# Unsupervised representation learning for single-cell transcriptomic and epigenomic data

Felix Raimundo

PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT

DE L'UNIVERSITÉ PSL

Préparée à MINES ParisTech

# Unsupervised representation learning for single-cell transcriptomic and epigenomic data
# Apprentissage non supervise pour l'epigenomique et transcriptomique en cellule unique


Soutenue par

**Felix Raimundo**

Le 2022/12/07


École doctorale n°621

**ISMME - Ingénierie des Systèmes, Matériaux, Mécanique, Énergétique**

Spécialité

**Bio-informatique**

Composition du jury :

| | |
|---|---|
| Jean-Philippe VERT<br>Owkin / Mines Paris - HDR | *Directeur de thèse* |
| Céline VALLOT<br>CNRS / Institut Curie - DR | *co-Directrice de thèse* |
| Stein AERTS<br>Univerity of Leuven - Full professor | *Rapporteur, Président du jury* |
| Olivier GANDRILLON<br>CNRS / ENS Lyon - DR | *Rapporteur* |
| Laura CANTINI<br>CNRS / ENS Paris - CR | *Examinatrice* |
| Antonio RAUSSELL<br>Université Paris Cité, MCU-PH | *Examinateur* |

MINES ParisTech | PSL

# Abstract

In recent years, single-cell transcriptomics and epigenomics have allowed biologist to observe tissues at a new resolution. Using these protocols, we are now able to observe the whole distribution of cell states within a tissue, instead of just measuring an aggregate cell state. With these new types of measurements has come the need for new statistical methods to analyze them. Indeed the previous generation of analysis tools were designed for a regime of few high quality samples, while these new measurements are much higher in quantity, but of significantly lower quality. This problem of low quality is even more pronounced for single-cell epigenomics protocols, due to cells only having two copies of the genome, compared to the hundreds of thousands of RNA molecules present in the cell. Since epigenomics and transcriptomics profiles are evaluated across a high number of variables, there has been a great interest in methods for reducing the dimension of the data.

This explosion of interest has led to numerous new algorithms and a thriving community of methods developers. Their work has however not yet been fully adopted by practicing bioinformaticians, either because they were not deemed reliable enough, or because they failed to properly answer biological questions. In this thesis, we measured how reliable these new methods are, as well as how they are affected by the steps preceding them. We found that the recent deep learning methods fail to outperform linear methods on current datasets, for most modalities. We further found showed, for epigenetic assays, that the feature engineering steps were more important than the dimension reduction algorithm, in order to obtain good representation of cells. We further attempted to develop a novel algorithm to learn embeddings of epigenomic measurements in an end-to-end fashion, learning at once both the low-dimension representation of the cells, as well as the epigenomic annotation.

# Résumé

Ces dernières années, la transcriptomique et l'épigénomique en cellule unique ont permis aux biologistes d'observer les tissus à une nouvelle résolution. Grâce à ces protocoles, nous sommes maintenant en mesure d'observer l'ensemble de la distribution des états cellulaires dans un tissu, au lieu de simplement leur agrégat. Avec ces nouveaux types de mesures, est apparu le besoin de nouvelles méthodes statistiques pour les analyser. En effet, la génération précédente d'outils d'analyse était conçue pour un régime de peu d'échantillons de haute qualité, alors que ces nouvelles mesures sont beaucoup plus importantes en quantité, mais de qualité nettement inférieure. Ce problème de faible qualité est encore plus prononcé pour les protocoles d'épigénomique en cellule unique, du fait que les cellules ne possèdent que deux copies du génome, par rapport aux centaines de milliers de molécules d'ARN présentes dans la cellule. Le profil transcriptomique et épigénomique des cellules étant mesuré en grande dimension, la communauté scientifique s'est beaucoup intéressée aux méthodes permettant de réduire la dimension des données.

Cette explosion d'intérêt a conduit à de nombreux nouveaux algorithmes et à une communauté florissante de développeurs de méthodes. Leurs travaux n'ont cependant pas encore été adoptés par les bioinformaticiens, soit parce qu'ils n'étaient pas jugés suffisamment fiables, soit parce qu'ils ne répondaient pas correctement aux questions biologiques. Dans cette thèse, nous avons tenté de mesurer la fiabilité de ces nouvelles méthodes, ainsi que la façon dont elles sont affectées par les étapes qui les précèdent. Nous avons en outre tenté de développer un nouvel algorithme pour apprendre des représentations de mesures épigénétiques de bout en bout, apprenant ainsi à la fois la représentation des cellules, ainsi qu'une annotation du génome.

# Acknowledgements

I would like to thank my two advisors, Jean-Philippe (JP) Vert and Céline Vallot for bearing with me for the past 3.5 years, and managing to train me as much as they could towards becoming a scientist. This thesis would not have been possible without their rigor, expertise and patience.
Thanks to Céline, I am now able to attend an epigenetics talk and understand most of the words spoken during a presentation, on good days I am now even able to arrange them in a way that somehow makes sense. Being part of her team also allowed me to see what real biology research looks like, and strive to understand how the data was generated, why it was generated, and how it can be exploited to answer scientific questions, instead of just treating it as a task on which to obtain a higher accuracy than competing scientists.
Thanks to JP, I learned how to go beyond naive applications of already existing solutions, and to instead formulate the problem at hand into a useful abstraction. He also had to bear with an engineer who considered maths as a chore to be feared, and turned him into someone who still fears large equations, but manages to understand them after some effort (even though RKHS remain a mystery to me).
Their joint supervision allowed me to progress on their respective fields of expertise, and created an environment in which I could thrive as a computational biologist, and for that I will always be thankful. They also had to deal with my original lack of organisation, which I hope got better by the end of my PhD, as well as bad sense of humor, which should deserve a prize.

I would also like to thank my former colleagues at Google, who helped on a technical level when my experiments didn't work or when I needed help understanding some papers, and equally helped me on a human level when I needed friends to talk to when my experiments didn't work. I would like to especially thank Felipe, Olivier(s), Laetitia, Marco, Neil, Johan, Nino, Leonard, Albert and Alex for standing me this long. The folks from TGIF also deserve a thank, but there are so many that I would be sure to forget some, so will rather let them recognize themselves.
The colleagues from Curie, were also instrumental in this thesis, I would like to thank Pacome and Melissa in particular for their help with anything Bio-

conductor related.

Arnaud, Chelfi, Sam, Marion, phh, nato, lau, Alexis, JB, Jon, Flo, Victor, and all my friends from Telecom also deserve their fair share of appreciation, without their support and patience when I complained about the state of my experiments, this PhD would not have been possible.

Finally I would like to thanks my family, and Adele who supported me even before this thesis, I am not the easiest person to be around (especially when things don't go well), and your support for all these years means the world to me.

# Contents

# Chapter 1

# Introduction

## 1.1 Motivation

For the past 20 years, new molecular techniques have been developed in genomics to allow biologists and medical practitioners to have a better understanding of human biology and disease. These techniques have been especially useful in the context of cancers, where they shed light on the existence of cancer subtypes based on more relevant criteria than just histological ones or the location of the tumor cells. Thanks to these molecular profiles of tumors, the medical community was able to identify specific characteristics (biomarkers) that were able to segretate patients into subtypes with different prognosis and underlying cancer biology, and to predict of how well the cancer of a specific patient would react to a specific treatment (e.g. in breast cancer, HER-2 positive breast cancer can be treated with the targeted therapy herceptin [1, 2]). In addition to serving as guides for making treatment decisions (which the ASCO produces every year [3]), these biomarkers have also been fundamental for developing better therapeutics. Indeed by understanding the differences specific of a category of cancer, we may gain insights in which mechanisms these cancers use to progress and how they may resist to treatment. A better understanding of the biology of cancer is thus of tremendous importance, both for treating patients today, and for developing the cures for the patients of tomorrow.

Unlike most other diseases, cancer has the specificity that the tumor cells accumulate a large quantity of mutations as they divide. This means that there is not in fact a *single* cancer in a patient, but multiple strains evolving in parallel and responding to selective pressure. This heterogeneity of the tumor cells can be of paramount importance when treating a patient, as the different cells may respond differently to treatment. In particular, the current hypothesis for why some cancers come back after an apparently successful chemotherapy [4], is that the chemotherapy was successful in eliminating most, but not all, of the tumor cells. The small population that survived, called chemo-resistant or chemo-persistent, then proceeds to proliferate leading to a relapse of the cancer. Even worse for the patient, all the new tumor cells will be derived from that population and a new round of chemotherapy will not be successful.

Identifying biomarkers for these chemo-resistant or chemo-persistent cells, is however complicated by the fact that they are generally a small fraction of all the tumor cells. Indeed most of the classical molecular assays rely on sequencing a whole sample at once and observing the aggregate behavior. While this approach can be successful if the characteristics are shared

by all the cells in the sample, they tend to struggle as the population of interest represents a small fraction of the cell population. Indeed, most current variant detection algorithm tend to struggle if a mutation is present in less than 5% of the sequenced DNA [5]. Furthermore, the characteristics that make a cell population resistant are not always genetic, and thus need to be measured with either gene expression (transcriptomic) or gene regulation (epigenetic) protocols. These protocols show a more refined view of the cells, and their measurements tend to be even more heterogeneous than the ones obtained by regular genetic measurement. Indeed, while most healthy cells in a tissue share the same genome, their transcriptome or epigenome vary across cell type and cell state, and this heterogeneity can be even more pronounced for diseased tissues such as cancer.

In the past 10 years, new protocols measuring molecular profiles of individual cells in a tissue have emerged. Thanks to these protocols the heterogeneity of complex tissues, such as tumors, can be studied at a higher resolution. These new protocols generate measurements for a few thousands, up to hundreds of thousands, of cells but are much more noisy than the previous protocols. This noise is due both to cost reasons, as well as to the small amount of biological material contained in a cell (e.g. just 2 copies of DNA per cell instead of thousands in a sample).

The data generated by these protocols contains thousands of cells measured noisily, most of which have previously never been identified due to their rarity. This creates challenges for analysing such data since the previous algorithms were designed for a regime of just a few dozens or hundreds of samples, measured at a much higher quality. Furthermore, since the goal of these experience is generally to discover new cell populations, whether diseased or healthy, interpreting whether these algorithms manage to correctly identify previously unknown cell populations or are incorrectly seeing patterns where there is nothing, is generally very tedious or impossible without further experiments.

In this thesis, we work on understanding how to make these algorithms more robust, so that when a biologist designs a new sequencing experiment they can be confident that the results that they obtained are likely to be correct. To give the reader the background necessary to understand the original contributions presented in the following chapters, this introduction provides a quick survey of existing technologies to map the transcriptome and epigenome at the single cell resolution (Section 1.2), followed by an overview of the main questions and challenges that arise when it comes to analyzing the resulting data (Section 1.3). We then review the main analytical tools that exist to map the raw experimental data to a representation of the biological state of each cell (Section 1.4), and conclude this introduction by summarizing our contributions in Section 1.5.

## 1.2 Mapping the transcriptome and epigenome at the single cell resolution

### 1.2.1 General principle

Molecular assays, such as RNA-seq, have been used extensively in the past 30 years to probe tissue and cell biology. Their protocol usually follows a similar set of steps, namely, first acquire

cells from a tissue, target the molecular characteristic of interest (e.g. transcription factor binding, histone modification or gene expression), generate short *DNA reads* corresponding to this molecular information, and finally sequence them.

This framework has been used to design a broad range of assays such as Hi-C [6], ATAC-seq [7, 8], CUT&Tag [9], ChIP-seq [10, 11], or RNA-seq [12]. While these protocols provide a detailed molecular profile of a tissue, often refered to as a "bulk" profile, it is important to understand that during the sequencing step, the reads of all the cells are pooled together. This means that final result is the sum of the reads of all the cells in the sample. For homogeneous samples, such as cell lines, this is perfectly acceptable, it can even be useful for some heterogeneous tissues such as cancer in the case of TCGA [13]. However, when studying tissues where the diversity of cells is the object of interest, these protocols tend to fall short. These protocols also tend to be relatively limited when trying to identify new cell types for which we have, by construction, no markers for isolating them. What's more, in the case of early embryo or tumor development, the diversity of cells contained in the tissue is so large that just observing the aggregate reads is not useful. This problem can be partly solved in some instances by experimentally isolating cells of interest, but this step can be very tedious, requires a lot of samples, a lot of time, and is not always applicable.

In order to solve this technical issue, a lot of research and engineering has been done in the past decade to improve the protocols. A review of the history of single-cell transcriptomics can be found in [14, 15].

The general principle underlying all the new protocols consists of adding a cell specific "barcode" at the end of the reads generated in a cell. Having that barcode allows reads to be sequenced together, since cell of origin can be obtained from the barcode. This requires generating the reads, and adding the barcode, in each cell separately which has been done differently in the protocols as technology advanced.

Currently, the two leading strategies for isolating the cells and running the reaction are: *(i)* isolating the cells in micro/pico wells which is used by protocols such as SMART-seq [16, 17, 18], *(ii)* using microfluidics systems to isolate the cells in water droplets, which was pioneered in The Fluidigm C1 [19] and has been further developed by InDrop [20] and DropSeq [21].

### 1.2.2 Brief overview of single cell RNA approaches, throughput versus coverage

Single-cell protocols first appeared for the study of gene expression, by extending RNA-seq. While the measurement of the transcriptome for just a few cells has been around for a few decades [22], [23] was one of the first to adapt the process for next generation sequencing platforms. The Linnarson lab then expended on their work with [24], this work was relevant as its stated goal was not to measure just a few cells, but to lay the ground work for sequencing large amounts of cells. The cell isolation was later improved by using robotics [25] and an early microfluidic system [19].

The field drastically changed with the introduction of massively parallel high throughput microfluidic droplet based systems such as InDrop [20] and DropSeq [21]. With these methods it became possible to sequence the transcriptome of thousands of cells at a time instead of just single plate by single plate. The emergence of easily accessible and usable commercial machines,

such as the ones developed by 10X, helped the technology penetrate many labs. The ability to automatically sequence large numbers of cells also opened the door for discovering rare cell types, or observing with better granularity the various steps involved in cell type differentiation.

While microfluidic approaches helped popularize the single-cell technology, they suffer from not always being able to capture all the mRNA species present in a cell. In parallel to the very high throughput methods, plate based technologies also improved with better automation and optimisation of molecular biology steps. The SMART-seq protocols [16, 17, 18] has been continuously improving in the last 5 years, and can now catch most of the mRNA species present in cells. This increased coverage, both in species of mRNA and in amount of mRNA caught per cell, can prove useful for having higher quality measurements of the cells in a sample, which could potentially allow to discover cell states that would not be otherwise distinguishable. This effort in increasing the coverage has also recently been pushed to sequence even non poly adenylated RNA in [26], thus opening the door for also measuring parts of the non coding transcriptome.

Whether to use high throughput or high coverage technologies is still up for debate in the community, it is currently unclear if one technology will become the standard or if they will co-exist and be used for different purposes. High throughput technologies are currently the ones being used in the various atlas efforts [27, 28, 29], as their stated goal is to map the whole variety of cells contained in an organism (either mice, fly or human). This choice between the two approaches will also be affected by the computational methods used in downstream applications. Indeed, if they succeed in leveraging a large amount of cells to correct the lower quality of each individual cell, they could potentially computationally correct the technical issues present in high throughput technologies. This is one of the reasons why deep learning based methods are currently seeing a surge of interest for analysis, as they should be the ones best positioned to use large quantities of cells. However if these methods fail in that task, it would be reasonable to assume that higher quality samples would become the standard in the end. Rigorous studies of the impact of the quality of the cells versus their quantity, is to the best of our knowledge currently not present to guide us in that choice, this is further complicated by the rapid development of both sequencing technologies and computational analysis methods.

Despite the many open questions in how best to conduct scRNA-seq experiments and analysis, it is a relatively mature field and has become a routine, and often necessary, step in many labs.

### 1.2.3 Emergence of single cell epigenome methods

The transcriptome can be considered a view of the phenotype of a cell, as the measured mRNA will be translated into proteins. This view is however limited, and does not allow measuring all the regulatory events that can drive cell fate. Indeed, it is recently becoming accepted in the community that cancers are not just defined by their gene expression, but that epigenetic events play a large role in the acquisition of cancer hallmark capabilities [30], tumor progression [31], resistance to therapy [32], and metastatis [33].

The single-cell study of the epigenome became popular with the advent of the scATAC-seq protocol [7, 34, 35], in particular [7, 34] allowed using the commercially available droplet machines to easily move to the single-cell level. This protocol allows observing which regions

of the DNA are accessible and slowly became the standard protocol for this task. It benefited from the fact that the experiments are relatively easy to perform [1], and has already led to a better understanding of the biology of triple negative breast cancer (TNBC) [36, 37]. Just like scRNA-seq, this protocol has also been included in commercially available kits sold by 10x, making the adoption of that technology easier for many labs.

The study of DNA methylation at the single-cell resolution has also been made possible by the scRRBS protocols [38, 39], which also led to better understanding of the role of methylation in cancer [40, 41]. This protocol is also interesting because it was the first one to leverage deep learning methods for its analysis with the DeepCpG algorithm of inference [42].

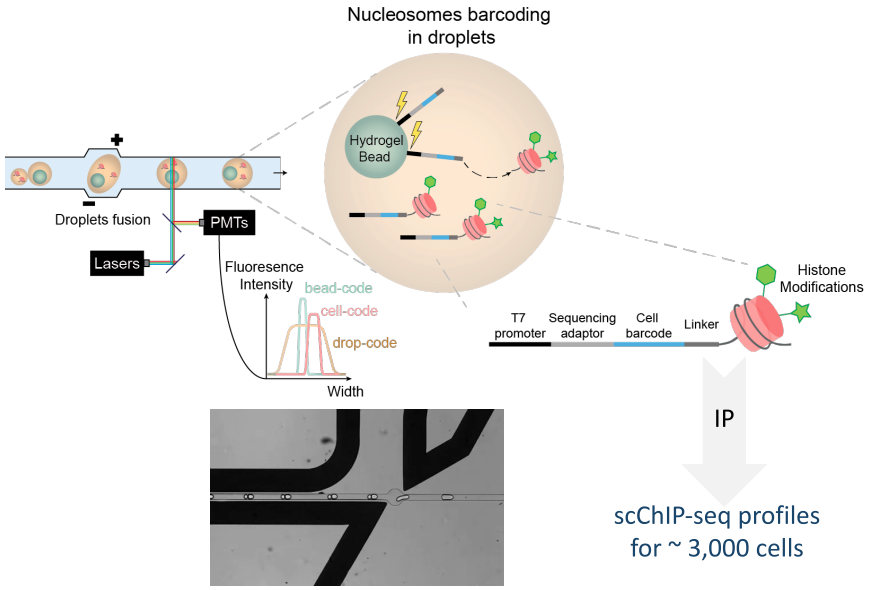


Figure 1.1: Reads generation in droplets for scChIP. The nucleosomes are bound to a hydrogel bead containing the cell barcode before being bound by the antibodies. Selection of nucleosomes with the targeted histone modification is done during the immunoprecipitation.

Histone modifications also play a very important role in gene regulation. Indeed, genome accessibility and structure is determined by these modifications, they also serve a role in either repressing or enhancing gene expression. The role of histone modifications in cancer has been shown in glioblastomas [43], cancer cell lines [44], and TNBC [45]. The measure of histone modifications at the single cell resolution was first introduced in [46], by adapting the ChIP-seq technology to single-cell, this work was further improved on by [47] and [48]. These technologies are all based on some variation of the ChIP-seq protocol, there are however other technologies for measuring histone modification based on the CUT&Tag technology, giving rise to the sc-CUT&Tag protocol developed in [9, 49]. These scCUT&Tag technologies are currently gaining traction as they are believed by some to have measurements of higher quality, this higher quality has however not been rigorously proven to the best of our knowledge.

[1] "It's so easy that even an undergrad can do them" according to Dr. Buenrostro on episode 53 of the "Epigenetics podcast" by Active Motif

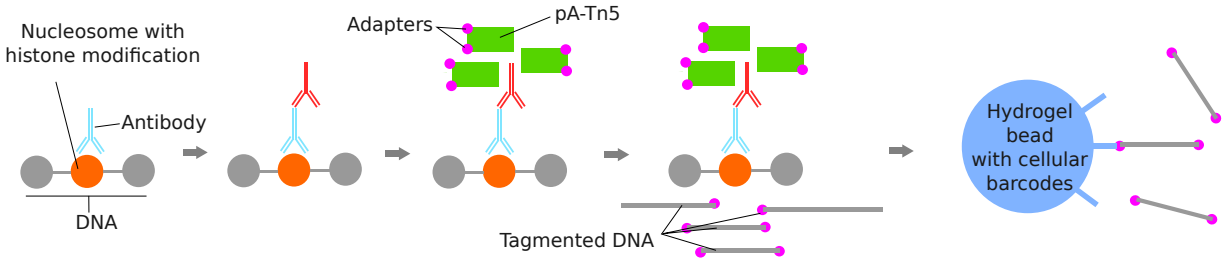


Figure 1.2: Reads generation in droplets for scCUT&Tag. The histone mark of interest is on the orange histone, which a specific (light blue) antibody will bind to. A second antibody (red), will then bind to the first antibody, and recruit pA-Tn5 transposomes (green) with adapters (pink). Once activated, the transposomes will generate tagmented DNA reads (grey) with the adapters, these reads will then be bound to the gel bead (blue) containg the cell specific barcode.
All these steps are happening in a droplet, thus allowing to barcode the reads in a cell specific fasion.

ChIP-seq and CUT&Tag both use antibodies targeting a specific histone modification, but differ in how the reads are generated.

- In the ChIP-seq protocol, the genome is first sheared into one to three nucleosome long parts, using either an enzyme or sonication. These small DNA segments are then introduced to antibodies targeting a specific histone modification forming DNA-antibody complexes, this step is called the immunoprecipitation (IP). These complexes are then isolated, the antibody removed, and the DNA segments left are sequenced. This means that in this protocols, reads are first generated by shearing the DNA, and are then filtered by using the antibodies. The single-cell version of this protocol adds the cell barcode after the shearing of DNA, the IP step is done on all the reads at once, as shown in Fig 1.1.

- In the CUT&Tag protocol, antibodies specific for the histone modification are introduced first, without shearing the DNA. A second special antibody is then introduced, it is specific for the heavy chain of the first one, this second antibody is modified to enhance pA-Tn5 transposome at antibody-bound sites. The transposome is then activated by introducing $Mg^{++}$, and will generate reads with a specific adapter at bound sites. These reads can then be purified with PCR, and sequenced afterwards. This means that in this protocols, all the reads produced are supposed to be from a location with the targeted histone modification, requiring no filtering step. The single-cell version of this protocol [49] uses a modification of the scATAC-seq 10x kit, where the barcodes are added just after the read generation, by binding to a bead with barcode adapters as shown in Fig 1.2.

Single-cell epigenetic protocols however suffer from an insurmountable drawback compared to scRNA-seq: there are only two copies of DNA in each cell compared to the hundreds of thousands of RNA molecules. This means that if we only manage to generate reads for 10% of the RNA molecules, we can still have a good estimations of the distribution of the RNA species in the cell, this is however untrue for epigenetic assays. Indeed if we fail to measure at a location in the two copies of DNA, we do not have access to the epigenetic state at that

location. This makes these modalities much more dependant on good analysis pipelines to infer the unobserved states.

## 1.3 Key challenges of single cell data analysis

By using single-cell sequencing protocols, scientists have been able to measure the states of thousands up to millions of cells [50] when we could only previously rely on just a few dozens or hundreds of measurements. This has opened new avenues of research as questions that were previously extremely hard to answer (e.g. cell differentiation fate) have now become common practice. Indeed, when we used to only have access to a summary statistics about the cells states contained in a sample, we now have access to the full distribution of these states.

While this increase in the amount and richness of data being generated in a single experiment has helped biology move forward, computational methods for analysis were not developed for these regimes. Indeed in previous experiments, measurements were few and of very high quality (millions of sequencing reads per samples), now these measurements are of low quality ($\sim$1.000-10.000s of reads per cell) but in drastically higher number. Furthermore, due to the low sequencing depth on each cell compared to their bulk counterpart, there is a much higher cell-cell variability between cells of the same type than there is between similar bulk measurements,

With the access to the full distribution of cells, scientists were able to discover new cell types that were previously lost in the aggregate, build better hierarchies of cell types, observe cell development and differentiation, as well as identify new biomarkers between these cells.

Being able to sequence large numbers of cells at the same time has also been used to run multiple examples in parallel instead of studying a single tissue. For example, studying the effect of different drugs on different cell types for a fraction of the cost with protocols such as Perturb-Seq [51] or databases such as the Connectivity Map [52].

All these innovations have created a lot of opportunities for asking, and answering, many scientific questions, but at the same time they have introduced many new tasks for which computational methods have to be updated or created.

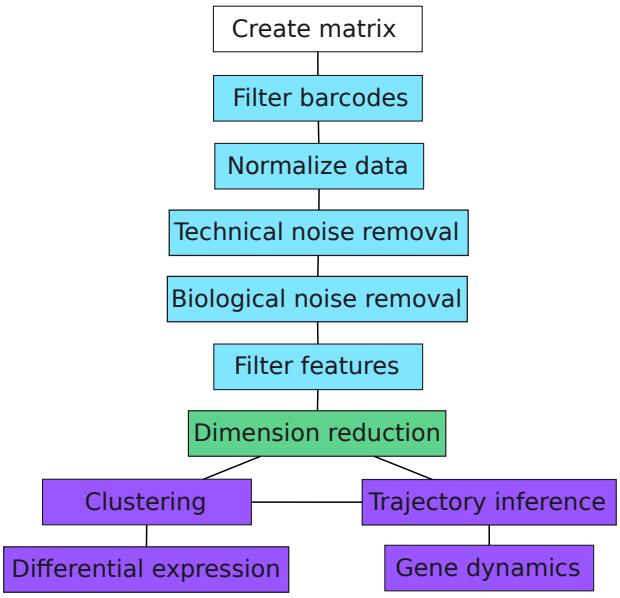


Figure 1.3: Organisation of the different steps in a standard single-cell analysis pipeline.

Standard computational analysis pipelines on a single modality use the following steps, as can be seen in [53, 54]. *(i)* filter barcodes, *(ii)* normalize the data, *(iii)* correct for technical noise, *(iv)* correct for biological noise, *(v)* filter relevant features, *(vi)* dimension reduction, *(vii)* clustering algorithm, *(viii)* differential expression analysis, *(ix)* trajectory inference, and *(x)* study gene dynamics. **These steps are presented here using the vocabulary of scRNA-seq in humans as it is currently the most commonly used protocol**, execept when specified otherwise.

For scRNA-seq we can easily represent each cell as a vector in $\sim$25.000 dimensions, one for each human gene, getting one vector per cell we thus obtain a matrix representing the data. The construction of that matrix for single-cell epigenetic protocols is explained in more details in section 1.4.2 and is the subject of Chapter 4.

*(i) filter barcodes*: sequencing protocols aim to add a unique barcode to each cell so that when sequencing all the reads at once, they can be demultiplexed. However it can happen that a well or a droplet does not contain a cell, the associated barcode will thus measure the background material and will not correspond to an actual cell. Two or more cells can also end up in the same droplet or well, the associated barcode will thus not measure an actual cell either. Furthermore cells can also have died (either because of the experiment or naturally), and will not behave like a normal functioning cell, thus the measurement will not be representative of the cells in the sample. All the barcodes associated with these problems need to be filtered out of the matrix, this can be done by looking at the number of genes expressed (very low or very high corresponding to either no cells or multiple cells), the number of reads sequenced (*ibid*), or the proportion of mitochondrial RNA measured (high is common for dead cells).

*(ii) normalize the data*: the number of sequenced reads is generally different for each barcode, this difference can be either technical (the reaction generated more reads in a specific bioreactor) or biological (a cell is more transcriptionaly active, or larger). In order to compare cells it is common to normalize them, in order to study the relative enrichment of particular genes, the

expression levels can thus be normalized in order to facilitate this analysis. This is usually done by counts per million (CPM) normalization, where all the features are multiplied by a common factor so that they sum to a fixed number. Afterwards the counts can also be log transformed by applying $\log(1+x)$ transformation. Both of the previous step, while common, are not used across all analysis pipelines, and are not used for all modalities. For epigenetic assays, using TF-IDF to normalize the data is also a common step.

*(iii) correct for technical noise*: due to the low sequencing depth per cell, some genes may be measured as having zero expression (or the equivalent for different protocols) despite actually being expressed in the cell. These technical zeroes are hard to distinguish from real biological zeroes, where the gene is actually not expressed, and are generally called dropout. Another source of technical noise is *"batch effect"*, where similar cells will have different measured expression due to the library preparation being done in different conditions or a different experiment. In particular, this *"batch effect"* is very common for plate-based protocols, where we can observe differences between plates even if they measure the same cell states. These systemic effects lead to spurious differential expression, or overclustering and generally needs to be corrected for. This effect also appears when trying to analyse experiments done in different conditions, *e.g.* diseased vs healthy, if the samples were acquired or sequenced in different experimental conditions (different day, different experimentalist, different hospital, etc …).

*(iv) correct for biological noise*: biological noise is noise that is not due to the measurement technology, but is actually present in the cells. A common example is the cell cycle that cells can be undergoing. This type of noise can introduce deferentially expressed genes between cells of the same type, which can pollute the downstream analysis. However, unlike technical noise, this noise is not always corrected for as it can be relevant for some experiments.

*(v) filter relevant features*: scRNA-seq can measure up to 25.000 different genes in humans, and epigenetic protocols can have up to 3.000.000 features, most experiments however usually have in the order of 10.000 cells. In order to both speed up the analysis tools, as well as removing genes believed not to be relevant for the experiment, it is standard to filter out a large amount of features. This is generally done by removing the ones that vary little across cells, or the ones with very low counts.

*(vi) dimension reduction*: even after feature selection, there are still a lot of features, an issue with high dimensional data is that the euclidean distances between the points tend to all be extremely similar: this is called the *curse of dimensionality*. This curse of dimensionality make all downstream applications that rely on a measure of distance or similarity between cells, such as clustering or trajectory inference, perform very poorly. In order to be able to run these algorithms properly we need to reduce the number of features even further using methods of *dimension reduction*. The various methods used in single-cell are presented in more details in Section 1.4. Besides their computational value, dimension reductions also allow to summarize the very high number of features measured in each cell into a few statistics, and the way in which this summary is computed can be of biological interest. For example PCA, a very popular dimension reduction, uses a linear combination of the features to create the reduced dimensions, these combinations can be interpreted as sets of coregulated expression programs. Reducing the data to 2D or 3D can also allow for visualization of the distribution of cells, which can be very useful for exploration.

