet cells. Axis values correspond to intensity values. Knowing that T-cells (right cluster) express CD3, they can be delineated from other lymphocytes (left cluster). The corresponding density estimates for both channels are shown along the left and top axis. The CD3 threshold is located in a valley between the density peaks from both populations.

The accurate determination of cell population frequencies is a key aim in FCM analysis. Using differential expression of one or more markers, it is possible to delineate cell populations of interest, a process commonly known as “gating”. This task usually comprises the manual inspection of bivariate density plots using marker channels that were selected from biological knowledge. The latter includes information about differences in expression levels of certain cell types and constitutes strong prior information for gating. Subsequently, for each sample, cell populations are identified by drawing regions of interest or setting channel thresholds (Figure 4.1). While some populations are easy to gate, populations with very small cell proportions (rare populations) can be challenging. Large populations can obscure rare ones, such that they do not necessarily appear as clusters or well pronounced density peaks, hence are difficult to detect. Gating is a labour-intensive and highly subjective process (Saeys et al., 2016). This is mainly due to the manual exploration of a large number of intensity plots and inconsistent views on at which position exactly to set gates. Furthermore, in large FCM studies, thousands of samples have to be gated, making it difficult to consistently gate all of them manually. In contrast, automated methods offer little to no bias and comparable variability (Aghaeepour et al., 2013; Finak et al., 2016). Thus, there has been substantial interest in developing methods that ease the process of identifying cell populations as much as possible, and a large variety of tools have been developed (Kvistborg et al., 2015; Saeys et al., 2016; Aghaeepour et al., 2013).

In general, automated tools for FCM gating can be distinguished into unsupervised and supervised ones. While the former do not require information about the target population to gate, i.e. discover hidden structures, the latter require characteristics such as cell membership as examples from which gates on unseen samples can be inferred. Common to both approaches is the required capability to resolve clusters in a high-dimensional space. For this task, there has been interest in the application of dimensionality reduction and clustering techniques, for example as presented in the past two chapters. A particular example is given by "viSNE" (Amir et al., 2013). Such methods are applicable for small, targeted FCM studies with a comparably small number of samples and cells per sample. Unfortunately, in complex FCM studies comprising a large number of markers and populations to gate, cells do not necessarily form distinct clusters that would be easily discoverable. An example is given by Figure 4.2 which shows the t-SNE (Van Der Maaten, 2014) representation of 19 marker channels of a large population of CD45

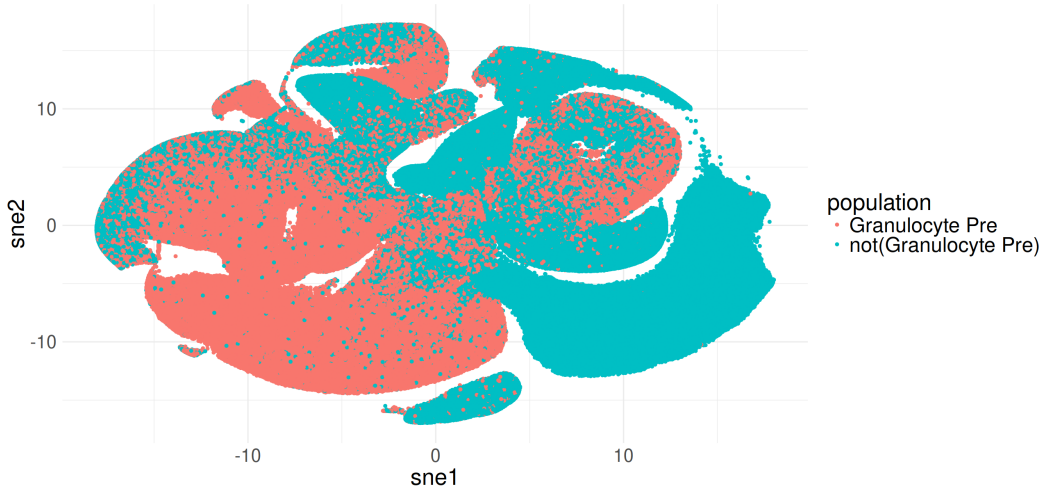

Fig. 4.2.: T-sne plot of a population of 345 637 CD45 cells (dots), colored by the child populations "Granulocyte Pre" and "not(Granulocyte Pre)" cells. Both populations do not form two distinct clusters in this representation.

cells. Although there is some clear structure, it is visible that both child populations are not distinct from each other. Consequently, an approach solely based on dimensionality reduction and clustering cannot gate both complex populations. Additionally, due to the large number of cells per sample (usually $> 1 \times 10^5$ cells), running t-SNE can take multiple hours per sample, making it highly resource-intensive for a large number of samples. Hence, more sophisticated techniques are required.

Several automated gating tools are based on techniques from machine learning. They aim to eliminate the traditional approach of inspecting bivariate channels plots by considering all channels at once instead of only two at a time. While these methods have shown very good performance on many data sets (Mair et al., 2016), they still suffer from two major drawbacks, especially pronounced for large sets of highly diverse FCM samples. First, in order to describe populations of interest, the fine-tuning of a set of hyperparameters is crucial and common to all tools, in particular in the context of small populations where the underlying machine learning task is quite challenging. Depending on the method, practitioners might not have the required knowledge to set those optimally (Kvistborg et al., 2015). In the presence of high sample diversity, fine-tuning such parameters is essential and might take a significant amount of time. Due to the difficulties in finding an optimal set of parameters, it is also complicated to compare such tools. Second, when incorporating machine learning, interpretability of results is limited (Lisboa, 2013), leading to a lack of general understanding of how

such methods work. Hence, it is problematic to verify gates from a biological standpoint (Kvistborg et al., 2015). Another aspect of FCM studies is quality checking of results, an essential step in the accurate identification of populations, and ensures that no wrong conclusions are drawn from the data in later steps. For that reason, and also because of the familiarity with the traditional approach of inspecting bivariate density plots, for quality checking, manual gating is the current standard practice.

FlowDensity (Malek et al., 2015) takes another point of view and tries to automate the threshold selection based on density shape features. The algorithm works in an unsupervised fashion. When customizing thresholds on a per-population level, one or more channels are inspected, and density features such as differences in extrema, slope changes, or the number of peaks are examined, generally based on a pre-determined manual gating hierarchy. Gates in the form of channel thresholds are estimated, from which sub-populations can be extracted (Figure 4.1). Provided hyper-parameters are appropriately chosen, flowDensity offers a state-of-the-art tool for the accurate identification of cell populations that matches what would have been obtained through manual analysis (Finak et al., 2016). Once the rules for each population are set, thresholds are automatically and individually set for each new data file, similar to the manual tweaking that operators tend to do, but in a data-driven fashion. As a result, flowDensity results are robust, reproducible and the approach performs better than the manual alternative it is designed to match (Aghaeepour et al., 2013). However, undertaking a supervised setup does require a significant time component in order to obtain the optimal results.

In this chapter, I present a novel software tool called "flowLearn" (availability: section A.1.2). Using density features, it works the same way as flowDensity, but does not require a practitioner to manually tune hyper-parameters for an optimized outcome. Rather, it works in a semi-supervised mode, and requires the gating of one or few characteristic samples by a human expert in the form of thresholds. These thresholds are then transferred to all samples in an automated way by means of so called derivative-based density alignments. In contrast to methods with a high-dimensional understanding of FCM data, the presented method does not have to deal with the problem of noise inherent to such spaces. It reduces the problem complexity by relying on traditional, prior biological knowledge.

Besides this highly efficient and effective mode, it also opens the way towards a quality control of samples which have already been gated: By comparing optimal (manually verified) and predicted gates, it can identify samples that stand out, for example due to biological variation or differences in laboratory sample preparation and analysis. Furthermore, it can be used to spot problems in existing gating hierarchies, offering the possibility to be used to identify problematic gates.

The absence of difficult-to-set hyper-parameters, combined with the use of a small amount of example gates makes this approach tractable. Results are easy to interpret and verify, and offer user-interactive adjustments on the predicted gates in specific circumstances. Using two large and diverse data sets for which accurately set and verified gates exist, and by comparing to two other recent state-of-the-art methods, DeepCyTOF (Li et al., 2017) and FlowSOM (Van Gassen et al., 2015), flowLearn's superiority is demonstrated for the classical and practically very relevant setting of comparably low dimensional data and imbalanced populations. It exhibits high to very high accuracy in terms of predicted cell frequencies and F_1-measures of identified populations, having low computational complexity.

4.2 Methodology

4.2.1 Pipeline

FlowLearn is based on four main steps that are depicted in Figure 4.3:

1. Input to the tool is a set of related FCS files (samples)

2. On each sample, it solves the task of extracting sub-populations from a given parent-population. To achieve this, marker channel densities of the desired population are calculated and a pairwise comparison of all densities is performed.

3. Based on the resulting distances, one or more prototype(s) (red) are selected, for which manual gating is performed.

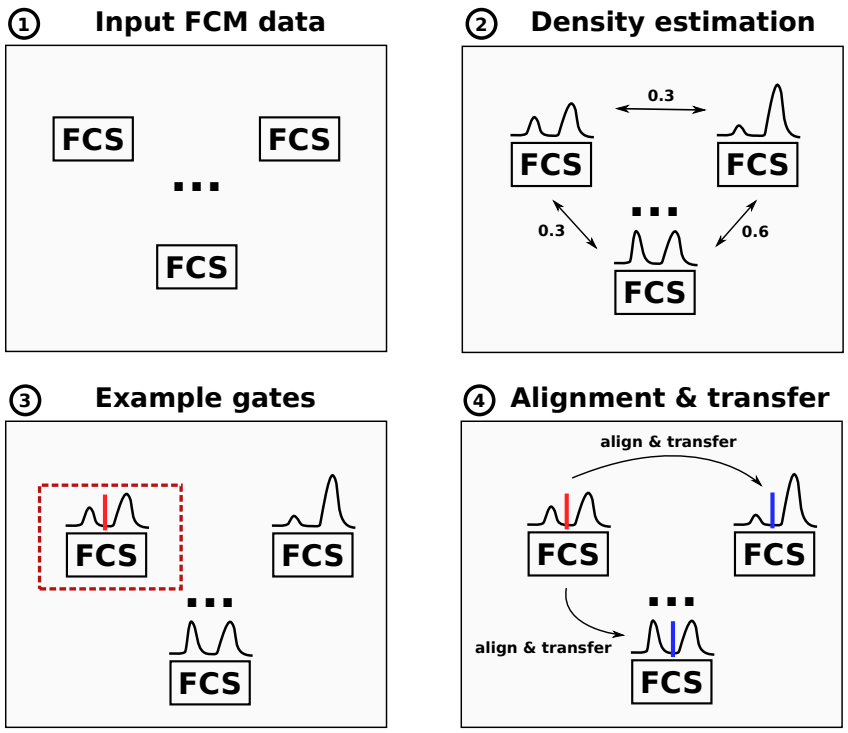

Fig. 4.3.: FlowLearn pipeline to gate one population in four consecutive steps.

4. Each prototype density is aligned to all other densities which makes it possible to predict thresholds by transfer between the aligned densities (blue).

A more detailed procedure on how to apply flowLearn to a set of samples and populations is given by Algorithm 1. Given that, in the following, the four steps will be described in-depth.

1: Input FCM data

A dataset $S = \{s_1, s_2, \ldots, s_N\}$ is defined as a collection of N related samples generated using the same panel of antibodies (markers). Each sample contains a set of cells with marker-based measurements of fluorescent intensities, and any subset of cells is called a population. In the following, a population with M cells is assumed, and each cell has measurements in the form of marker intensities. A marker channel for sample s_i is defined as the set $C_i = \{c_i^1, \ldots, c_i^M\}$ of marker intensity measurements for all M cells. The set of marker channels is experiment-specific, and used across all samples. Furthermore, a sample-specific channel threshold t_i splits up

Algorithm 1 Applying flowLearn to gate all populations in a given set of FCS files: First, densities are calculated once for all channels. Second, prototype densities are identified and manually gated. Third, all other densities are aligned to the prototypes and thresholds are transferred. This is repeated for each population of interest, starting with all cells.

Input: set of N FCS files, gating hierarchy H

$parentPopulation \leftarrow$ all cells
$childPopulation \leftarrow$ next in H
while not all populations in H gated **do**
    for all FCS files s_i **do** ▷ Calculate densities once
        $Z_i \leftarrow$ cells matching $parentPopulation$ in s_i
        for all channels j **do**
            $d_i^j \leftarrow$ density on j-th channel for cells in Z_i
        end for
    end for
    for all considered channels j **do** ▷ Predict thresholds
        $D_j \leftarrow$ distances between all densities d_i^j, $\forall$ samples i
        $P_j \leftarrow n_p$ prototype densities, w.r.t. D_j
        let expert set reference threshold(s) on each density in P_j
        for all non-prototype densities m **do**
            $p \leftarrow$ nearest prototype w.r.t. D_j
            $a \leftarrow$ DDTW alignment between m and p
            transfer reference threshold(s) from p to m using a
        end for
    end for
    gate $childPopulation$ using reference and predicted thresholds
    $parentPopulation \leftarrow childPopulation$
    $childPopulation \leftarrow$ next in H
end while

a given channel into two distinct sub-populations C_{i_1} and C_{i_2}, such that $C_i = C_{i_1} \cup C_{i_2}$ and $C_{i_1} \cap C_{i_2} = \emptyset$. A gate is defined by a set of thresholds for one or more channels and can delineate sub-populations from a given parent population. Last, a gating hierarchy is the consecutive extraction of sub-populations, starting with one root population that usually contains all cells in the FCS file. It is assumed that the gating hierarchy for a dataset is known, meaning that for each parent population, the set of channels to gate is given.

2: Density estimation

Given a parent cell population, from which sub-populations should be gated, for each sample and channel to gate, flowLearn uses kernel density estimation (such as in Venables and Ripley, 2013) to generate a unique density profile over a regular grid of $G = 512$ points (Algorithm 1, lines 4–9). For the purpose of gating, a density profile has to fulfill two main properties. First, cell populations have to be sufficiently captured by individual density peaks. Second, the density should be smooth enough such that alignment techniques in later steps do not fail, i.e. variations not caused by the presence of a cell population should not be captured. More formally, all cells from a given marker channel are binned to an equally spaced grid with G bins. Next, a Gaussian kernel function K is used to calculate a density e for each bin x:

$$e(x) = \frac{1}{G \cdot h} \sum_{j=1}^{G} K\left(\frac{x - x_j}{h}\right) \tag{4.1}$$

The bandwidth h is determined by Silverman's rule of thumb (Silverman, 1986). Empirical observations showed that this choice captures cell populations well, but in some cases also captures unwanted noise. To account for that, the density is smoothed to avoid the matching of spurious density peaks in subsequent alignments. While a more careful choice of bandwidth may help, a simpler solution is given by the use of smoothing splines (Hastie et al., 2009) that allow for a more direct control over the smoothness. This is achieved by finding a cubic spline function d that minimizes

$$\sum_{j=1}^{G} (d(x) - e(x)) + \lambda \cdot \int d''(x)^2 dx \tag{4.2}$$

Here, the parameter λ controls the degree of smoothness in terms of the second derivative of d. While $\lambda = 0$ allows for any function d, choosing larger values increases smoothness. Further empirical observations showed that, in flowLearn, a default value of $\lambda = 0.4$ resulted in the best performance on all data sets.