*(vii) clustering algorithm*: since the reads of cells have to all be merged together during the

sequencing step, we do not know their cell type identity, nor do we know which hierarchy of cell types are in the sample. In order to identify the different cell types contained in the sample, we have to identify the different groups of cells contained in the sample with computational methods. Since the previous step allows us to get rid of the curse of dimensionality, we can run standard clustering algorithms to group the cells by similarity. By using algorithms with hierarchical structure (e.g. hierarchical clustering with Ward's linkage), or by iteratively reclustering the data, we can build such a tree structure and identify potential new cell types as has been done in [21]. In a more general fashion this step allows to identify the different groups of cells contained in a sample. In the case of cancer samples, this can be used to identify different groups of tumor cells.

*(viii) differential expression analysis*: identifying the groups of similar cells in a sample, while useful, does inform us on the specific types of cells contained in the sample, nor on how they differ. For scRNA-seq, this allows us to identify which genes are more or less expressed in each cluster. These deferentially expressed genes can be used to characterize the clusters, and can also be used in conjunction with external databases such as the Gene Expression Ontology to identify the cell type of the cluster. This step can be used in the case of cancer samples, where we would like to understand which genes are upregulated in chemo-resistant cells. Alternatively this can also be used to identify if the tumor cells correspond to a known cancer type for which a treatment is known.

*(ix) trajectory inference*: the two previous steps assume that cells are in distinct groups, however cells can also differentiate into a different cell type. This happens both in healthy cells, where a stem cell will differentiate into a regular cell (after asymmetric division), or in the case of cancer development. Understanding the trajectory that cells follow while undergoing differentiation, as well as the intermediary states that they go through is both of scientific interest, as well as of practical interest. Indeed if we can identify the steps by which cells enter diseased states, we can design interventions in order to stop them following that path.

*(x) gene dynamics*: trajectory inference allows us to obtain an ordering of the cell states, called pseudotime, it however does not inform us on how gene expression changes over that pseudotime. Running algorithms such as TradeSeq [55] estimates how genes are expressed as a function of pseudotime for each trajectory.

We can observe that steps *(i)-(v)* serve the purpose of cleaning up the data, that step *(vi)* builds a low dimension summary of the data, and that steps *(vii)-(x)* use this representation in order to extract biological knowledge from the experience. Since the last steps rely on that representation, a poor representation will lead to poor conclusions, and a great representation can shine line on unknown cell types and lead to new discoveries.

Note that most software packages perform multiple of these steps, and that they are not always as cleanly divided as how we have presented them (*e.g.* scVI [56] computes the size factor correction in the same step as it builds the dimension reduction). This division of the steps however provides a useful abstraction for understanding the analysis pipeline.

This importance of representations is not exclusive to single-cell biology, but is shared by most modern analysis software. A recent example is the focus on "foundational models" [57] in the field of natural language processing [58], where a lot of effort is spend on building a tool that can build useful representations, and where all downstream tasks (e.g. translation,

sentiment analysis, summarization, etc …) benefit from this model.

## 1.4 Building useful representations of cells

In the previous section we introduced the different steps commonly used in single-cell data analysis, and showed that the step of building a representation of cells is a crucial one. In this section we will delve into how these representation are commonly obtained for single-cell data.

### 1.4.1 General principles for dimension reduction

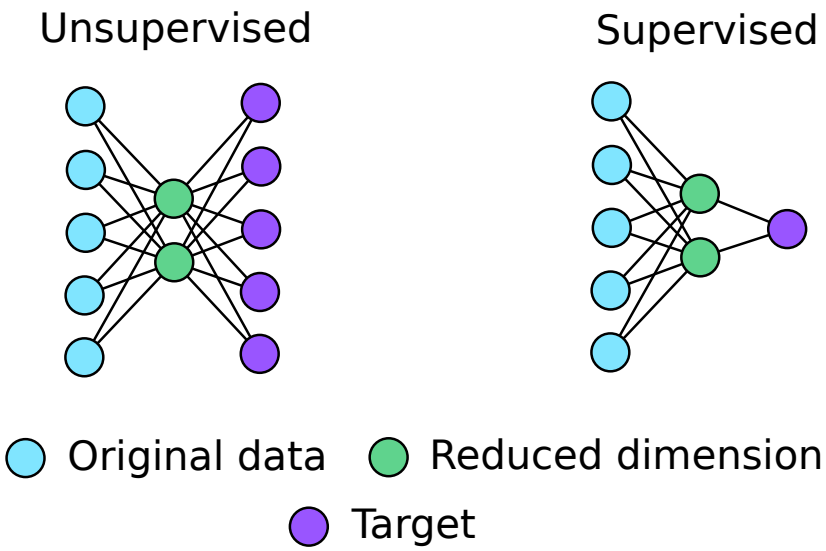


Figure 1.4: Representation of the difference between supervised and unsupervised learning for obtaining embeddings of the data.

While a complete introduction on machine learning is beyond the scope of this manuscript, we will briefly introduce the differences between supervised machine learning and unsupervised machine learning, as well as the problems raised by the latter. More complete texts on these concepts can be found in [59, 60, 61, 62].

Machine learning, and statistical learning, techniques can broadly be divided into two categories: supervised learning, and unsupervided learning.

In the case of supervised learning, each datapoint is mapped to a corresponding categorical or scalar value (which can be multidimensional or a combination of both), the goal is to learn how to estimate this value from the data. This is done by approximating this mapping with a function of the input data, generally taken from a restricted family of functions (*e.g.* linear functions or neural networks), the specific function being defined by a set of parameters. In order to select the parameters specifying that function, we define another function, called *objective function* to be minimized, such as the mean squared error between the predictions of the function and the actual values, and select the parameters that minimize that objective function.

In the case of single-cell data, supervised learning can be used to learn how to predict cell types (reviewed in [63]), whether a cell responds to treatment, or to predict how an intervention will change gene expression or morphology.

Once the function has been learned, it can be used for predicting the same variable on new data for which a mapping is not provided, the original data on which the model has been trained being called *training data*. The actual parameters of the function can also be of interest, for example one could be interested in how a change in the input data (*e.g.* expression of a gene) will affect the prediction (*e.g.* response to treatment).

One advantage of supervised learning is that obtaining a function that makes good prediction is generally the end goal. It is thus easy to evaluate which algorithm is the best by comparing how good their predictions are. Furthermore since the mapping could be generated for the data used to learn the function, it means that there is a procedure (which may be long and costly) to generate such a mapping, the performances of the algorithms can thus be evaluated on new data if needed.

While supervised learning in itself does not directly provide a lower dimension representation of the original data, some families of functions of functions can be used for that purpose. For example standard neural networks, such as multi layer perceptrons (MLP), compute the mapping in multiple stages, the stage just before the actual prediction is generally used as representation for the data.

Alternatively, when there are no mapping associated with the data, the objective function has to be a function of the data. For our purpose we can broadly separate the unsupervised methods into two categories: *generative models* and non generative models.

In the case of a generative model, one makes the assumption that the *observed variables* (*e.g.* measured gene expression) are generated from a stochastic process based on some *hidden variables* that are not measured (*e.g.* unknown cell type) and some parameters that are unknown. One then has to specify the probability of observing a set of observed variables, given some hidden variable and some parameters. The *likelihood* function is the probability of observing the data, viewed as a function of the parameters. The goal is then to find the associated hidden variable for each sample, as well as the parameters of the stochastic process, which maximize the likelihood of observing the data (called *evidence* in that context). If computing this evidence is not tractable (which is generally the case), one can optimize a tractable approximation of that objective function, such as the *evidence lower bound* (ELBO).

A naive example of a generative model for gene expression would be the following:

- **observed variable**: gene expression of the cells in $\mathbb{R}^{20.000}$.
- **hidden variable**: discrete cell type a categorical variable with $K$ values, and a uniform prior over these categories.
- **stochastic process and parameters**: given a cell type $k \in 1 \dots K$, the gene expression is sampled according to a multivariate normal distribution with a diagonal covariance matrix, parameterized by its $\mu_k \in \mathbb{R}^{20.000}$ and variance $\sigma_k \in \mathbb{R}_{+*}^{20.000}$

This models the data as a mixture of gaussians, and the objective function is classically the ELBO which could be optimized with the *expectation maximization* (EM) algorithm. In this

case the hidden variable is only one dimensional (cell type), but the hidden variable can be multidimensional and continuous, as in the case of the Variational Autoencoders (VAE) such as scVI [56], peakVI [64], and SCALE [65] or in the case of Latent Dirichelet Allocation used in cisTopic [66].

Specifying a generative model and its associated stochastic process is however not necessary. The dimension reduction can be done using an encoder function that maps the data to a lower dimension, and a decoder function that maps this reduced space back to the original data. By applying the encoder followed by the decoder one compresses and then decompresses the data. Given a function that computes the error between the original data and this process, called the *reconstruction error*, one can jointly optimize the encoder and decoder such that this error is minimized.

By using a linear functions for the encoder and decoder, and the mean squared error for the reconstruction error we obtain a formulation the classical algorithm Principal Component Analysis used in Seurat [67] and scran [54]. If we use MLPs for the encoder and decoder, and the mean squared error for the reconstruction error, we obtain a standard autoencoder which is used in DCA [68][2].

While unsupervised approaches have the benefit of not relying on a mapping to build the dimension reduction (also called *embedding*), its objective function is generally not the object of interest. Indeed, having a model with a low reconstruction error or ELBO provides no guarantees that the embedding obtained by this model will have any biological relevance or practical use. Furthermore, not only is a good value on the objective function no guarantee that a model is superior to another, but also not all unsupervised method even optimize for the same objective function: how should one then compare the reconstruction error of PCA against the ELBO of a VAE ?

The recent increase in the use of deep learning based methods [69, 70] further increases that problem. Indeed, while classical models such as PCA are fairly easy to train and do not require the user to specify a lot of *hyperparameters*[3], their deep learning based counterparts have a lot of hyperparameters to select. These hyperparameters can have a large influence on the biological explanatory power of the models as we show in Chapter 3 and Chapter 4, and selecting the correct value (or at least a non suboptimal one) is not trivial. Indeed not only is a better value of the objective function not a guarantee of better biological performances, some hyperparamter values tend to naturally artificially increase the objective function (*e.g.* using PCA with a number of dimensions equal to the original dimension just applies the identify which has zero reconstruction error despite being useless as a dimension reduction).

Some attempts have been made at identifying heuristics for comparing between models such as the *silhouette score*, but they have not been shown to be successful to the best of our knowledge. The new multiomics protocols[4] may offer new heuristics from the field of contrastive learning to solve this problems such as the *neighborhood score* used in Chapter 4, in [64, 72], or the the new community benchmark openproblems.bio.

[2]DCA uses by default a negative binomial reconstruction error, but can be run with the mean squared error.

[3]Hyperparameters are parameters about the model which are generally not learned. This can be the number of dimensions for PCA, the number of hidden layers in a VAE, the type of objective function, etc…

[4]Protocols measuring multiple modalities on each cell, such as RNA and ATAC or RNA and CUT&Tag [71]

As we saw in Section 1.3, the standard analysis pipeline for single-cell data analysis heavily relies on unsupervised methods, both for the dimension reduction and for its clustering step. This reliance on methods that are very hard to evaluate for two of its most crucial steps introduces liabilities for the whole analysis and the interpretations of the experiment. Indeed if the putative cell types obtained from the clustering step are incorrect either by *overclustering*[5] or *underclustering*[6], we end up looking for biomarkers between incorrect groups and thus either missing important biological insights or coming up with incorrect conclusions.

### 1.4.2 The case of single-cell epigenome: how to build the count matrix ?

Unlike transcriptomic assays where building a vector representation of a cell from the reads is natural, by having a feature for every gene, epigenetic assays can have reads at every location of the genome. Since the human genome contains about 3 billion base pairs and 20 thousand genes, this means that we cannot naively have a feature for every position. Furthermore, while the scientific community has a good idea of which parts of the genome are genes, there is no consensus on which parts of the genome are important for epigenetic assays, as these regions change depending on the assays, change depending on the tissue being studied, and that identifying these regions is often the goal of the experiment.

In the context of bulk epigenetic assays three main different strategies have been adopted for turning this distribution of reads on the genome into a vector representation:

- Discretize the genome into a set of (usually non overlapping) fixed sized windows covering the whole genome, called *bins*, and count how many reads are sequenced into each window. This has been used successfully in [73, 74].

- Use a predefined set of windows, not necessarily of the same size nor covering the whole genome, and count how many reads are sequenced into each window. Such an annotation can be a set of already identified enhancers, promoters, trascription start sites or genes.

- Identify computationally which regions of the genome are locally highly enriched in reads, and count how many reads are sequenced in each of these regions. The identification of these regions is done with a tool called a *peak caller* such as MACS2 [75].

These 3 approaches share the fact that they start by defining which regions of the genome will be kept, and then reads being sequenced in these regions are counted to build a vector representation of the data.

The third approach, identifying highly enriched regions, is the one currently being recommended in ENCODE4 ChIP-seq best practices. In particular since peak callers are just a heuristic and can have false positives, it is common to use multiple peak callers (SPP [76],

[5]Overclustering: creating too many clusters of cells, thus separating cells of the same cell type into different groups.

[6]Underclustering: creating not enough clusters, thus merging together cells of different cell types

PeakSeq [77], and GEM [78] for ENCODE3) and only keep the regions identified by multiple ones using either some ad hoc overlapping criterion or the ireproducible discovery rate [79] (used in ENCODE3).

One reason why peak callers are so successful in the case of case of bulk analysis is that the number of reads sequenced per sample is recommended to be above 20 millions, thus limiting the impact of sequencing noise. The sequencing noise can further be limited by being measured directly in some protocols, ChIP-seq and CUT&Tag in particular, thus if a region is enriched at similar levels in the experiment and in the measured background noise it can be removed from the analysis.

Single-cell epigenetic assays do not currently measure this background noise, what's more the standard number of reads per cell is generally in the order of a few thousand (for high quality experiments), rather than tens of millions, even worse there are only two copies of DNA per cell (at least in mouse and humans) instead of the millions that are available in a bulk sample. These issues make using a peak caller on each cell separately obviously impossible.

In order to use peak callers on single-cell data, we can aggregate all the reads together to build a *pseudo-bulk* on which peaks caller can be used. This suffers from two drawbacks: there are still no measurements to estimate the background noise, and the sample contains a mixture of different cell types. While the lack of background noise estimation can lead to calling regions which are not actually enriched, we can still identify the relevant peaks, we just end up with more noise in the data. However, since epigenetic modification can be cell type specific, some regions that are specific to rare cell types may not contain enough reads to appear as enriched in the pseudo-bulk, these regions will thus not be used as features and we may not be able to identify these cells later in the analysis. Because of this problem, peak callers on the pseudo-bulk are rarely used in recent analysis pipelines.

This only leaves the first two options: bins or annotations. The annotation approach obviously suffers from the fact that by specifying the regions of interest before analyzing the data, we cannot discover new regulatory regions and is thus not commonly used in analysis pipelines.

By elimination, we are left with the bins solution for matrix construction, but then comes the question of which size of bins to use. Where for bulk data the choice of bin size is generally limited by the computer hardware, there are actual data limitations for single-cell. Indeed bulk analysis has shown that the size of the enriched regions can be as small as 5.000 base pairs (5kbp) for some histone modification marks (H3K4me1 or H3K4me3) or as large as 200kbp (H3K27me3). By selecting too small a bin size, we can end up splitting a single functional region into multiple features and introduce noise in the analysis, as well has having a lot of bins with very few reads and introduce a lot of technical zeroes. Alternatively by selecting too large a bin size, we can end up merging regions together which have different functional role and missing on important biological insight.

This step of matrix construction is often not treated at all in the recently published methods for single-cell epigenetic data analysis, with the methods assuming that they are given a count matrix already built. This leads to a new hyperparameter that is often ignored, while its influence on downstream performances has not been characterized.

Understanding the role of this matrix construction, and how to make it reliable is the topic of Chapter 4.

## 1.5 My contributions

In the previous sections we have highlighted two problems that we believe to be important for ensuring that single-cell analysis pipelines can reliable discover relevant biology from the experimental data: the role of hyperparameters for scRNA-seq dimension reduction, and the role of matrix construction for epigenetic assays.

In my doctoral studies I attempted to make advances on these problems and this led to the following works:

- While [80] was the first to highlight the role of hyperparameters for VAEs, the authors only studied one class of algorithms, and this was done only on synthetic data. We decided to extend their work to the 7 most popular dimension reduction package, as well as using real experimental data of varying complexity to have a more complete picture of the role of hyperparameters. We also studied different heuristics for selecting hyperparameters, and how they performed for clustering cell types. This led to a first author publication in *Genome Biology* in 2020 [81], a first author publication at the Learning Meaningful Representation for Life workshop at NeurIPS in 2019, and is presented in Chapter 3
- We studied the role of matrix construction for the analysis of single-cell histone post translation modifications, as well as benchmarked which algorithms provided the best dimension reduction for that protocol. Since measurement of histone post translation modification at the single-cell level is relatively recent [46, 47, 71] there was no consensus on which algorithms to use. This led to a first author manuscript [82], and is presented in Chapter 4.
- We completed a review of the advances in machine learning for single-cell biology. This led to a co-first author publication in *Current Opinion is System Biology* in 2021 [69] and is presented in Chapter 2.
- I have also worked on designing an algorithm for jointly identifying the relevant regions of the genome and computing embeddings for single-cell epigenetic data, thus completely bypassing the need for selecting a matrix construction algorithm. While this project shows some promising early results on simulated data, it is still a work in progress and is presented in Chapter 5

# Chapter 2

# Machine learning for single cell genomics data analysis

## 2.1 Introduction

With single-cell omics technologies getting wide-spread adoption, computational methods are urgently needed to process the large amounts of data they produce [83]. Machine learning (ML) approaches have recently demonstrated their fantastic potential to automatically process and learn from large amounts of high-dimensional data in fields such as computer vision or natural language processing [84]. They are therefore seen by many as a promising way to infer biological knowledge and develop predictive models from single-cell omics data, which provide high-dimensional characterization of large quantities of cells. Not surprisingly, the development of ML approaches to analyze single-cell omics data has been a very active field of research recently.

In this review we survey recent advances in ML approaches developed to analyze single-cell transcriptomic and epigenomic data, focusing mainly on articles published in the last two years (2019-2020). This period witnessed active developments of new methods, in particular based on deep learning, to automatically extract information from large sets of single-cell data, tackling important problems such as batch normalization, multimodal data integration, automatic cell type classification, trajectory inference or gene network reconstruction. It is also a period where systematic benchmarks started to highlight the practical challenges associated to these methods, as well as their potential. With this review we hope to give the reader enough entry points to that fast-moving literature in order to grasp the current state-of-the-art and join its future developments.

## 2.2 From raw data to useful representations

Raw single-cell transcriptomic count data, as well as their epigenomic counterparts, provide a high-dimensional and noisy description of each cell by assessing the activity of thousands of genes or DNA loci simultaneously. Transforming raw count data to a lower-dimensional representation of each cell using dimension reduction (DR) technique is a useful step to re-

move technical noise and prepare data for visualization, classification or further analysis tasks (Figure 2.1).

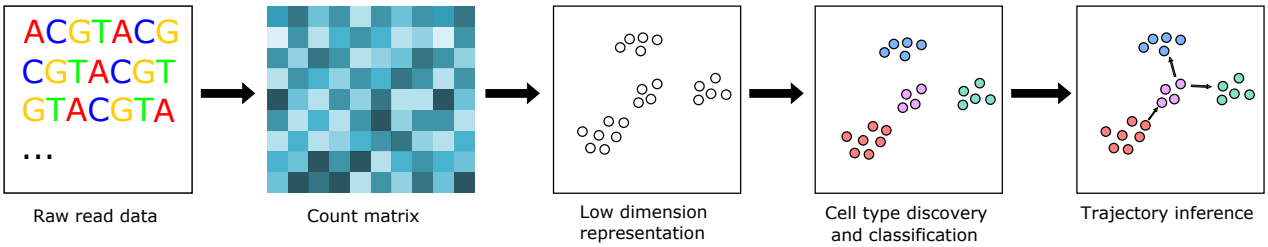


Figure 2.1: Standard analysis pipelines using a single modality of single-cell omics data start by turning the raw sequencing reads into a matrix of cells×feature counts. This matrix is then used for dimension reduction, representing each cell by a vector of lower dimension (embedding). The embedding is then used as starting point for subsequent tasks such as visualization, cell type discovery, or trajectory inference.

While early and widely-used methods such as scran [54] and Seurat v2 [67] use standard principal component analysis (PCA) on log-transformed count data for DR, many new DR models have been proposed specifically for scRNA-seq data recently. A common theme has been to replace the implicit Gaussian noise assumption of PCA by explicit statistical models of raw count data, modelling for example overdispersion and zero-inflation due to dropout in the matrix factorization-based model ZinbWave [85], or heavy-tailed count distribution in the nonparametric Bayesian model of [86]. Several groups have also investigated the potential of (variational) autoencoders ((V)AE), a very popular class of deep learning-based DR models. In short, a (V)AE learns a low-dimensional representation of input data (cell transcriptomes in our case) that is sufficient to reconstruct the input data, using flexible neural network models to go from the input to the compressed representation (encoding), and from the representation to the input data (decoding). Several (V)AE models for scRNA-seq data have been proposed recently, include scVI [56], DCA [68], SAVER [87] and scVAE [88]. Methods using hyperbolic geometry have also recently been developed [89, 90]. These models differ from each other by some modelling assumptions, such as the statistical model for count data in the decoder, or the prior distribution of the low-dimensional representation, but otherwise follow a similar architecture. An interesting property of these models is their computational scalability, as they are typically implemented with deep learning libraries designed to train models with millions or more input points.

Have deep learning-based (V)AE definitively imposed themselves as the best DR approach for scRNA-seq data? The answer is not so simple. Besides requiring large number of cells to learn parameters, (V)AE performance was shown to be very sensitive to arbitrary parameter choices [80], and [81] highlighted that with datasets of a few hundreds or thousands cells simpler models remain competitive and easier to use. The practical difficulty to correctly train complex ML models is not specific to (V)AE: another example is the ”art of training” the popular t-distributed stochastic neighbour embedding (tSNE) model for visualizing scRNA-seq in two dimensions [91], that requires specific initialization and choices of hyperparameters. Once correctly trained, tSNE reaches the same performance as uniform manifold approximation and

projection (UMAP), a model proposed to improve tSNE mapping of scRNA-seq [92, 91]. This highlights, again, both the potential and the difficulty to train some modern ML-based models, and raises in particular important concerns about making sure that all published results are reproducible and not overfitted to a given experiment.

Several DR methods for single-cell epigenomic data have also been proposed recently, either based on standard PCA models [93, 94], matrix factorization with latent Dirichlet allocation [95], or a VAE [65]. A recent benchmark highlights the importance of preprocessing, in particular how reads are binned into regions of interest and counted, for the success of these methods [96].

One interesting idea to use complex models on small datasets is to leverage larger, already annotated, datasets to learn the embedding, using techniques from the field of transfer learning or domain adaptation. Embeddings learned by PCA and non-negative matrix factorisation (NMF) on datasets such as the Human Cell Atlas (HCA) have successfully been used in both scATAC-seq [97] and scRNA-seq [98, 99] on new unseen datasets and cell types, as well as used for denoising the new dataset [100]. Similarly the embeddings learned by (demoising) AEs on one dataset, have been shown to be useful on other datasets, both for clustering [101, 102, 103, 104] and surface protein prediction [105]. One limitation of these methods is that the embedding is only learned on a single dataset, and applied to another dataset, without analyzing both in parallel. This limits the ability to train models on multiple datasets and thus truly leverage the mass of experiments in databases such as HCA.

The result of the DR is often fed to standard clustering algorithms, as reviewed in [106], in order to identify cell types, with these algorithms also being extremely sensitive to hyperparameter choices [107]. Once the cells are clustered, differential expression tools, benchmarked in [108], can be used to identify *de novo* marker genes.

The cells can also be matched to known cell types either by querying a reference database with tools such as Cell BLAST [109], scMap [110], scQuery [111] or CellFishing.jl [112] or by using standard supervised learning techniques as benchmarked in [63]. However these methods can be sensitive to batch effects, whose corrections are the subject of the following section.

## 2.3 Batch correction and integration of heterogeneous scRNA-seq data

Instead of analyzing data of a single experiment, much can be gained by jointly analyzing single-cell transcriptomic data of many experiments, potentially coming from different labs, using different technologies, and following different experimental protocols. ML models are likely to benefit from analyzing more cells, but the risk of capturing batch effects and other confounding factors instead of biological knowledge is large and considered one of the grand challenges of scRNA-seq data analysis [83]. A number of models have been proposed to specifically perform jointly DR on heterogeneous scRNA-seq data, build a global graph or construct a common gene expression matrix, aiming to capture biology and ignore confounding effects (see Figure 2.2 and [113] for a comprehensive benchmark).

A first group of models learn a low-dimensional representation over a common space that is invariant to technical confounders. Among those, SAUCIE [114] and scDGN [115] are deep-learning based, SAUCIE is an AE trained with a specific regularisation penalty on the latent

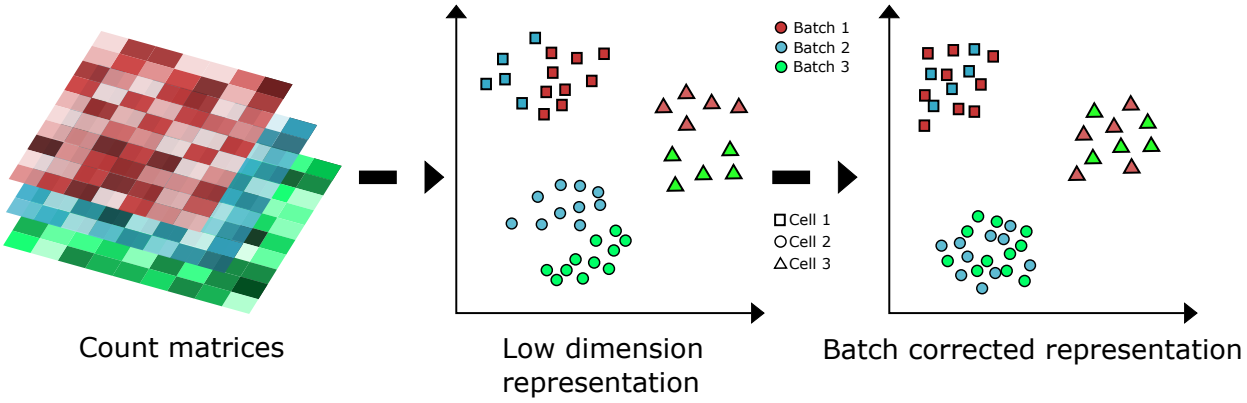


Figure 2.2: Different experiments of a similar modality (e.g., scRNA-seq) containing different number of cells can be integrated into a single unified view. At first, cells of the same type are separated by their batch, but after correction are perfectly merged together.

codes to remove batch effects, and scDGN is a supervised adversarial neural network model trained to accurately classify cell types and discriminate against batches. scMC [116], Harmony [117] and SMNN [118] rely on a linear transformation to a lower dimensional space, clustering (shared nearest neighbour scheme, soft k-means or supervised mutual nearest neighbours) and post-processing of the low dimensional embeddings to both account for cell-cell similarities and remove batch-specific variations. Other models have an objective to build a joint graph connecting all measured cells, such as scPopCorn [119] which relies on PageRank and graph-k partitioning, and Conos [120] which exploits cell-cell similarity matrices and mutual nearest neighbours. These graph-based models allow for tasks such as cell annotation and information propagation along the network. However, the methods previously described hinder interpretability as they do not enable studying differentially expressed genes leveraging the multiple datasets. A third group of models attempt to tackle this problem by correcting for batch effects on the original count data. Among them, scAlign [121] uses paired AEs with a common latent space that conserves the cell-cell distances estimated in the count data, while BERMUDA [122] instead uses a regularisation penalty on cell clusters from different batches in the latent space, and scGen [123] combines VAEs and latent space vector arithmetics. scVI [56] and trVAE [124] are so-called *conditional* VAE approaches that condition the decoder on an auxiliary batch variable to correct the data in the latent space. Based on variants of nearest neighbour search, scMerge [125] combines mutual nearest clusters and RUV-III factor analysis to remove unwanted factors from the count data, and Scanorama [126] and Seurat v3 [127] rely on linear projection to a low-dimensional space and an efficient (mutual) nearest neighbour search to obtain matched cells in low-dimensional space that are used to build translation vectors in the high-dimensional space.

All methods cited above offer batch correction for scRNA-seq data, while scMC has also been proposed for scATAC-seq integration and SAUCIE for single-cell CyTOF measurements. While most methods need shared cell types across datasets to build anchor cells, SAUCIE, scPopCorn and scMerge can be used without. Finally, almost half of the methods are able to scale to datasets containing hundreds of thousands of cells.

## 2.4 Learning trajectories, dynamics and regulation

Besides capturing the cellular heterogeneity of tissues and identifying cell types, single-cell omics data offers the possibility to learn about *dynamical* processes that shape this heterogeneity, such as cell cycle, differentiation, proliferation or tumorigenesis. From a data analytical point of view, this raises the question of inferring a dynamical model or at least the cellular trajectories from a snapshot of cells scattered at different time points along the dynamics. Since the first algorithms such as Monocle [128] were published in 2014 to infer trajectories and order cells using the notion of pseudotime, dozens of methods have been proposed. Recently proposed methods include GrandPrix [129], an efficient implementation of the Gaussian process latent variable model (GPLVM) to estimate pseudotimes and their uncertainty; STREAM [130], which estimates a low-dimensional set of curves, called the principal graph, to describe the cells' pseudotime, trajectories and branching points; PAGA [131], a graph-based method to compute a graph representation of a set of cells, allowing visualization and dynamical interpretation at different resolutions; TinGa [132], which builds a graph to fit the single-cell omics data as well as possible using the Growing Neural Graph (GNG) algorithm; or Monocle 3 [133], the latest version of Monocle with new features such as learning trajectories with loops or point of convergence and better scalability. To help users choose a particular method for a given problem, [134] published an impressive benchmark of trajectory inference methods, comparing 45 published algorithms on 110 real and 229 synthetic datasets. While no clear winner emerges in all situations, the benchmark is useful to understand the strengths and weaknesses of different methods in different settings.

A related problem is to infer the relationships between populations of cells captured at different time points along a dynamic process, such as developmental processes after induced pluripotent stem cell reprogramming observed through scRNA-seq profiles captured at half-day intervals [135]. In that paper the authors develop a method, called Waddington-OT, to relate the populations of cells at different time points using the concepts and tools of optimal transport (OT), a mathematically well-established and fast-growing field in ML [136], particularly well adapted to compare populations of cells and model their evolution. With ImageAEOT, [137] show how OT combined with an autoencoder allows to predict the lineages of cells using time-labeled single-cell images.

While trajectory inference implicitly allows us to predict the future evolution of cells, some algorithms have also been proposed to explicitly infer the *velocity* of each individual cell's transcriptomic profile. Following the pioneering work of [138], [139] proposed scVelo, a likelihood-based dynamical model for velocity inference from the ratio of spliced and unspliced mRNA. [140] propose another kernel-based velocity estimator, and show how gene regulatory networks (GRN) can be automatically inferred, although with modest accuracy, by training a sparse regression model to predict the velocity from gene expression levels. Another recent attempt to reconstruct GRN and more general gene networks from scRNA-seq data with an ML approach is the convolutional neural network for coexpression (CNNC) approach of [141], who represent each gene pair as a scatter plot of their expression levels across cells and train a standard CNN for 2D images on the resulting plots to learn pairwise relationships.

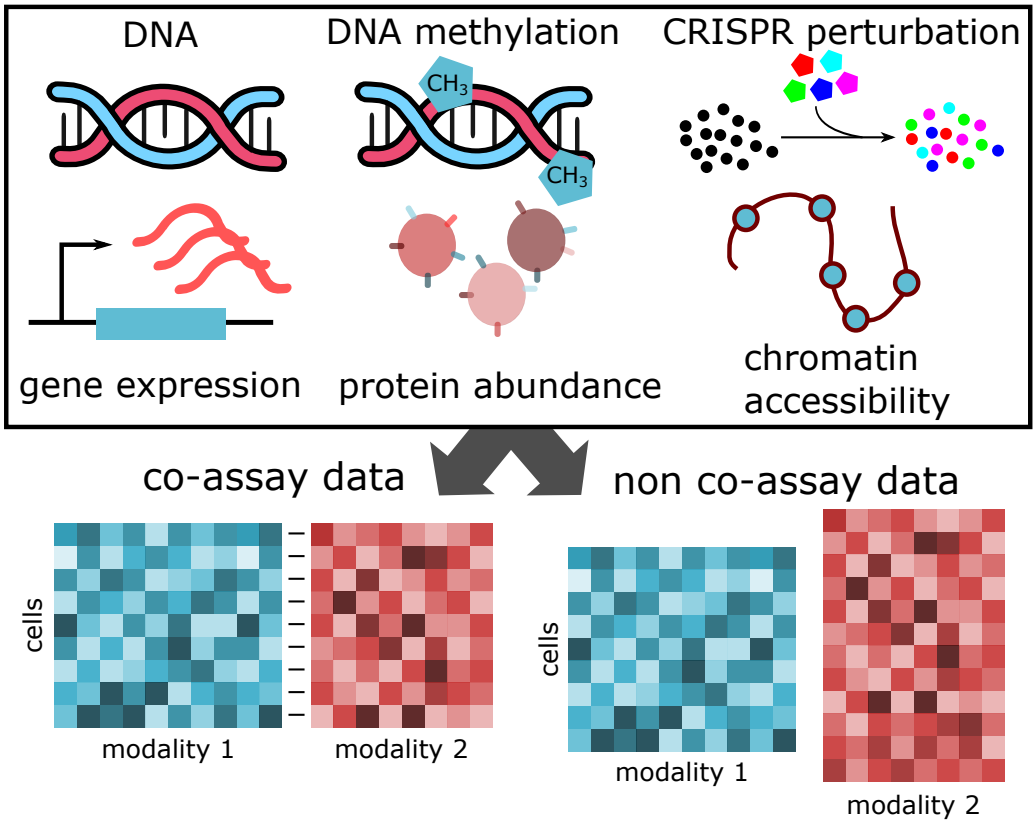


Figure 2.3: Single-cell modalities can take various forms, such as DNA, DNA methylation, CRISPR perturbations, transcriptomics, proteomics or chromatin accessibility. ML models developed for single-cell multimodal data integration assume that the correspondences between cells are either known (co-assay data) or not (non co-assay data) across modalities. In the case of non co-assay data, additional supervision signal might be used, such as cell types, correspondences between features or anchor cells.

## 2.5 Multimodal data integration

An important problem in single-cell omics data analysis is to integrate several modalities together, in order to enhance the performance of downstream tasks such as cell type labelling, identification of subpopulations, visualisation or regulatory network inference, as reviewed in [142, 143]. Several ML approaches have been developed for that purpose, for instance by characterizing cells across measurements, projecting multiple measurements into a common latent space or learning the missing modalities. Transcriptomics is typically one of the modalities that is integrated, together with chromatin accessibility [144, 127, 145], DNA [146], DNA methylation [147, 127], proteomic data [148, 149, 144, 150, 151] or CRISPR perturbations [152, 153].

A first category of models assume that the correspondences between cells are known across modalities, with direct applications to co-assay data (Figure 2.3). Such methods learn a joint representation of each cell or a cell-cell similarity matrix that is used for downstream analyses by exploiting variants of VAEs such as totalVI [154] and scMVAE [145], matrix

factorisation-based models such as scAI [151] and MOFA+ [155], or k-nearest neighbour prediction to learn cell-specific modality weights as Seurat v4 [156]. A second category of models do not require co-assays within individual cells and can be applied to independent multi-omics datasets originating from different cells. Current deep learning-based methods either rely on a pair of VAEs whose latent spaces are coupled through a specific penalty (K. D. Yang et al., arxiv.org/abs/1902.03515), or on learning low-dimensional representations minimising a tSNE loss for each view, coupled through a learned matching matrix (UnionCOM [149]). Other methods rely on NMF, to learn a low-dimensional space composed of specific and common factors (LIGER [150]), or cluster representatives of subpopulations of cells (DC3 [157]). MMD-MA [148, 158] learns a joint latent representation where different modalities have a similar distribution using the theory of kernel methods. SCOT [144] uses OT to learn a joint distribution between cells from both views. clonealign [146] models the association between copy number features and gene expression leveraging mean field variational Bayes inference. While these methods can in theory be applied to any bi-modal omics dataset, hyperparameter selection is difficult when no co-assay data is available for MMD-MA, SCOT and UnionCOM. Among models that do not require co-assay data, some use weak supervision such as SCIM [147], an adversarial AE model that assumes that the cell types are known for a fraction of the cells and Seurat v3 [127], a canonical correlation analysis (CCA)-based model that relies on building anchor cells using mutual nearest neighbours. Applied to single-cell CRISPR screenings, scMAGeCK [153] relies on statistical analyses and MUSIC [152] on topic modeling in order to link gene perturbations to cell phenotype. Finally, it is worth mentioning that some models require features to have a one-to-one correspondence between views [146, 127, 150, 152, 153], which may not be the case systematically.

While the diversity of models is large, most of them rely on finding a joint low-dimensional space that can be later used on downstream tasks. Most models combine two modalities and a few enable the integration of more than two, such as UnionCOM, MOFA+ and DC3, the latter also incorporating scHiC or bulk HiChIP datasets. Finally, the scalability of the models evolve conjointly with single-cell technologies, nowadays being able to handle tens or hundreds of thousands of cells [145, 147].

## 2.6 Conclusion

Researchers are facing an exponential growth of approaches to deal with single-cell genomics data, with over 800 tools (scrna-tools.org) published for scRNA-seq analysis so far, many of which being based on ML approaches. A vast majority of ML-based tools have been straightforwardly imported from other fields, with some features unsuited for genomic challenges and to the reality of biological data - thereby not maximising their performance. In particular, a number of parameters, which have a strong impact on performance, need extensive training to be properly tuned, which is often unrealistic in the case of genomic data. It also raises questions of reproducibility that the scientific community should address, defining for example the processed datasets and variables that should be shared, i.e., random seed values or reduced dimensional spaces, in addition to the raw data. Whether ML models will in the near future make up for the current technical limitations of single cell genomics approaches - e.g dropouts, batch effects - remains uncertain. If current single-cell omics achieve genome-wide

characterization of the transcriptome or epigenomes for example, these methods do not yet achieve single-locus/single-cell resolution due to the dropouts within datasets, leaving room for experimental and computational optimisation.

# Chapter 3

# Tuning parameters of dimensionality reduction methods for single-cell RNA-seq analysis

## 3.1 Introduction

Single-cell RNA sequencing (scRNA-seq) is a powerful technology to characterize the transcriptomic profile of individual cells within a population [159]. By allowing researchers to identify cell types based on their transcriptomic signatures instead of pre-defined markers, it is rapidly establishing itself as a standard tool to answer a variety of biological questions, ranging from characterizing the heterogeneity of complex tissues [160, 161] to discovering new cell types [21] or elucidating cell differentiation processes [135].

The analysis of scRNA-seq data raises, however, a number of challenges. Due to the small amount of RNA available in each individual cell, and to the technical difficulty to analyze thousands (or millions) of cells in parallel, raw scRNA-seq data have been found to be subject to a number of biases including low sequencing depth, over-dispersion and zero inflation of read counts, or sensitivity to batch effects [162, 163]. Many computational methods have therefore been developed in recent years to take into account the specificities of scRNA-seq data and address the issues of data normalization, cell type identification, differential gene expression analysis, cell hierarchy reconstruction, or gene regulatory network inference (see [164, 53] for recent reviews). In order to help practitioners choose an analysis pipeline among the many available, several studies have benchmarked algorithms and softwares for applications such as dimensionality reduction [165], clustering [166, 167], differential expression [108], or trajectory inference [134].

One shared caveat by these benchmarking efforts, however, is that the methods tested are run with their default parameters. This may not reflect what an educated user would do in practice, and does not address the practical questions of (i) whether parameter tuning is relevant at all for a given method, and (ii) how to tune parameters if needed. Recently, [80] highlighted the relevance of these questions, showing that variation autoencoders (VAE) algorithms for dimension reduction (DR) of scRNA-seq work very well once properly tuned, but are extremely sensitive to changes in parameters and can dramatically fail if not properly tuned.

Here, we propose to challenge this issue of parameter tuning focusing on methods for DR of scRNA-seq data, not only because they can be directly useful for visualization purpose, but also because DR is a common first step for most downstream applications such as cell type identification or trajectory inference [164, 53]. We propose a new benchmark protocol for DR methods, composed of ten scRNA-seq datasets of various complexity mixing experimentally characterized populations of cells, where we measure the quality of a DR method by its ability to map cells of a given cell type near each other in the representation space. Using this protocol, we benchmark five popular and representative DR methods, combining both PCA-based methods, a matrix factorization method, and VAE methods, systematically varying their tunable parameters. The resulting ∼1.5 million experiments reveal not only the performance of DR methods using their default parameters, but also their performance if parameters are properly tuned. We find in particular that principal component analysis (PCA)-based methods like scran [168] and Seurat [67] are competitive with default parameters but do not benefit much from parameter tuning, while more complex models like ZinbWave [169], DCA [68], and scVI [56] can reach better performance but after parameter tuning. We propose and evaluate two strategies to tune parameters automatically, either by changing the default parameters or by optimizing a heuristic on each new dataset. In spite of promising results for some of the methods like ZinbWave, both strategies sometimes identify very suboptimal parameters, suggesting that parameter tuning for complex DR models on dataset without ground truth annotation remains an important but largely open problem.

## 3.2 Results

### 3.2.1 A benchmark of DR methods for scRNA-seq data

A DR method takes a scRNA-seq dataset as input and maps each individual cell to a point in $d$-dimensional *representation space*, where downstream applications such as cell type prediction or lineage reconstruction are performed. In order to empirically assess the quality of DR methods and the influence of parameter tuning, we propose a benchmark protocol, summarized in Figure 3.1, where we collected ten diverse scRNA-seq datasets with experimentally validated cell types, and evaluate five representative DR methods, tested on a large parameter sweep, according to their ability to map cells of a given origin near to other cells of the same cell type.

Table 3.1 and Figure 3.2 summarize the main features of the ten datasets.

Each dataset contains hundreds to thousands of experimentally characterized cell types, either derived from known cell lines or purified by FACS. The ten datasets vary in the technology used (10x, CEL-Seq2 or Smart-Seq2), the organism of origin (human or murine), and the overall biological complexity of the mixture. More precisely, the first five datasets (Zhengmix4eq to Zhengmixun8eq) are *in silico* mixtures of FACS purified human immune cell populations from [170] produced with 10x, comprising either equal mixes of four, five and eight cell populations, or unequal mixes of four and eight cell populations. The datasets with five and eight cell populations are particularly challenging, since they both contain five closely related T-cell populations. The next four datasets (sc_10x to sc_celseq2_5cl) are *in vivo* mixtures of three or five human cell lines from [167], sequenced by CEL-Seq2 or 10x. Finally, the last dataset is

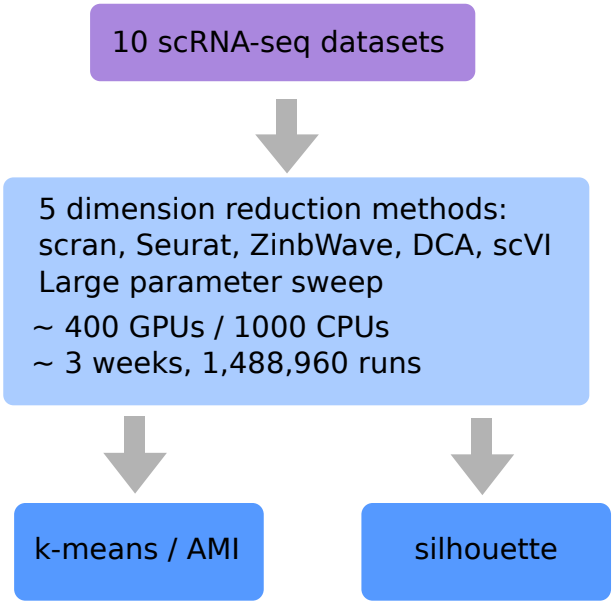


Figure 3.1: Overview of the benchmark protocol. We ran five representative DR methods, systematically varying their parameters on a large grid of values, on ten scRNA-seq datasets with known cell identity. We evaluate their ability to map cells of a given identity near other cells of the same identity, as measured by the silhouette score and the AMI after $k$-means clustering in the representation space.

an *in silico* mixture of four FACS purified mouse tissues from [171] produced with Smart-Seq2, where we selected tissues with no overlap in cell types.

We use this benchmark to evaluate the performance of five popular computational pipelines for DR of scRNA-seq data: scran [168], Seurat [67], ZinbWave [169], DCA [68], and scVI [56]. These pipelines are all publicly available as R or Python packages, and can process datasets containing thousands of cells in a reasonable amount of time (less than 12 hours on a GPU/CPU with 10 cores) . They all implement processing steps speficic to scRNA-seq data together with representative DR methods including principal component analysis (PCA) for scran and Seurat, matrix factorization for ZinbWave, and (variational) autoencoders for DCA and scVI. While these pipelines also implement various downstream tasks such as cell clustering or differential expression analysis, we restrict our analysis to the DR step.

To quantify the ability of a DR method to map biologically similar cells to similar locations in the representation space, we use two complementary measures: the silhouette, on the one hand, and the adjusted mutual information (AMI) when the cells are clustered with the $k$-means algorithms, on the other hand (see details in Material and Methods). Both measures vary between 0 for a random embedding to 1 for an embedding that perfectly preserves the cell type information. AMI directly measures how well a particular clustering algorithm ($k$-means) recovers known cell types, and is therefore a good proxy for the performance of cell type identification as a downstream task of DR. Silhouette is a measure agnostic to any particular clustering algorithm, and measures how close a cell is to other cells of the same type compared to cells of different types; for the silhouette to be large, cell types must not only be separated,

| Dataset | Technology | Organism | N cells | Cell types | N types | Ref. |
|---|---|---|---|---|---|---|
| Zhengmix4eq | 10x | Human | 3,909 | B (24.5%), Monocytes (24.5%), cytoT (25.5 %), rT (25.5 %) | 4 | [170] [166] |
| Zhengmix4uneq | 10x | Human | 6,345 | B (15%), Monocytes (30%), cytoT (8 %), rT (46 %) | 4 | [170] [166] |
| Zhengmix5eq | 10x | Human | 4,876 | hT (20%), mT (20%), cytoT (20%), nT (20%), rT (20%) | 5 | [170] |
| Zhengmix8eq | 10x | Human | 3,908 | B (12.5 %), Monocytes (14.5 %), hT (10 %), NK (15 %), mT (12.5 %), cytoT (10 %), nT (13 %), rT (12.5 %) | 8 | [170] [166] |
| Zhengmix8uneq | 10x | Human | 6,350 | B (7.5%), Monocytes (15%), hT (8%), NK (4%), mT (15%), cytoT (4%), nT (23%), rT | 8 | [170] |
| sc_10x | 10x | Human | 902 | H1975 (34.5 %), H2228 (35 %), HCC827 (30.5 %) | 3 | [167] |
| sc_10x_5cl | 10x | Human | 3,918 | A549 (32 %), H1975 (11 %), H2228 (19.5 %), H838 (22.5 %), HCC827 (15 %) | 5 | [167] |
| sc_celseq2 | CEL-Seq2 | Human | 274 | H1975 (41 %), H2228 (29.5 %), HCC827 (29.5 %) | 3 | [167] |
| sc_celseq2_5cl | CEL-Seq2 | Human | 895 | A549 (36 %), H1975 (14.5 %), H2228 (14 %), H838 (22 %), HCC827 (13.5 %) | 5 | [167] |
| TabulaMuris | Smart-Seq2 | Murine | 12,081 | Brain (36.5 %), Intestine (31.5 %), Skin (19 %), Spleen (13 %) | 4 | [171] |

Table 3.1: Benchmark datasets. The first five datasets are derived from [170] and [166], and contain CD19+ B cells (B), CD14+ monocytes (Monocytes), CD4+ helper T cells (hT), CD56+ natural killer cells (NK), CD4+/CD45RO+ Memory T Cells (mT), CD8+/CD45RA+ Naive Cytotoxic T Cells (cytoT), CD4+/CD45RA+/CD25- Naive T cells (nT), and CD4+/CD25+ Regulatory T Cells (rT). The four next are from [167] and contain the five following cell lines: A549, H1975, H2228, H838, and HCC827. The last one is from [171].

but also form compact clusters far from each others.

### 3.2.2 Performance of five popular DR methods with default parameters

We first assess the performance of each method with its default parameters, except for the dimension of the representation space which we arbitrarily set to 10 for all methods. Indeed, the performance scores (AMI and silhouette) strongly vary with the dimension (Figure S11 and S12), so fixing the dimension allows to compare more fairly the different DR methods. Figure 3.3.A (with "default" legend) shows the performance reached by each method on each dataset, in terms of AMI (left) and silhouette (right), and Table 3.2 summarizes the mean performance reached by each method over the datasets.

As expected, Figure 3.3.A clearly shows that for all methods, the performance varies across datasets, in a rather consistent manner. In terms of AMI, the four cell lines datasets tend to be the easiest (AMI>0.9 for most methods), followed by TabulaMuris and the two Zhengmix mixtures of four cell lines (AMI in the range 0.7∼0.9 for most methods), followed by the three Zhengmix mixtures containing the five closely related T-cell populations (AMI in the range 0.1∼ 0.7 for most methods). The silhouette scores overall follow the same trend, although the difference between the first two groups of datasets is less pronounced. Interestingly, we see that in cases where several DR methods allow to almost perfectly cluster the cell types, such as sc_10x or sc_10x_5cl, the silhouette is usually far from 1 and varies across methods,

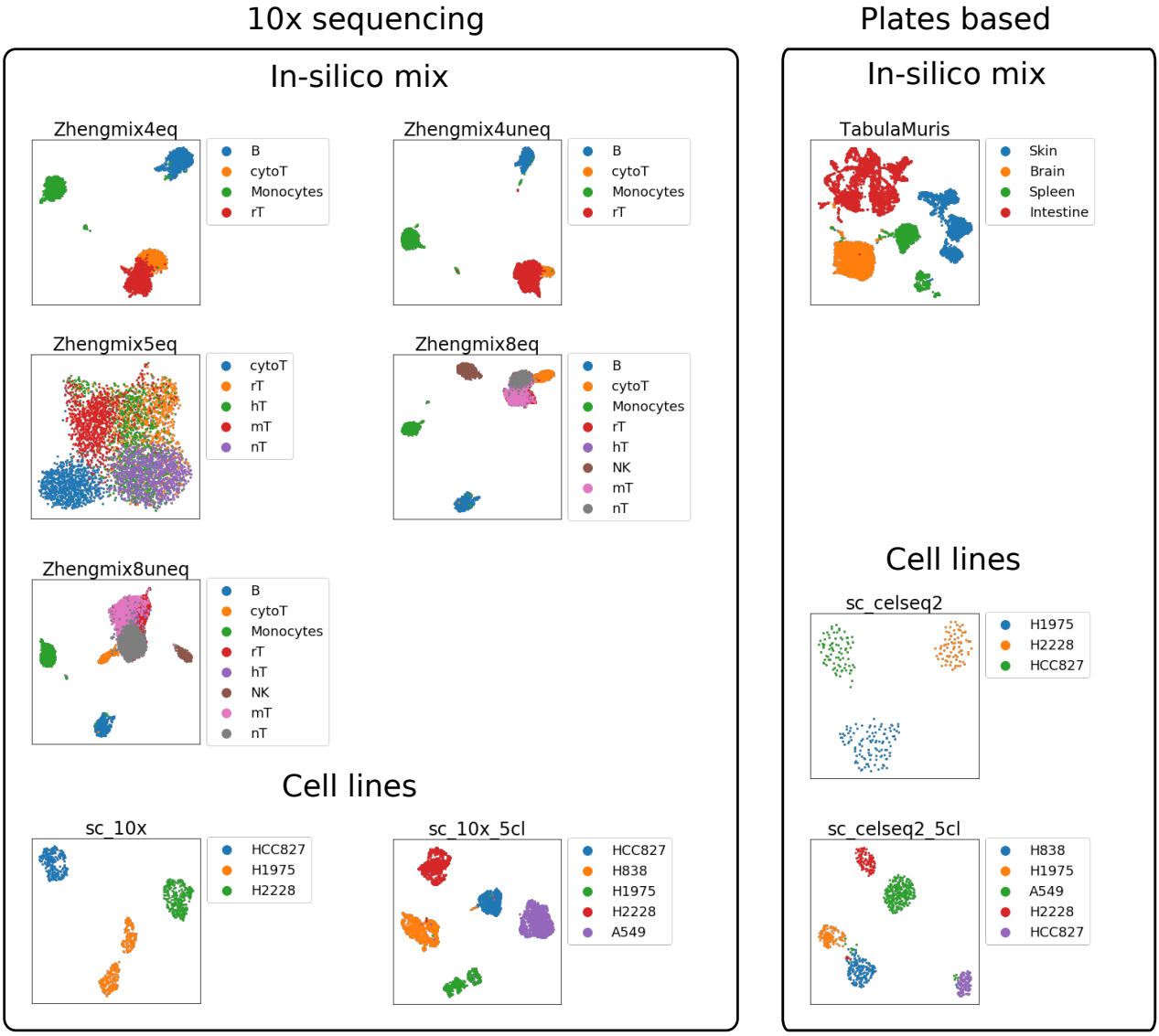


Figure 3.2: UMAP representation of the ten scRNA-seq datasets, run after processing of the count matrices with Seurat with default parameters.

illustrating the complementarity of both measures.

Besides variations across datasets, we also observe variations across DR methods. As shown in Table 3.2, scran has the best AMI on average (mean AMI=0.84), followed by Seurat, ZinbWave and DCA (mean AMI=0.75∼0.79), but this ranking is not statistically significant (p-value $> 0.05$ for Wilcoxon one-way test), while scVI is clearly behind (mean AMI=0.56) (p-value $< 0.05$ for all methods). As suggested in Figure 3.3.B on Zhengmix8eq, for example, scVI with default parameters does not manage to clearly isolate the three non-T cell clusters, resulting in errors in $k$-means clustering. In terms of silhouette, all methods are very similar (mean silhouette=0.36∼0.39), except for ZinbWave which is clearly behind (p-value $< 0.05$ for all methods). The reason why ZinbWave tends to have a correct AMI but a poor silhouette is sug-

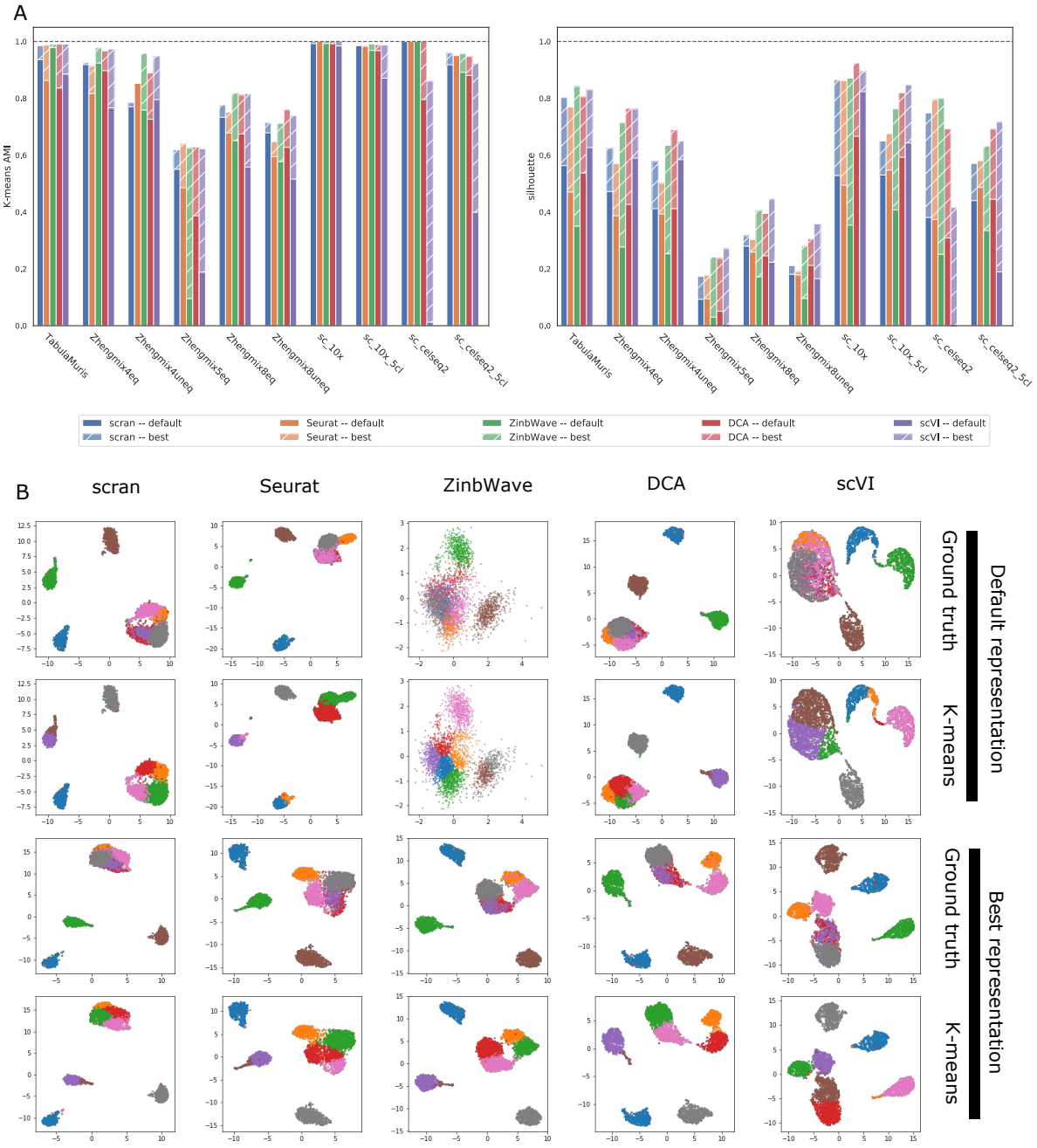


Figure 3.3: Performance of five DR pipelines (scran, Seurat, ZinbWave, DCA and scVI) with default parameters and a dimension of 10 (legend "default") or after parameter optimization (legend "best") on our benchmark of ten datasets. **A.** AMI (left) and silhouette (right) reached by each method on each dataset. **B.** UMAP representation of Zhengmix8eq after DR by each method (in column) using default parameters (top two rows) of after parameter optimization (bottom two rows). In each row, cells are colored either based on their true cell type (rows 1 and 3) or based on a $k$-means clustering.

| Method | Mean AMI | | | | Mean silhouette | | | |
|---|---|---|---|---|---|---|---|---|
| | Default | Best | ANOVA AMI heuristic | silhouette heuristic | Default | Best | ANOVA AMI heuristic | silhouette heuristic |
| scran | **0.840** | 0.868 | 0.841 | 0.741 | 0.362 | 0.547 | 0.396 | 0.494 |
| Seurat | 0.788 | 0.860 | 0.814 | 0.683 | 0.369 | 0.543 | 0.490 | 0.373 |
| ZinbWave | 0.780 | **0.896** | **0.851** | **0.825** | 0.249 | 0.609 | **0.562** | **0.591** |
| DCA | 0.758 | 0.885 | 0.837 | 0.583 | **0.396** | **0.639** | 0.403 | 0.381 |
| scVI | 0.560 | 0.872 | 0.709 | 0.510 | 0.384 | 0.621 | 0.482 | 0.318 |

Table 3.2: Mean performance on the ten datasets of each method in terms of AMI and silhouette. The "Default" columns correspond to the performance of each method using its default parameters, with a dimension of 10. The "Best" column corresponds to the best performance reached after varying the parameters. The "ANOVA AMI heuristic" column corresponds to the performances of the new default parameters described in section 3.2.4. The "silhouette heuristic" column corresponds to the performance of the heuristic described in section 3.2.5

gested by Figure 3.3.B, where we see on Zhengmix8eq that ZinbWave (with default parameters) produces a representation good enough for $k$-means to correctly recover most of the cell types, but where the different clusters look much less compact and separated from each other than with other DR methods. The p-values for the comparisons can be found in Figures S13-S14.