In the following, a smoothed density for channel C_i is denoted as $d_i = g(C_i)$, where g estimates and smoothens a density from a given channel and parent population in sample s_i. In the resulting distributions, pronounced cell populations are visible as density peaks (Figure 4.1). All further processing is done on the N densities for each channel only. For clustering in the next step, pairwise L_1 distances between all densities are calculated as well.

3: Example gates

The alignment (step 4) of two channel densities can be inaccurate if they are very different from each other. In that case, thresholds might be transferred wrongly. To prevent this, flowLearn selects prototypes that define sample groups (Algorithm 1, line 12). Parameterized with the number of prototypes n_p, it clusters all densities for each channel and selects n_p samples that represent each set accurately. It was found empirically that for this task, a k-medoids clustering (Friedman et al., 2001) with $k = n_p$ worked best. In this approach, similar densities are characterized by their absolute difference (L_1 distance). Experiments using other distances for clustering, for example the alignment distance, showed worse results. The number of prototypes should be chosen according to the sample diversity in the data set, and desired gating accuracy. While often $n_p = 1$ shows very good results (section 4.4) , with increasing complexity, n_p should be increased accordingly. It is worth to note that, first, n_p can be different for different channels, depending on the data complexity, and second, prototypes for different channels are not necessarily from the same sample.

Next, having the prototypes identified, it is the task of an expert analyst to set gates as accurate as possible, resulting in prototype thresholds (Algorithm 1, line 13). A gate that delineates two populations can be given by lower and upper thresholds for at least one channel. In Figure 4.1, the T-cell population (right cluster) is defined by a lower threshold on the CD3 channel, meaning that all cells with higher CD3 intensity belong to the target population.

The CD3 threshold is located in a valley between the density peaks from both populations. If necessary, an upper threshold can be set as well, even though it is not necessary in this particular example. Also, here the SSC-A channel is irrelevant for gating. A gate can be defined by more than one channel, where two channels are usually chosen for the ease of visualization and interpretability. A limitation is that prototype thresholds have to be perpendicular to the channel axes, though current efforts are focused on addressing this.

4: Alignment and gate transfer

Given a prototype sample with density d_p, a known prototype threshold t_p, and another sample's density d_i, flowLearn is a function $f(d_i, d_p, t_p) = \hat{t}_i \approx t_i$, where t_i is the unknown true threshold in d_i. In order to estimate $\hat{t}_i$, it is necessary to align d_i and d_p. For this task, Dynamic Time Warping (DTW, Sakoe and Chiba, 1978; Ratanamahatana and Keogh, 2004) is a common choice. Given a distance matrix D, in which the entry $D_{m,n}$ corresponds to the quadratic difference between the m-th and n-th element of d_i and d_p, respectively, DTW finds a warping path w through D. The k-th entry $w_k = D_{m,n}$ corresponds to the alignment of the respective elements in d_i and d_p. The goal is to find

$$w^* = \arg\min_w \frac{1}{K} \sqrt{\sum_{k=1}^{K} w_k} \tag{4.3}$$

The optimal path w^* has to fulfill boundary, continuity and monotonicity constraints and can be found elegantly using Dynamic Programming (Sakoe and Chiba, 1978). An example of an optimal warping path is given by Figure 4.4.

In flowLearn (Algorithm 1, lines 14–18), instead of using distances between absolute density values, local derivatives are used instead. This technique is commonly known as Derivative Dynamic Time Warping (DDTW, Keogh and Pazzani, 2001) and offers improvements of the original DTW algorithm for the current application. DDTW puts more focus on aligning shapes. More specifically, because of differences in the size of cell populations, the associated density peaks can locally differ in height, which constitutes a problem for regular DTW: because it only compares absolute height values,

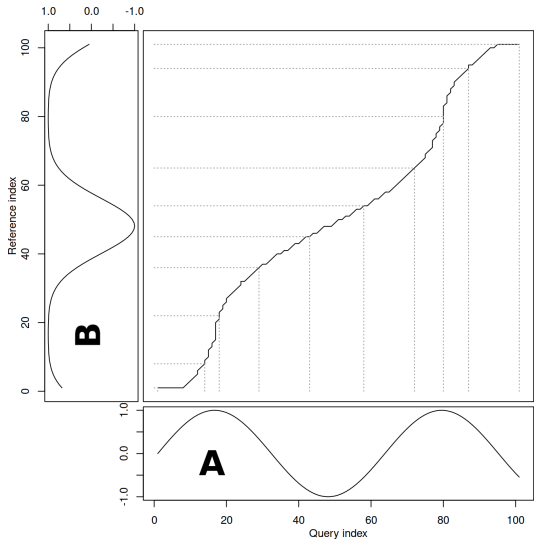

Fig. 4.4.: Example of a DTW alignment between two sequences $A(x) = \sin(x)$, $B(x) = \sin(x + \frac{3}{4}\cos(x))$. The warping path shows how the stretched peaks in B are matched with those in A.

local differences make it difficult to align the corresponding density regions (Keogh and Pazzani, 2001). This effect can be remedied by considering derivatives instead. In the case of aligning FCM densities, this is the preferred way, because thresholds are mostly characterized by density features such as extrema or slope changes (Malek et al., 2015). For DDTW, flowLearn uses the average of two slopes as a robust estimate (Keogh and Pazzani, 2001) of the local derivative of a density d:

$$d'_i = \frac{(d_i - d_{i-1}) + (d_{i+1} - d_{i-1})\ /2}{2} \tag{4.4}$$

Last, to disallow arbitrary warping paths that would align large parts of one density to only small parts of another one (singularities), flowLearn uses a step pattern (Myers, 1980) to restrict the direction of the warping path and effectively constraints its slope.

Once the alignment between two densities is calculated, thresholds can be transferred. Illustrated in Figure 4.5, each point in the red-dashed prototype density d_p is matched with a point in the solid-black test density d_i. This makes it possible to transfer the known thresholds t_p to predicted thresholds $\hat{t}_i$. The resulting threshold prediction on the right contains both t_i (red) and $\hat{t}_i$ (blue) with a nearly perfect match. Using flowLearn, it is also possible to gate rare populations by the alignment of density tails. Rare populations usually result in flat parts of the density, although peaks do not need to be very pronounced in order to be aligned, when there is another, larger density

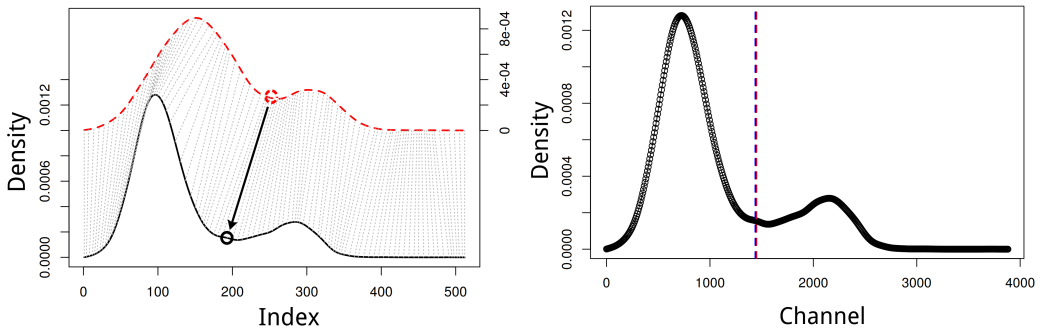

Fig. 4.5.: Example of how thresholds are transferred using DDTW alignments. Top: The red-dashed prototype density d_p is aligned to the solid-black test density d_i. The transfer of threshold location is indicated by an arrow. Bottom: Test density d_i with predicted (blue) and true (red) thresholds, $\hat{t}_i$ and t_i, respectively. Both thresholds match up well.

peak that can be aligned properly. Only a few cells of another rare population are needed in order for the density to extend into a tail, containing the rare population. That the tail is aligned correctly follows directly from the correct alignment of the adjacent, larger peak.

It is worth mentioning that, in the context of FCM gating, the idea of alignment has already been proposed in the technology dubbed per-channel basis normalization (Hahne et al., 2010). Here, specific density landmarks are identified and aligned. By using parameter-less shape matching for arbitrary shapes, flowLearn improves upon this. In particular, it can take into account rare populations, which are not easily captured by using parameterized density models as in per-channel basis normalization.

4.2.2 Evaluation measures

FlowLearn is evaluated by assessing its performance per predicted population: For that task, two measures are used. First difference percentages in median cell frequencies

$$d_f = \frac{\text{median}|f_i - \hat{f}_i|}{\text{median } f_i} \tag{4.5}$$

are calculated, where for each sample i, f_i and $\hat{f}_i$ are cell frequencies of a given population with regard to the ground truth and predicted gates, respectively. Second, F_1-measures for all ground truth and predicted popu-

lations are calculated: Given one sample and population, for one or more channels, true and predicted thresholds, t_i and $\hat{t}_i$, define a gate. Furthermore, a sub-population is defined by all cells that fall within the gate, i.e. have larger/smaller density than the lower/upper threshold for each involved channel. For the sub-population, the true and predicted set of cells, S and $\hat{S}$, are known. Then the performance on this particular population and sample is defined by

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4.6}$$

where precision and recall are calculated in terms of S and $\hat{S}$. While analysts are mostly interested in the accuracy of cell frequencies, using the F_1-score for evaluation provides an additional, more informative measure. It is worth to note, that even though thresholds might differ significantly, the agreement of population memberships as measured by the F_1-score can still be high. This is the case when thresholds differ in regions with only a low number of cells.

4.2.3 Quality checking cell population thresholds

Thresholds might be predicted wrongly either because of sample diversity (i.e. failed alignments with too different prototypes), or because of gates that were set wrongly in the first place. The prediction capabilities and evaluation measures of flowLearn can alternatively be used as a tool for detecting such irregularities in a given set of samples and already existing gates. Samples with deviant F_1-scores can be an indication of unexpected biological variation, differences in reagent preparation and analysis, or wrongly set thresholds. FlowLearn give clues to an expert to further analyze identified, possibly problematic samples or gates. Given an input dataset with existing gates in the form of channel thresholds, quality checking is performed as follows. Similar to Algorithm 1, for each analyzed population:

1. Use flowLearn to identify n_p prototype densities.

2. Let flowLearn gate all other non-prototype samples.

3. Compare predicted populations with ones defined by existing gates.

4. Samples with significant deviation from the average F_1-score suggest problems or give hint to unexpected biological variation.

FlowLearn provides the functionality to identify such outlier samples. Considering the distribution of F_1-scores in a given population, it uses the rule of thumb (Upton and Cook, 1996) that outliers are given by samples with $F_1 < Q_1 - 1.5 \cdot \mathrm{IQR}$, where Q_1 is the 25%-quantile and IQR is the interquartile-range of the distribution of F_1-values.

4.2.4 Implementation and computational complexity

FlowLearn is available as an `R`-package. By default, using the `density` function, flowLearn uses a smoothed Gaussian kernel density estimate with a granularity of $G = 512$ points. This setting is a good compromise between speed and accuracy: For coarse densities, the alignment is fast, however prediction accuracy will suffer. The subsequent spline fitting is performed using `R`'s `smooth.spline` function. For prototype selection, the function `pam` of the package `cluster` is used. FlowLearn provides separate methods to identify prototype densities and to predict other samples from there. This way, it can be integrated into existing analysis pipelines.

Computational time complexity of the gating of one population can be broken down into multiple steps. First, density calculation and spline fitting (Algorithm 1, lines 4–9) is performed in $\mathcal{O}\left(N \cdot (M + G \log G)\right)$ and $\mathcal{O}\left(N \cdot G\right)$, respectively. The distance matrix calculation and subsequent prototype selection (Algorithm 1, lines 11–12) is done in $\mathcal{O}\left(N^2\right)$. By using warping constraints, DDTW takes place in $\mathcal{O}\left(N \cdot G^2\right)$. Consequently, flowLearn has quadratic time and memory complexity with respect to the number of samples, however practical requirements are very low.

4.3 Study design and evaluation data sets

To demonstrate flowLearn's capabilities to accurately identify populations on a large number of samples, it was evaluated on two distinct data sets (Mice, FlowCAP). Both data sets were used in real-world applications, contain diverse samples, and were augmented with carefully set gates in the form of channel thresholds. For each sample and cell population in both data sets, flowLearn identified n_p prototypes, and predicted gates on the remaining test samples using only the prototypes as a reference. For each population, median cell frequencies and F_1-scores are reported to assess flowLearn's performance on these data. Furthermore, the results on the Mice data set were compared with two recent state-of-the-art gating tools for identifying cell populations, DeepCyTOF (Li et al., 2017) and FlowSOM (Van Gassen et al., 2015). Next, the two evaluation data sets are introduced.

4.3.1 Mice data

As part of a large study to identify gene-immunophenotype associations in mice (Brown and Moore, 2012), 2665 FCS files from mice bone marrow sample were gated, first manually and then using flowDensity, and independently verified. By looking at the variability of resulting cell proportions, the flowDensity gates were found to be superior to manual gates. Hence, flowDensity gates were used as the gold standard for this study. In the following experiments, these curated thresholds are referred to as true gates. The mean cell frequencies of 16 cell populations (relative to the parent population) range from 0.2% to 50%, covering a wide biological diversity, including very rare populations.

4.3.2 FlowCAP data

FlowLearn was evaluated on a second dataset from the FlowCAP (Flow Cytometry: Critical Assessment of Population Identification Methods) consortium (Finak et al., 2016). FlowCAP provides the means to objectively test methods for the identification of cell populations, and puts out state-of-the-art data sets with which it is possible to compare tools to manual analysis by

experts. In the context of the FlowCAP-III competition, seven participating centers were given the task to analyze three samples. For each sample, three replicates were analyzed, and for each replicate, sub-populations in four datasets (B cells, T cells, Regulatory T cells (T-reg), Dendritic Cells (DC)) were identified independently from each center. Manual gating was performed by a central site. For each dataset, a total of 63 FCS files were available for evaluation. Since flowLearn currently uses channel thresholds, only populations with rectangular gates were used, 48 in total. Mean cell frequencies (relative to the parent population) range from 0.4% to 70%. Even though, all centers were advised to follow reagent and analysis standardization, technical variation between centers was still large (Finak et al., 2016), leading to a greater sample diversity in this data set. This effect is also pronounced in the densities estimated by flowLearn, i.e. the sample diversity is captured in the densities as well.

4.4 Results

4.4.1 Mice data

For each population of the Mice data set (2665 samples of clean CD45 cells), a small number of $n_p \in \{1, 2, 5, 10, 50\}$ prototypes were gated (using flowDensity and manual curation). These thresholds were transferred to all samples by the described flowLearn protocol. Those thresholds are referred to as the predicted ones. Figure 4.6 shows the result of $n_p = 1$, where results on all 2664 test samples per population are represented by boxplots. Distributions of true and predicted cell frequencies match up well, with mean $d_f = 0.05$ ($\min d_f = 0.0001$, $\max d_f = 0.22$), taken over all populations. Also, predicted gates are accurate with respect to extracted cell populations, specifically it is $\text{median}(F_1) > 0.99$ for the majority (9/16) of populations and $\text{median}(F_1) > 0.90$ for the large majority (15/16). Rare populations such as T cells (4% of parent) and Plasma cells (0.2% of parent) show high performance. For some populations, a minority of samples with lower F_1-scores exist, including outliers (represented as dots) with poor F_1-score. Choosing the number of prototypes $n_p > 1$ increases performance significantly (Figure 4.7), especially for populations with initially low performance. Exemplary, for the HFC population, choosing $n_p = 10$ increases the result to $\text{median}(F_1) = 0.94$.

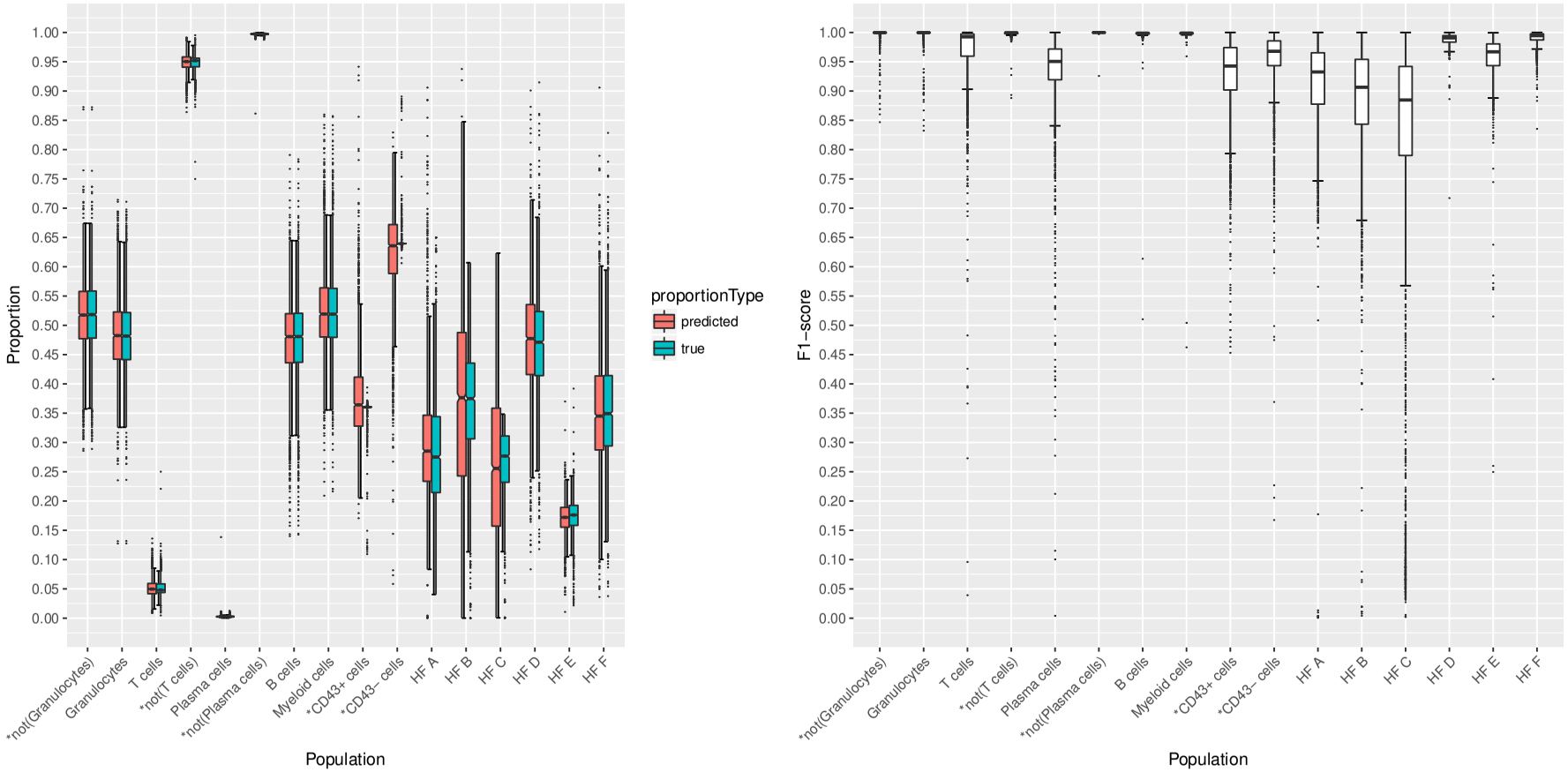

Fig. 4.6.: Results on the Mice data set using one prototype. Left: True and predicted cell frequencies for each population. Right: F_1-scores for each population (2665 samples). Outliers are shown as single dots. Populations HFA to HFF stand for Hardy-Fraction A to F. Populations denoted with an asterisk are non-biological (technical) populations.

	Mice	DC	T-cells	T-reg	B-cells
n_p	1	7	7	7	7
$\min d_f$	0.0001	0.01	0.001	0.001	0.002
$\text{mean}\, d_f$	0.05	0.11	0.13	0.14	0.08
$\max d_f$	0.22	0.62	0.13	0.14	0.22

Tab. 4.1.: For each dataset, minimum, average and maximum difference percentages in predicted cell frequencies.

4.4.2 FlowCAP data

On the FlowCAP data, flowLearn was run using $n_p \in \{1, 4, 7, 11, 20\}$ prototype(s). As the data set contained only 63 samples, $n_p = 7$ prototypes were chosen, i.e. one prototype per center. Table 4.1 shows a statistic for difference percentages in cell frequencies for all datasets. The average differences were low, ranging from 5 to 14 percent. While for some populations, differences were negligible, for other populations, differences were large. Furthermore, a summary of F_1-scores is shown in Figure 4.7, where the minimum and average population median (F_1)-scores, depending on n_p is displayed for each dataset. Performance depended on both the population and more strongly on the chosen number of prototypes. Considering all datasets, for $n_p = 7$ (number of centers), 11/48 populations achieve median$(F_1) > 0.99$ and median$(F_1) > 0.90$ for 36/48 populations. Results on other populations are not as good when only few prototypes ($n_p < 7$) are chosen, and are generally not as good as on the Mice data set. In general, choosing more prototypes yields higher F_1 values (Figure 4.7). Especially for populations that perform poorly, increasing n_p significantly increases performance. Detailed results for all FlowCAP datasets can be found in section A.8.

4.4.3 Runtime

For the Mice data (16 populations), gating and evaluating all 2665 samples using one prototype took one hour (1.3 seconds per sample). By exploiting parallelism, practical runtimes are low. When using n_p prototypes, flowLearn will perform n_p alignments per sample per population. Increasing n_p also linearly increases runtime. Again, for the Mice data, using $n_p = 2$ took 62 minutes ($n_p = 10$: 66 min, $n_p = 50$: 111 min). In all experiments, using a Dell M3800 laptop (Intel® Core™ i7-4712HQ processor), resident

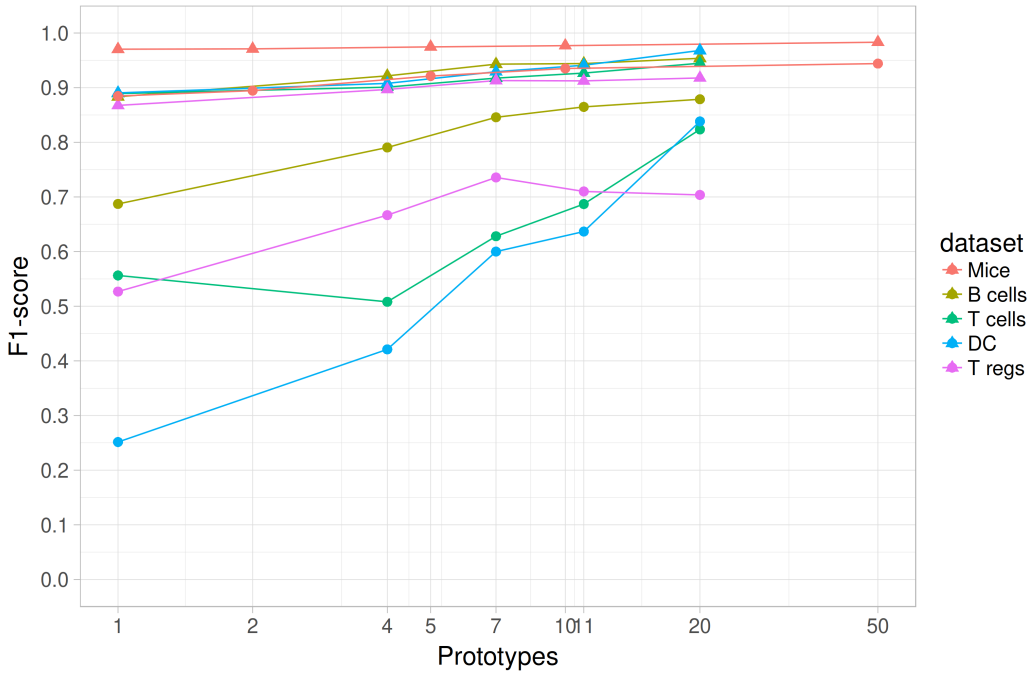

Fig. 4.7.: Minimum (circle) and mean (triangle) population performances for all datasets, depending on the chosen number of prototypes.

memory usage never exceeded 5 GB of RAM. These results indicate that flowLearn can be run on recent consumer laptops without problems.

4.4.4 Comparison to nearest-neighbor gating

To assess the influence of the use of DDTW alignments on the result, flowLearn was compared to nearest-neighbor gating, exemplary for populations from the Mice dataset (similar results were obtained on all FlowCAP datasets) and different numbers of prototypes. More specific, the thresholds were transferred statically, i.e. no alignment was performed and threshold values were simply copied from the nearest prototype. Table 4.2 shows the average and maximal increase in F_1-score and decrease of its standard deviation when comparing flowLearn to nearest-neighbor gating. The average F_1 score over all populations is already good without any alignment, i.e. median ΔF_1 is small. However, for some difficult to gate populations, the increase is large ($\max \Delta F_1 = 12.2\%$), especially for the choice of very few prototypes. In the case of a larger number of prototypes, the benefit of using alignments is small. More prototypes increase their average similarity to neighboring samples such that a static threshold transfer gives better results.

n_p	max ΔF_1	median ΔF_1	max $\Delta\sigma$	median $\Delta\sigma$
1	+12.2%	+1.0%	−36.8%	−17.8%
2	+6.4%	+0.7%	−56.1%	−11.9%
5	+4.7%	+0.4%	−65.1%	−14.9%
10	+4.9%	+0.3%	−43.8%	−15.5%
50	+3.1%	+0.1%	−33.7%	−7.8%

Tab. 4.2.: Increase of F_1 score and decrease of standard deviation σ per population for different numbers of prototypes, when comparing flowLearn to nearest-neighbor gating.

For all choices of prototype counts, there is a large decrease in standard deviation of F_1-scores. Given that most populations have good performance already, this observation indicates that especially poorly performing samples throughout all populations can strongly benefit from correct alignments. Furthermore, when replacing the prototypes selected by flowLearn with randomly selected ones, performance decreases and variability increases in all settings. This indicates that not only the number of prototypes is important but also the correct selection thereof. Last, besides nearest-neighbor gating, several other machine learning techniques such as generalized linear models and support vector machines were tested and delivered comparable results. One important disadvantage of such methods is the limitation to multiple prototypes which are needed for training the model. Because of kernelized approaches, such techniques do not work with only one prototype, as this is the case with flowLearn.

4.4.5 Comparison to DeepCyTOF and FlowSOM

FlowLearn was compared to two recent state-of-the-art methods for population identification, DeepCyTOF (Li et al., 2017) and FlowSOM (Van Gassen et al., 2015). DeepCyTOF exhibited very high F_1-scores on multiple data sets, being better than all best-performing methods on data from the FlowCAP-I competition (Aghaeepour et al., 2013). In a recent study (Weber and Robinson, 2016), FlowSOM outperformed 18 competing tools. Both methods were run on the Mice data set and results were compared to the ones obtained by flowLearn (Figure 4.8). Since these methods return disjoint clusters (only one cluster ID per cell), suitable populations are given by all leaf populations in the gating hierarchy. The F_1-scores were evaluated in the same manner as for flowLearn. DeepCyTOF predicted seven out of ten populations with median $F_1 > 0.75$, with mean $F_1 = 0.79$. FlowSOM was able to achieve

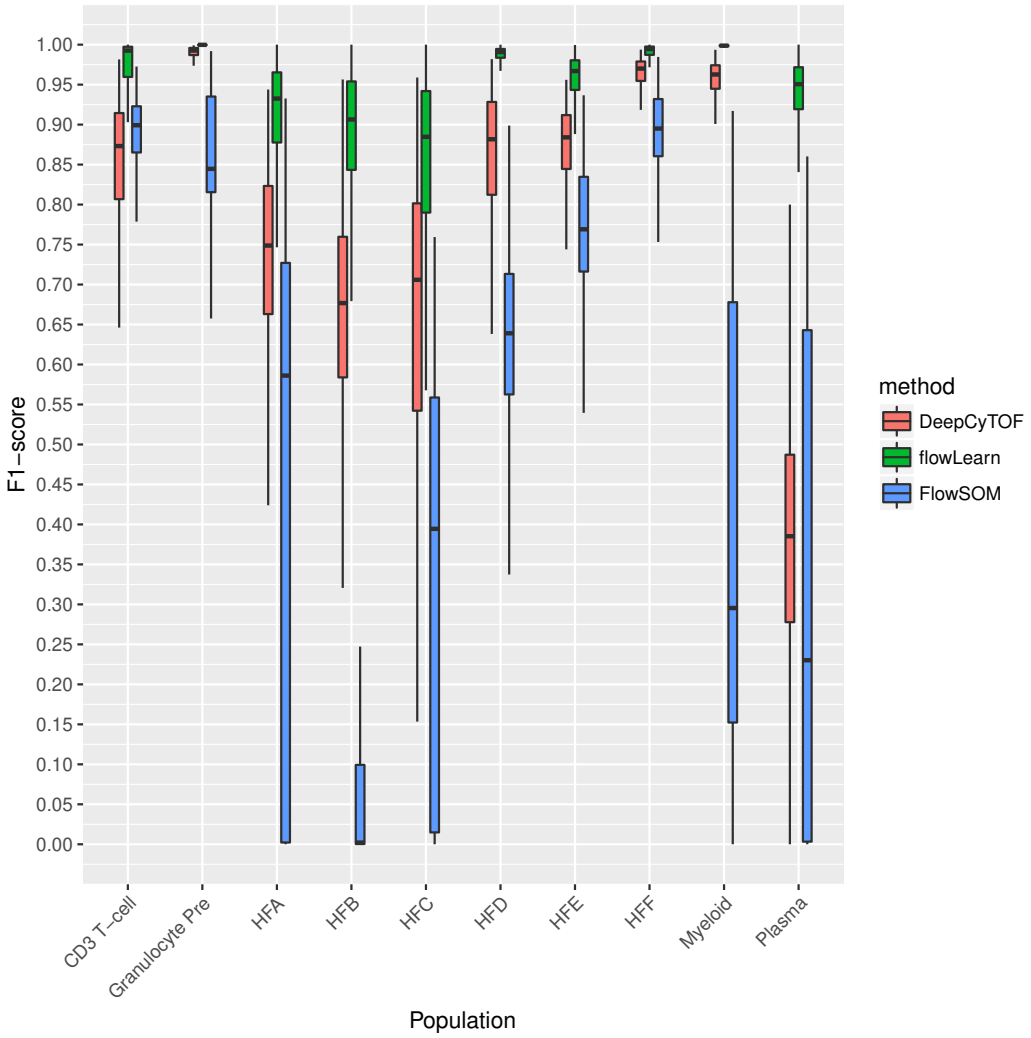

Fig. 4.8.: F_1-scores obtained by running flowLearn, DeepCyTOF, and FlowSOM on the leaf populations of the Mice data set.

median $F_1 > 0.75$ for four out of ten considered populations. Other populations were subpar. The FlowSOM average performance over all considered populations and samples, mean $F_1 = 0.53$, is similar to results obtained previously (Weber and Robinson, 2016). Using the same populations, and using $n_p = 1$ only, flowLearn achieved mean $F_1 = 0.94$, demonstrating flowLearn's superiority on this data.