While these relative performances hold for a representation in dimension 10, the performance of some methods fluctuates with the dimension of the embedding space (Supplementary Figures S11 and S12). While scran and Seurat are rather insensitive to increase in dimension after 10, DCA's mean AMI tends to increase in higher dimensions, while ZinbWave's and scVI's mean AMI decrease with the dimension, suggesting that different methods need more or less dimension to capture the same biology.

This average performance of DR methods hides important variations across datasets, as visualized in Figure 3.3.A. For example, we see that scVI has specifically poor performance compared to other methods on the two CEL-Seq2 datasets, which may be due to a particularity of this technology or to the fact that both datasets (sc_celseq2 and sc_celseq2_5cl) have a relatively small number of cells. The difference in AMI across methods, and the good behavior of the linear models underlying scran, Seurat and ZinbWave, is most visible on the "difficult" Zhengmix mixtures containing the five closely related T-cell populations, while the difference in silhouette, and the good behaviour of the nonlinear models underlying DCA and scVI, is more visible in the "easy" datasets where all methods have an AMI above 0.7.

### 3.2.3 Performance reachable across a parameter sweep

While using a computational pipeline with default parameters is often the method of choice for practitioners, there is little guarantee that default parameters are adapted to all situations. In particular, the performance of deep learning-based methods for scRNA-seq analysis was shown to be highly sensitive to choices of parameters [80]. In order to assess the performance of each DR method if all parameters were properly tuned in a dataset-specific way, we now run each method by sweeping all tunable parameters across a large grid of values, as summarized in Table 3.3, and compute the performance reached by each method after cherry picking *a*

*posteriori* the best parameters. Note that the resulting performance is therefore an upper bound on the performance that each method can reached if parameters are tuned without knowing the ground truth.

| Method | Parameters | Values |
|---|---|---|
| scran | Size factors normalization | { **True**, False } |
| | ERCC counts normalization | { **True**, False } |
| | Assay type | { **logcounts**, counts } |
| | High variance genes | { 100, 300, **500**, 1000, 2000, 3000 } |
| | Dimension of latent space | { 2, 8, 10, 16, 32, **50**, 64, 128} |
| Seurat | Normalization method | { **LogNormalize**, CLR } |
| | Criteria for high variance genes | { **vst**, mvp, dist } |
| | High variance genes | { 100, 300, 500, 1000, **2000**, 3000 } |
| | Dimension of latent space | { 2, 8, 10, 16, 32, **50**, 64, 128} |
| ZinbWave | Gene covariates | { True, **False** } |
| | Epsilon (regularizer) | { 200, 500, **1000**, 2000 } |
| | High variance genes | { **100**, 300, 500, 1000, 2000, 3000 } |
| | Dimension of latent space | { **2**, 8, 10, 16, 32, 50, 64, 128} |
| DCA | Dispersion and reconstruction | { **zinb-conddisp**, zinb, nb-conddisp, nb } |
| | Batch normalization | { **True**, False } |
| | Dimension of the latent space | { 2, 8, 10, 16, **32**, 50, 64, 128 } |
| | Number of training epochs | { 20, 50, 100, 200, **300**, 500, 1000 } |
| | Normalize counts | { **True**, False } |
| | Scale variance | { **True**, False } |
| | Log normalization | { **True**, False } |
| | Dropout rate | { **0**, 0.1 } |
| | Number of hidden neurons | { **64**, 128, 256 } |
| | Random seed | { **0**, 1, 2, 3, 4 } |
| scVI | Number of hidden neurons | { 64, **128**, 256 } |
| | Number of training epochs | { **20**, 50, 100, 200, 300, 500, 1000 } |
| | Learning rate | { 1e-2, **1e-3**, 1e-4 } |
| | Dropout rate | { 0, **0.1** } |
| | Layers | { **1**, 2 } |
| | Dimension of the latent space | { 2, 8, **10**, 16, 32, 50, 64, 128 } |
| | Dispersion | { **gene**, gene-cell } |
| | Reconstruction loss | { nb, **zinb** } |
| | Random seed | { **0**, 1, 2, 3, 4 } |

Table 3.3: Description of parameter sweep. For each method (first column), we vary a number of tuneable parameters (second column) systematically over a grid of values (third column). The bold value in the third column is the default value.

In total, sweeping across the grid of parameters results in 384 different runs per dataset for scran and ZinbWave, 288 for Seurat, 40,320 for scVI, and 107,520 for DCA, hence a total of 1,488,960 DR experiments. Running all experiments took several weeks on a dedicated cluster

of 1,000 CPUs and 400 GPUs. Out of these runs 60% ran correctly for scran, 99.97% for Seurat, 96.51% for ZinbWave, 97.96% for DCA, and 100% for scVI. The low number of runs for scran is mostly due to the absence of ERCC in all the 10X datasets, and because the size factor computation on TabulaMuris failed. The failures for ZinbWave and DCA were due to memory issues (either of the GPU or CPU). There was a single failure for Seurat whose cause has not been identified.

We report in Figure 3.3.A and Table 3.2 (with "Best" legend) the best value reached across the parameter sweep on each dataset, in addition to the performance reached with the default parameters. Overall, we see that for all methods, a gain can result from parameter tuning compared to using default parameters. For AMI, the mean gain across datasets ranges from 0.311 for scVI to 0.028 for scran, while for for silhouette, it ranges from 0.185 for scran to 0.360 for ZinbWave. Seurat and scran are the methods that benefit least from parameter tuning, suggesting that default parameters are already good choices across most datasets. Autoencoder-based DCA and scVI benefit more for parameter tuning, and outperform scran and Seurat in mean AMI after parameter tuning, confirming the importance of parameter tuning for these models [80]. Since the number of parameters tested for these models is also two orders of magnitude larger than for Seurat, scran and ZinbWave, the "best" performance after cherry-picking the best parameters may be over-optimistic for DCA and scVI. As for ZinbWave, a good choice of parameters leads to the best mean AMI across methods (0.896), and the largest improvement in silhouette compared to default parameters.

More precisely, for all cell line datasets and for TabulaMuris, parameter tuning allows all method to reach an almost perfect AMI, including methods like scVI that have a very poor performance with default parameters on sc_celseq2 and sc_celseq2_5cl. On the same datasets, parameter tuning brings an important improvement to the silhouette score of 0.2 to 0.6 to all methods. After parameter tuning, both encoder-based methods (DCA and scVI) tend to outperform ZinbWave, which tends to outperform both PCA-based methods scran and Seurat in terms of silhouette. This highlights the possibilities of nonlinear DR methods to perform DR even on simple datasets, but the need to correctly tune parameters in order to reveal their full potential.

On the immune cell datasets, we see again that tuning parameters allows to boost performance and bridge important gaps between methods in terms of silhouette, and that after parameter optimization both autoencoder-based methods slightly outperform ZinbWave, which slightly outperforms both PCA-based methods. The AMI performance of all methods is also improved by parameter optimization for all methods but scran, and we see no clear and consistent winner after parameter optimization. This suggests that simple PCA-based methods like scran, even with default parameters, are good enough to match the performance of more complex models after parameter tuning in terms of AMI; however the better silhouette of more complex models once tuned may be an advantage for other downstream applications beyond clustering by $k$-means. The benefits of parameter tuning is further illustrated in Figure 3.3.B, which shows a UMAP visualization of the representation space learned by the different methods with the default parameters (bottom two rows) or after parameter tuning for AMI (top two rows), for the Zhengmix8eq dataset. For ZinbWave and scVI, which strongly benefit from parameter tuning in this case, we see that the dataset looks very different with the default parameters or after parameter optimization, the different cell types being but better separated

in the later case.

Regarding the absolute performance reached across the datasets, it is interesting to note that the best AMI scores for Zhengmix5eq and Zhengmix8uneq (the two hardest datasets) are only between 0.6 and 0.7, when they are above 0.9 for Zhengmix4eq (the easiest one). This suggests that if current DR methods are good at identifying sufficiently different cell types, they have difficulties to differentiate very similar cell types, a variety of T cells in our case, even after parameter optimization.

### 3.2.4 Influence of parameters on performance

Having shown that parameter tuning has the potential to boost the performance of all methods for all datasets compared to using default parameters, we now investigate in more details the influence of each parameter on the performance. For that purpose, we estimate the mean contribution of each parameter value on the performance (AMI or silhouette) with a factorial analysis of variance (ANOVA, see Methods) procedure. We find that all parameters of all methods, except for "gene covariate" for ZinbWave on the silhouette, have a significant influence on both AMI and silhouette (ANOVA p-value $< 0.05$), and summarize in Table S11 and Table S12 the potential effect of tuning each parameter by comparing the best and worse contributions to performance among the values it can take. We see that some parameters can have a very important effect, such as proper log normalization in scran which on average can boost the AMI by 0.50, or the choice of dimension in the latent space for ZinbWave that can boost the silhouette by 0.56 on average. If we arbitrarily define a parameter as "influential" if its potential effect is more than 0.05 on AMI or silhouette, we see that in addition to the dimension of the representation space which is influential for all methods, scran, Seurat and ZinbWave have one influential parameters (log normalization for scran; normalization method for Seurat; number of top genes for ZinbWave), DCA has two (batch normalization and normalize counts), and scVI has three (dispersion, number of training epochs and learning rate), suggesting that more care in parameter tuning is needed for the autoencoder-based methods than for the matrix factorization-based methods.

As shown in Table S11 and 3.3, the best parameter values in terms of mean contribution to the performance are not always the default parameters of each method. This suggests that the best parameter values identified by our analysis, which we refer to below as "ANOVA AMI heuristic" when we pick the parameter values that have the largest positive influence on AMI, may be interesting to use as new default parameters for each method. To test this hypothesis, we report the performance of each DR method using the ANOVA AMI heuristic as default parameters in Figure 3.3.A, and summarize the mean performance across datasets in Table 3.2.

We see that, on average, all methods benefit from the ANOVA AMI heuristic compared to the existing default parameters, particularly scVI, DCA and ZinbWave in terms of AMI, and particularly ZinbWave, Seurat and scVI in terms of silhouette. In particular, ZinbWave outperforms all other methods, both in AMI and in silhouette, with the ANOVA AMI heuristic. Interestingly, we see in Figure 3.3 that for all methods, the AMI increases on almost all datasets with the ANOVA AMI heuristic. Of course these promising results should be taken with care, given that we evaluate the performance of the ANOVA AMI heuristic on the datasets used to perform the ANOVA, but they suggest a systematic approach for method developers to set

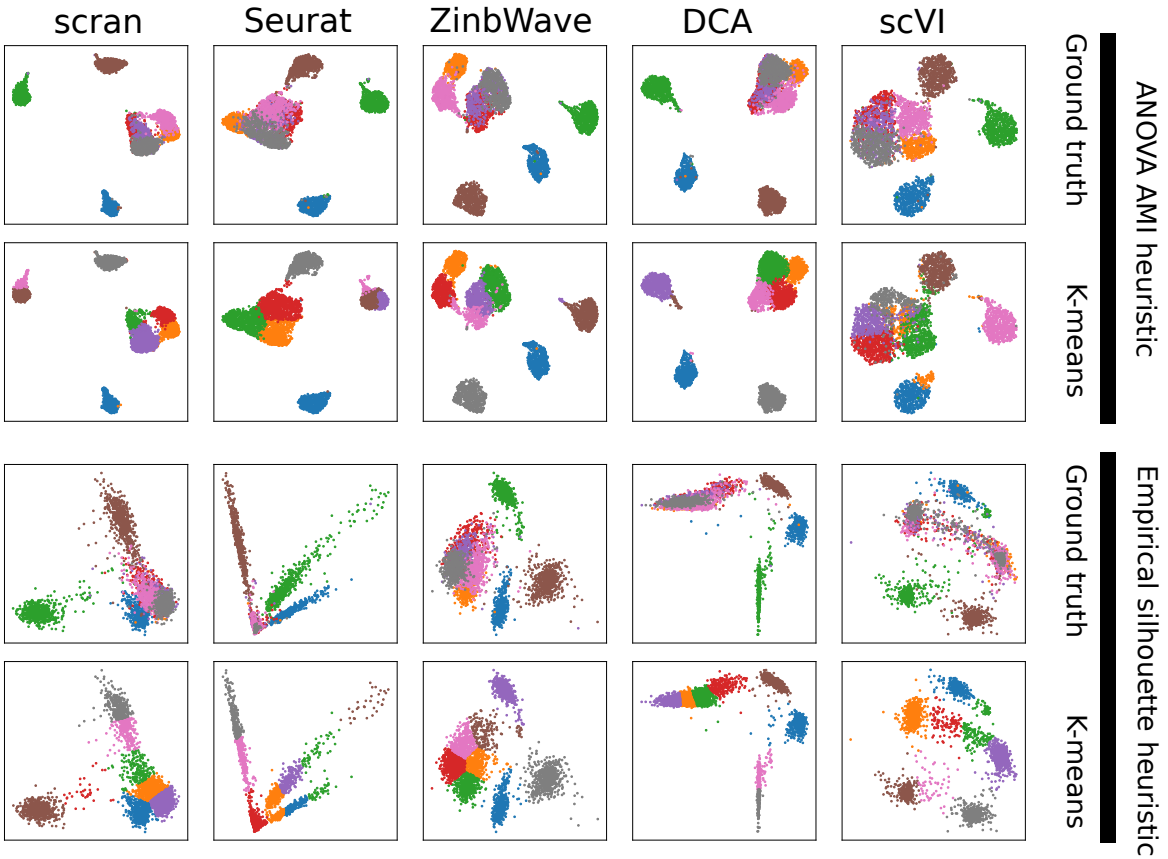


Figure 3.4: UMAP representation of Zhengmix8eq after DR by each method (in column) using the ANOVA AMI (top two rows) or empirical silhouette (bottom two rows) heuristic to tune parameters. In each row, cells are colored either based on their true cell type (rows 1 and 3) or based on a $k$-means clustering.

default parameters.

We now investigate in more details to what extent further performance gain may result from parameter tuning on each dataset separately, as opposed to setting new default parameter values common to all datasets. For that purpose we estimate again the contribution of each parameter in the final AMI and silhouette, allowing a different contribution in different datasets by adding interaction terms in the factorial ANOVA model between the dataset, on the one hand, and the tunable parameters, on the other hand (see Methods). Note that we remove from this analysis a few parameters that were not tested on all datasets: ERCC for scran on 10x datasets, and sum factor normalization for scran on TabulaMuris. Tsbles S1- S10 summarize the contributions of each parameter in each dataset for AMI and silhouette, as estimated from the factorial ANOVA with interactions. All interactions between dataset and parameters are significantly non zero (p-value $< 0.05$), except for the interaction between dataset and the "gene covariate" parameter of ZinbWave, showing that the influence of most parameters on the final performance is not the same across datasets. To assess whether dataset-specific parameter tuning is useful, we now check for each parameter whether the default value provided by the

ANOVA AMI heuristic, which identifies the best values on average, is also the best or within 0.05 of the best for both AMI and silhouette on *all* datasets. Based on this criterion, we find that for the AMI scran, Seurat and ZinbWave have one parameter that benefits from dataset-dependent tuning (number of top genes for scran and Seurat; dimension of the latent space for ZinbWave), DCA has two (scale variance and dropout rate), and scVI has three (dimension of the latent space, number of hidden neurons and number of training epochs); and for the silhouette, scran, Seurat, ZinbWave and scVI have one parameter that benefits from dataset-dependent tuning (dimension of the latent space for scran and Seurat; number of top genes for ZinbWave; number of training epochs for scVI), and DCA has two (scale variance and dropout rate). Table S11 and Table S12 detail the potential gain in dataset-specific tuning for each parameter of each method. This therefore confirms the potential benefit of tuning parameters on each dataset, particularly for autoencoder-based methods.

### 3.2.5 Tuning parameters in practice

Having shown that DR methods benefit from various degrees of parameter tuning, we now discuss the question of how this can be done in practice. Indeed, our strategy so far to identify the best parameters and evaluate their influence on performance is only possible when one knows the true cell type for each cell in the population, but such an oracle is usually not available in practice. In the absence of such information, one must therefore rely on quantitative heuristics or qualitative validation by domain experts, e.g., by looking at the distribution of cells in the representation space and assessing whether it shows some promising structure such as clusters.

As a first step towards an automated way to tune parameters in a dataset-specific way, we now propose a simple quantitative and objective heuristic to tune parameters, which we call the *silhouette heuristic*, and evaluate its performance on our benchmark. The silhouette heuristic measures how well a distribution of cells in the representation space looks like a possible clustering of distinct cell types. Given a set of cells in a representation space, typically a dataset of cells processed by a DR method with some parameter values, the silhouette heuristic first runs a $k$-means clustering algorithm on the cells in the representation space, and then computes the silhouette score of the dataset with respect to the cell types assigned by the $k$-means clustering. In particular, if the $k$-means clustering identifies the true cell types, then the silhouette heuristic boils down to the silhouette score with respect to the true cell types. To tune parameters for a DR method on a dataset, we then just compute the silhouette heuristic over a grid of candidate parameter choices, and select the values that maximize it. Here we chose $k$ to be the true number of cell populations, which we already know in advance. In real applications that number may not be known and has to be estimated with prior knowledge or other heuristics. Thus our heuristic here only works if the practitioner already knows the true number of populations.

Figure 3.4 shows the performance all methods on all datasets when parameters are tuned by maximizing the silhouette heuristic, and Table 3.2 summarizes the mean performance across datasets.

We can see that the silhouette heuristic works very well for ZinbWave, where it always identifies parameters equal or very close to the best ones in terms of silhouette, and is comparable to the default parameters in terms of AMI. It also works well for all methods for simple

datasets like sc_10x, sc_10x_5cl and sc_celseq2_5cl, where it also identifies parameters equal or close to the best ones for the silhouette score. As shown by the good AMI performance, these are cases where the initial $k$-means clustering recovers the correct clustering with good accuracy. However, there are also cases of surprising failures on easy datasets, for example for DCA on TabulaMuris, where the parameter set selected by the silhouette heuristic has a very bad AMI and silhouette with respect to the true labels, probably because the initial clustering selected by the silhouette heuristic completely fails to identify the cell types but nevertheless leads to a good empirical silhouette, while simply using the default parameters gives an almost perfect clustering in terms of AMI and a decent silhouette. On the more challenging immune cell datasets, on the other hand, the silhouette heuristic does not seem to be useful (except for ZinbWave). It leads to worse parameters than the default ones for all methods but ZinbWave on the difficult Zhengmix8uneq and Zhengmix5eq datasets, except for scran on the later one. For the easier Zhengmix4eq and Zhengmix4uneq, it leads to better parameters for all methods but scVI. In summary, we see that automatically tuning parameters to try to increase the silhouette using the silhouette heuristic only works well on relatively simple cases, up to possible dramatic errors, but on more challenging situations where there is no clear separation between cell types then it can lead to disastrous choices by overfitting a bad initial clustering. ZinbWave is an exception where, in our benchmark, the silhouette heuristic gives consistently good results. Proposing other heuristics that really help tune parameters is an important open challenge.

## 3.3 Discussion

In this study we have systematically compared the performances of five representative and popular DR methods over ten datasets with known experimental ground truth, representing various levels of biological complexity. Importantly, we have extensively investigated how the choice of parameters for these methods influence their performances, and discussed various ways to properly tune parameters. This can inform practitioners about both the capacity of these methods, as well as on the amount of work required to properly tune them.

When properly tuned, we did not observe huge differences in performance between the methods, particularly in terms of AMI. Both PCA-based methods (scran and Seurat) are nevertheless outperformed by ZinbWave and both autoencoder-based methods (DCA and scVI), particularly in terms of silhouette. On the other hand, we also found that autoencoder-based methods have more parameters that require careful tuning than ZinbWave and PCA-based methods. We illustrated with the silhouette heuristic that automatic parameter tuning is not always easy when the true cell types are not known, and can lead to disastrous results. Interestingly, a similar conclusion was reached by [80], who highlighted the impact of parameters choices in a variational autoencoder-based model, and the need to tune them.

We benchmarked DR methods using downstream analysis-agnostic metrics, mostly for simplicity and because of a lack of ground truth, other than simulations, for tasks such as trajectory inference. It would be interesting to investigate in further studies how well our metrics translate to downstream applications: in particular which metric out of the AMI and silhouette correlates best with downstream performances.

An interesting result of our study is the large drop in performance, for all methods, on the immune cell mix data compared to the cell lines. This shows that good performances in

the later does not necessarily translate to good performances in the former. In particular the performances on Zhengmix8eq and Zhengmix8uneq showed that the methods failed to properly separate the various T-cells populations, probably due to their relative similarity compared to the other cell populations present in the datasets. Being able to separate similar populations is of utmost relevance when investigating, for example, early tumor development, where the tumor cell population still displays a transcriptome very similar to that of cells of the organ of origin. For example in the case of triple negative breast cancer, in which tumor cells originate from normal luminal cell populations, it is crucial to be able to distinguish the various states the tumor cells undergoes towards full transformation, in order to properly target these cells at an early stage of the disease. Our study shows that efforts are still needed to develop methods able to robustly discover cell populations in complex mixtures.

## 3.4 Material and methods

### 3.4.1 Datasets

The four cell lines datasets come from [167] and were downloaded from https://github.com/LuyiTian/ the TabulaMuris dataset is an in silico mixture containing all the cells from four tissues sequenced with Smart-Seq2 from the Tabula Muris consortium [171] which was downloaded with the TabulaMurisData Bioconductor package. Zhengmix4eq, Zhengmix4uneq, and Zhengmix8eq come from [166] and was downloaded from the DuoClustering2018 Bioconductor package, and we generated Zhengmix5eq and Zhengmix8uneq following the same procedure inorder to have more complex datasets.

All ten datasets were subject to the same quality control pipeline, using scater [168]. We removed cells three median absolute deviations (MAD) under the mean in counts and expressed genes, as well as those three MAD above the mean in percentage of mitochondrial reads.

### 3.4.2 Performance metrics

Given a set of cells with given ground truth labels, we consider two metrics to measure how well a mapping of those cells in a representation space fits the ground truth labels.

The first metric is the *silhouette*, defined as the average over all cells of each cell silhouette. The silhouette of a given cell $x$ is defined as

$$\text{silhouette}(x) = \frac{b(x) - a(x)}{\max(a(x), b(x))},$$

where $a(x)$ is the average Euclidean distance between $x$ and the other cells of the same class, and $b(x)$ is the average Euclidean Euclidean distance between $x$ and cells in the closest different class. We used the implementation of scikit-learn [172].

The second metric is the AMI [173], which measures how well the ground truth clustering matches the clustering found by a $k$-means algorithm in the representation space. The AMI is formally defined as:

$$\text{AMI} = \frac{\text{MI} - \mathbb{E}\left[\text{MI}\right]}{\text{mean}\left(H(U), H(V)\right) - \mathbb{E}\left[\text{MI}\right]},$$

where MI is the mutual information between both clustering $U$ and $V$ and $H$ is the entropy function. We used scikit-learn's implementation with default parameters for both the $k$-means algorithm (setting $k$ equal to the ground truth number of classes), and to compute the AMI.

### 3.4.3 Statistical analysis

To analyze results we used R to perform T-tests and run ANOVA analysis with the AovSum function from the FactoMineR package [174]. All parameter values were turned into factors for the ANOVA the analysis.

To compare the methods presented in Table 3.2, as shown in Figures S13-S14, we used the wilcoxon function from the scipy package [175], with default parameters, except for alternative which was set to "greater" in order to have a one-way test. Note that we only had 10 samples, which is small for that test.

### 3.4.4 Computational methods

The five DR methods were downloaded from their canonical package manager in June 2019. We followed either the tutorials or vignettes available for each methods to use them. The selection of parameters to tune was based on the arguments of the functions called in these tutorials. Methods dependant on a random seed, DCA and scVI, were run on five seeds and we averaged their metrics in order to reduce the effect of a single good or bad seed.

## Availability of data and materials

The code and data used in this manuscript are available at `https://github.com/google-research/google-research/tree/master/scrna_benchmark`

# Chapter 4

# Best practices for single-cell histone post translation modification analysis

## 4.1 Introduction

Posttranslational modifications (PTM) of histone proteins are important epigenetic events that modulate chromatin structure, nucleosome positioning and transcription. They are involved in numerous biological processes, including DNA repair [176], development [177, 178] and cancer [179]. With the recent advent of high-throughput technologies to measure histone PTM at the single-cell level (scHPTM), such as single-cell chromatin immunoprecipitation followed by sequencing (scChIP-seq) [46] and single-cell cleavage under targets and tagmentation (scCUT&Tag) [49], it is now feasible to explore the diversity of histone PTM in complex biological samples with an ever-increasing level of details [47, 9, 71, 180]. ScHPTM has already allowed new biological insights such as epigenetic factors involved in cancer response to chemotherapy [45], and is likely to be relevant for years to come.

While scHPTM has great potential, it is also a relatively recent approach which comes with numerous challenges that need to be addressed in order to fully deliver its promise of capturing biologically relevant information from raw experimental data. In this work, we leave aside the question of which technology to use to generate scHPTM data, and focus instead on two important questions for practitioners, namely, 1) how to design experiments, in particular to choose a good trade-off between number of cells and coverage, and 2) how to computationally analyze the raw experimental data and transform them in biologically relevant representations, where subsequent analysis such as cell classification or lineage inference become feasible. While both questions have been investigated through systematic benchmarks and comparisons for more mature single-cell technologies such as single-cell RNA-seq (scRNA-seq) and single-cell sequencing assay for transposase-accessible chromatin (scATAC-seq) [96, 81, 167, 165, 108], we are not aware of any similar study conducted for the burgeoning field of scHPTM, leaving experimentalists without rational guidelines on how to design their scHPTM experiments and analyze the data they produce.

Given the similar nature of raw experimental data between scHPTM and scATAC-seq, namely, sequencing reads capturing an epigenomic signal distributed in specific regions over the whole genome, it would seem natural to use the same computational methods to analyze

scHPTM and scATAC-seq data. However, both modalities differ in many aspects. First, the actual distribution of reads can be drastically different between scHPTM and ATAC-seq. Indeed ATAC-seq reads are known to cluster in relatively small, ∼1k base pairs (kbp), regions [181], whereas the regulatory regions for scHPTM vary much more widely in size (e.g., between 5kbp and 2000kbp for H3K27me3 [181]) and their locations can vary depending on the histone mark - from enhancers (H3K27ac) to gene body (H3K36me3) or intergenic regions (H3K27me3). Second, with current technologies, the number of sequenced reads in scHPTM is generally between a few hundred and a few thousand per cells, compared to several thousands for scATAC-seq and tens of thousands for scRNA-seq. Such a low coverage leads to only about 1% of the expected enriched regions to contain at least one read per cell (compared to 1-10% for scATAC-seq and 10-45% for scRNA-seq [96]). Thus one can not assume that what is true for scATAC-seq or RNA-seq holds for scHPTM.

To start filling this gap, we perform in this chapter a large-scale computational study to evaluate the impact and best choices for the number of cells, coverage per cell, cell selection, matrix construction algorithm, feature selection and dimension reduction algorithm. To quantify the impact of each of these factors, we use two single-cell multi-omics datasets where, in addition to scHPTM, a second modality is measured for each cell (gene expression or cell surface proteins); we then assess how well the cell-to-cell similarity induced by scHPTM data analysis agrees with the one induced by the co-assay [64, 72]. The analysis of more than 10.000 computational experiments allows us to clarify the impact of different experimental choices and data processing factors for scHTPM data, and suggest practical guidelines. We found that LSI-based methods performed the best amongst the current existing pipelines, that their performances plateau around 6.000 cells, that feature selection generally degrades their performances, and that building the matrices with binsizes in the order of 100kbp is generally beneficial.