Furthermore, runtimes of DeepCyTOF, FlowSOM and flowLearn were compared. On the Mice data, using ten populations, not including the one-time investment for training the deep network, DeepCyTOF was able to classify each sample in an average of one second. Using the same populations, FlowSOM was able to gate one sample in an average of 11 seconds which is in accordance with results reported in (Van Gassen et al., 2015). On the same

data, flowLearn predicted 2664 samples for one population in 1.37 minutes. For comparability, predicting ten populations with flowLearn takes 13.7 minutes, 0.31 seconds per sample. This does not include the time spent for providing manual gates to flowLearn.

4.5 Discussion

On two state-of-the-art data sets, flowLearn achieves median(F_1)-measures exceeding $F_1 > 0.99$ for 20/64 (31%), and $F_1 > 0.90$ for 51/64 (80%) of all analyzed populations. It predicts populations with low bias and variance, using only few examples (Mice: $n_p = 1$, FlowCAP: $n_p = 7$). Hence, it is possible to gate very few training files to obtain excellent results for most cell populations for thousands of additional files. However, there are populations, for which it is difficult to predict gates ($F_1 < 0.9$). Possible causes were identified: First, it was found that flowLearn can identify possible sample-based irregularities, in particular samples with densities that are significantly different from all other densities in a given dataset. Such differences might exist due to either biological diversity or anomalous sample reagent preparation and analysis. For example, the HFC population of the Mice data set has many wrongly predicted samples with low F_1-score (Figure 4.6). Here, flowLearn identified all samples with $F_1 < 0.56$ as outliers. In all included box plots, these are shown as individual dots. By visualizing sample densities from that population, and coloring each sample by F_1-score in Figure 4.10, it is visible that samples with F_1-score below the identified threshold form a distinct region, but not necessarily a separate cluster. Furthermore, a closer look at wrongly predicted HFC samples explains their poor performance. While in the large majority of samples, the HFC population is very well pronounced, in samples with low F_1-score, it is either completely missing or pronounced only very weakly. This leads to wrongly set thresholds in the training data, or failed DTW alignments. Detecting such irregularities confirms flowLearn's capabilities of being used as an appropriate tool for quality checking.

Next, the sub-optimal performance of some populations in the FlowCAP data set was analyzed. Here, all datasets exhibit a wide diversity in terms of samples, centers and gates (Finak et al., 2016). It was observed that this variability has impact on the ability of flowLearn to correctly predict gates.

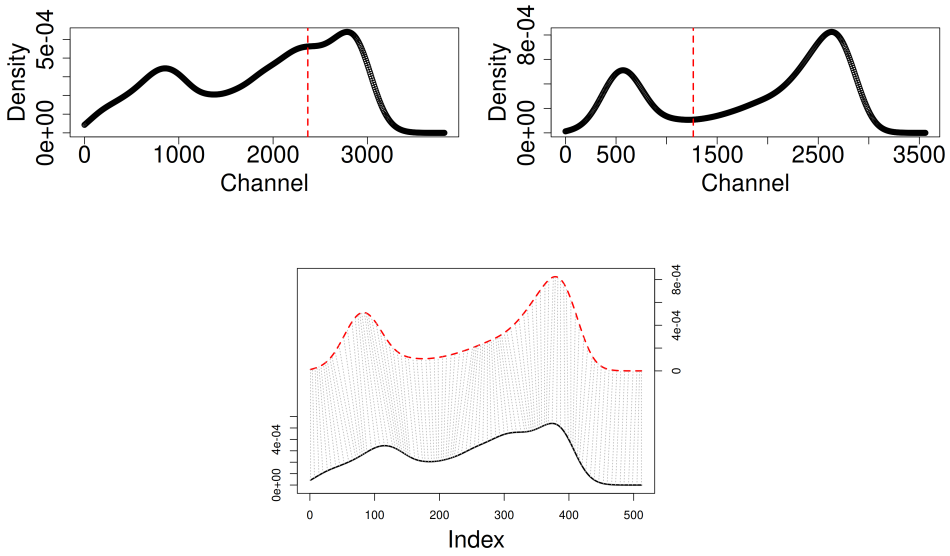

Fig. 4.9.: Differences in samples and gates in the FlowCAP data set, exemplary for one channel of the CD4 Effector population: Density plots show the HFC density from two samples (top left, top right). It is visible that the true thresholds (red) are much different from each other. The alignment between the two densities is shown on the right.

An example is shown in Figure 4.9, which shows the same T-cell / CD4 Effector population from two different FCS files and that densities and gates for the displayed channel are very different from each other. Despite the difference in both densities, the alignment looks correct. However, because of the difference in threshold locations, a threshold transfer will incorrectly predict the resulting gate. Similar effects on most other low-performing populations from the FlowCAP data set were observed. Reasons for wrongly predicted gates include differences in thresholds, failed alignments due to large differences in samples, and also few failed alignments even though samples were similar.

To prevent wrong alignments due to large sample differences, flowLearn's clustering approach is essential and picks prototypes that are representative for all other samples. In some cases (especially in the presence of high sample diversity) it can happen that a prototype is wrongly aligned to all other samples, resulting in reduced prediction performance. The number of prototypes is important as well. It was shown that flowLearn can accurately predict gates using only one prototype, but again, when there is high sample diversity, performance can be improved significantly by using more. The data might be generated in experiments in which samples can be categorized, for example into healthy/diseased or wild type/knockout,

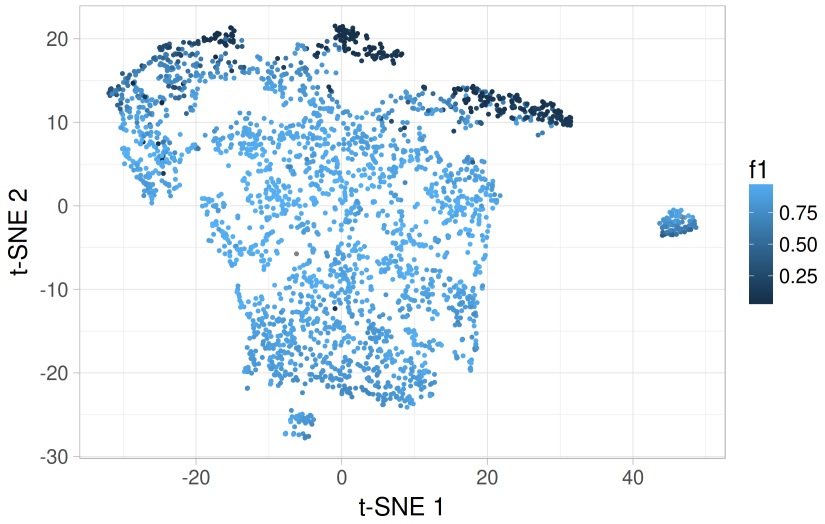

Fig. 4.10.: T-SNE projection of Mice densities (HFC population) from one channel, colored by F1-score according to predictions using $n_p = 1$. Samples with low performance cluster together.

or samples were analyzed in different centers. In such cases where there is prior knowledge about the data, the number of prototypes should be chosen accordingly. Generally, performance depends on the heterogeneity of the observed densities. At present, the number of prototypes is set manually within the pipeline, but this choice could be automated in order to guarantee a sufficient sample homogeneity within every single prototype cluster.

Last, when compared to two recent state-of-the-art methods, DeepCyTOF and FlowSOM, flowLearn showed superior performance in terms of F_1-scores. While all methods aim to solve the same problem of identifying cell populations, it is worth to highlight that depending on the application, one might be better suited than another. FlowSOM has the advantage that it does not require any prior knowledge about the data at hand, in particular no manual gating is needed. Being completely unsupervised, however it is expected that it cannot perform as well as its (semi-)supervised counterparts. Both DeepCyTOF and FlowSOM can be used for high-dimensional data such as from mass cytometry. In the case of DeepCyTOF, if addressing the difficult problem of automatically finding good hyper-parameters and network architectures (Klein et al., 2016), it may be used for end-to-end machine learning, i.e. directly inferring biological variables from a given set of cells, without gating (Mair et al., 2016). In contrast, flowLearn automates the prevalent paradigm of gating using channel thresholds for lower-dimensional data. As shown in section 4.4.4, the selection of markers and channels constitute a

strong biological prior. This condition gives our tool an advantage over other tools that have to search a much larger function space. At the same time, being based on channel thresholds makes our results interpretable and adjustable, an advantage that is not directly given for other tools. Furthermore, DeepCyTOF and FlowSOM return assignments of disjoint cell populations. In a gating hierarchy, cells can have multiple labels, and assignment with such methods is difficult. Last, it is worth to note that in the right hands, by carefully choosing method parameters, both DeepCyTOF and FlowSOM may yield better results. In contrast, flowLearn's parameters are robust and tuning is not necessary.

4.6 Summary

In this chapter, I have presented flowLearn, a software tool that is able to accurately identify FCM cell populations. In a quality checking setting, flowLearn can also be used to identify both anomalous samples and aberrant thresholds from existing gatings. Using simple density alignments, on two diverse data sets it demonstrated good to excellent performance on a wide variety of populations, including very rare ones. This can be achieved using as few as only one gated sample, keeping invested resources for gating at a minimum level. Furthermore, on a large set of bone marrow samples, I have shown that flowLearn is superior to DeepCyTOF and FlowSOM, two top-performing methods according to recent comparisons. On highly diverse FCM samples such as from different datasets or centers, flowLearn shows its limitations, although in general, choosing more prototypes can increase performance significantly. The correct choice of reference densities is essential for prediction.

In the future I will investigate more options to choose better prototypes, for example by including alignment properties or using averaged densities instead of the current prototype-based approach. It would also be beneficial to include confidence measures for the suitability of a given prototype, for example based on alignment distances. With that, one could judge the number of needed prototypes as well. Furthermore, the use of alignments of two-dimensional densities is of high interest and would enable the usage of arbitrarily shaped gates. Being based on the `R` ecosystem, flowLearn is ready to be included into existing FCM analysis pipelines, and offers improvements thereof in terms of gating quality and resource investment.

5 Conclusion

In this thesis, I discussed three topics concerning the computational analysis of biological data, in particular clustering and classification problems. First, in metagenomics, I presented a machine learning pipeline for taxonomy-independent, sequence composition based binning of contigs to their originating and mostly unknown genomes. Specifically, a suitability analysis of various techniques, algorithms and their parameters was performed. First, it was shown how the setting of window parameters in oligonucleotide frequency computation heavily influences the resulting representation. Second, the choice of non-linear dimensionality reduction techniques was demonstrated to be essential when the goal is to obtain a data representation that focuses on compact and separated clusters. And last, favorable algorithms to separate such clusters into individual taxonomic units were evaluated, resulting in a fully automated pipeline. I showed on both theoretical and real-world metagenomes of varying sizes that the proposed method works well. It is difficult though, to achieve high binning performance using a taxonomy-independent, sequence composition based approach only. It is therefore desirable to further integrate techniques to refine a particular binning or, more importantly, combine different approaches and integrate auxiliary knowledge. This can open the way towards accurate and precise tools for de-novo binning.

Based on the results of metagenomic binning, I presented a second topic, namely single-cell genome contamination detection. Here, similar to binning, the task is to separate different species in a given de-novo assembly. However, due to the much lower and possibly imbalanced number of included species, it is inherently different. I created the software acdc as a fully automated tool to detect and remove contaminants with possibly unknown taxonomy. In particular, the method combines reference-based and mostly reference-free algorithms to find foreign DNA not belonging to a target organism. Using different models of cluster validity assessment, the tool provides interpretable confidence measures to an end-user who can further inspect results using an interactive interface or apply the software in a fully automated fashion, for example by integrating it into existing quality assurance pipelines. In both cases, auxiliary database knowledge can

be included to improve detection and removal accuracy. On a large set of single-cell assemblies, acdc was shown to achieve high performance in terms of precision and recall. Furthermore, it may be used to find impurities in metagenome-assembled genomes, as demonstrated on data from the CAMI challenge. It can also detect the presence of horizontally transferred, genetic material such as DNA from phages or plasmids. Currently, in collaboration with the Joint Genome Institute, we are evaluating the possibility to include acdc into their single-cell sequencing production pipeline. In the future, it may be viable to look at different methods for the representation of genomic sequences, since the current approach is not able to resolve inter-species differences. Here, both single-cell contamination detection and metagenomics may benefit from such improvements.

As it can be used to isolate single cells from metagenomes, flow cytometry as the third part of this thesis can be seen as a connecting element. A crucial task in FCM analysis is the delineation of cell populations from a given parent population. For this task, I developed flowLearn as a semi-supervised utility that allows for the fast and precise identification of cell populations. While for FCM gating, multivariate methods render a promising future, the current paradigm of bivariate gating still constitutes a strong biological prior. In that setting, the manual tuning of thresholds is time-consuming and subjective. In contrast, with flowLearn I showed that the use of density alignments with prototypes can serve as an automation of this process. Depending on the complexity of the gated population, often only one prototype has to be manually gated in order to automatically predict the same population on thousands of other samples with good to excellent performance. This was demonstrated on a large and diverse set of FCM samples. Furthermore, it was shown that flowLearn outperformed two current state-of-the-art tools for gating. However, gates based on thresholds may not be suitable for populations that do not allow for rectangle gates. To overcome this limitation, the use of two-dimensional alignments is a promising extension of flowLearn in the future.

The application of a variety of techniques from machine learning to the three presented topics emphasizes the importance of a good choice of involved methods and parameters. While the presented grouping problems appear very similar to each other, seemingly unimportant specifics can change major parts of the approach for solving them. For example, clustering single cells for contamination detection is related to metagenomic binning

as both problems can be broken down to the assignment of sequences to clusters. However, due to the different numbers of involved genomes in both problems, the clustering task is inherently different and requires problem-specific choices of methods and tuning of parameters. In flow cytometry, cell populations can be seen as high-dimensional clusters as well. But the application of dimensionality reduction for that task works only to a small extent. Hence, the accurate identification of cell populations required a more thorough choice of the pipeline. In the same way, the integration of auxiliary data requires careful attention. It is the question, how to integrate more information, for example for metagenomics, such that it will help to craft high-precision binning tools. These points underline that there seldom is a silver bullet in machine learning. Nonetheless, machine learning has shown good results not only in the context of this thesis.

With ideas for metagenomic binning, the development of acdc, and flowLearn, based on various machine learning techniques, this thesis provided possible solutions for open problems around grouping methods for the annotation and sorting of single cells. It was demonstrated that the contributions perform well on many kinds of data and I hope that an adoption of the tools will enable analysts to more efficiently advance research in their respective fields.

A

Appendix

A.1 Software Availability

A.1.1 Acdc

Acdc source code is available on GitHub (`https://github.com/mlux86/acdc`).
A web version of acdc has been provided within the Bielefeld University Bioinformatics Services (BiBiServ) infrastructure (`https://bibiserv.cebitec.uni-bielefeld.de/acdc`).
Evaluation data and accompanying acdc results are available from Bielefeld University (`http://doi.org/10.4119/unibi/2904577`).

A.1.2 FlowLearn

FlowLearn is available as an `R` package on GitHub (`https://github.com/mlux86/flowLearn`).

A.2 List of genomes used in the NCBI-9 data set

NCBI accession	Species
NC_014471.1	*Ignisphaera aggregans DSM 17230*
NC_009954.1	*Caldivirga maquilingensis IC-167*
NC_011766.1	*Desulfurococcus kamchatkensis 1221n*
NC_017461.1	*Fervidicoccus fontis Kam940*
NC_018719.1	*Candidatus Nitrososphaera gargensis Ga9.2*
NC_014222.1	*Methanococcus voltae A3*
NC_007681.1	*Methanosphaera stadtmanae DSM 3091*
NC_021355.1	*Methanobrevibacter sp AbM4*
NC_018706.1	*Acinetobacter baumannii TYTH-1*

Tab. A.1.: List of genomes used in the NCBI-9 data set

A.3 Results on the CAMI data

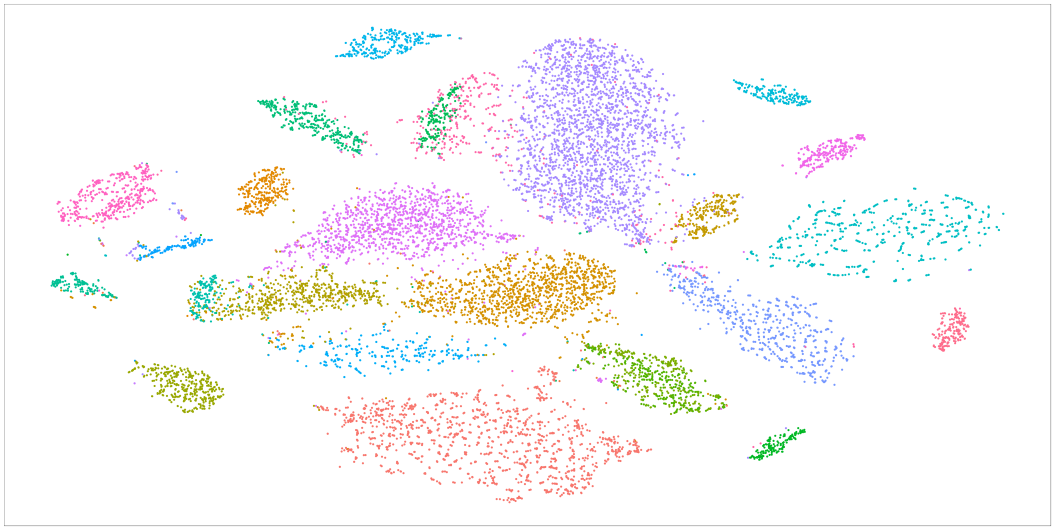

Fig. A.1.: Data representation using t-SNE of the CAMI-30 low complexity metagenome, colored by gold standard binning.

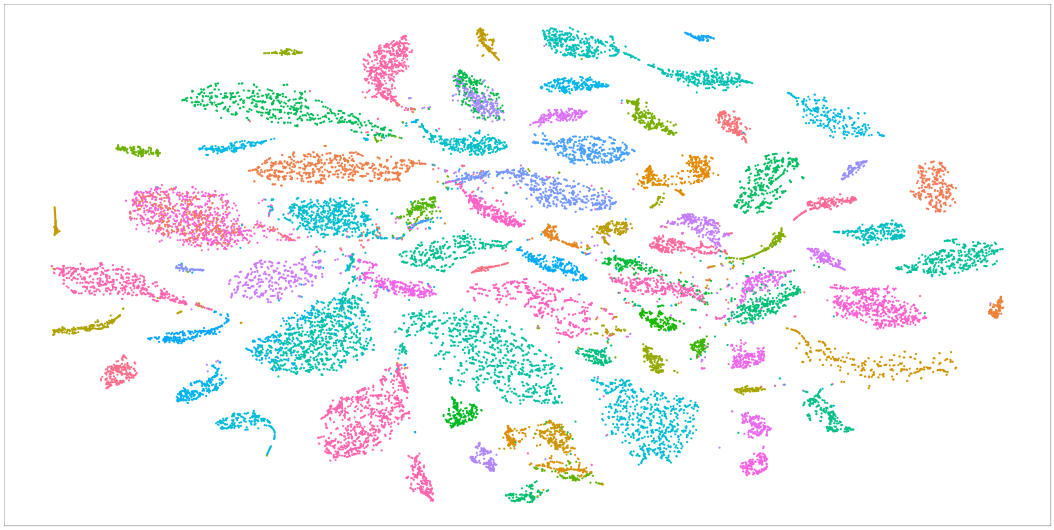

Fig. A.2.: Data representation using t-SNE of the CAMI-93 medium complexity metagenome, colored by gold standard binning.

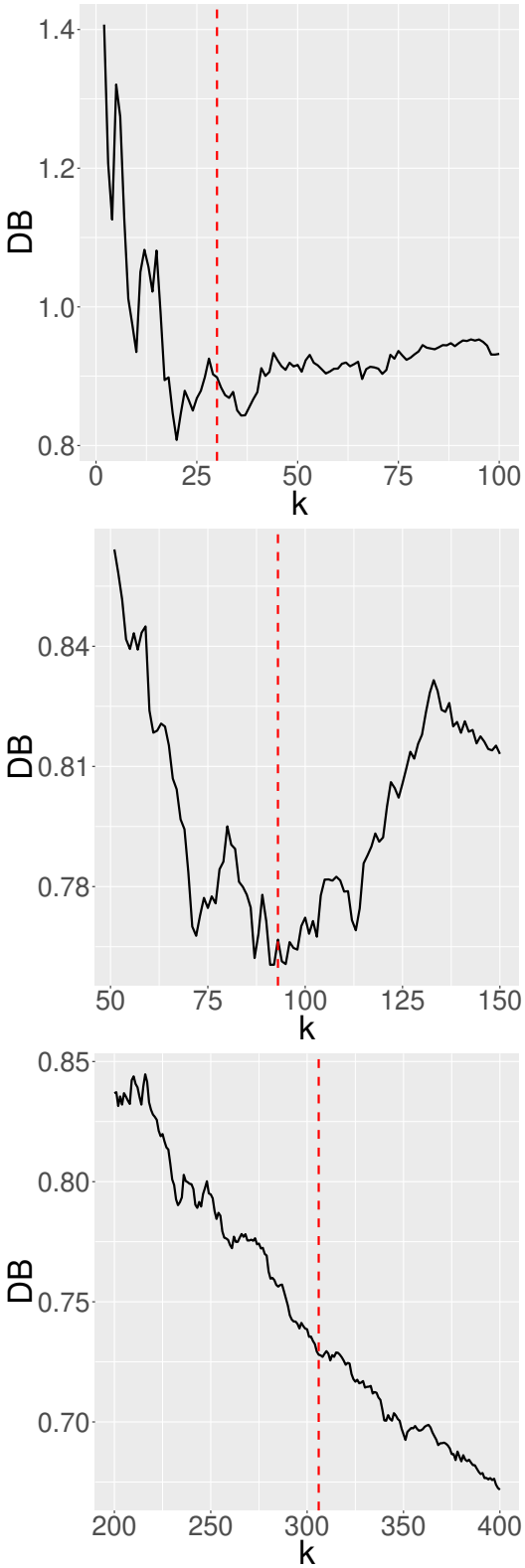

Fig. A.3.: Estimated number of clusters for the `CAMI-30`, `CAMI-93`, `CAMI-306` data sets (top, middle, bottom, respectively). Dashed red lines indicate the true number of clusters.

A.4 Acdc parameters

Method	Parameter description
Data pre-processing	Given a target of n data points (by default, $n = 1000$), the window width is fixed as $w = \sum_i l_i / n$, where l_i is the length of contig i. Default choices of $\Delta w = w/2$ and $k = 4$ (tetramer frequencies) are robust. For contigs with $l_i < w$, the window width is taken as large as possible ($w = l_i$).
t-SNE	The parameter $\theta = 0.5$ is a trade-off between speed and accuracy. The perplexity is set to $\text{perp}(n) = \lfloor \log(n)^2 \rfloor$. It can be seen as an effective neighborhood size that controls the graininess of clusters. A small number of data points n receives a small perplexity whereas with growing n the perplexity saturates.
DIP	The significance level which is uncritical as it is $\alpha = 0$ in the large majority of significant cases. Furthermore, the DIP split threshold, i.e. the percentage of data points, for which multimodality was detected, can be seen as a control of detection precision. A default value of $q_{dip} = 0.001$ was found to work very well throughout all tested data sets.
CC	The number of clusters found depends on the underlying graph. In acdc, the graph is constructed by connecting each data point to its m mutual nearest neighbors. The parameter m can be interpreted as a rough guide for the minimum number of data points contained in a separate cluster. To be able to detect also very small contamination, a default value of $m = 9$ is used.
Bootstrapping	The number of bootstraps is set to $B = 10$. Setting B to a larger number will result in more accurate confidence estimations at the cost of a longer runtime.
Kraken	The only parameter required by Kraken is the database to be used. It can be specified as a parameter to acdc as well.
RNAmmer	16S rRNA gene sequence prediction using RNAmmer does not require any parameters.

Tab. A.2.: Description of parameters for various techniques used in acdc.

A.5 Evaluation of the optimal number of nearest neighbors m

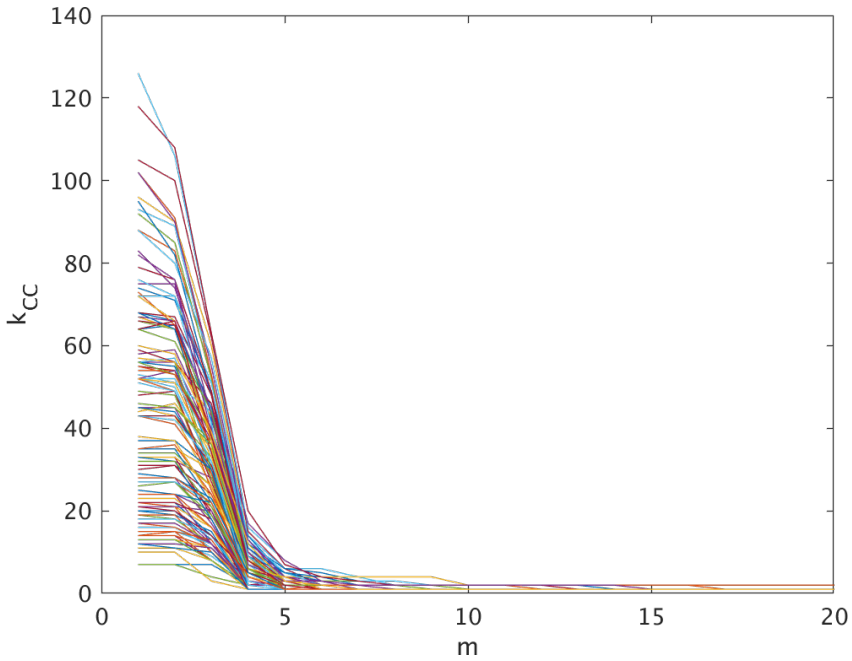

Fig. A.4.: Evaluation of different values of m. Here, on a set of 201 clean assemblies, the default value of $m = 9$ in acdc is robust and determined as the 90%-quantile of the distribution of m' for which the underlying graph does not dissolve into individual connected components anymore.

A.6 Description of the `simulated` data set

Level of relatedness	Species (NCBI accession)
Kingdom	• Acetobacter pasteurianus (NC_013209) • Ammonifex degensii (NC_013385) • Borrelia crocidurae (NC_017808)
Phylum	• Peptoclostridium difficile (NC_017174) • Bacillus cereus (NC_016779) • Faecalitalea cylindroides (NC_021019)
Class	• Staphylococcus haemolyticus (NC_007168) • Pediococcus claussenii (NC_016605) • Listeria monocytogenes (NC_021823)
Order	• Enterococcus faecium (NC_017960) • Pediococcus claussenii (NC_016605) • Weissella koreensis (NC_015760)
Family	• Lactococcus garvieae (NC_015930) • Streptococcus dysgalactiae (NC_022532)
Genus	• Streptococcus agalactiae (NC_021486) • Streptococcus mutans (NC_017768) • Streptococcus gallolyticus (NC_017576)
Species	• Streptococcus suis (NC_012926) • Streptococcus suis ST3 (NC_015433) • Streptococcus suis TL13 (NC_021213)

Tab. A.3.: Description and availability of the `simulated` data set.