# Results

### 4.1.1 Benchmarking methods for scHPTM analysis

Irrespective of the technology used, most protocols for scHPTM analysis produce sequencing reads which, after being mapped to a reference genome, indicate where on the genome a given PTM mark is likely to be present in each individual cell under study. A number of computational steps are then applied to transform these raw data into a useful representation of each individual cell, where downstream applications such as cell classification or trajectory inference are performed. Here we focus on computational frameworks that produce a representation of each cell as a vector of moderate dimension (typically, 10 to 50 dimensions), which has been found to be a powerful approach for scRNA-seq data analysis [53] and is currently the *de facto* standard for scATAC-seq and scHPTM as well [53]. Going from the mapped read to a vector representation for each cell involves a number a steps that we investigate in this study (Figure 4.1A), including 1) the binning of the mapped reads into regions in order to create a cell×region count matrix to summarize the raw data, 2) various quality control (QC) preprocessing operations to filter out low-quality cells and regions, and 3) an embedding method to build the representation of each cell from the preprocessed count matrix. Each step can be performed in many different ways, and we propose a benchmark to assess the impact of each choice at each

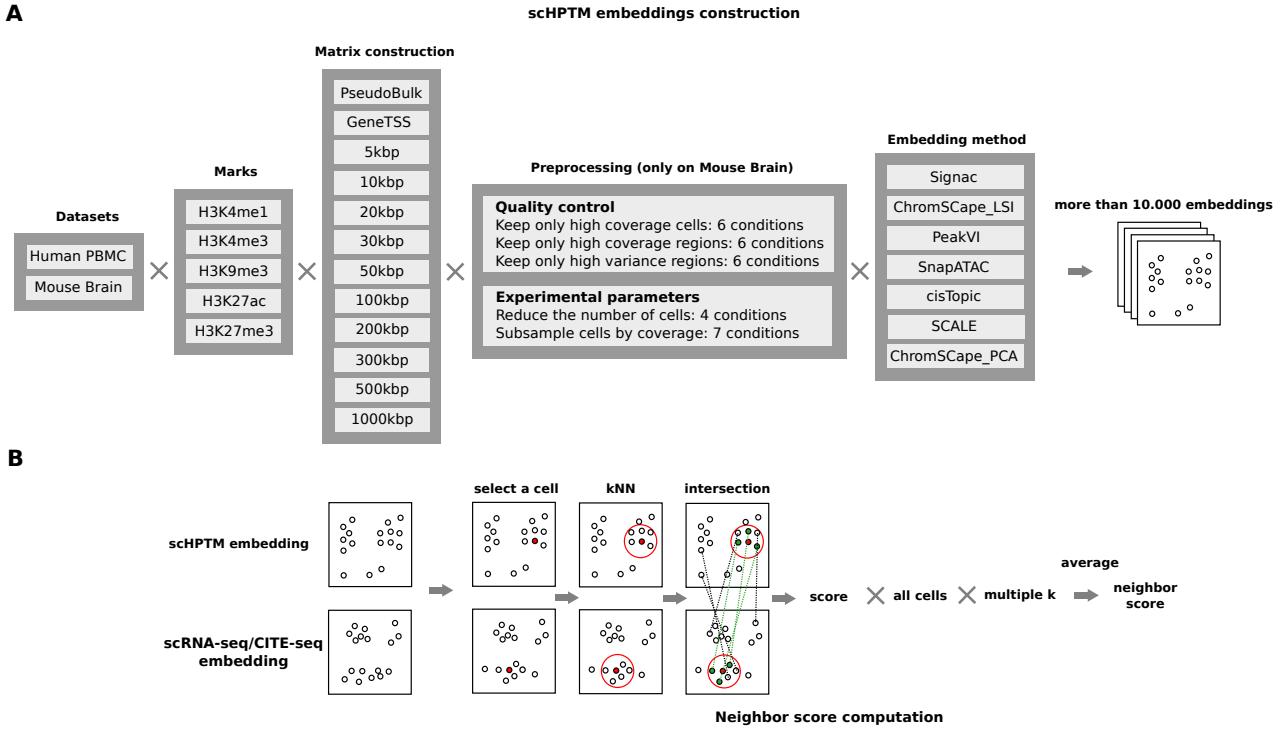


Figure 4.1: Overview of the evaluation protocol. **A.** We build the count matrix using different bin sizes as well as a GeneTSS annotation and peaks called on the pseudo bulk (only for the human PBMC dataset). We then simulate in-silico different experimental conditions for studying the role of the number of cells in a dataset, and the effect of the coverage per cell, as well as different feature selection strategies. Afterwards we run 7 different dimension reduction methods to obtain the cell representations. **B.** In order to compute the neighbor score, we start by selecting a cell, we then build the $k$NN graph for a value of $k$ (5 in the figure), we then compute the size of the intersection between the neighborhood of the cell in the two embeddings (3 cells in the figure) and divide it by $k$ to obtain the score for one cell and one value of k (score of 0.6 in the figure). We then compute and average this score over all the cells, to have an neighbor score for a given value of $k$, that score is then further averaged over different values of $k$ (0.1%, 0.3%, 0.5%, 1%, 3%, 5% and 10% of the number of cells in the experience) to obtain the final neighbor score

step on the final cell representation (Figure 4.1).

In order to evaluate the impact of each decision on the quality of the final representation, we need a way to quantify the quality of that representation. For that purpose, we rely on two datasets produced with multiomics co-assays (Table 4.1), where two modalities are measured simultaneously in each cell. More precisely, we consider a mouse brain dataset from [71] where five histone marks (H3K4m1, H3K4me3, H3K9me3, H3K27ac, and H3K27me3) are assessed by scHPTM jointly with scRNA-seq-based gene expression, and a human peripheral blood mononuclear cell (PBMC) dataset from [180] where the same five histone marks are assessed by scHPTM jointly with CITE-seq-based cell surface proteins. For both datasets, we use a unique representation of the second modality (respectively, scRNA-seq and CITE-seq) using

a well-established method as a reference, and compare each representation obtained from the scHPTM data to that reference using a neighbor score that assesses to what extent similar cells in the scHPTM representation are similar in the reference representation of the second modality. The neighbor score varies between 0 when both representations disagree completely to 1 when both representations are identical (see Methods and Figure 4.1B). This evaluation has been previously used in [64, 72] and is currently the standard for evaluating modality alignment tasks in recent community benchmarks such as https://openproblems.bio/.

| Tissue | Source | Co-assay | Mark | Number of cells |
|---|---|---|---|---|
| Mouse brain | [71] | RNA-seq | H3K4me1 | 12,962 |
| | | | H3K4me3 | 7,465 |
| | | | H3K9me3 | 12,044 |
| | | | H3K27ac | 11,749 |
| | | | H3K27me3 | 6,534 |
| Human PBMC | [180] | CITE-seq | H3K4me1 | 12,770 |
| | | | H3K4me3 | 10,386 |
| | | | H3K9me3 | 8,304 |
| | | | H3K27ac | 15,609 |
| | | | H3K27me3 | 8,232 |

Table 4.1: Description of the co-assay datasets used for this study

.

For each dataset and each histone PTM mark, we systematically vary the choices that we can make in each step of the computational pipeline that goes from the mapped reads to the scHPTM representation of each cell, and measure the quality of the final representation with the neighbor score to assess the impact of the choices.

More precisely, for the first step that bins mapped reads to regions in order to build a first cell×region count matrix, we consider three different strategies that represent the various approaches used in practice for the analysis of epigenetic assays: 1) discretizing the whole genome into "bins" of fixed size, and trying different sizes following a logarithmic progression between 5kbp and 1000kbp, 2) counting the reads into bins based on genes and transcription start sites annotations (GeneTSS), 3) counting the reads sequenced in the peaks identified from the corresponding pseudo bulk using MACS2 [75] (PseudoBulk), which we only do with the human PBMC dataset that is distributed in a format that allows us to build the pseudo bulk and use it for peak calling. With these matrices, we attempt different feature selection approaches to select only a subset of regions to keep: 1) selection of highly variable regions using Seurat's [182] FindVariableFeatures function (variable features), 2) selection of regions with high coverage (top features). The first feature selection method is the current standard in scRNA-seq, and the second approach is the recommended one in Signac [183] for analyzing scATAC-seq. We further study the role of filtering cells based on their coverage, which is part of the standard analysis steps. For filtering strategy, we test six different fractions of regions or cells filtered. We also simulate different experimental conditions *in silico* in order to evaluate the role of the number of cells in an experiment, as well as the importance of their coverage.

Finally, we consider seven popular methods for analyzing the count matrices: cisTopic [66], Signac [183] SnapATAC [184], PeakVI [64], SCALE [65], and ChromSCape [94] with TF-IDF (ChromSCape_LSI) and count per million (CPM) normalization (ChromSCape_PCA).

This leads us to test 11970 (out of which 11080 ran successfully, failures were generally due to memory issues on small bin sizes and GeneTSS annotation, a precise percentage of successful runs can be found in Table S13-S14) combinations of mark, dimension reduction method, matrix construction, cell selection, feature selection, number of cells and coverage conditions. We then analyze the impact of each decision choice and experimental condition by assessing statistically how the neighbor score of the representation varies with the decision.

### 4.1.2 LSI based methods outperform other methods on Mouse brain data

We first focus on the influence of the embedding methods on the quality of the final representation. The seven methods we selected implement a broad range of algorithms that are currently used for the analysis of scATAC-seq and scHPTM data. More precisely, ChromSCape_PCA is a simple use of PCA after count per million (CPM) normalization, which serves as baseline. ChromSCape_LSI and Signac implement two variants of the latent semantic indexing (LSI) algorithm, which consists in transforming the count matrix with TF-IDF and applying PCA on that matrix. They have been used to analyze scHPTM data [94, 180], and differ in the fact that ChromSCape_LSI weights the principal components by their eigenvalues, as is standard to do with PCA, while Signac does not and instead whitens the data representation. They implement variants of the algorithm used in Cusanovich2018 [185, 93, 186], which was found with SnapATAC and cisTopic to be among the best methods for scATAC-seq data analysis in [96]. SnapATAC computes the Jaccard similarity between all the cells, and runs kernel PCA on this similarity matrix. cisTopic binarizes the count matrix and then applied latent Dirichlet allocation (LDA) on this modified matrix. Finaly SCALE and PeakVI both implement a variational autoencoder (VAE) with a product of Bernoulli likelihood function. They differ in the fact that SCALE uses a mixture of gaussian prior where PeakVI uses a unimodal gaussian prior. Furthermore PeakVI computes corrections for the size factor of each cell as well as for the accessibility of each DNA region. We run all methods with their default parameters. In particular, we chose to keep the default number of dimensions for all methods, this choice was made because some methods offer their own heuristics for deciding the number of dimensions, and we did not want to disadvantaging them by using a dimension they do not consider optimal. Indeed, PeakVI by default uses the square root of the square root of the number of regions, and cisTopic trains model for multiple dimensions, and chooses one based on an elbow rule of its ELBO. Signac uses a dimension of 50 by default, SnapATAC, SCALE, and ChromSCape have a default dimension of 10.

Figures 4.2A and S15 summarize the performance of each embedding method on the different histone PTM marks in the mouse brain and human PBMC datasets, respectively. In those plots, we summarize the performance of each embedding method by reporting the best performance achieved by each embedding method across all possible matrix construction choices, without performing any additional QC processing such as cell or feature selection. This allows us to quantify the best possible result that each embedding method can reach without setting

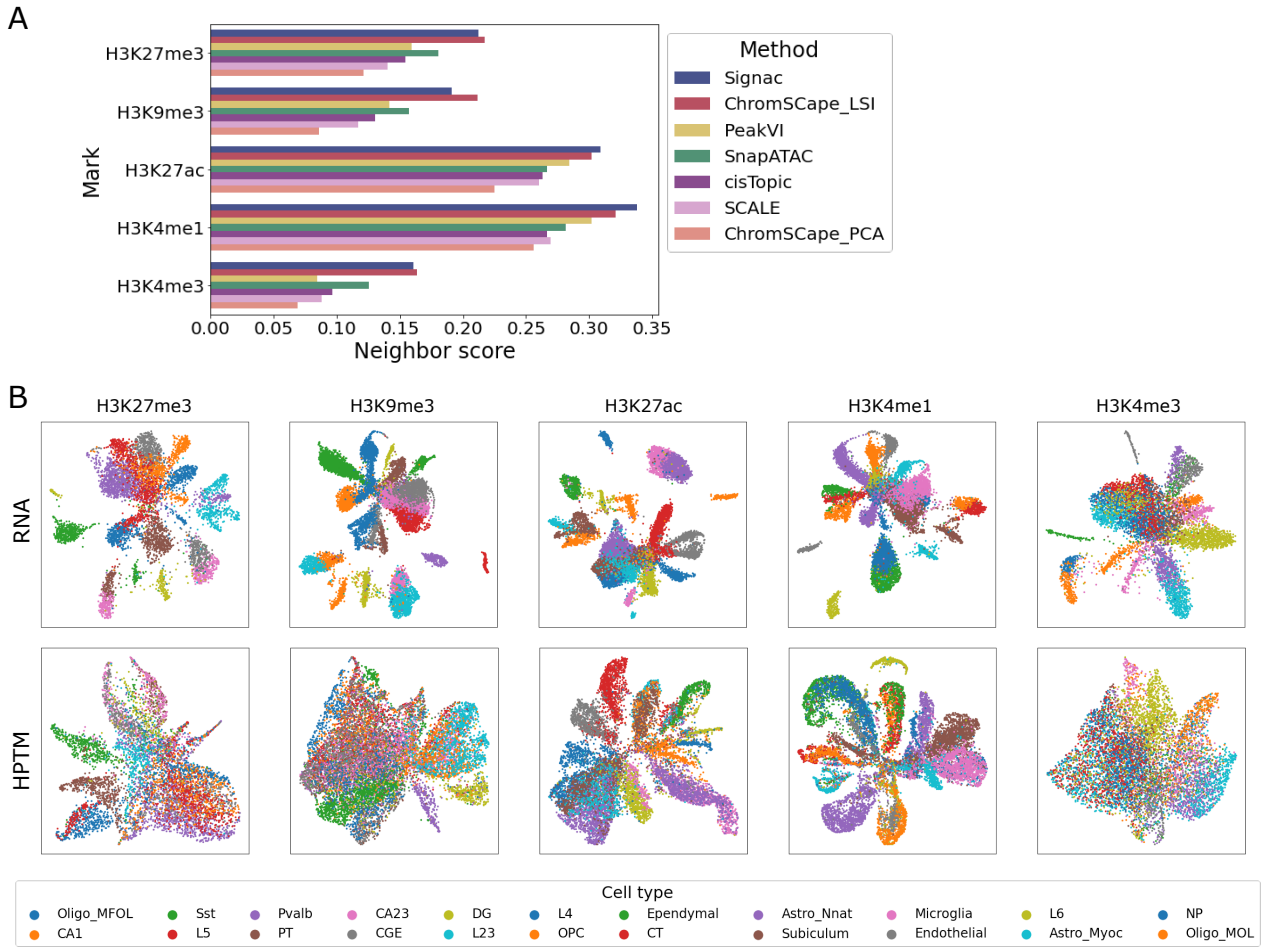


Figure 4.2: **A.** Best performances of the different representation methods on the mouse brain dataset. **B.** UMAP representation of the different samples in the mouse brain dataset, the first row is the RNA co-assay processed with PCA using the scanpy best practices, the second row is the scHPTM assay processed with ChromSCape_LSI using the matrix construction algorithm with the best neighbor score, both colored by the labels of [71] obtained from the scRNA-seq co-assays.

an arbitrary feature engineering pipeline that could advantage some methods over others. We see that the neighbor scores vary roughly in the range 0.05∼0.35 across methods, datasets and marks. As can be seen in Figures 4.2B, where we visualize the embeddings obtained by ChromSCape_LSI on different marks on the mouse brain dataset, this corresponds to a fairly good agreement with scRNA-seq embedding in terms of recovering major cell types, particulary for H3K27ac (score=0.302) and H3K4me1 (score=0.321). Interestingly, we observe differences in the neighbor scores of different marks across methods in the mouse brain dataset, with H3K4me1 and H3K27ac (score=$0.291 \pm 0.028$ and $0.273 \pm 0.026$, respectively) significantly (p=0.008, see Table S16 for all pairwise comparison p-values) higher than H3K9me3 and H3K27me3, and H3K4me3 (score=$0.148 \pm 0.040$, $0.169 \pm 0.033$ and $0.112 \pm 0.035$, respectively). Note that this does not necessarily mean that some marks are more informative than others in general, but

rather than they are less directly linked to expression than others. A similar trend is visible but weaker on the human PBMC dataset (Figure S15), where in particular the scores on H3K27ac and H3K4me1 are lower than on the mouse brain dataset (scores=$0.113 \pm 0.031$ and $0.150 \pm 0.021$, respectively), and only H3K4me1 has a significantly higher scores ($p<0.05$) than the other marks (see Table S17). This difference between the mouse brain and human PBMC datasets could be caused by the differences in co-assay, by the relative complexity of the cell types, or by the quality of the experiments.

The performance of each method on each histone PTM mark of the mouse brain datasets is shown in Figure 4.2 and Table S15. We see that the two best performing methods on the mouse brain datasets are consistently ChromSCape_LSI and Signac, which are significantly better than all other methods (see Table S19 for p-values of pairwise comparisons). They are followed by SnapATAC and PeakVI (except on H3K4me3), then cisTopic, SCALE, and ChromSCape_PCA. SnapATAC is significantly better than cisTopic and SCALE, while ChromSCape_PCA is significantly worse than all other methods. Both top performing methods (ChromSCape_LSI and Signac) implement LSI, suggesting that LSI-based method have an advantage over other approaches. Surprisingly, though, while ChromSCape_LSI also performs well on the human PBMC dataset, Signac does not (Figure S15). This may be due to the lower coverage of the human PBMC dataset than of the mouse brain data, and on the detrimental effect of the whitening operation specific to Signac, as studied in more details in the supplementary text. On the PBMC dataset, ChromSCape_PCA again performs poorly compared to other methods, while the differences between other methods and between marks are overall less pronounced than on the mouse brain dataset.

Since the four methods ChromSCape_PCA, ChromSCape_LSI, Signac and SnapATAC all implement a form of PCA after applying to the count data matrix a specific data transformation, the difference in their performance highlights the importance of this data transformation choice. Simply normalizing the counts by CPM, as ChromSCape_PCA does, leads to poor performances, while normalizing the count data by Jaccard similarity (SnapATAC) or TF-IDF (ChromSCape_LSI and Signac) is consistently better. This seems to be specific to scHTPM, since methods using CPM normalization are competitive with the ones using TF-IDF or kernel PCA on the Jaccard similarity on scATAC-seq data [96].

We also find that cisTopic is not among the best performing methods for the analysis of scHPTM, while it was identified by [96] as one of the best tools for analysing scATAC-seq. On the other hand, LSI is extremely competitive for both modalities. This shows that while scHPTM and scATAC-seq have some similarities, one should be careful before extrapolating good practices from one modality to the other. Finally, the more recent VAE-based methods, PeakVI and SCALE, are overall not competitive with the more classical LSI-based ones. As we show below, this may be due to the relatively small size of the datasets used.

### 4.1.3 The count matrix construction strongly influences the quality of the representation

We now investigate the influence of the count matrix construction method (i.e., how the raw reads are mapped to regions) to obtain relevant embeddings of scHTPM datasets. For that purpose, we explore the performance of the different embedding methods as a function of

the matrix construction parameter, again without further preprocessing such as cell or feature selection. We show the results in Figures 4.3 and S16 for the mouse brain and human PBMC datasets, respectively.

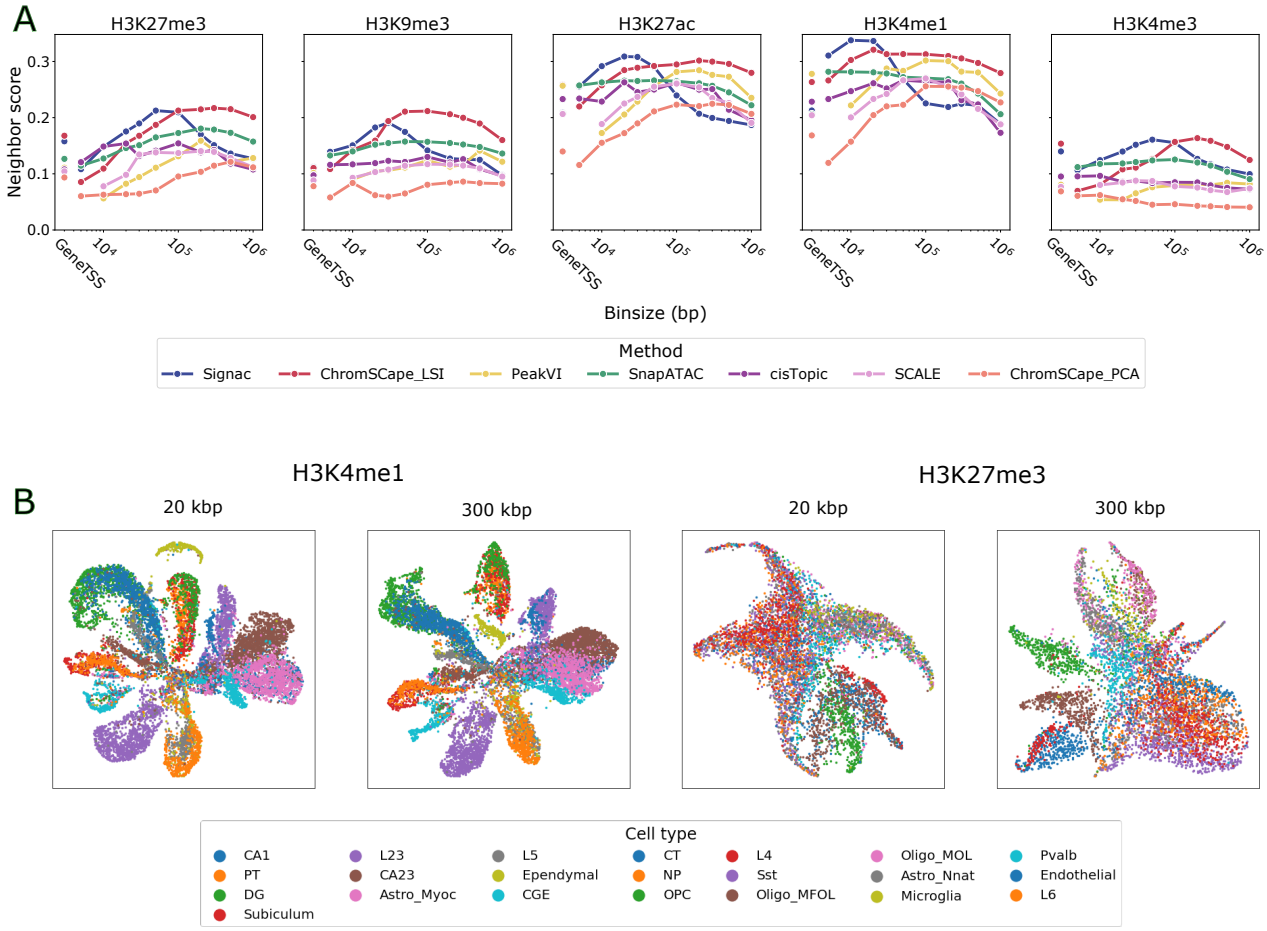


Figure 4.3: **A.** Performances of the 7 dimension reduction algorithms on the 5 marks in the mouse brain dataset, as a function of the matrix construction. **B.** UMAP projecttion of H3K4me1 and H3K27me3 using ChromsSCape_LSI using bins of 20kbp and 300kbp, colored by the labels of [71] obtained from the scRNA-seq co-assays.

We see that matrix construction has overall a strong influence on the quality of the representations. For most methods and marks, the performance first increase when the bin size increases, then decrease after a peak. This effect is visually more pronounced on the mouse brain data, and in particular for repressive marks (H3K27me3 and H3K9me3). In order to quantify this effect, we report the ratios between the best and worst performing matrix construction for each method and mark in Table S22 for the mouse brain dataset and in Table S23 for the human PBMC dataset. In the human PBMC dataset, we can see that the ratio between the best and worst feature engineering can reach up to 7.64 (PeakVI on H3K4me1), this is mostly due to the very poor performances of using a GeneTSS annotation on this dataset as can be seen in Fig S16.

In the mouse brain dataset, we can see that this ratio is on average above 2 for H3K27me3 in Fig S21 and reaches 2.8 in the case of PeakVI. The lowest ratio is 1.2 (ChromSCape_LSI on H3K4me1), which is still an increase in performance of 20%. While that ratio is on average higher for the best performing methods (ChromSCape_LSI and Signac), it is mostly due to the fact that their best performances are higher than the other methods, more than it is due to an extreme sensitivity to matrix construction. Indeed we can see that for all marks, ChromSCape_LSI has a very large range of matrix construction parameters that are extremely competitive. We can also note that by choosing an average performing method (e.g. SnapATAC or PeakVI) and an appropriate matrix construction parameter, we can always beat the best performing methods (ChromSCape_LSI or Signac) if they are run with a suboptimal parameter for matrix construction.

We see on the mouse brain dataset that performances reach a level close to their maximum for smaller binsizes for enhancing marks (H3K27ac, H3K4me1 and H3K4me3) than for repressive marks (H3K27me3 and H3K9me3), and that, except for Signac, the range of appropriate bin size is relatively large (e.g. 50kbp-1000kbp for H3K27me3 or 10kbp-200kbp for H3K4me1). Furthermore, except for Signac, that range is relatively stable across methods for each bin size. We investigate in more details the reason why Signac behaves so distinctively in the supplementary text (Fig 4.8), and show in particular that the fact that it uses a whitening step and a relatively high embedding dimension by default makes it capture more noise than, e.g., ChromSCapte_LSI, for large bin sizes.

It is interesting to note that the choice of using the GeneTSS annotation is usually not competitive compared to using an appropriate binsize. The fact that H3K4me3 is an exception to that rule is consistent with the fact that this mark is known to be particularly enriched around genes and TSSs. We can also see in Fig S16 that the PseudoBulk annotation is also generally not competitive, with a less pronounced effect for H3K4me1 and H3K4me3. This is consistent with the fact that these marks tend to have small peaks, which are easier to identify with peak calling algorithms than larger ones.

It is interesting to note that the range of appropriate bin sizes usually includes 100kbp and can even go up to 500kbp, which would a priori be considered too large to keep biological relevant information. In particular in [47], the authors made the choice of 5kbp for H3K4me3 and 50kbp for H3K27me3. And in [71] the authors made the choice of 5kbp for all marks, except for H3k4me3 for which it was 1kbp. Here we find that, to reach a maximal concordance between epigenomic and transcriptomic embeddings, the appropriate bin size is one, or two, orders of magnitude larger than the ones used in previous studies. The choice of larger bins also reduces the number of dimensions, leading to less memory usage and less sparsity in the matrix which could be beneficial for differential enrichment analysis.

### 4.1.4 Selecting high coverage cells does not strongly influence the performances

In a standard QC pipeline, poorly covered cells can be filtered out before performing dimensionality reduction and subsequent analysis on the highest quality cells. Such selection step often leads to a trade-off between keeping a high number of cells to maximise the discovery rate of rare cell states, and the keeping only highly-covered cells to maximise the quality of

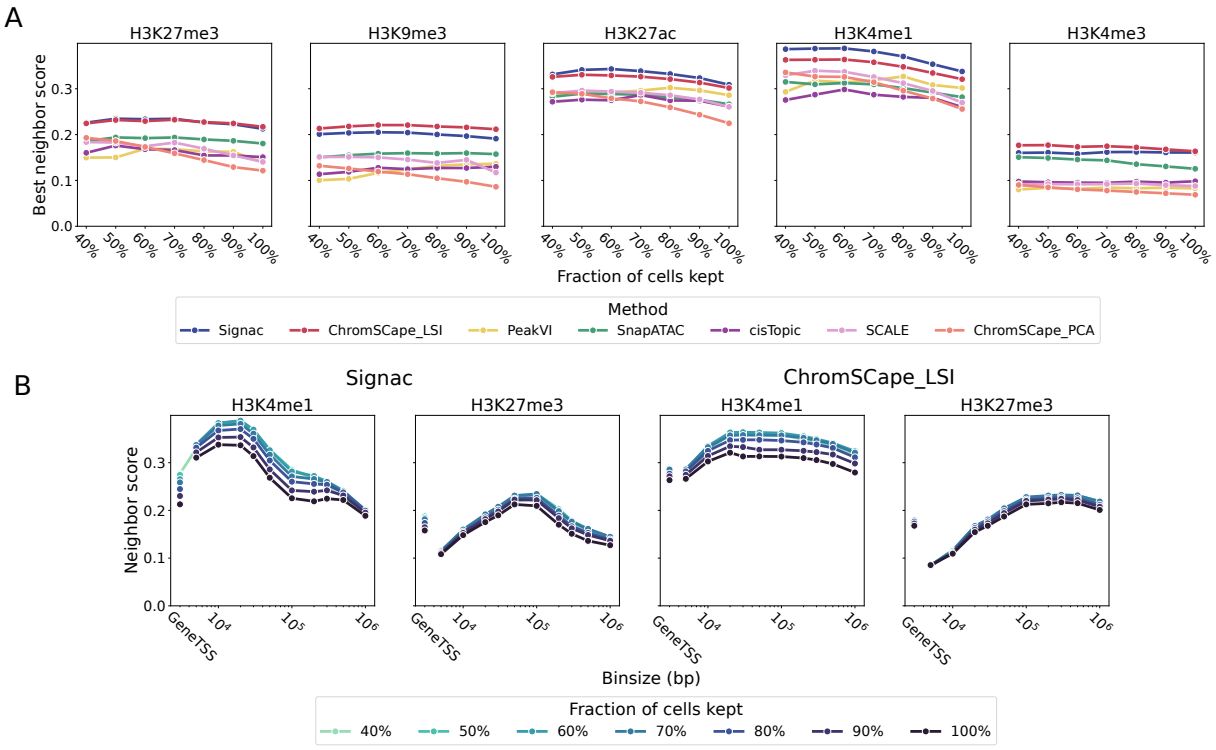


Figure 4.4: **A**. Each point corresponds to the best performance across matrix construction of a given method and a given coverage threshold, for the 7 methods, 5 marks, and 7 coverage conditions. **B**. Performances of Signac and ChromSCape_LSI as a function of matrix construction on H3K4me1 and H3K27me3 for different coverage thresholds.

the embedding. We now assess how selection of cells based on coverage affects the quality of the embedding, by applying different thresholds for coverage selection and measuring neighbor scores across methods.

As shown on Fig 4.4, there is overall a minor gain in performance when applying more stringent QC criteria on cell coverage. Across histone marks, we observe a maximum gain of 15% and 13% in performance for H3K4me1 when using the best performing methods ChromSCape_LSI or Signac respectively (Table S24). Across methods, we observe that the highest gains in performances are observed for the low performing methods identified above Table S25. ChromSCape_PCA and SCALE benefit from a 41% and 21% gain respectively whereas ChromSCape_LSI only benefits from an average 8% gain.

We can also observe that the gains in performances from cell selection have a larger effect on H3K27me3, H3K27ac and H3K4me1 than on the other marks.