A.7 Description of the `mix` data set

Species name	Ref.	Strain availability
Herbinix luporum SD1D^T	Koeck et al., 2015	Prof. Dr. W. Schwarz, Prof. Dr. W. Liebel, Dr. V. Zverlov, Dr. D. Koeck, Technische Universität München, Institute for Microbiology, Munich, Germany
Clostridium sp. hoe 37/3	NA	
Propionispora sp. 2/2-37	Koeck et al., 2015	
Proteiniborus sp. DW1	NA	Prof. Dr. H. König, Dr. K.G. Cibis, Johannes Gutenberg-University, Institute for Microbiology and Wine Research, Mainz, Germany
Peptoniphilaceae sp. SG1.4B	Cibis et al., 2016	
Methanobacterium formicicum MF^T	Maus et al., 2014	
Methanobacterium formicicum Mb9	NA	
Sporanaerobacter sp. PP17-6a	NA	Dr. M. Klocke and Dr. S. Hahnke, Leibniz-Institut für Agrartechnik Potsdam-Bornim e.V. (ATB), Department of Bioengineering, Potsdam, Germany
Methanobacterium bourgensis HAW	NA	Prof. Dr. Scherer, Dr. S. Off, Dr. Y.S. Kim, University of Applied Sciences Hamburg (HAW), Faculty Life Sciences / Research Center 'Biomass Utilization Hamburg', Hamburg, Germany

Tab. A.4.: Description and availability of the `mix` data set. Non-available references are denoted by 'NA'.

A.8 Results on the FlowCAP data set

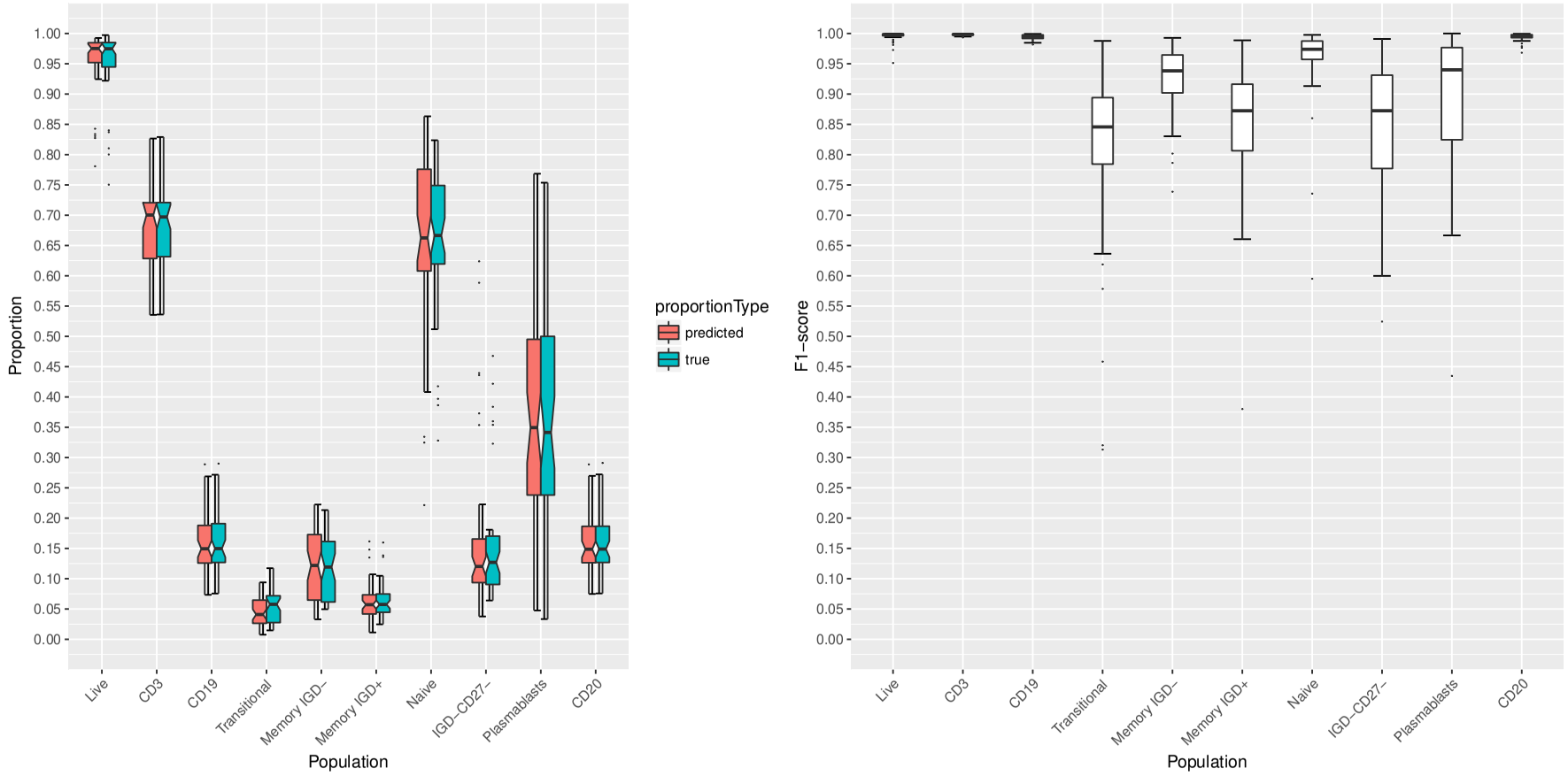

Fig. A.5.: Results on the FlowCAP B-cell dataset for $n_p = 7$. Left: True and predicted cell frequencies for each population. Right: F_1-scores for each population (63 samples). Outliers are shown as single dots.

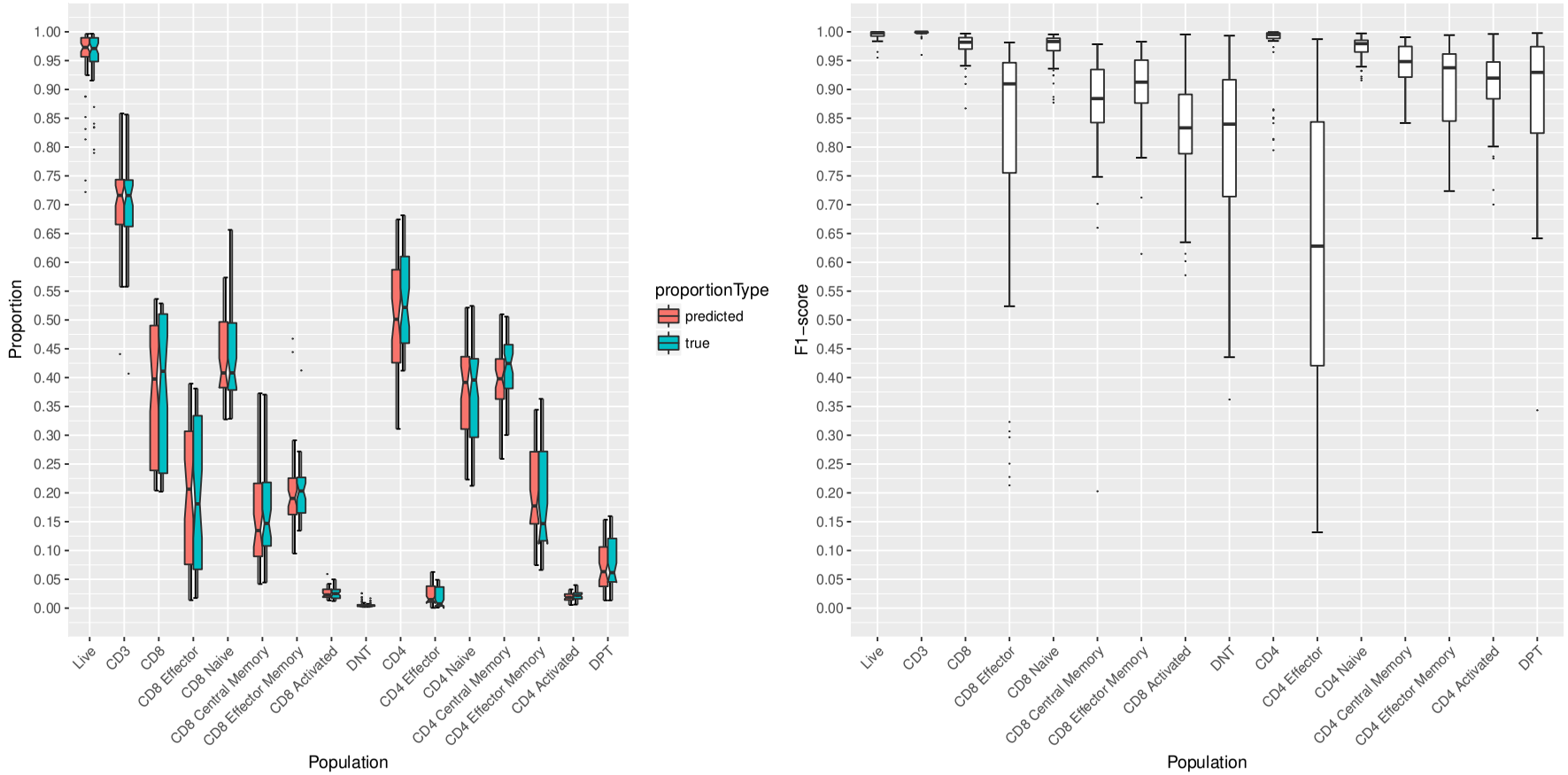

Fig. A.6.: Results on the FlowCAP T-cell dataset for $n_p = 7$. Left: True and predicted cell frequencies for each population. Right: F_1-scores for each population (63 samples). Outliers are shown as single dots. Missing boxes indicate aborted computation due to failure.

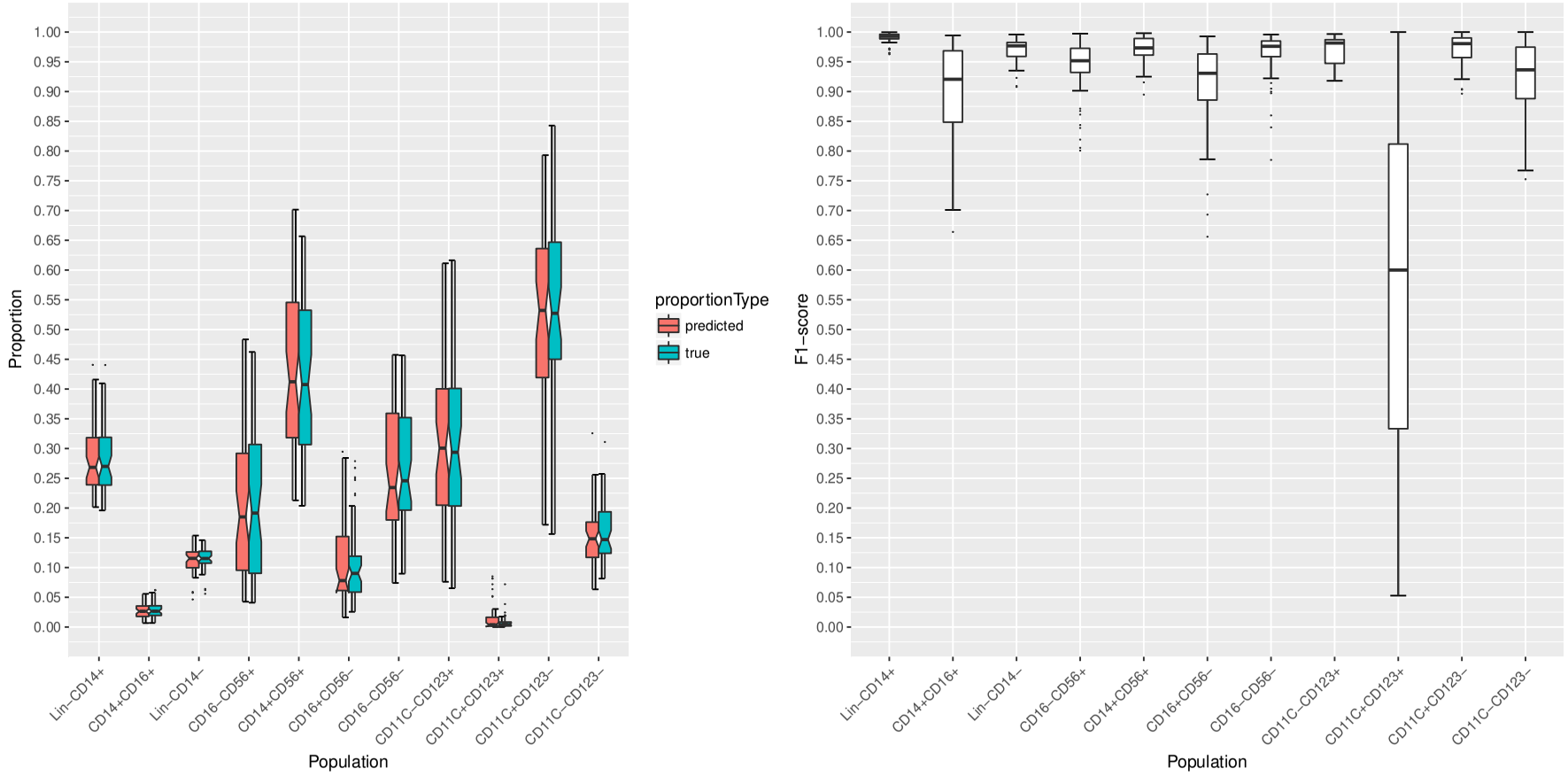

Fig. A.7.: Results on the FlowCAP DC dataset for $n_p = 7$. Left: True and predicted cell frequencies for each population. Right: F_1-scores for each population (63 samples). Outliers are shown as single dots. Missing boxes indicate aborted computation due to failure.