### 4.1.5 Feature selection decreases the quality of the embedding

Another QC criteria used in single-cell analysis is the selection of features - genomic regions for single-cell epigenomics datasets - prior to dimensionality reduction. Two standard approaches are (i) the selection of regions with the highest coverage or (ii) the selection of regions that have a highly variable enrichment score across cells. Such a selection step is relatively

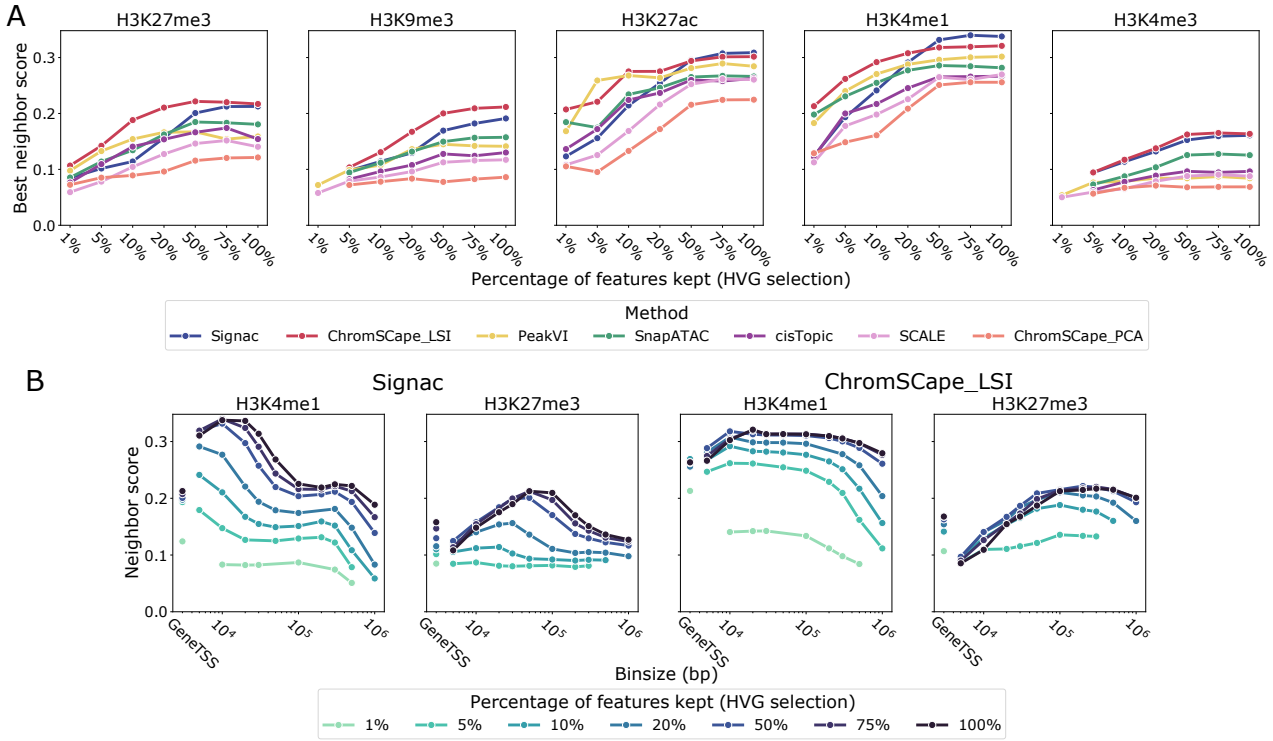


Figure 4.5: Role of feature selection, using the Highly Variable Gene (HVG) method used for scRNA-seq on the mouse brain dataset.**A**. Each point corresponds to the best performance across matrix construction of a given method and a given percentage of features kept, for the 7 methods, 5 marks, and 7 feature selection conditions. **B**. Performances of Signac and ChromSCape_LSI as a function of matrix construction on H3K4me1 and H3K27me3 for different feature selection thresholds.

common, but there is currently no consensus for scHPTM analysis on whether such selection is beneficial, and which of the two methods is optimal.

To address this question, we compare the maximal neighborhood scores for all methods with various feature selection thresholds, when we select features based on variability (HVG) or coverage. The results are shown on Figures 4.5.A and S17 respectively, for the mouse brain dataset. We observe consistently that feature selection is generally detrimental to the performances, in the sense that for both methods, the more regions we keep the better the performances are. As shown on Figure 4.5.B for Signac and ChromSCape_LSI, this trend is in fact not only true when we look at the best performance reached over different bin sizes in the matrix construction step, but also when we look at each bin size individually.

Feature selection has been shown to increase performances for scRNA-seq in [81] and is part of the guidelines for scATAC-seq [183]. Our results show that, contrary to scRNA-seq and scATAC-seq, feature selection is detrimental to the analysis of scHPTM data, and we therefore recommend not to use it.

### 4.1.6 Performances reach a plateau near 6000 cells

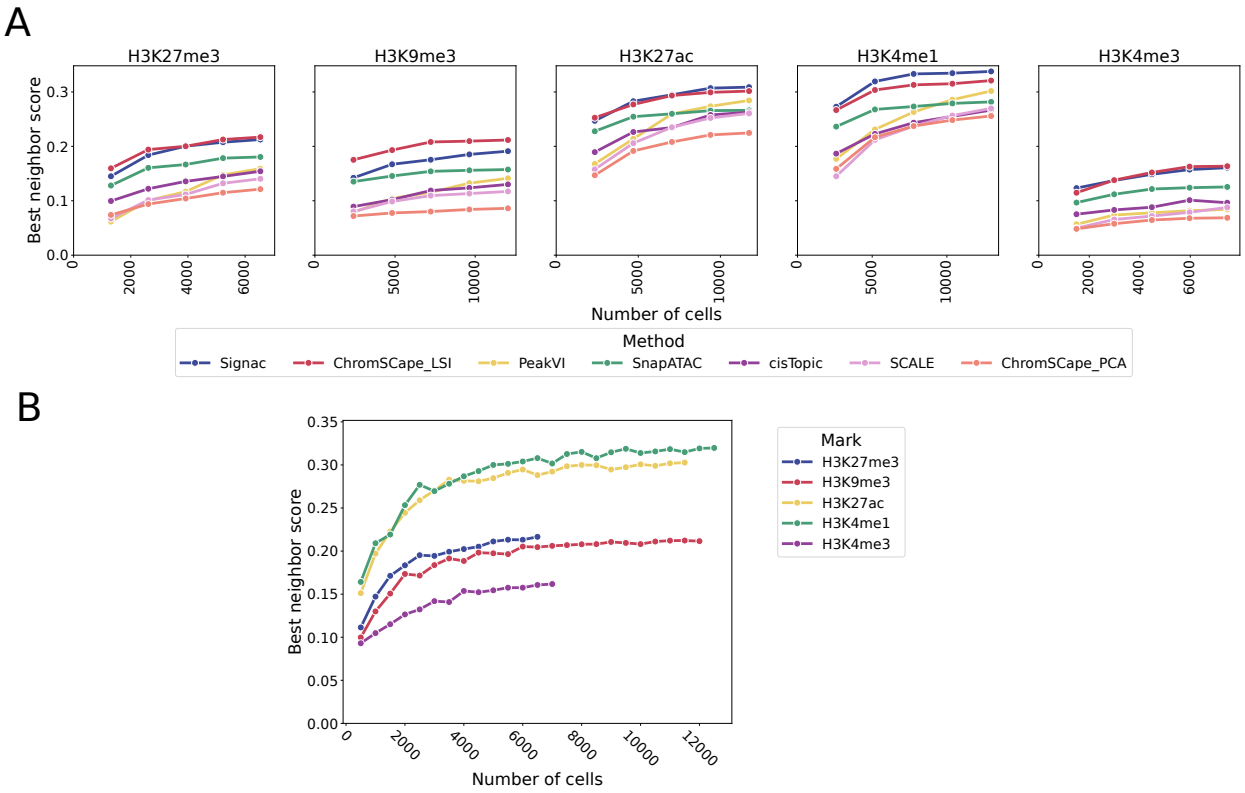


Figure 4.6: Effect of downsampling uniformly at random the number of cells in the experiment. Each point corresponds to the best performance across matrix construction. **A.** Performances the 7 methods, on the 5 marks of the mouse brain dataset and on 5 sizes of dataset (by increase of 20% of the dataset size). **B.** Performances of ChromSCape_LSI on the 5 marks, using an increase of 500 cells per step.

While computational parameters can have an important role in the quality of the representation [81], experimental ones also have a strong influence. In this section we look at the role of the number of cells on such representations, in order to help practitioners design their experiments. For that purpose, we systematically downsample each dataset by randomly selecting a subset of cells of various size, and assess the quality of the representation obtained from the downsampled datasets. We show on Figure 4.6.A the best performance reached across matrix construction for each method on each mark, as a function of the size of the downsampled dataset, for the mouse brain dataset. We further add a finer grained sweep over dataset size for ChromSCape_LSI, by increasing the size of the datset by 500 cells per step as can be seen in Fig 4.6.B.

We see that all methods, on all datasets, benefit from an increase in the number of cells. However it is interesting to note that the benefit from a larger number of cells is diminishing as the number of cells increases. Indeed we can observe that the performances seem to reach a plateau in performance near the 6.000 cells mark. While we can see that the performances

keep on increasing after this plateau, these gains are much smaller than the ones achieved by increasing the cell count before.

PeakVI is an exception to that observation, and we can see that its performances have not yet reached this plateau, see Table S27. This is consistent with the intuition that deep learning based models require a large amount of data to achieve their full performances, and in the datasets used in this chapter, this full performance does not seem to have been achieved. The gains in performances are also quite important, with an average increase of 34% by increasing the number of cells by 150%, and 18% by increasing the number of cells by 66%.

On the other hand the more standard methods, such as LSI or kernel PCA, reach their peak performances around 6.000 cells, and only gain an average of 5% in performances by going from 6000 of to to 10000 cells. Since these methods are the best performing ones in regime tested in this chapter (less than 12.000 cells), it means that practitioners can sequence less cells while keeping relatively good performances. The case of ChromSCape_LSI is shown in more details in Fig 4.6.B, where the dimishing return effect of adding more cells is very pronounced. It also allows us to confirm that the difference in performances between the enhancing and repressive marks is not due to the number of cells present in the datasets, as we see a clear separation of 3 groups: H3K27ac and H3K4me1 having the best performances, H3K9me3 and H3K27me3 following them, and finally H3K4me3 having the worst performances. The plateauing effect can also be seen for all of these marks, leading us to believe that it is not specific to just some marks.

It is however very possible that given more cells (>12.000) PeakVI and SCALE could outperform these methods and lead to better representations. This would be consistent with the behaviour of this family of methods on other modalities such as text of images, even though we can only conjecture that this would happen.

While the increase in the number of cells leads to observable and consistent gains in the quality of the representation, it is noteworthy that these gains have a lower influence than the use of an optimal matrix construction algorithm. It is also important to note that the performances of the current best methods do not strongly benefit from such an increase in the number of cells as can be seen in Table S26, meaning that practitioners may work on relatively small samples while maintaining state of the art performances.

### 4.1.7 Trade-off between coverage and cell number

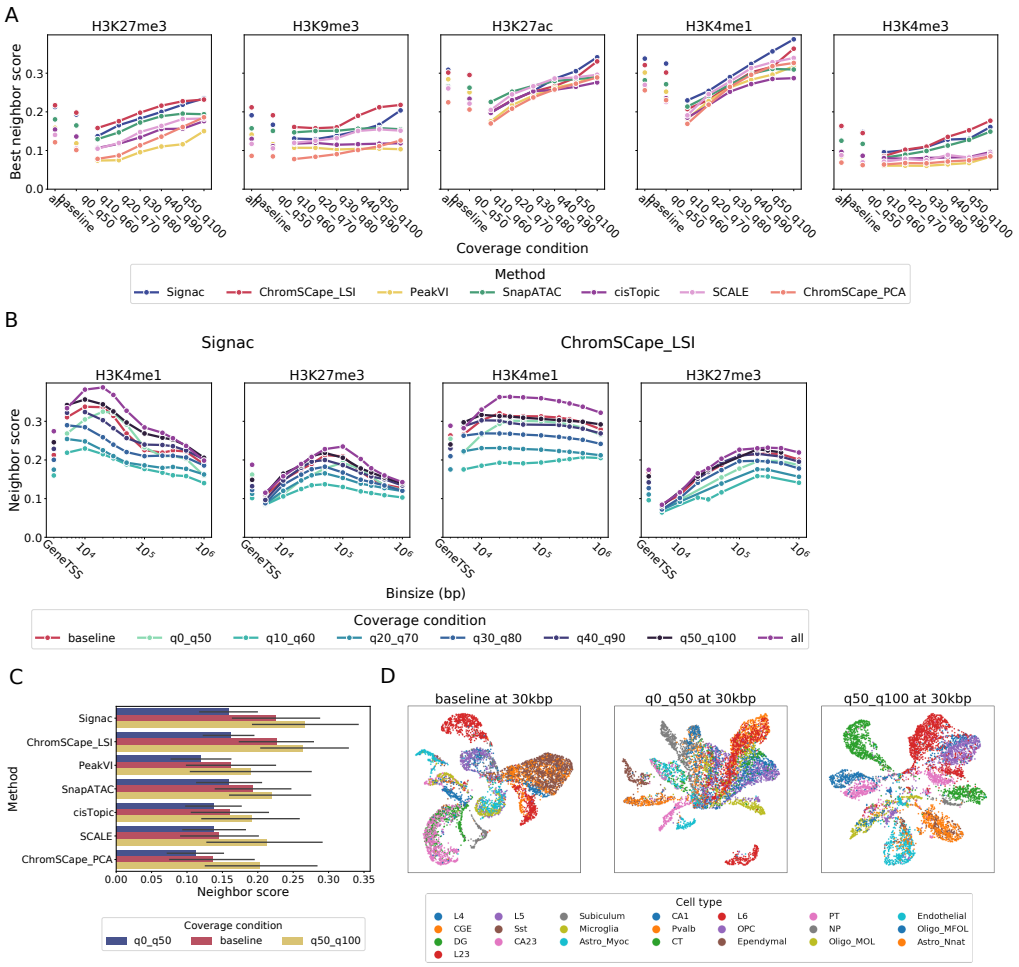


Figure 4.7: Study of the effect of cell coverage on the performances of the representations. The all condition contains all the cell as a reference, the baseline condition contains only 50% of the cells uniformly sampled at random, the other 6 conditions contain 50% of the cells, but are sorted by coverage. We order the cells by how much reads they contain and take all the cells from the bottom n% up to $n + 50\%$ in order to have the same amount of cells in all conditions. **A** Best performance across matrix construction, measured for all of the 7 methods, 5 marks of the mouse brain dataset and 8 coverage conditions. **B**. Performances of Signac and ChromSCape_LSI on H3K4me1 and H3K27me3 as a function of matrix construction. **C**. Average best performance of the 7 methods across all marks, for the lowest covered cells, random cells, and highest covered cells. **D.** UMAP projection of H3K4me1 at different coverage qualities, using ChromSCape_LSI across at 30kbp, colored by the labels of [71] obtained from the scRNA-seq co-assays.

Another important factor that practitioners can influence when designing their experiment is the coverage per cell. This parameter is rather interesting because there is a trade off between

the number of cells in an experiment and the number of reads per cells. In this section we study the role of coverage and how it influences the performances obtained by different computational methods.

In order to evaluate the effect of coverage, we select 50% of the cells, but constrain them to have similar coverage. For example in the q0_q50 condition we take the 50% cells with the lowest coverage per cell, and in q50_q100 we take the cells with the highest coverage. In a more general way, we sort the cells by coverage per cell, and select the cells whose coverage falls between the $n$-th percentile, and $n+50$-th percentile. This allows us to have all conditions with the same amount of cells, and just study the effect of coverage. We also have a condition where we just sample half of the cells at random, including all coverage, in order to have a baseline to compare against. That protocol is summarized in Fig 4.7. This approach of sampling the cells by coverage instead of downsampling the reads per cell has the advantage that it does not make any assumption on the data generation process. Indeed here all the observed cells are real cells, instead of cells that are modified with computational assumptions.

First we can observe that the performances of all methods increase as we increase the coverage, which was expected. However unlike the number of cells in an experiment seen in Fig 4.6, the positive effect of more reads per cells does not plateau and is almost a straight line in the case of H3K4me1 as we can see in Fig 4.7. If we look at the differences in performances between the least and most covered cells, summarized by mark in Table S28 and by method in Table S29, we can see an increase of at least 35%. That increase even goes as far as 107% for H3K4me3 with ChromSCape_LSI. Looking at the difference between the baseline and the high coverage cells, that increase is still in the order of 15%. It is interesting to notice that this effect on performances is larger than the one obtained by an increase in the number of cells, furthermore as we can see in Fig 4.7 this gain is not yet completely achieved in our protocol. That effect being specifically noticeable for H3K27me3. This also agrees with the results on the section studying the role of selecting cells by coverage, where we identified that the marks benefiting the most from this selection were H3K27me3, H3K27ac, and H3K4me1.

The effect of coverage is also consistent across methods, with Signac, ChromSCape_LSI an ChromSCape_PCA benefiting the most form this increase in coverage, above 60%. The first two already being the best performing methods allows to fully reap the benefits from a high coverage.

We thus recommend increasing coverage as much as possible when designing experiments, as it is the experimental step that has the strongest impact on performances that we could identify, and it is reasonable to assume that the current protocols are far from reaching a plateau in performance gains.

## 4.2 Discussion

In this chapter we studied the role of experimental parameters, cell and feature selection, matrix construction, and dimension reduction on the quality of the representation that they lead to. We decided to focus on the quality of the dimension reduction as it is generally the input of most downstream tasks such as clustering, cell type identification, differential enrichment or trajectory inference, a good representation is thus beneficial to all these tasks.

Unlike other benchmarks [96, 81, 165] we decided not to measure the quality of the representation based on the ability of clustering algorithms to retrieve known cell types. This is due to the lack of high quality labels for scHPTM data. One possibility for obtaining labels could be to use the labels derived from the co-assay, however this would rely on computational methods instead of an orthogonal protocol and thus would not allow us to truly measure the quality of the representation. An alternative would be to use FACS-sorted data, whose label are not computational in nature, however no such data exists that presents interesting complexity to our knowledge. FACS-sorted label also suffer from being discrete in nature, which may not be informative to how well we could represent continuous state transition in cell differentiation. The last alternative would be to use simulation data, but not only is there no such simulation tool accepted in the community, performances on simulated data may not be transferable to real data. Instead we decided to evaluate how well the representation in scHPTM locally agrees with a reference co-assay. This approach allows us to be independent of labels, as well as working for continuous cell types. This score makes the assumption that cells that are similar in regulation, measured by scHPTM, should at least locally be similar in expression (or surface proteins in the case of the human PBMC dataset). While we know that this assumption holds better for enhancing marks (e.g. H3K4me1 or H3K27ac), than repressive marks (H3K27me3), it is reasonable and the best we can have without labels. We can also note that this approach has already been successfully used in scATAC-seq in [72, 64].

While we expected the choice of matrix construction algorithm to have an impact, that impact is larger than what we expected a priori. Indeed as is shown in Table S20 the performances using the best binsize can be up to 80% better than using the worst one. We also saw in Fig 4.3 that not all methods can achieve good performances for a large range of binsizes, indeed Signac in particular accepts a smaller range than other methods such as ChromSCape_LSI. Surprisingly the ranges of binsize are larger than what could be expected a priori, and we were also surprised to find that enhancing marks such as H3K4me1 benefited from very large binsizes (up to 200kbp) despite being known to have small peaks (in the order of a few kbps [181]). The fact that GeneTSS and PseudoBulk annotations were in general not competitive is also not sompething that was previously rigorously established in the litteratire to the best of our knowledge.

It was also interesting to note that, except for PeakVI, the performances of the different methods tended to stagnate when increasing the number of cells. This is likely due to the relatively low complexity of the models used. However more complex models such as PeakVI or SCALE did not manage to outperform these low complexity ones. One could imagine that these models could show better performances with larger datasets, such as cell atlases, however they do not seem appropriate for experiments as they are currently designed.

On the other hand, we found that the performances of all methods largely benefited from being run on high coverage cells, and that these performances did not stagnate on the available data. This leads us to recommend increasing the coverage when designing experiments since it seems that there is still room for improving representations this way.

We were also surprised to observe that feature selection, using either a variance or a coverage criteria, almost always had a negative impact on the performances. This may be due to the excessively low coverage per cells compared to other protocols where this procedure can be beneficial (see [81] for scRNA-seq).

To the best of our knowledge, this manuscript provides the first comprehensive study on how to both design the experiment, build the matrix, and analyse scHPTM data. We hope that the large effect of matrix construction that we were able to identify will lead the community to pay more attention to this crucial, if overlooked step. Furthermore by testing the algorithms and pipelines most likely to work on scHPTM data, we hope to save the community some time by avoiding having to discover which already existing algorithms work best.

## 4.3 Material and Methods

### 4.3.1 Matrix construction

We downloaded the mouse brain dataset from GSE152020 (https://www.ncbi.nlm.nih.gov/geo/query/acc.
The data come in count matrix format, with 5kbp bins for all marks except for H3K4me3 which is in 1kbp bins. The larger binsizes were obtained by merging the original bins together to form new bins using a custom script, available at https://github.com/vallotlab/benchmark_scepigenomics. The GeneTSS annotation comes from the ChromSCape package, and the matrix was done by merging the bins containing the regions in that annotation using the same custom script. We keep all the cells present in that matrix, as the original authors already applied QC steps on the cells.

The human PBMC dataset data was downloaded from https://zenodo.org/record/5504061, the data was processed from the fragment files. We used ChromSCape for generating 5kbp matrices and then used our custom script to generate the other matrices similarly to the mouse brain dataset. The PseudoBulk annotation was obtained by turning the fragment files into bams, calling the peaks using MACS2, and then merging them using bedtools. We then select only the cells used in the original paper analysis, by keeping only the barcodes present in the rds objects on zenodo.

### 4.3.2 In silico modifications

Using the matrices generated in the previous section, we modified them in order to both simulate experimental conditions, as well as apply standard bioinformatics steps. Feature selection was done using Seurat's `FindVariableFeatures` for the HVG selection and ChromSCape's `find_top_features` for the top regions selection, ran using our `filter_sce.R` script. For selecting only the high coverage cells we sorted cells by coverage, and kept only the most covered ones, the relevant script is `filter_cells_quality.R`. For studying the role of the number of cells, we sampled cells at ramdom without replacement from the matrice, the relevant script is `sample_cells.R`.

### 4.3.3 scRNA-seq and CITE-seq processing

The scRNA-seq matrix for the mouse brain dataset was processed using the scanpy [187] package, and following their best practice notebook (https://scanpy-tutorials.readthedocs.io/en/latest/pbmc3
We have previously shown in [81] that the algorithms used in that package are robust and per-

form well. The CITE-seq matrix for the human PBMC dataset was extracted from the rds objects and processed with standard PCA.

### 4.3.4 Representations for scHPTM

For computing the representations using the different methods, we used the implementation from the original packages, except for SnapATAC for which we used the reimplementation of [96] as it allowed a nicer API for running a large number of jobs, their implementation can be found on their github https://github.com/pinellolab/scATAC-benchmarking. For cisTopic, we ran the `runWarpLDAModels` method from the cisTopic Bioconductor package (version 0.3.0) and followed the steps from [96]. For Signac we followed the scATAC-seq best practices vignette (https://satijalab.org/signac/articles/pbmc_vignette.html) and used the Signac CRAN package (version 1.7.0). For ChromSCape_LSI and ChromSCape_PCA we processed the matrix with the `tpm_norm` and `TFIDF` methods respectively, then called the `pca` method, and removed the first principal component, all the methods were callled from the ChromSCape Bioconductor package (version 1.6.0). For PeakVI we followed the tutorial on the package website https://docs.scvi-tools.org/en/0.15.1/tutorials/notebooks/PeakVI.html using the scvi-tools (version 0.15.1) [188] pip package. Since SCALE did not have an API for calling their model, we modified the `main.py` script from the `scale` python package (version 1.1.0), so that it does not remove cells.

The scripts for processing used for all R methods ate in the `R_analysis.R` script, PeakVI and SCALE are respectively `peakVI_process.py` and `scale_process.py` scripts.

The R methods were run on CPUs with 4 cores and 32GB of RAM, the deep learning ones (PeakVI and SCALE) V100 GPUs.

### 4.3.5 Neighbor score computation

To compute the neighbor score of an scHPTM representation, we first compute the $k$ nearest neighbor graphs (kNNG) for values of $k$ ranging from 0.1% up to 10% of the cells present in the dataset. We then compute the representation for the second modality using scanpy [187], whose algorithm (PCA) has been identified in [81] as being the most reliable for achieving good representations for scRNA-seq. We then compute kNNG on this second representation, count the number of common neighbors in the kNNG for each cell, divide by $k$, and average over the cells. This gives a score between 0 and 1, where 1 means that the two representations perfectly agree on which cells are similar, and a score of 0 means complete disagreement. We further average that score across the various values of $k$, which were selected to be 0.1%, 0.3%, 0.5%, 1%, 3%, 5% and 10% of the cells contained in the assay, in order to take into account the multiple possible levels of similarity. Two completely random representations would have a score of 0.05 given the values of $k$ that we selected.

# Supplementary material

## Supplementary text

### Role of whitening in LSI

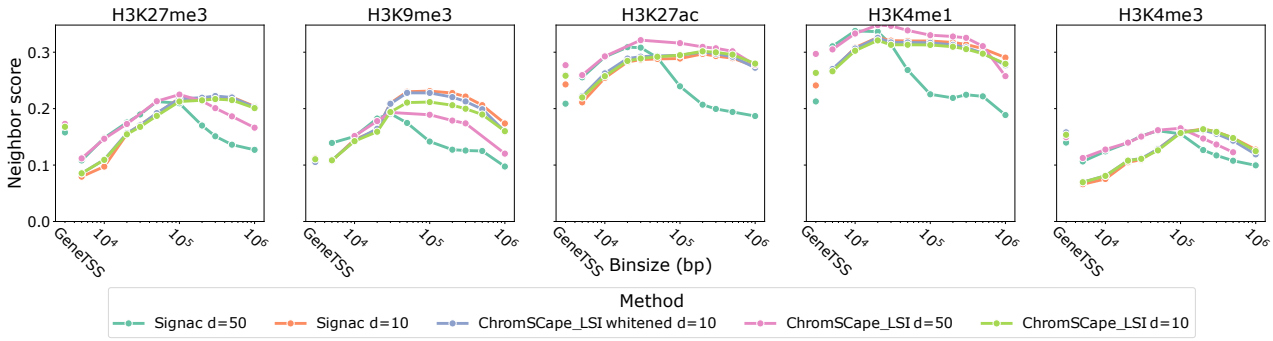


Figure 4.8: Performances of Signac, ChromSCape_LSI, and modified ChromSCape_LSI on the 5 marks in the mouse brain dataset, as a function of the matrix construction

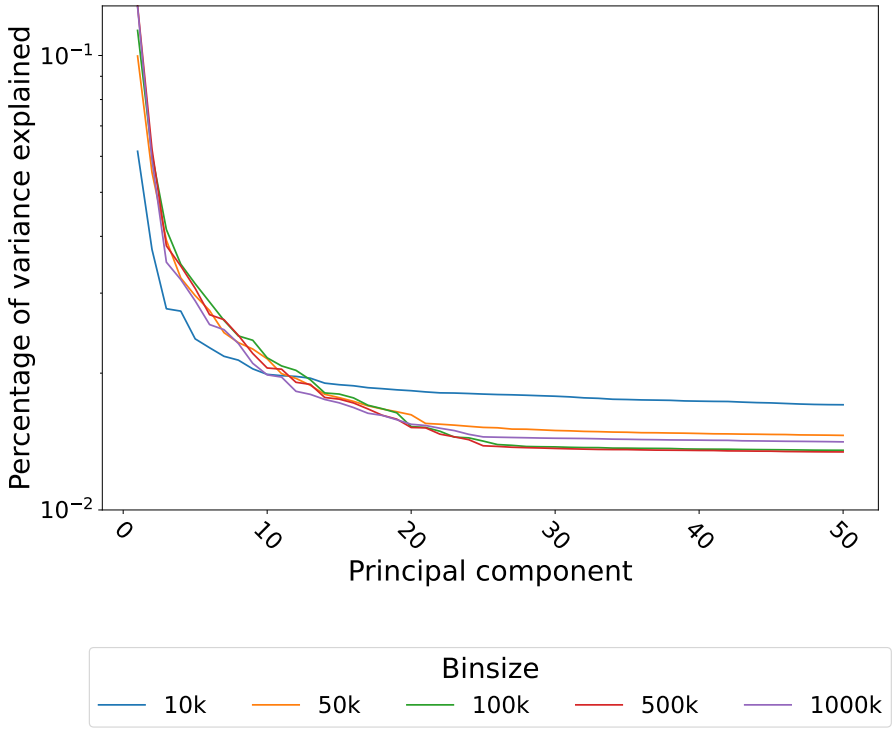


Figure 4.9: Percentage of variance explained by each principal component of LSI on H3K4me1 of the mouse brain dataset.

We could see in Fig 4.3 that Signac's best performances are achieved for a smaller binsize

than the other methods, it is especially surprising because it is supposed to implement the same algorithm as ChromSCape_LSI. In this section we investigate the reason behind this difference.

A close look at the implementation of Signac shows that it does not implement the standard LSI algorithm, but instead a whitened version of it; the principal components (PC) are not weighted by their explained variance. Furthermore the default dimension used is 50 instead of 10 for ChromSCape_LSI. Both methods also remove the first PC, as it is assumed to be mostly driven by the coverage per cell. In order to understand the cause of the shift in optimal bin size we ran Signac with dimension 10, and ran ChromSCape_LSI at dimension 50 and modified it to use whitening.

By comparing the three conditions with a dimension of 10, we can see that the difference in performances between Signac and ChromSCape_LSI in the previous sections can mostly be explained by the number of PCs, indeed except in the case of H3K9me3 the performances of the two are almost identical as can be seen in Fig 4.8.

However we can see that in higher dimensions, 50 PCs, the performances are very different. While the top performances are comparable, with a slight advantage for ChromSCape_LSI, Signac has a much tighter range of binsizes with good performances. This can be explained by the fact that in large dimensions, the later PCs explain less variance than the early ones, and not weighing appropriately induces noise. We can see that when the binsize is small, the explained variance per PC is less concentrated in the first PCs than in the later ones, this is shown in Fig 4.9. This explains why Signac has a tighter range of good binsizes, as its whitening makes the later PCs as important as the first ones.

We can thus recommend not whitening when using LSI, as it makes the method less robust to the choice of binsize.

# Chapter 5

# Single cell peak calling

## 5.1 Introduction

As mentioned before, the issue of feature engineering has a very strong impact on the quality of the representations obtained for single-cell epigenomics. In particular, in current approaches to analyze single-cell epigenomic data like histone modifications protein bindings, the regions of DNA that are taken into account to define bins where reads are counted are specified as input, and the functional ones are identified via differential enrichment analysis after the embedding step. However, the choice of regions provided as inputs remains dependent on a user's prior knowledge, e.g., relying on prior functional analysis of the genome or on peaks detected in bulk experiments, and may not be adapted to discovering novel biology and cell types in single-cell data. In this chapter, I present an attempt to automatically learn these regions from the data, and thus have an algorithm that jointly learns the cell embeddings, as well as a genome annotation. **This work is in progress, and I therefore only present below the main ideas and the model I started to develop, together with preliminary experimental results.**

By considering each cell as a mixture of different profiles (the regulatory topics of *cisTopic* [95]), we can infer the unobserved regulatory state of each cell. Furthermore, since we formulate the profile as a set of enriched locations, we can use them for understanding co regulated parts of the genome. The model we propose also benefits from being extremely parameter efficient, indeed its parameters grow with the number of peaks we consider, instead of the number of bins used to represent the data, thus allowing us to look at a finer resolution than other methods while still being able to avoid overfitting.