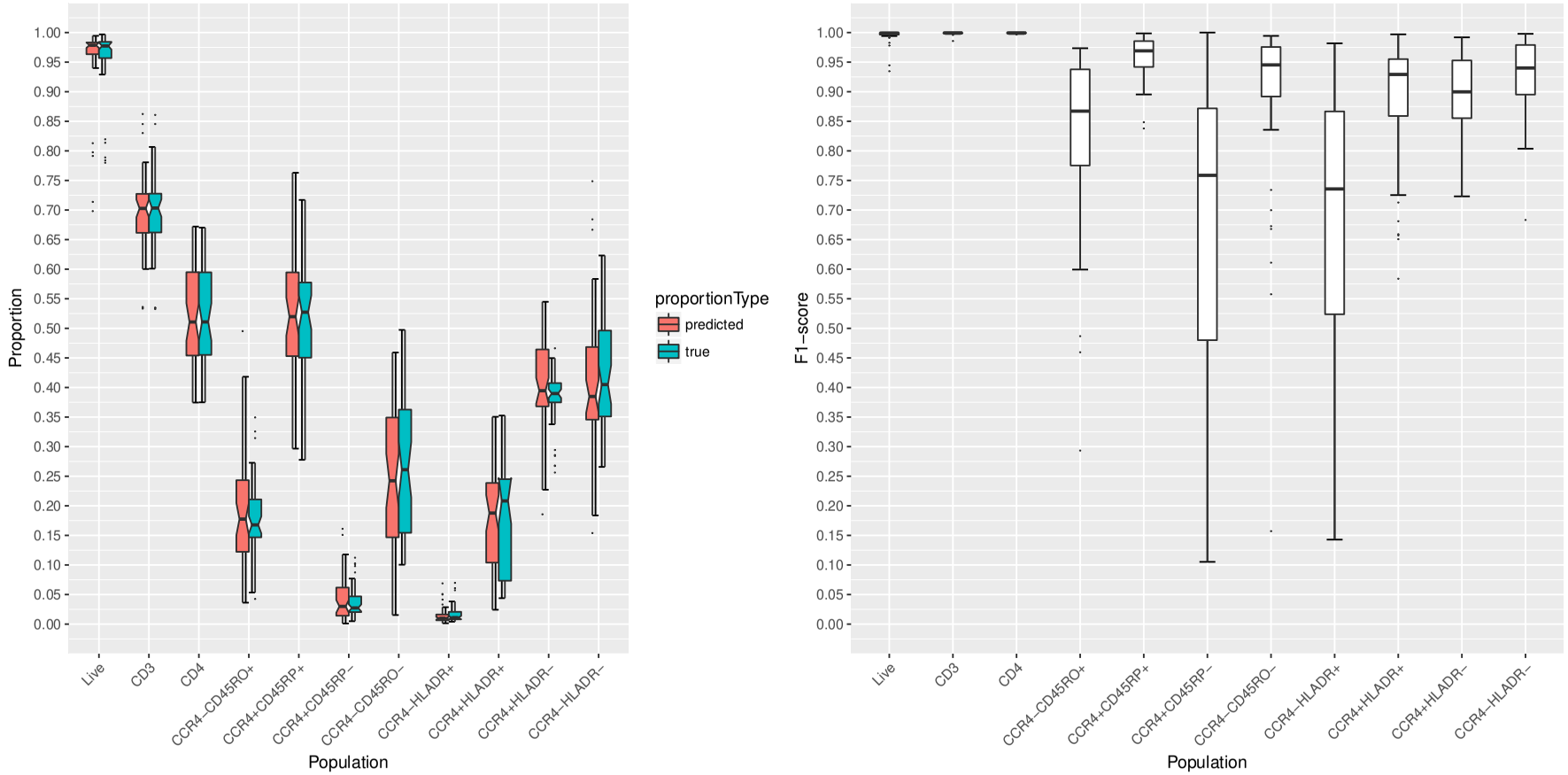

Fig. A.8.: Results on the FlowCAP T-reg dataset for $n_p = 7$. Left: True and predicted cell frequencies for each population. Right: F_1-scores for each population (63 samples). Outliers are shown as single dots.

Bibliography

Aghaeepour, Nima, Greg Finak, Holger Hoos, et al. (2013). „Critical assessment of automated flow cytometry data analysis techniques“. In: *Nature methods* 10.3, pp. 228–238 (cit. on pp. 7, 73, 75, 91).

Aghaeepour, Nima, Pratip Chattopadhyay, Maria Chikina, et al. (2016). „A benchmark for evaluation of algorithms for identification of cellular correlates of clinical outcomes“. In: *Cytometry Part A* 89.1, pp. 16–21 (cit. on p. 71).

Albertsen, Mads, Philip Hugenholtz, Adam Skarshewski, et al. (2013). „Genome sequences of rare, uncultured bacteria obtained by differential coverage binning of multiple metagenomes“. In: *Nature biotechnology* 31.6, pp. 533–538 (cit. on pp. 13, 14).

Amir, El-ad David, Kara L Davis, Michelle D Tadmor, et al. (2013). „viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia“. In: *Nature biotechnology* 31.6, pp. 545–552 (cit. on p. 73).

Ander, Christina, Ole B Schulz-Trieglaff, Jens Stoye, and Anthony J Cox (2013). „metaBEETL: high-throughput analysis of heterogeneous microbial populations from shotgun DNA sequences“. In: *BMC bioinformatics* 14.Suppl 5, S2 (cit. on p. 45).

Arthur, David and Sergei Vassilvitskii (2007). „k-means++: The advantages of careful seeding“. In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, pp. 1027–1035 (cit. on pp. 22, 28).

Bankevich, Anton, Sergey Nurk, Dmitry Antipov, et al. (2012). „SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing“. In: *Journal of Computational Biology* 19.5, pp. 455–477 (cit. on p. 61).

Bellman, Richard, Richard Ernest Bellman, Richard Ernest Bellman, and Richard Ernest Bellman (1961). *Adaptive control processes: a guided tour*. Vol. 4. Princeton University Press Princeton (cit. on p. 16).

Ben-Hur, Asa, Andre Elisseeff, and Isabelle Guyon (2001). „A stability based method for discovering structure in clustered data“. In: *Pacific symposium on biocomputing*. Vol. 7, pp. 6–17 (cit. on pp. 24–26).

Beyer, Kevin, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft (1999). „When is “nearest neighbor” meaningful?“ In: *International conference on database theory*. Springer, pp. 217–235 (cit. on p. 17).

Bishop, Christopher M (2006). *Pattern recognition and machine learning*. springer (cit. on p. 22).

Blainey, Paul C (2013). „The future is now: single-cell genomics of bacteria and archaea“. In: *FEMS microbiology reviews* 37.3, pp. 407–427 (cit. on pp. 4–7, 44, 71).

Bottou, Leon and Yoshua Bengio (1995). „Convergence properties of the k-means algorithms“. In: *Advances in neural information processing systems*, pp. 585–592 (cit. on p. 22).

Bowers, Robert M, Nikos C Kyrpides, Ramunas Stepanauskas, et al. (2017). „Minimum information about a single amplified genome (MISAG) and a metagenome-assembled genome (MIMAG) of bacteria and archaea“. In: *Nature biotechnology* 35.8 (cit. on pp. 6, 44).

Bremges, Andreas (2016). „Assembling the microbial dark matter“. PhD thesis (cit. on p. 1).

Bremges, Andreas, Irena Maus, Peter Belmann, et al. (2015). „Deeply sequenced metagenome and metatranscriptome of a biogas-producing microbial community from an agricultural production-scale biogas plant“. In: *Gigascience* 4.1, p. 33 (cit. on p. 1).

Brown, Michael and Carl Wittwer (2000). „Flow cytometry: principles and clinical applications in hematology“. In: *Clinical chemistry* 46.8, pp. 1221–1229 (cit. on p. 71).

Brown, Steve DM and Mark W Moore (2012). „The International Mouse Phenotyping Consortium: past and future perspectives on mouse phenotyping“. In: *Mammalian Genome* 23.9-10, pp. 632–640 (cit. on p. 86).

Camacho, Christiam, George Coulouris, Vahram Avagyan, et al. (2009). „BLAST+: architecture and applications“. In: *BMC bioinformatics* 10.1, p. 1 (cit. on pp. 45, 55, 66).

Charles, Trevor C and D Marco (2010). „The potential for investigation of plant-microbe interactions using metagenomics methods“. In: *Metagenomics: Theory, methods and applications*, pp. 102–107 (cit. on p. 11).

Cibis, Katharina Gabriela, Armin Gneipel, and Helmut König (2016). „Isolation of acetic, propionic and butyric acid-forming bacteria from biogas plants“. In: *Journal of biotechnology* (cit. on p. 108).

Clingenpeel, Scott, Patrick Schwientek, Philip Hugenholtz, and Tanja Woyke (2014a). „Effects of sample treatments on genome recovery via single-cell genomics“. In: *The ISME journal* 8.12, pp. 2546–2549 (cit. on p. 62).

Clingenpeel, Scott, Alicia Clum, Patrick Schwientek, Christian Rinke, and Tanja Woyke (2014b). „Reconstructing each cell's genome within complex microbial communities—dream or reality?“ In: *Frontiers in microbiology* 5 (cit. on p. 62).

Eberwine, James, Jai-Yoon Sul, Tamas Bartfai, and Junhyong Kim (2014). „The promise of single-cell sequencing“. In: *Nature methods* 11.1, pp. 25–27 (cit. on p. 43).

Ester, Martin, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. (1996). „A density-based algorithm for discovering clusters in large spatial databases with noise.“ In: *Kdd*. Vol. 96. 34, pp. 226–231 (cit. on p. 23).

Estivill-Castro, Vladimir (2002). „Why so many clustering algorithms: a position paper“. In: *ACM SIGKDD Explorations Newsletter* 4.1, pp. 65–75 (cit. on pp. 21, 50).

Finak, Greg, Marc Langweiler, Maria Jaimes, et al. (2016). „Standardizing flow cytometry immunophenotyping analysis from the human immunophenotyping consortium“. In: *Scientific reports* 6 (cit. on pp. 73, 75, 86, 87, 93).

Forbes, Jessica D, Natalie C Knox, Jennifer Ronholm, Franco Pagotto, and Aleisha Reimer (2017). „Metagenomics: the next culture-independent game changer“. In: *Frontiers in microbiology* 8 (cit. on p. 4).

Frank, Daniel N and Norman R Pace (2008). „Gastrointestinal microbiology enters the metagenomics era“. In: *Current opinion in gastroenterology* 24.1, pp. 4–10 (cit. on p. 11).

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2001). *The elements of statistical learning*. Vol. 1. Springer series in statistics Springer, Berlin (cit. on p. 80).

Gawad, Charles, Winston Koh, and Stephen R Quake (2016). „Single-cell genome sequencing: current state of the science“. In: *Nature reviews. Genetics* 17.3, p. 175 (cit. on pp. 44, 45).

Geer, Lewis Y, Aron Marchler-Bauer, Renata C Geer, et al. (2009). „The NCBI biosystems database“. In: *Nucleic acids research* 38.suppl_1, pp. D492–D496 (cit. on pp. 28, 61).

Gies, Esther A, Kishori M Konwar, J Thomas Beatty, and Steven J Hallam (2014). „Illuminating microbial dark matter in meromictic Sakinaw Lake“. In: *Applied and environmental microbiology* 80.21, pp. 6807–6818 (cit. on pp. 4, 11).

Gill, Steven R, Mihai Pop, Robert T DeBoy, et al. (2006). „Metagenomic analysis of the human distal gut microbiome“. In: *science* 312.5778, pp. 1355–1359 (cit. on p. 4).

Gisbrecht, Andrej, Barbara Hammer, Bassam Mokbel, and Alexander Sczyrba (2013). „Nonlinear dimensionality reduction for cluster identification in metagenomic samples“. In: *Information Visualisation (IV), 2013 17th International Conference*. IEEE, pp. 174–179 (cit. on p. 30).

Goodwin, Sara, John D McPherson, and W Richard McCombie (2016). „Coming of age: ten years of next-generation sequencing technologies“. In: *Nature Reviews Genetics* 17.6, pp. 333–351 (cit. on p. 1).

Gori, Fabio, Dimitrios Mavroedis, Mike SM Jetten, and Elena Marchiori (2011). „Genomic signatures for metagenomic data analysis: Exploiting the reverse complementarity of tetranucleotides“. In: *Systems Biology (ISB), 2011 IEEE International Conference on*. IEEE, pp. 149–154 (cit. on pp. 15, 16).

Gu, Zuguang, Roland Eils, and Matthias Schlesner (2016). „HilbertCurve: an R/Bioconductor package for high-resolution visualization of genomic data“. In: *Bioinformatics* 32.15, pp. 2372–2374 (cit. on p. 69).

Hahne, Florian, Alireza Hadj Khodabakhshi, Ali Bashashati, et al. (2010). „Per-channel basis normalization methods for flow cytometry data“. In: *Cytometry Part A* 77.2, pp. 121–131 (cit. on p. 83).

Handelsman, Jo (2004). „Metagenomics: application of genomics to uncultured microorganisms“. In: *Microbiology and molecular biology reviews* 68.4, pp. 669–685 (cit. on p. 4).

Harbom, Lise J, William D Chronister, and Michael J McConnell (2016). „Single neuron transcriptome analysis can reveal more than cell type classification“. In: *Bioessays* 38.2, pp. 157–161 (cit. on p. 1).

Hartigan, John A and PM Hartigan (1985). „The dip test of unimodality“. In: *The Annals of Statistics*, pp. 70–84 (cit. on pp. 50, 51).

Hastie, Trevor, Robert Tibshirani, Jerome Friedman, et al. (2009). *The elements of statistical learning*. Vol. 2. 1. Springer (cit. on pp. 16, 21, 22, 46, 55, 79).

Hedlund, Brian P, Jeremy A Dodsworth, Senthil K Murugapiran, Christian Rinke, and Tanja Woyke (2014). „Impact of single-cell genomics and metagenomics on the emerging view of extremophile “microbial dark matter”“. In: *Extremophiles* 18.5, pp. 865–875 (cit. on p. 5).

Hess, Matthias, Alexander Sczyrba, Rob Egan, et al. (2011). „Metagenomic discovery of biomass-degrading genes and genomes from cow rumen“. In: *Science* 331.6016, pp. 463–467 (cit. on p. 11).

Huang, Weichun, Leping Li, Jason R Myers, and Gabor T Marth (2012). „ART: a next-generation sequencing read simulator“. In: *Bioinformatics* 28.4, pp. 593–594 (cit. on p. 61).

Jain, Anil K (2010). „Data clustering: 50 years beyond K-means“. In: *Pattern Recognition Letters* 31.8, pp. 651–666 (cit. on pp. 13, 46).

Jain, Anil K and Richard C Dubes (1988). *Algorithms for clustering data*. Prentice-Hall, Inc. (cit. on p. 20).

Kaleem, Zahid, Eric Crawford, M Hanif Pathan, et al. (2003). „Flow cytometric analysis of acute leukemias: diagnostic utility and critical analysis of data". In: *Archives of pathology & laboratory medicine* 127.1, pp. 42–48 (cit. on p. 1).

Kalogeratos, Argyris and Aristidis Likas (2012). „Dip-means: an incremental clustering method for estimating the number of clusters". In: *Advances in neural information processing systems*, pp. 2393–2401 (cit. on pp. 50–52).

Keogh, Eamonn J and Michael J Pazzani (2001). „Derivative dynamic time warping". In: *Proceedings of the 2001 SIAM International Conference on Data Mining*. SIAM, pp. 1–11 (cit. on pp. 81, 82).

Klein, Aaron, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter (2016). „Fast bayesian optimization of machine learning hyperparameters on large datasets". In: *arXiv preprint arXiv:1605.07079* (cit. on p. 95).

Koeck, Daniela E., Wolfgang Ludwig, Gerhard Wanner, et al. (2015). „Herbinix hemicellulosilytica gen. nov., sp. nov., a thermophilic cellulose-degrading bacterium isolated from a thermophilic biogas reactor". In: *International Journal of Systematic and Evolutionary Microbiology* 65.8, pp. 2365–2371 (cit. on p. 108).

Koonin, Eugene V (2011). *The logic of chance: the nature and origin of biological evolution*. FT press (cit. on p. 48).