This chapter is organised in the following fashion:

- In section 5.2 we introduce a reformulation of the standard feature engineering, as a convolution with a set of masks.
- In sections 5.3 and 5.4, we present the problem of topic modelling, and how it related to single-cell epigenomics, we then introduce the model used in *cisTopic* [95], as we will base our model on it.
- In section 5.5, we present our extension of the LDA model, and how we expect it to be relevant for single-cell epigenomics.

- In sections 5.6 and 5.7, we present how to evaluate the correctness of our genome annotation, as well as the simulation tool we used to run our experiments.
- Finally we present our preliminary results and experiments, as well a future work.

## 5.2 Count matrix as a masking algorithm

The main insight of that project is that all feature engineering approaches count the number of reads in different regions of the genome, what they differ on is on the regions they chose to count into. More formally assuming that we concatenate all the chromosomes in order to have a one dimension distribution:

1. Let $(s_i, e_i)_{i=1..M}$ be the set of the $M$ regions where reads are counted, where $s_i$ is the starting position and $e_i$ the end position of the $i$-th region.
2. Let $d$ be the observed distribution of reads for a cell, where $d(g) = 1$ if there is a read at position $g$ and $d(g) = 0$ otherwise.
3. Let $\mathbb{I}_{[a,b]}$ be the indicator function of $[a, b]$, equal to 1 in that interval, and zero otherwise.
4. For each region do: $x_i = \sum_{g=1}^{G} d(g) \times \mathbb{I}_{[s_i,e_i]}(g)$

This way we obtain a vector representation $x$ of dimension $M$ from the distribution of reads of a cell. All the algorithms used for dimension reduction could be reformulated to work on the raw reads distribution by simply adding this step. In particular for probabilistic models, it is noteworthy that they use this $x$ representation as input, and this is also the representation that they compute the likelihood of.

As we mentioned before, this current formulation suffers from the fact that the existing algorithms are not able to select the regions themselves and have to take them as input. This is due to the fact that this formulation is not easy to optimize as it is not differentiable.

However we could note that the $M$ regions look a lot like the parameters of a mask, and we could probably modify it to be easily optimized. Indeed each mask only has two parameters, its start $s$ and end $e$, which can also be parameterized as a position $\mu$ and a width $\sigma$. If we further modify the mask to be a Gaussian instead of an indicator function, we obtain a smooth, and differentiable, set of masks with just as many parameters, and the following steps.

1. Let $(\mu_i, \sigma_i)_{i=1..M}$ be the set of the $M$ regions where reads are counted.
2. Let $d$ be the observed distribution of reads for a cell, where $d(g) = 1$ if there is a read at position $g$ and $d(g) = 0$ otherwise.
3. For each region do: $x_i = \sum_{g=1}^{G} d(g) \times \mathcal{N}(g; \mu_i, \sigma_i)$

This approach gives us a function that takes into parameters the $(\mu_i, \sigma_i)_{i=1..M}$, the observed reads for a cell, and returns a count vector for that cell. Calling that function on each cell and keeping the same regions, naturally allows us to build a count matrix. This however has the

benefit that this function is differentiable with regard to its parameters, meaning that if the algorithm it is plugged into is also differentiable, the two steps together stay differentiable which paves the way to an optimization of the $(\mu_i, \sigma_i)_{i=1..M}$ parameters by gradient-based continuous optimization of an objective function.

This masking function also has the benefit that its number of parameters is only a function of the number of regions that we want to look at, these regions can be located anywhere on the genome and be of arbitrary size, thus bypassing the main issue of the strategy of using fixed sizes bins. We can think of these Gaussians as peaks that would be identified with a peak caller, but which can be included in an algorithm that jointly learns the representation and the peaks.

Once we thought of this masking function, we realized that it could trivially be extended to express a distribution over the genome by adding weights to each regions, which will be used later.

## 5.3 Topic modelling

Since the current models used in the single-cell epigenomics literature are based on models from the topic modeling literature, we will here briefly present the similarities in modeling. The field of topic modelling is used to analyse a set of textual documents (e.g. newspaper articles) or sentences, and identify which topics (e.g. sports, science, New York City, etc …) are treated in them. The documents are represented as either a list (ordered) or a set (unordered) of words taken from a limited vocabulary. The use of a set is generally called the bag of words representation, as it loses the information of the location of the words in a sentence or a document. In this context the following analogies are done:

- A document (or sentence) corresponds to a cell.
- The vocabulary $\mathbf{V} = \{w_1, ..., w_M\}$ of size M, corresponds to the M regions of interest $\mathbf{R} = \{r_1, ..., r_M\}$.
- Observing the word $w_j$ in a document corresponds to observing a read sequenced in region $r_j$.
- The count matrix $X$ of words observed in each document corresponds to the count matrix $X$ of number of reads sequenced in regions in each cell. This representation is a bag of words one, because there is sense in saying that a read comes before another one.

In this context, using PCA (or equivalently SVD) embedding on the count matrix is referred to as Latent Semantic Indexing (LSI), and is the method used in *Cusanovitch2018* (after TF-IDF transformation), and has been identified by [96] and us [82] to be the best performing method for scATAC-seq and scHPTM respectively.

## 5.4 Background: Latent Dirichlet Allocation

Our model takes inspiration from the Latent Dirichlet Allocation (LDA) [189] used in *cisTopic* [95]. In particular we heavily rely on the notion of topics, and representing a cell as a mixture of topics, we also use a similar optimization procedure as the one presented in [189].

In this section we will introduce the LDA algorithm, using the epigenetics terminology instead of the text documents one, this will serve as an introduction to our model as well as help understand which modifications we made to LDA.

The LDA algorithm is a generative probabilistic model, as presented in Section 1.4.1:

- Observed variables: the $N$ reads in a cell $\mathbf{r}$
- Parameters: the probability of sequencing a read at a given location, if we only sequenced one read from a cell of regulatory topic $k$, the probabilities for each of these topics are $(\beta_k)_{k=1..K}$.
- Unobserved variables: the mixture of topics for a give cell $\theta$, and the regulatory topic from which each read is sampled $(z_n)_{n=1..N}$.

,

Given these observed variables, parameters, and unobserved variables, the generative process is the following.

For each cell $\mathbf{r}$

1. Choose $N \sim \text{Poisson}(\xi)$, the number of reads in the cell.
2. Choose $\theta \sim \text{Dir}(\alpha)$, the mixture of regulatory topics for the cell.
3. For each of the N reads $r_n$:
   (a) Choose a regulatory topic $z_n \sim \text{Multinomial}(\theta)$.
   (b) Choose a read $r_n$ from $p(r_n|z_n, \beta)$, a multinomial probability on the M regions, conditioned on the topic $z_n$

Some simplifying assumptions are made in this model, the number of regulatory topics $K$ is assumed known and fixed. The location probabilities are parameterized by a $K \times M$ matrix $\beta$ where $\beta_{i,j} = p(r_n^j = 1|z^i = 1)$ and $\sum_j \beta_{i,j} = 1$, and the Dirichlet parameter $\alpha \in \mathbb{R}_+^K$ are both treated as fixed quantities to be estimated. The Poisson distribution, which would probably be a log-normal for scATAC-seq and scChIP-seq, is not critical, and actually not even used in the model. This gives us the following equations:

$$p(\theta|\alpha) = \frac{\Gamma(\alpha_0)}{\sum_{k=1}^K \Gamma(\alpha_k)} \prod_{k=1}^K \theta_k^{\alpha_k - 1}, \text{ with } \alpha_0 = \sum_{k=1}^K \alpha_k, \tag{5.1}$$

where $\alpha \in \mathbb{R}_+^*$ and $\Gamma$ is the Gamma function.

Given the parameters $\alpha$ and $\beta$, we get the following joint distribution:

$$p(\theta, \mathbf{r}, \mathbf{z}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z|\theta)p(r_n|z, \beta), \tag{5.2}$$

and by integrating over $\theta$ and summing over $\mathbf{z}$ we get the following marginal distribution:

$$p(\mathbf{r}|\alpha, \beta) = \int p(\theta|\alpha) \left( \prod_{n=1}^{N} \sum_{z} p(z_n|\theta)p(r_n|z_n, \beta) \right) d\theta, \tag{5.3}$$

which is known to be intractable [190].

The key inference problem in that situation is to solve the posterior distribution of the hidden variables given a cell:

$$p(\theta, \mathbf{z}|\mathbf{r}, \alpha, \beta) = \frac{p(\mathbf{r}, \theta, \mathbf{z}|\alpha, \beta)}{p(\mathbf{r}|\alpha, \beta)} \tag{5.4}$$

Since the normalizing factor $p(\mathbf{r}|\alpha, \beta)$ is intractable for exact inference, we need to use an approximate inference strategy, in *cisTopic* the authors chose to use a collapsed Gibbs sampler from [191], however in [189] the choice was to use a variational expectation maximisation (VEM) scheme. Since the posterior $p(\mathbf{r}|\alpha, \beta)$ is not tractable, the solution is to approximate it with variational posterior $q$ taken from family of function called variational family. The variational family selected in [189] is the following:

$$q(\theta, \mathbf{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{5.5}$$

where the Dirichlet parameter $\gamma$ and the multinomial parameters $(\phi_n)_{n=1..N}$ are free variational parameters.

Instead of optimizing the quantity in 5.4, we can optimize for an approximation of the logarithm of that quantity called the expectation lower bound (details of the derivation can be found in [189]).

$$\text{ELBO}(q, \alpha, \beta) = \mathbb{E}_q \left[\log p(\mathbf{r}, \mathbf{z}, \theta|\alpha, \beta)\right] - \mathbb{E}_q \left[\log q(\mathbf{z}, \theta|\mathbf{r})\right] \tag{5.6}$$

where our goal is to find the parameters of $q$, $\beta$, and $\alpha$ that optimize that function. It is important to note that if the variational function $q$ is differentiable with regards to its parameters, the ELBO can be optimised with gradient based methods.

## 5.5 Proposed modification of LDA

We propose to make a modification of this model that would allow it to be run on the raw data, by making it aware of the notion of peaks using the insight about masking from 5.2 which will drastically reduce the number of parameters to fit.

$$\beta_i \propto \sum_{l=1}^{L} \eta_{i,l} \times \mathcal{N}(\mu_l, \sigma_l), \text{ where } \beta_i = [\beta_{i,1}, .., \beta_{i,G}]^T \tag{5.7}$$

which means that each coordinate $\beta_{i,j}$ has the following likelihood:

$$\beta_{i,j} \propto \sum_{l=1}^{L} \eta_{i,l} \mathcal{N}(j|\mu_l, \sigma_l) \text{ with } \eta \in \Delta^L, \mu_l \in [1..G], \text{and } \sigma_l \in \mathbb{R}^+ \tag{5.8}$$

With (5.7) modification, we have done two conceptual modifications:

- All of the $(\beta_i)_{i=1..K}$ are defined by a shared set of $L$ peaks, each of them with their own mean $\mu_l$ (location) and variance $\sigma_l$ (width). The difference between the $(\beta_i)_{i=1..K}$ is due to the contributions of each peak to the topics, expressed by $(\eta_{i,l})_{i=1..G,l=1..L}$.
- Each of the peaks contributes to a local region in the genome, thus making the coordinates of the $(\beta_i)_{i=1..K}$ dependant on each other.

This modification has multiple advantages:

- This reduces the number of parameters of the $(\beta_i)_{i=1..K}$ from $KM$ to $KL$, the number of parameters no longer directly depends on the number of bins, thus allowing us to use bins as small as we want.
- This allows us to identify the peaks directly as the $(\mathcal{N}(\mu_l, \sigma_l))_{l=1..L}$.
- This allows us to reconstruct the $(\beta_i)_{i=1..K}$ as bulk profiles of the different regulatory topics.
- This keeps all the interpretation benefits from *cisTopic* while working directly on the raw data, and learning the peaks at the same time.

There is however one drawback of this formulation, as the inference strategies used in [66] and [189] cannot easily be adapted. We chose to keep the variational inference approach from [189], but chose to use neural networks as the variational family, thus leading to a modified VAE. While the choice of a VAE framework seemed reasonable at the time, the variational family we chose to use caused us a lot of problems, which will be expanded upon later.

This extension of *cisTopic* can be considered an embedding method that also performs the role of peak caller. We can note that this algorithm could as well be applied to cohorts of bulk profiles , such as the ones of ENCODE or TCGA, and learn a relevant genome annotation for them at the same time.

## 5.6 Evaluating a peak caller

Since we were tackling the problem of peaking calling, we had to identify how their accuracy was measured. Understanding how to compare peak callers was especially important because we would have to develop a new metric; indeed, we were the first one to treat the issue of peak calling in the single cell context. According to the evaluation used in Zerone [192] and other peak callers, they can be evaluated with the following strategies in bulk:

- Irreproducible Discovery Rate (IDR) [193]: Which is based on using multiple replicates of the same bulk experiment and evaluating the concordance between the peaks.
- Using known consensus motifs for transcription factor (TF) bindings sites as ground truth, and evaluate if the regions containing these motifs are discretised.
- Using known properties of the mark being evaluated. For example, H3K6me3 is known to bind to transcribed genomic regions, thus using a bulk profile of RNA we look at the variance explained in the RNA counts by the number of reads in the H3K6me3 region identified.
- Zerone defines its own metric as looking at the number of reads located in a discretised regions, as a function of the number of discretised regions. This is based on the idea that the caller should identify regions with strong read support first, and that the more regions it calls, the less support they will have. However this metric is trivially bypassed by calling the whole genome since it will catch all the reads in a single region.

The method based on IDR may be used between cells of known similar type, or between the known bulk of a cell type and the profiles infered for each cell of that type. However, having low sequencing depth may cause a high IDR if most peaks have no reads.

The method based on motifs is not useful in the single-cell scenario (except for detecting false positives), because the motifs are shared by all the cells, we should thus call the same regions for all cells. However, they are presenting this as a simple classification task and measure precision, recall and F1 (ways to account on how to compare a distance between chr1:1-100 and chr1:2-101 are not mentioned).

Using the properties of the studied mark may be interesting, however it relies on having access to two signals for each cells, which to the best of our knowledge is currently not the case for scChIP-seq. It could be done for scATAC-seq and scRNA-seq, but it requires that measuring the two signals has no deleterious effect (i.e. a read with two marks will only contribute to one signal).

Standard pipelines as used by ENCODE are described here.

The evaluation we selected was the formulation as a classification task. This choice was made because if we generate simulation data, it is relatively easy to measure, and is also the most appropriate for the single cell context.

## 5.7 Creation of a simulation tool

As we saw in the previous subsection, evaluating peak calling is not a trivial task, and there are no agreed upon ground truth dataset for benchmarking purposes. Furthermore since our goal is to jointly learn the peaks as well as a representation for the cells, we would need a ground truth for the two tasks.

To do so, we chose to build a simulation tool that would allow us to generate cells of different cell types following the following algorithm:

1. Choose N the number of cells.

2. Choose M the number of reads per cell.

3. Choose K the number of cell types.

4. Choose G the size of the genome (in regions).

5. Choose L the number of peaks.

6. Sample $(\mu_l)_{l=1..L}$ the locations of the peaks, uniformly across the genome.

7. Sample $(\sigma_l)_{l=1..L}$ the widths of the peaks.

8. For each cell type $k$, sample $(\eta_{k,l})_{l=1..L} \in \Delta^L$, thus building $\beta_k$ the probability of observing a read at each location of the genome for that cell type.

9. For each cell:

   (a) Select a cell type $k$.

   (b) Sample M times from $\beta_k$.

This simulation tool can be, and has been, trivially extended to support different distributions for the number of reads per cell, or cells that have continuous cell types.

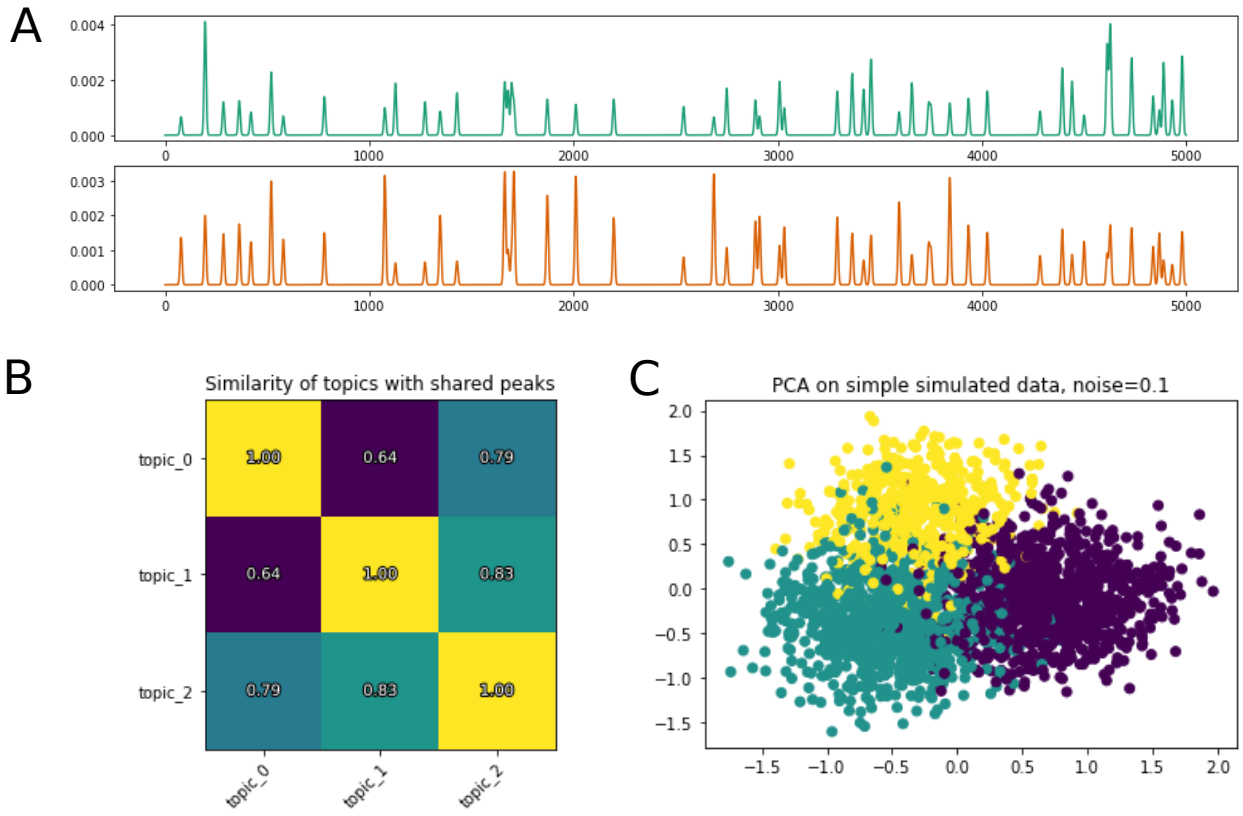


Figure 5.1: **A.** Example distributions of reads over the genome for two different cell types, obtained with our simulation tool. **B.** Example correlation matrix between the underlying distributions obtained for three different cell types, correlations around 0.7 are what we observe empirically for bulk distributions of H3K27me3 on ENCODE. **C.** Example PCA obtained on simple simulated data, we can see that while cell types are distinct, they are not linearly separable.

Having a shared set of peaks across all cell types allows us to simulate data that is not trivially separable. This is also coherent with real experimental data where the cell types differ in the amount of enrichment in their various regions, instead of just having a drastically different set of regions.

With this simulation tool, illustrated in Fig 5.1, we have a ground truth for peaks location, thus allowing us to evaluate the accuracy of the peaks learned by our model; we also have labels for each cell, which are used to evaluate the quality of the representation.

## 5.8 Preliminary results

We implemented the model presented in 5.5 using the JAX [194] framework, as well as three competing baselines: PCA, PeakVI [64] and SCALE [65].

We obtained very encouraging results on very simple simulated data with just 10 peaks (2 regions wide) and 100 regions, by beating all baselines (AMI of 0.7 for our model versus 0.5 for

LSI). This was an important sanity check as we were trying to fit a mixture of 1D gaussians using gradient descent based optimisation procedures, which are known not to always work on non convex optimisation problems. In particular the issue of non convexity leads to the problem that if two gaussians end up at the same location, both of them will usually be stuck at that location. This problem can be solved by having more gaussians in the model than we expect to have in the data, indeed if multiple of them end up merged together, we still have enough to fit the data. The merged gaussians can be cleaned up in post processing, allowing us to maintain an interpretable model. Another technique we used was to initialize the locations of the gaussians using the EM algorithm, this allowed us to have our gaussians start at reasonable locations, and speed up the convergence of the algorithm.

However, the promising results we obtained on toy data failed to translate to more complex data with thousand of peaks. The leading problem we encountered was that while our model seemed to train properly, i.e., its training loss was decreasing as the training was progressing, the quality of its representation suddenly dropped at some point in the training, as illustrated in Fig 5.2. This was rather surprising, as our model actually managed to learn the peaks from the data, with recall as high as 0.98, which meant that our model was indeed learning non trivial properties of the data.

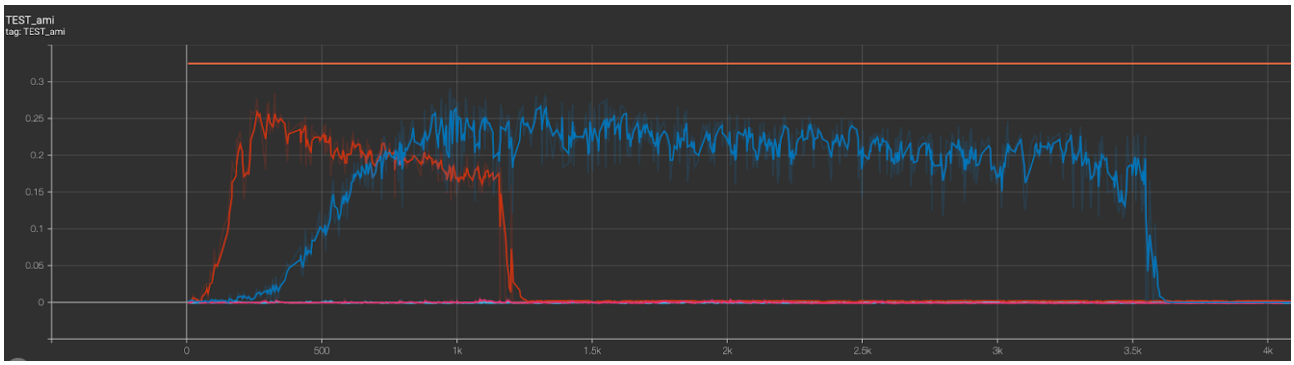


Figure 5.2: AMI of the embeddings as a the training progresses. The orange line corresponds to a PCA baseline, the other curves correspond to different runs of our model with different learning rates. This experiment is done on 100 peaks and 5.000 regions, and is used for showing the failure modes of our model.

A possible explanation for this phenomenon is in the architecture of our encoder model. Indeed, while the decoder part of our model is very parameter-efficient and designed in a principled manner, we have used an extremely naive and strongly over-parameterized variational family for the encoder. For example, in experiments with 100 peaks, 3 cell types, 5,000 regions, 3,000 cells, and 50 reads per cell, we have around 500 parameters for the decoder, but close to 100,000 parameters for our encoder as it was a naive multi layer perceptron. We are thus in a regime where the decoder manages easily to learn the location of the peaks in the data, but where the encoder may completely fail to train and gave random embeddings.

## 5.9 Future work

As explained in the previous paragraph, we believe that implementing a new encoder with less parameters may be the key to obtain good results with our model. This could be done, e.g., by using a set learnable masks as we presented in 5.2, these masks can even be the same peaks as the ones used in the decoder. We hope that it should allow the model to learn reasonable embeddings and thus achieve good performances on more complex data.

Once the model works consistently on simulated data, our plan is to use the same evaluation as we did in [82] so that this model can be validated on real data. If it obtains performances comparable, or better, than the ones of LSI we would then investigate if the peaks learned from the data differ in a useful way from the ones obtained by differential enrichment analysis. We could also extend this model to do differential enrichment analysis in a similar fashion as [195] extended scVI [56]. Conditioning on the batch id, is also an obvious improvement that should help handle batch correction.

Another interesting venue we would like to pursue is to use this model on large cohorts of bulk data. Indeed, while this model was developed with single-cell data in mind, it can directly be used on bulk data provided that there is enough samples. A direct application would be to run it on ENCODE or eGTEX, as it would learn the peaks directly from the data instead of relying on an ad-hoc peak merging step between samples.

# Chapter 6

# Conclusion

The main aim of this thesis was to focus on improving the quality of embeddings for cells, which we believe to be of paramount importance for the field of single-cell. This realisation and goal is obviously shared by many of the groups with a computational component, however their focus is usually more on the side of developing new methods instead of understanding how best to use the already existing ones.

In this final chapter we will present the reasoning behind the work that was done during this thesis, both published, and unpublished.

## 6.1 Role of the published contributions

My initial goal was to focus on scHPTM from the start, as the Vallot lab was among the first labs to develop methods for the mapping of histone modifications in single cells (see [47, 45]). However since this thesis was done between industry (Google) and academia (Institut Curie), being able to exchange data between the two was a problem at the start, thus leaving me unable to initially work on the data produced in the Vallot lab.

We instead chose to initially focus on a problem that could be tackled with public data, and could leverage the computational infrastructure of the industry partner. At the time public single-cell data was mostly scRNA-seq data, which was also a protocol commonly used by the academic partner, so we decided to work on scRNA-seq data. By doing the literature review to familiarise myself with the field, I realised that while methods development was a very active and prolific domain, most biology-driven paper tended to rely on older methods to analyze scRNA-seq datasets (i.e., some variation of PCA). This was surprising because according to the evaluation section of the papers of newer methods, they usually outperformed their PCA baseline. Such behaviour could be explained by the fact that practitioners were relying on established methods until the methods development domain matured and reached a consensus on which methods worked best, or it could mean that these methods were not yet robust enough to be used in routine by biologists. There were already a few robust studies on the relative advantages of clustering methods with [166, 167], as well as of the various embedding methods [165], but they suffered from either (1) using data not representative of real life applications (e.g. limited to cell lines), (2) were only using older methods, or (3) used the methods in a default way (i.e. run with default parameters). While (1) and (2) were mostly due to the fact

that the papers were a few years old at the time, (3) was harder to solve, as testing multiple parameters can drastically increase the computational cost of the experiments. However since we had access to a compute infrastructure orders of magnitude larger than what academic labs usually have, we had a great opportunity to solve that problem.

Indeed the effect of hyperparameters is known to have a strong influence in the deep learning literature, and designing algorithms to identify the best ones is a very active and important field of research [196, 197] in that community. This problem was first tackled in the single-cell community by [80], who indeed showed that the deep learning-based methods for embedding are very sensitive to parameter tuning. This paper was however limited to just one class of models (naive VAEs), on simulated data, and only tried to vary a few hyperparameters. These limitations were perfectly reasonable as the paper was intended to raise awareness on the issue, not to fully quantify the influence of the hyperparameters. This problem of sensitivity to hyperparameters is relatively common in the machine learning literature, where by properly tuning their method, and poorly tuning the baselines, authors can show better performances in their results section, which do not translate to the real world.

We thus decided to leverage our infrastructure to extensively quantify the sensitivity to hyperparameters of the various algorithms used in the scRNA-seq domain. We selected datasets of increasing complexity to understand in which case each method would be the most relevant. We further created new datasets following the same procedure as [166] (presented in Appendix C.1) in order to have even more challenging conditions than the ones used in other benchmarks. We then selected seven software pipelines representative of the various classes of algorithms used in the field, identified which parameters were commonly changed, and tried all combinations of reasonable values. This gave us close to 1.5 million runs, whose analysis is done in Chapter 3.

This initial project allowed us to show that the recent deep learning-based methods were indeed very sensitive to hyperparameters. This problem was further exacerbated by the fact that distinguishing between a model with good performances and bad performances based on the model loss is currently impossible. This is a bad look for deep learning-based model, because the search for good hyperparameters is akin to finding a needle in a haystack (as there are few good hyperparameter combinations amongst all possible combinations), without knowing the difference between them (because model evaluation is impossible). Furthermore, we showed that PCA-based methods were extremely robust to hyperparameters, and could pretty much be ran without any hyperparameter tuning, while also having very competitive performances compared to the former type.

With this project done, I felt confident in having gotten a good understanding of the literature for obtaining embeddings of cells. We put this to use with other colleagues by writing a review of the recent progresses in machine learning for single-cell [69], were I was responsible for the dimension reduction part. This gave me a good excuse to dig broader into all the methods developed in 2019 and 2020 and I noticed that most recent methods were mostly variations of some form of VAE. While these variations did marginally increase the performances, it felt like there was no new discoveries to make deep learning suddenly work. Finding a new architecture for scRNA-seq seemed unlikely to succeed as it is in its nature "just" tabular data. And until the problem of model selection is solved, or until scientists manage to leverage atlases for training larger models, this seemed like a risky area of research.

Epigenetic data on the other hand is not "just" tabular data. Indeed if one were to switch the ordering of two genes in a transcriptomic profile seen as a vector of counts, this would not change anything about how we understand the data. On the other hand if one were to switch the ordering of the different regions in the genome in an epigenetic profile, this would obviously feel wrong. This feeling is caused by the fact that epigenetic marks (either histone modifications or DNA methylation) are known to be deposited in a sequential fashion, for example PRC2 methylates histones one after the other along the genome. Treating these regions as independent, when we know that their modifications are locally correlated, thus misses an important inductive bias that could be used for designing new models. This problem of how to represent epigenetic data in a machine learning friendly way (i.e. a vector) is thus not just a data processing problem, but is a strong modeling choice. When looking at the literature for representations of single-cell epigenomic data, I found that all methods bypassed that very important step and just assumed that the data was already in an appropriate vector representation (except for AtacWorks [198]). This led me to start two projects related to the question of turning epigenomic data into vectors. The first one was to develop a dimension representation method that could work on the raw data without having to rely on a previous step of feature engineering. Indeed replacing feature engineering by a model that learns it from the data is one of the biggest success of machine learning methods, this is how convolutional neural networks (CNNs) [199] replaced handcrafted filters and managed to become the standard for image analysis in the early 2010s [200]. This project is still in progress and represented most of my PhD work, it is briefly presented in 5.