Koonin, Eugene V, Kira S Makarova, and L Aravind (2001). „Horizontal gene transfer in prokaryotes: quantification and classification". In: *Annual Reviews in Microbiology* 55.1, pp. 709–742 (cit. on p. 68).

Kouchaki, Samaneh, Santosh Tirunagari, Avraam Tapinos, and David L Robertson (2016). „Local binary patterns as a feature descriptor in alignment-free visualisation of metagenomic data". In: *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on*. IEEE, pp. 1–6 (cit. on p. 69).

Kvistborg, Pia, Cécile Gouttefangeas, Nima Aghaeepour, et al. (2015). „Thinking outside the gate: single-cell assessments in multiple dimensions". In: *Immunity* 42.4, p. 591 (cit. on pp. 73–75).

Laczny, Cedric C, Nicolás Pinel, Nikos Vlassis, and Paul Wilmes (2014). „Alignment-free Visualization of Metagenomic Data by Nonlinear Dimension Reduction". In: *Scientific reports* 4 (cit. on pp. 13, 14, 30, 46).

Lagesen, Karin, Peter Hallin, Einar Andreas Rødland, et al. (2007). „RNAmmer: consistent and rapid annotation of ribosomal RNA genes". In: *Nucleic acids research* 35.9, pp. 3100–3108 (cit. on p. 56).

Land, Miriam, Loren Hauser, Se-Ran Jun, et al. (2015). „Insights from 20 years of bacterial genome sequencing". In: *Functional & integrative genomics* 15.2, pp. 141–161 (cit. on p. 16).

Lee, John A and Michel Verleysen (2007). *Nonlinear dimensionality reduction*. Springer (cit. on p. 17).

Levandowsky, Michael and David Winter (1971). „Distance between sets“. In: *Nature* 234.5323, pp. 34–35 (cit. on p. 24).

Li, Huamin, Uri Shaham, Kelly P. Stanton, et al. (2017). „Gating Mass Cytometry Data by Deep Learning“. In: *Bioinformatics* (cit. on pp. 76, 86, 91).

Lisboa, Paulo JG (2013). „Interpretability in Machine Learning–Principles and Practice“. In: *International Workshop on Fuzzy Logic and Applications*. Springer, pp. 15–21 (cit. on p. 74).

Liu, Yanchi, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu (2010). „Understanding of internal clustering validation measures“. In: *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, pp. 911–916 (cit. on p. 46).

Maaten, Laurens Van der and Geoffrey Hinton (2008). „Visualizing data using t-SNE“. In: *Journal of Machine Learning Research* 9.2579-2605, p. 85 (cit. on p. 18).

Maier, Markus, Matthias Hein, and Ulrike Von Luxburg (2007). „Cluster identification in nearest-neighbor graphs“. In: *ALT*. Vol. 7. Springer, pp. 196–210 (cit. on pp. 52, 53).

Mair, Florian, Felix J Hartmann, Dunja Mrdjen, et al. (2016). „The end of gating? An introduction to automated analysis of high dimensional cytometry data“. In: *European journal of immunology* 46.1, pp. 34–43 (cit. on pp. 74, 95).

Malek, Mehrnoush, Mohammad Jafar Taghiyar, Lauren Chong, et al. (2015). „flowDensity: reproducing manual gating of flow cytometry data by automated density-based cell population identification“. In: *Bioinformatics* 31.4, pp. 606–607 (cit. on pp. 75, 82).

Mande, Sharmila S, Monzoorul Haque Mohammed, and Tarini Shankar Ghosh (2012). „Classification of metagenomic sequences: methods and challenges“. In: *Briefings in bioinformatics* 13.6, pp. 669–681 (cit. on pp. 12, 45).

Martinetz, Thomas M, Stanislav G Berkovich, and Klaus J Schulten (1993). „'Neural-gas' network for vector quantization and its application to time-series prediction“. In: *IEEE transactions on neural networks* 4.4, pp. 558–569 (cit. on p. 22).

Maus, Irena, Robbin Stantscheff, Daniel Wibberg, et al. (2014). „Complete genome sequence of the methanogenic neotype strain Methanobacterium formicicum MFT“. In: *Journal of biotechnology* 192, pp. 40–41 (cit. on p. 108).

Maus, Irena, Daniela E Koeck, Katharina G Cibis, et al. (2016). „Unraveling the microbiome of a thermophilic biogas plant by metagenome and metatranscriptome analysis complemented by characterization of bacterial and archaeal isolates“. In: *Biotechnology for biofuels* 9.1, p. 171 (cit. on p. 1).

Myers, Cory S (1980). „A comparative study of several dynamic time warping algorithms for speech recognition“. PhD thesis. Massachusetts Institute of Technology (cit. on p. 82).

Naeem, Raeece, Mamoon Rashid, and Arnab Pain (2013). „READSCAN: a fast and scalable pathogen discovery program with accurate genome relative abundance estimation“. In: *Bioinformatics* 29.3, pp. 391–392 (cit. on p. 45).

Nature (Jan. 2013). „Nature Method of the Year 2013“. In: *Nature Methods* 11.1. Editorial, pp. 1–1 (cit. on p. 43).

Navin, Nicholas E (2015). „The first five years of single-cell cancer genomics and beyond“. In: *Genome research* 25.10, pp. 1499–1507 (cit. on p. 43).

Nikolenko, Sergey I, Anton I Korobeynikov, and Max A Alekseyev (2013). „BayesHammer: Bayesian clustering for error correction in single-cell sequencing“. In: *BMC genomics* 14.1, S7 (cit. on p. 2).

Parks, Donovan H, Michael Imelfort, Connor T Skennerton, Philip Hugenholtz, and Gene W Tyson (2015). „CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes“. In: *Genome research* 25.7, pp. 1043–1055 (cit. on p. 45).

Parks, Donovan H, Christian Rinke, Maria Chuvochina, et al. (2017). „Recovery of nearly 8,000 metagenome-assembled genomes substantially expands the tree of life.“ In: *Nature microbiology* (cit. on p. 11).

Qin, Junjie, Ruiqiang Li, Jeroen Raes, et al. (2010). „A human gut microbial gene catalog established by metagenomic sequencing“. In: *nature* 464.7285, p. 59 (cit. on p. 11).

Raes, Jeroen, Ivica Letunic, Takuji Yamada, Lars Juhl Jensen, and Peer Bork (2011). „Toward molecular trait-based ecology through integration of biogeochemical, geographical and metagenomic data“. In: *Molecular systems biology* 7.1 (cit. on p. 11).

Rand, William M (1971). „Objective criteria for the evaluation of clustering methods“. In: *Journal of the American Statistical association* 66.336, pp. 846–850 (cit. on p. 25).

Ratanamahatana, Chotirat Ann and Eamonn Keogh (2004). „Everything you know about dynamic time warping is wrong“. In: *Third Workshop on Mining Temporal and Sequential Data*. Citeseer, pp. 22–25 (cit. on p. 81).

Rinke, Christian, Patrick Schwientek, Alexander Sczyrba, et al. (2013). „Insights into the phylogeny and coding potential of microbial dark matter“. In: (cit. on pp. 1, 11, 12, 43–45, 62).

Saeys, Yvan, Sofie Van Gassen, and Bart N Lambrecht (2016). „Computational flow cytometry: helping to make sense of high-dimensional immunology data“. In: *Nature Reviews Immunology* 16.7, pp. 449–462 (cit. on p. 73).

Sakoe, Hiroaki and Seibi Chiba (1978). „Dynamic programming algorithm optimization for spoken word recognition“. In: *IEEE transactions on acoustics, speech, and signal processing* 26.1, pp. 43–49 (cit. on p. 81).

Salter, Susannah J, Michael J Cox, Elena M Turek, et al. (2014). „Reagent and laboratory contamination can critically impact sequence-based microbiome analyses“. In: *BMC biology* 12.1, p. 87 (cit. on p. 44).

Sczyrba, A, P Hofmann, P Belmann, et al. (2017). „Critical Assessment of Metagenome Interpretation- a benchmark of computational metagenomics software. bioRxiv: 099127“. In: (cit. on p. 29).

Sedlar, Karel, Kristyna Kupkova, and Ivo Provaznik (2017). „Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics“. In: *Computational and structural biotechnology journal* 15, pp. 48–55 (cit. on pp. 12, 13, 16).

Shaham, Uri and Stefan Steinerberger (2017). „Stochastic Neighbor Embedding separates well-separated clusters“. In: *arXiv preprint arXiv:1702.02670* (cit. on p. 19).

Shapiro, Howard M (2005). *Practical flow cytometry*. John Wiley & Sons (cit. on pp. 1, 71).

Shi, Jianbo and Jitendra Malik (2000). „Normalized cuts and image segmentation“. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8, pp. 888–905 (cit. on p. 23).

Silverman, Bernard W (1986). *Density estimation for statistics and data analysis*. Vol. 26. CRC press (cit. on p. 79).

Speicher, Michael R (2013). „Single-cell analysis: toward the clinic“. In: *Genome Med* 5.8, pp. 1–3 (cit. on p. 43).

Steinbach, Michael, Levent Ertöz, and Vipin Kumar (2004). „The challenges of clustering high dimensional data“. In: *New directions in statistical physics*. Springer, pp. 273–309 (cit. on p. 17).

Sukhdeo, Kumar, Rosanto I Paramban, Jason G Vidal, et al. (2013). „Multiplex flow cytometry barcoding and antibody arrays identify surface antigen profiles of primary and metastatic colon cancer cell lines“. In: *PloS one* 8.1, e53015 (cit. on p. 1).

Swan, Brandon K, Manuel Martinez-Garcia, Christina M Preston, et al. (2011). „Potential for chemolithoautotrophy among ubiquitous bacteria lineages in the dark ocean“. In: *Science* 333.6047, pp. 1296–1300 (cit. on p. 43).

Tarjan, Robert (1972). „Depth-first search and linear graph algorithms“. In: *SIAM journal on computing* 1.2, pp. 146–160 (cit. on pp. 24, 52).

Teeling, Hanno, Anke Meyerdierks, Margarete Bauer, Rudolf Amann, and Frank Oliver Glöckner (2004). „Application of tetranucleotide frequencies for the assignment of genomic fragments“. In: *Environmental microbiology* 6.9, pp. 938–947 (cit. on pp. 15, 48).

Tennessen, Kristin, Evan Andersen, Scott Clingenpeel, et al. (2015). „ProDeGe: a computational protocol for fully automated decontamination of genomes“. In: *The ISME journal* (cit. on pp. 45, 62, 65).

Thomas, Torsten, Jack Gilbert, and Folker Meyer (2012). „Metagenomics-a guide from sampling to data analysis“. In: *Microbial informatics and experimentation* 2.1, p. 3 (cit. on p. 12).

Tibshirani, Robert, Guenther Walther, and Trevor Hastie (2001). „Estimating the number of clusters in a data set via the gap statistic“. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.2, pp. 411–423 (cit. on p. 25).

Upton, Graham and Ian Cook (1996). *Understanding statistics*. Oxford University Press (cit. on p. 85).

Van Der Maaten, Laurens (2014). „Accelerating t-SNE using tree-based algorithms.“ In: *Journal of machine learning research* 15.1, pp. 3221–3245 (cit. on pp. 18, 19, 31, 46, 49, 60, 73).

Van Gassen, Sofie, Britt Callebaut, Mary J Van Helden, et al. (2015). „FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data“. In: *Cytometry Part A* 87.7, pp. 636–645 (cit. on pp. 76, 86, 91, 92).

Venables, William N and Brian D Ripley (2013). *Modern applied statistics with S-PLUS*. Springer Science & Business Media (cit. on p. 79).

Vendramin, Lucas, Ricardo JGB Campello, and Eduardo R Hruschka (2010). „Relative clustering validity criteria: A comparative overview“. In: *Statistical analysis and data mining: the ASA data science journal* 3.4, pp. 209–235 (cit. on pp. 13, 46).

Von Luxburg, Ulrike (2007). „A tutorial on spectral clustering“. In: *Statistics and computing* 17.4, pp. 395–416 (cit. on pp. 23, 52).

Von Luxburg, Ulrike et al. (2010). „Clustering stability: an overview“. In: *Foundations and Trends in Machine Learning* 2.3, pp. 235–274 (cit. on p. 26).

Wang, Yi, Henry Chi Ming Leung, Siu Ming Yiu, and Francis Yuk Lun Chin (2014). „MetaCluster-TA: taxonomic annotation for metagenomic data based on assembly-assisted binning“. In: *BMC genomics* 15.1, S12 (cit. on pp. 13, 14).

Weber, Lukas M and Mark D Robinson (2016). „Comparison of clustering methods for high-dimensional single-cell flow and mass cytometry data“. In: *Cytometry Part A* 89.12, pp. 1084–1096 (cit. on pp. 91, 92).

Wood, Derrick E and Steven L Salzberg (2014). „Kraken: ultrafast metagenomic sequence classification using exact alignments“. In: *Genome biology* 15.3, R46 (cit. on pp. 2, 45, 55).

Woyke, Tanja, Alexander Sczyrba, Janey Lee, et al. (2011). „Decontamination of MDA reagents for single cell whole genome amplification“. In: *PloS one* 6.10, e26161 (cit. on pp. 44, 49).

Yi, Gangman, Sing-Hoi Sze, and Michael R Thon (2007). „Identifying clusters of functionally related genes in genomes“. In: *Bioinformatics* 23.9, pp. 1053–1060 (cit. on p. 2).

Zhang, Xiaoyan, Sadie L Marjani, Zhaoyang Hu, et al. (2016). „Single-cell sequencing for precise cancer research: progress and prospects“. In: *Cancer research* 76.6, pp. 1305–1312 (cit. on p. 1).

Webpages and images

3i, Project (2017). *Infection, Immunity, Immunophenotyping*. URL: `http://www.immunophenotype.org/` (visited on Sept. 5, 2017) (cit. on p. 1).

Erbe, Eric (2005). *E coli at 10000x*. URL: `https://commons.wikimedia.org/wiki/File:E_coli_at_10000x,_original.jpg` (visited on Sept. 15, 2017) (cit. on p. 5).

Illumina (2016). *Single-Cell Research – An Overview of Recent Single-Cell Research Publications Featuring Illumina Technology*. URL: `https://www.illumina.com/content/dam/illumina-marketing/documents/products/research_reviews/single-cell-sequencing-research-review.pdf` (visited on Sept. 5, 2017) (cit. on p. 1).

ImageJ (2005). *E coli at 10000x*. URL: `https://commons.wikimedia.org/wiki/File:FluorescentCells.jpg` (visited on Sept. 15, 2017) (cit. on p. 7).

Peaco, Jim (2001). *Grand prismatic spring*. URL: `https://commons.wikimedia.org/wiki/File:Grand_prismatic_spring.jpg` (visited on Sept. 15, 2017) (cit. on p. 4).

List of Figures

List of Tables

Colophon

This thesis was typeset with LaTeX 2ε. It is based on the *Clean Thesis* style developed by Ricardo Langner.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.