The second one was to study the effect of the various data engineering methods on the quality of the embedding obtained by existing dimension reduction methods. Indeed it was perfectly possible that this problem of vector representation did not have a large impact on the performances, or that if it did there existed a heuristic that worked well enough most of the time so that this was not a real problem (e.g. peak calling on pseudo-bulk). Furthermore, since the modality of single-cell histone modification was relatively new, there was not yet a lot of methods developed, nor knowledge of which class of algorithms were likely to succeed. This felt like a great opportunity to just use all classes of algorithms that were looking reasonable, and thus save the research community some time by not having to rediscover what works and what doesn't.

However evaluating the quality of the representations was more complicated for scHPTM than it was for scRNA-seq, since there are no accepted high quality annotated datasets for scHPTM. At that time I had a colleague (Laetitia Meng-Papaxanthos) working on multimodal data integration [1], and I took an interest in the type of metrics she was using to evaluate alignment algorithms. Indeed methods for alignment are evaluated on co-assays where we know which measurement in one modality corresponded to which one in the second modality (see https://openproblems.bio/benchmarks/multimodal_data_integration/). This is done by first getting a low dimension representation of each modality, then running the alignment algorithm (generally Procrustes or Gromov-Wassertein), and finally measuring whether cells that were

[1] multimodal data integration consists of having two experiments with different protocols (e.g. ATAC-seq and RNA-seq) on the same biological sample. The goal is to predict which states in the first modality corresponds to which states in the second modality. This allows to act as if the measurement was done with a co-assay.

similar in one modality, are still similar in the second modality. This metric makes sense because a good alignment algorithm should be able to match similar cells from one modality to another, and the fact that it measures this for multiple cells at each time instead of one by one is done in order to take into account the stochasticity and noise inherent in the measurements. This metric however makes a strong assumption: that the embeddings in the two modalities are good enough to be aligned, that is to say that the two embeddings manage to capture the biology contained in the data. If one of the two embeddings is of low quality, the two modalities obviously become unalignable, even with the best alignment algorithm. This insight can be leveraged to evaluate the quality of the embeddings. Indeed if we already know the matching between the two modalities, by using a co-assay, we can see how good the embeddings are by measuring how "alignable" they are. This method of evaluating the quality of representations was discovered in parallel by [64, 72], and is the one we used in Chapter 4 for evaluating scHPTM embedding methods without having any labels. This insight will also be explored again in the Perspectives.

Now that we had a way to evaluate the quality of an embedding without requiring any label, we identified a few datasets containing the most common histone marks and selected two of them: [71] as it was very high quality data with a robust scRNA-seq co-assay, and [180] as it contained data in raw format allowing us to test more feature engineering strategies. We then identified which methods were likely to be used for analysing this type of data, identified the various feature engineering used, the various quality controls used, the effect of the number of cells in a experiment and tested them all at the same time. This was made possible since we had access to a large computation infrastructure, once the code for using this infrastructure was completed we could very simply increase the number of conditions we tried, thus allowing us to test new conditions by simply adding them to a configuration file. This project is currently [82], in review at Genome Biology. Our goal there was to save the community some time by avoiding a year where papers using obvious algorithms had to be published, followed by a benchmark a few years later. Instead we directly did the benchmark with the methods most likely to be used, this also allowed us to hammer in the message that feature engineering is almost as important as the embedding method used, thus attracting the attention of the community to the often dismissed problems.

## 6.2 Perspectives

### 6.2.1 Impact of deep learning

I started my PhD thesis in 2019, just as deep learning started to enter the domain of single-cell data science. During these 3 years we saw an explosion of methods and ideas on how to analyse this kind of data, this explosion was further encouraged by the progresses made on the data generation side, with the generalisation of co-assays, new epigenomic assays, spatial transcriptomic, and large atlases.

I felt that while this opened great opportunities on the methods side, people actually analyzing data were left with poor guidance on which methods actually worked in the lab, and which ones only worked on the datasets they were evaluated on. This feeling led me to try to consolidate knowledge and provide robust best practices for using the tools that currently

exist.

I also tried to tackle often overlooked issue of building the count matrix from epigenomic datasets, and I hope that my published contribution will bring some more attention of the methods developers to that problem.

The parallel with the 2010s era of deep learning for natural language processing and computer vision is however hard not to see. The domain currently lacks its "ImageNet" moment where large high quality datasets are available, and researchers can confidently use models developed by other labs. While the field is currently in a "Wild West" era, pushing the frontier on both the experimental technologies and analysis tools, we can reasonably expect it to mature in the coming years as model evaluation becomes easier. When that time comes we will be able to leverage the information contained in the new atlases, reuse models across experiments, use model to reliably predict perturbations, and hopefully single-cell data science will become as the experimental part.

### 6.2.2 Computational biology versus deep learning

An interesting lesson I learned during this PhD is the incredibly large difference between *machine learning research* and *scientific research using machine learning*. In the first case the goal is to design a better algorithm for a well formulated problem (e.g. predicting whether an image contains a cat or a dog, beating Tetris), and there is generally well accepted datasets and metrics in the community. When starting a machine learning research project, the authors can afford to only focus on their algorithm without questioning the validity of the metrics used, or whether the datasets are representative of real world tasks or of good quality. This means that the evaluation and validation of their new methods is generally just a matter of importing the competing methods, and showing that your method has a bigger number than them.

In the case of machine learning applied to scientific research, the design of the algorithm is generally just a small part of the work. Indeed, the end goal is to answer a scientific question, the researchers thus need to identify or produce data for answering that question, and if it is not possible they need to either ask a different question or find relevant proxy tasks. Once the data has been produced or sourced publicly, the researcher then needs to understand how the data production affected the measurements, since most measurements are noisy and biased in various ways (e.g. computational labels, batch effect, poor antibody …). After designing and running their algorithm, they need to show that not only the algorithm is correct, but that its result are scientifically relevant. This task is much harder here because the measurements are usually done in order to discover new things. Evaluating whether or not an algorithms discovers new things, and whether or not these discoveries are relevant and correct is extremely tricky. Indeed since the experiment is made to discover new insights, we *a priori* neither know how many insights there are in the data nor what they are. Validating the correctness of new algorithms is thus complicated because there is generally no known "ground truth" to compare against. Indeed if there were a ground truth, it would mean that we already knew all the insights in the data (or at least a way to obtain them), in which case there would be no point in doing the experiment as we would already have an answer to our question.

This problem, while it could easily be expected, really surprised me during my thesis. Being more familiar with machine learning research, I was surprised to discover the amount of work

needed on top of the algorithm design. This made all my projects longer than expected, indeed in the case of the paper presented in Chapter 3 I naively expected that the work would simply consist of running different pipelines, measuring the results, and reporting which one had the best performances. Since my background was in computer science and I worked as a software engineer before my PhD, I expected to be able to finish it in a just a few months, as I believed that the project would just require some pretty simple coding tasks. This project ended up taking a whole year, with at least 3 months spent on identifying the relevant metrics and dataset.

Making sure that the metrics used to evaluate algorithms are actually linked to the usefulness of that algorithm is a problem that followed me for the whole thesis. As presented in Chapter 5, there is no agreed upon definition of what peaks are, building an algorithm to detect them would thus end up involving proving that what we detect are indeed peaks, and if not that it is indeed useful.

In the work presented in Chapter‘4, I believe that we managed to identify a very useful metric to evaluate multiomics assays. We used that metric to evaluate modalities for which no cell type labels were available, and still managed to obtain useful conclusions. I also expect this metric to become useful for day to day analysis of multiomics assays in the future. Indeed since it can be used ”out of the box”, as long as we have two measurements per cell, it can be extended to any situation where we would want to validate that the embeddings are correct. Using two measurement on each sample, and using this information to evaluate (or learn) embeddings is becoming common in the machine learning research literature and is called *contrastive learning*. We hope that by using that, we could mitigate the issue of hyperparameter sensitivity of deep learning models, and thus making it actually useful and robust in real projects. This could at least allow practitioners to know when their embeddings are of poor quality, which would already be a great step forward. Using this metric as part of the objective function that deep learning methods use (e.g. as a regularisation) could even improve their performances further, as they would be optimizing for a biologically relevant function instead of a theoretically relevant one.

Overall, during this thesis I have spent more time than expected, and came to really enjoy, the work of evaluating the metrics used in machine learning for single-cell data analysis. Indeed, one of the biggest strengths of deep learning is its ability to optimize the model for a given objective function, this can be an issue if that objective is uncorrelated with actual biology, but could be a major asset should we manage to build the right objective functions.

# Bibliography


[1] Sandra M. Swain, José Baselga, Sung-Bae Kim, Jungsil Ro, Vladimir Semiglazov, Mario Campone, Eva Ciruelos, Jean-Marc Ferrero, Andreas Schneeweiss, Sarah Heeson, Emma Clark, Graham Ross, Mark C. Benyunes, and Javier Cortés. Pertuzumab, trastuzumab, and docetaxel in her2-positive metastatic breast cancer. *New England Journal of Medicine*, 372(8):724–734, 2015. PMID: 25693012.

[2] Adedayo A Onitilo, Jessica M Engel, Robert T Greenlee, and Bickol N Mukesh. Breast cancer subtypes based on er/pr and her2 expression: comparison of clinicopathologic features and survival. *Clinical medicine & research*, 7(1-2):4–13, 2009.

[3] Harold J. Burstein, Christina Lacchetti, and Jennifer J. Griggs. Adjuvant endocrine therapy for women with hormone receptor–positive breast cancer: Asco clinical practice guideline focused update. *Journal of Oncology Practice*, 15(2):106–107, 2019. PMID: 30523754.

[4] Li Ding, Timothy J Ley, David E Larson, Christopher A Miller, Daniel C Koboldt, John S Welch, Julie K Ritchey, Margaret A Young, Tamara Lamprecht, Michael D McLellan, et al. Clonal evolution in relapsed acute myeloid leukaemia revealed by whole-genome sequencing. *Nature*, 481(7382):506–510, 2012.

[5] Chang Xu. A review of somatic single nucleotide variant calling algorithms for next-generation sequencing data. *Computational and structural biotechnology journal*, 16:15–24, 2018.

[6] Erez Lieberman-Aiden, Nynke L Van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R Lajoie, Peter J Sabo, Michael O Dorschner, et al. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *science*, 326(5950):289–293, 2009.

[7] Jason D Buenrostro, Beijing Wu, Howard Y Chang, and William J Greenleaf. Atac-seq: a method for assaying chromatin accessibility genome-wide. *Current protocols in molecular biology*, 109(1):21–29, 2015.

[8] Jason D Buenrostro, Beijing Wu, Ulrike M Litzenburger, Dave Ruff, Michael L Gonzales, Michael P Snyder, Howard Y Chang, and William J Greenleaf. Single-cell chromatin accessibility reveals principles of regulatory variation. *Nature*, 523(7561):486–490, 2015.

[9] Hatice S Kaya-Okur, Steven J Wu, Christine A Codomo, Erica S Pledger, Terri D Bryson, Jorja G Henikoff, Kami Ahmad, and Steven Henikoff. CUT&Tag for efficient epigenomic profiling of small samples and single cells. *Nature Communications*, 10(1):1–10, 2019.

[10] Bing Ren, François Robert, John J Wyrick, Oscar Aparicio, Ezra G Jennings, Itamar Simon, Julia Zeitlinger, Jorg Schreiber, Nancy Hannett, Elenita Kanin, et al. Genome-wide location and function of dna binding proteins. *science*, 290(5500):2306–2309, 2000.

[11] Yuval Blat and Nancy Kleckner. Cohesins bind to preferential sites along yeast chromosome iii, with differential regulation along arms versus the centric region. *Cell*, 98(2):249–259, 1999.

[12] Ugrappa Nagalakshmi, Zhong Wang, Karl Waern, Chong Shou, Debasish Raha, Mark Gerstein, and Michael Snyder. The transcriptional landscape of the yeast genome defined by rna sequencing. *Science*, 320(5881):1344–1349, 2008.

[13] Cancer Genome Atlas Research Network Tissue source sites: Duke University Medical School McLendon Roger 1 Friedman Allan 2 Bigner Darrell 1, Emory University Van Meir Erwin G. 3 4 5 Brat Daniel J. 5 6 M. Mastrogianakis Gena 3 Olson Jeffrey J. 3 4 5, Henry Ford Hospital Mikkelsen Tom 7 Lehman Norman 8, MD Anderson Cancer Center Aldape Ken 9 Alfred Yung WK 10 Bogler Oliver 11, University of California San Francisco VandenBerg Scott 12 Berger Mitchel 13 Prados Michael 13, et al. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455(7216):1061–1068, 2008.

[14] Valentine Svensson, Roser Vento-Tormo, and Sarah A Teichmann. Exponential scaling of single-cell rna-seq in the past decade. *Nature protocols*, 13(4):599–604, 2018.

[15] Sarah Aldridge and Sarah A Teichmann. Single cell transcriptomics comes of age. *Nature Communications*, 11(1):1–4, 2020.

[16] Daniel Ramsköld, Shujun Luo, Yu-Chieh Wang, Robin Li, Qiaolin Deng, Omid R Faridani, Gregory A Daniels, Irina Khrebtukova, Jeanne F Loring, Louise C Laurent, et al. Full-length mrna-seq from single-cell levels of rna and individual circulating tumor cells. *Nature biotechnology*, 30(8):777–782, 2012.

[17] Simone Picelli, Omid R Faridani, Åsa K Björklund, Gösta Winberg, Sven Sagasser, and Rickard Sandberg. Full-length rna-seq from single cells using smart-seq2. *Nature protocols*, 9(1):171–181, 2014.

[18] Michael Hagemann-Jensen, Christoph Ziegenhain, Ping Chen, Daniel Ramsköld, Gert-Jan Hendriks, Anton JM Larsson, Omid R Faridani, and Rickard Sandberg. Single-cell rna counting at allele and isoform resolution using smart-seq3. *Nature Biotechnology*, 38(6):708–714, 2020.

[19] Philip Brennecke, Simon Anders, Jong Kyoung Kim, Aleksandra A Kołodziejczyk, Xiuwei Zhang, Valentina Proserpio, Bianka Baying, Vladimir Benes, Sarah A Teichmann, John C

Marioni, et al. Accounting for technical noise in single-cell rna-seq experiments. *Nature methods*, 10(11):1093–1095, 2013.

[20] Allon M Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A Weitz, and Marc W Kirschner. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, 2015.

[21] Evan Z Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R Bialas, Nolan Kamitaki, Emily M Martersteck, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, 2015.

[22] James Eberwine, Hermes Yeh, Kevin Miyashiro, Yanxiang Cao, Suresh Nair, Richard Finnell, Martha Zettel, and Paul Coleman. Analysis of gene expression in single live neurons. *Proceedings of the National Academy of Sciences*, 89(7):3010–3014, 1992.

[23] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B Tuch, Asim Siddiqui, et al. mrna-seq whole-transcriptome analysis of a single cell. *Nature methods*, 6(5):377–382, 2009.

[24] Saiful Islam, Una Kjällquist, Annalena Moliner, Pawel Zajac, Jian-Bing Fan, Peter Lönnerberg, and Sten Linnarsson. Characterization of the single-cell transcriptional landscape by highly multiplex rna-seq. *Genome research*, 21(7):1160–1167, 2011.

[25] Diego Adhemar Jaitin, Ephraim Kenigsberg, Hadas Keren-Shaul, Naama Elefant, Franziska Paul, Irina Zaretsky, Alexander Mildner, Nadav Cohen, Steffen Jung, Amos Tanay, et al. Massively parallel single-cell rna-seq for marker-free decomposition of tissues into cell types. *Science*, 343(6172):776–779, 2014.

[26] Fredrik Salmen, Joachim De Jonghe, Tomasz S Kaminski, Anna Alemany, Guillermo E Parada, Joe Verity-Legg, Ayaka Yanagida, Timo N Kohler, Nicholas Battich, Floris van den Brekel, et al. High-throughput total rna sequencing in single cells using vasa-seq. *Nature Biotechnology*, pages 1–14, 2022.

[27] Gökcen Eraslan, Eugene Drokhlyansky, Shankara Anand, Evgenij Fiskin, Ayshwarya Subramanian, Michal Slyper, Jiali Wang, Nicholas Van Wittenberghe, John M Rouhana, Julia Waldman, et al. Single-nucleus cross-tissue molecular reference maps toward understanding disease gene function. *Science*, 376(6594):eabl4290, 2022.

[28] Tabula Sapiens Consortium*, Robert C Jones, Jim Karkanias, Mark A Krasnow, Angela Oliveira Pisco, Stephen R Quake, Julia Salzman, Nir Yosef, Bryan Bulthaup, Phillip Brown, et al. The tabula sapiens: A multiple-organ, single-cell transcriptomic atlas of humans. *Science*, 376(6594):eabl4896, 2022.

[29] Preetish Kadur Lakshminarasimha Murthy, Vishwaraj Sontake, Aleksandra Tata, Yoshihiko Kobayashi, Lauren Macadlo, Kenichi Okuda, Ansley S Conchola, Satoko Nakano, Simon Gregory, Lisa A Miller, et al. Human distal lung maps and lineage hierarchies reveal a bipotent progenitor. *Nature*, 604(7904):111–119, 2022.

[30] Douglas Hanahan. Hallmarks of cancer: new dimensions. *Cancer discovery*, 12(1):31–46, 2022.

[31] Samra Turajlic, Andrea Sottoriva, Trevor Graham, and Charles Swanton. Resolving genetic heterogeneity in cancer. *Nature Reviews Genetics*, 20(7):404–416, 2019.

[32] Sydney M Shaffer, Margaret C Dunagin, Stefan R Torborg, Eduardo A Torre, Benjamin Emert, Clemens Krepler, Marilda Beqiri, Katrin Sproesser, Patricia A Brafford, Min Xiao, et al. Rare cell variability and drug-induced reprogramming as a mode of cancer drug resistance. *Nature*, 546(7658):431–435, 2017.

[33] Jocelyn F Chen and Qin Yan. The roles of epigenetics in cancer progression and metastasis. *Biochemical Journal*, 478(17):3373–3393, 2021.

[34] Ansuman T Satpathy, Jeffrey M Granja, Kathryn E Yost, Yanyan Qi, Francesca Meschi, Geoffrey P McDermott, Brett N Olsen, Maxwell R Mumbach, Sarah E Pierce, M Ryan Corces, et al. Massively parallel single-cell chromatin landscapes of human immune cell development and intratumoral t cell exhaustion. *Nature biotechnology*, 37(8):925–936, 2019.

[35] Xi Chen, Ricardo J Miragaia, Kedar Nath Natarajan, and Sarah A Teichmann. A rapid and robust method for single cell chromatin accessibility profiling. *Nature communications*, 9(1):1–9, 2018.

[36] Shaokun Shu, Hua-Jun Wu, Y Ge Jennifer, Rhamy Zeid, Isaac S Harris, Bojana Jovanović, Katherine Murphy, Binbin Wang, Xintao Qiu, Jennifer E Endress, et al. Synthetic lethal and resistance interactions with bet bromodomain inhibitors in triple-negative breast cancer. *Molecular cell*, 78(6):1096–1113, 2020.

[37] Yuanyuan Zhang, Hongyan Chen, Hongnan Mo, Xueda Hu, Ranran Gao, Yahui Zhao, Baolin Liu, Lijuan Niu, Xiaoying Sun, Xiao Yu, et al. Single-cell analyses reveal key immune cell subsets associated with response to pd-l1 blockade in triple-negative breast cancer. *Cancer Cell*, 39(12):1578–1593, 2021.

[38] Hongshan Guo, Ping Zhu, Xinglong Wu, Xianlong Li, Lu Wen, and Fuchou Tang. Single-cell methylome landscapes of mouse embryonic stem cells and early embryos analyzed using reduced representation bisulfite sequencing. *Genome research*, 23(12):2126–2135, 2013.

[39] Hongshan Guo, Ping Zhu, Fan Guo, Xianlong Li, Xinglong Wu, Xiaoying Fan, Lu Wen, and Fuchou Tang. Profiling dna methylome landscapes of mammalian cells with single-cell reduced-representation bisulfite sequencing. *Nature protocols*, 10(5):645–659, 2015.

[40] Kevin C Johnson, Kevin J Anderson, Elise T Courtois, Amit D Gujar, Floris P Barthel, Frederick S Varn, Diane Luo, Martine Seignon, Eunhee Yi, Hoon Kim, et al. Single-cell multimodal glioma analyses identify epigenetic regulators of cellular plasticity and environmental stress response. *Nature genetics*, 53(10):1456–1468, 2021.

[41] Alessandro Pastore, Federico Gaiti, Sydney X Lu, Ryan M Brand, Scott Kulm, Ronan Chaligne, Hongcang Gu, Kevin Y Huang, Elena K Stamenova, Wendy Béguelin, et al. Corrupted coordination of epigenetic modifications leads to diverging chromatin states and transcriptional heterogeneity in cll. *Nature communications*, 10(1):1–11, 2019.

[42] Christof Angermueller, Heather J Lee, Wolf Reik, and Oliver Stegle. Deepcpg: accurate prediction of single-cell dna methylation states using deep learning. *Genome biology*, 18(1):1–13, 2017.

[43] Steven J Wu, Scott N Furlan, Anca B Mihalas, Hatice S Kaya-Okur, Abdullah H Feroze, Samuel N Emerson, Ye Zheng, Kalee Carson, Patrick J Cimino, C Dirk Keene, et al. Single-cell CUT&Tag analysis of chromatin modifications in differentiation and tumor progression. *Nature biotechnology*, 39(7):819–824, 2021.

[44] Winnie WI Hui, Angela Simeone, Katherine G Zyner, David Tannahill, and Shankar Balasubramanian. Single-cell mapping of dna g-quadruplex structures in human cancer cells. *Scientific Reports*, 11(1):1–7, 2021.

[45] Justine Marsolier, Pacôme Prompsy, Adeline Durand, Anne-Marie Lyne, Camille Landragin, Amandine Trouchet, Sabrina Tenreira Bento, Almut Eisele, Sophie Foulon, Léa Baudre, Kevin Grosselin, Mylène Bohec, Sylvain Baulande, Ahmed Dahmani, Laura Sourd, Eric Letouzé, Anne-Vincent Salomon, Elisabetta Marangoni, Leïla Perié, and Céline Vallot. H3K27me3 conditions chemotolerance in triple-negative breast cancer. *Nature Genetics*, 54(4):459–468, 2022.

[46] Assaf Rotem, Oren Ram, Noam Shoresh, Ralph A Sperling, Alon Goren, David A Weitz, and Bradley E Bernstein. Single-cell ChIP-seq reveals cell subpopulations defined by chromatin state. *Nature Biotechnology*, 33(11):1165–1172, 2015.

[47] Kevin Grosselin, Adeline Durand, Justine Marsolier, Adeline Poitou, Elisabetta Marangoni, Fariba Nemati, Ahmed Dahmani, Sonia Lameiras, Fabien Reyal, Olivia Frenoy, et al. High-throughput single-cell ChIP-seq identifies heterogeneity of chromatin states in breast cancer. *Nature Genetics*, 51(6):1060–1066, 2019.

[48] Wai Lim Ku, Kosuke Nakamura, Weiwu Gao, Kairong Cui, Gangqing Hu, Qingsong Tang, Bing Ni, and Keji Zhao. Single-cell chromatin immunocleavage sequencing (scChIC-seq) to profile histone modification. *Nature methods*, 16(4):323–325, 2019.

[49] Marek Bartosovic, Mukund Kabbe, and Gonçalo Castelo-Branco. Single-cell CUT&Tag profiles histone modifications and transcription factors in complex tissues. *Nature Biotechnology*, 39(7):825–835, 2021.

[50] Junyue Cao, Diana R O'Day, Hannah A Pliner, Paul D Kingsley, Mei Deng, Riza M Daza, Michael A Zager, Kimberly A Aldinger, Ronnie Blecher-Gonen, Fan Zhang, et al. A human cell atlas of fetal gene expression. *Science*, 370(6518):eaba7721, 2020.

[51] Atray Dixit, Oren Parnas, Biyu Li, Jenny Chen, Charles P Fulco, Livnat Jerby-Arnon, Nemanja D Marjanovic, Danielle Dionne, Tyler Burks, Raktima Raychowdhury, et al.

Perturb-seq: dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *cell*, 167(7):1853–1866, 2016.

[52] Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, et al. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 171(6):1437–1452, 2017.

[53] Malte D Luecken and Fabian J Theis. Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol. Syst. Biol.*, 15(6):e8746, 2019.

[54] Aaron TL Lun, Davis J McCarthy, and John C Marioni. A step-by-step workflow for low-level analysis of single-cell rna-seq data with bioconductor. *F1000Research*, 5, 2016.

[55] Koen Van den Berge, Hector Roux de Bézieux, Kelly Street, Wouter Saelens, Robrecht Cannoodt, Yvan Saeys, Sandrine Dudoit, and Lieven Clement. Trajectory-based differential expression analysis for single-cell sequencing data. *Nature communications*, 11(1):1–13, 2020.

[56] Romain Lopez, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053, 2018.

[57] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[58] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[59] Kevin P. Murphy. *Machine learning : a probabilistic perspective*. MIT Press, Cambridge, Mass. [u.a.], 2013.

[60] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Springer, 2006.

[61] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[62] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[63] Tamim Abdelaal, Lieke Michielsen, Davy Cats, Dylan Hoogduin, Hailiang Mei, Marcel JT Reinders, and Ahmed Mahfouz. A comparison of automatic cell identification methods for single-cell rna-sequencing data. *bioRxiv*, page 644435, 2019.

[64] Tal Ashuach, Daniel A. Reidenbach, Adam Gayoso, and Nir Yosef. PeakVI: A deep generative model for single-cell chromatin accessibility analysis. *Cell Reports Methods*, 2(3):100182, 2022.

[65] L. Xiong, K. Xu, K. Tian, Y. Shao, L. Tang, G. Gao, M. Zhang, T. Jiang, and Q. C. Zhang. SCALE method for single-cell ATAC-seq analysis via latent feature extraction. *Nat. Commun.*, 10(1):1–10, 2019.

[66] Carmen Bravo González-Blas, Liesbeth Minnoye, Dafni Papasokrati, Sara Aibar, Gert Hulselmans, Valerie Christiaens, Kristofer Davie, Jasper Wouters, and Stein Aerts. cisTopic: cis-regulatory topic modeling on single-cell ATAC-seq data. *Nature Methods*, 16(5):397–400, 2019.

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

[68] Gökcen Eraslan, Lukas M Simon, Maria Mircea, Nikola S Mueller, and Fabian J Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):390, 2019.

[69] Félix Raimundo, Laetitia Meng-Papaxanthos, Céline Vallot, and Jean-Philippe Vert. Machine learning for single-cell genomics data analysis. *Current Opinion in Systems Biology*, 26:64–71, 2021.

[70] Romain Lopez, Adam Gayoso, and Nir Yosef. Enhancing scientific discoveries in molecular biology with deep generative models. *Molecular Systems Biology*, 16(9):e9198, 2020.

[71] Chenxu Zhu, Yanxiao Zhang, Yang Eric Li, Jacinta Lucero, M Margarita Behrens, and Bing Ren. Joint profiling of histone modifications and transcriptome in single cells from mouse brain. *Nature Methods*, 18(3):283–292, 2021.

[72] Han Yuan and David R Kelley. scBasset: sequence-based modeling of single-cell ATAC-seq using convolutional neural networks. *Nature Methods*, pages 1–9, 2022.

[73] David R Kelley, Jasper Snoek, and John L Rinn. Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome research*, 26(7):990–999, 2016.

[74] Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18(10):1196–1203, 2021.

[75] Yong Zhang, Tao Liu, Clifford A Meyer, Jérôme Eeckhoute, David S Johnson, Bradley E Bernstein, Chad Nusbaum, Richard M Myers, Myles Brown, Wei Li, et al. Model-based analysis of ChIP-Seq (MACS). *Genome Biology*, 9(9):1–9, 2008.

[76] Peter V Kharchenko, Michael Y Tolstorukov, and Peter J Park. Design and analysis of chip-seq experiments for dna-binding proteins. *Nature biotechnology*, 26(12):1351–1359, 2008.

[77] Joel Rozowsky, Ghia Euskirchen, Raymond K Auerbach, Zhengdong D Zhang, Theodore Gibson, Robert Bjornson, Nicholas Carriero, Michael Snyder, and Mark B Gerstein. Peak-seq enables systematic scoring of chip-seq experiments relative to controls. *Nature biotechnology*, 27(1):66, 2009.

[78] Yuchun Guo, Shaun Mahony, and David K. Gifford. High resolution genome wide binding event finding and motif discovery reveals transcription factor spatial binding constraints. *PLOS Computational Biology*, 8(8):1–14, 08 2012.

[79] Timothy Bailey, Pawel Krajewski, Istvan Ladunga, Celine Lefebvre, Qunhua Li, Tao Liu, Pedro Madrigal, Cenny Taslim, and Jie Zhang. Practical guidelines for the comprehensive analysis of chip-seq data. *PLoS computational biology*, 9(11):e1003326, 2013.

[80] Qiwen Hu and Casey S Greene. Parameter tuning is a key part of dimensionality reduction via deep variational autoencoders for single cell rna transcriptomics. In *PSB*, pages 362–373. World Scientific, 2019.

[81] F. Raimundo, C. Vallot, and J.-P. Vert. Tuning parameters of dimensionality reduction methods for single-cell RNA-seq analysis. *Genome Biol.*, 21:212, 2020.

[82] Felix Raimundo, Pacôme Prompsy, Jean-Philippe Vert, and Celine Vallot. Best practices for single-cell histone modification analysis. *bioRxiv*, 2022.

[83] D. Lähnemann, J. Köster, E. Szczurek, D. J. McCarthy, S. C. Hicks, M. D. Robinson, C. A. Vallejos, K. R. Campbell, N. Beerenwinkel, A. Mahfouz, L. Pinello, P. Skums, A. Stamatakis, C. S. O. Attolini, S. Aparicio, J. Baaijens, M. Balvert, B. D. Barbanson, A. Cappuccio, G. Corleone, B. E. Dutilh, M. Florescu, V. Guryev, R. Holmer, K. Jahn, T. J. Lobo, E. M. Keizer, I. Khatri, S. M. Kielbasa, J. O. Korbel, A. M. Kozlov, T. H. Kuo, B. P. F. Lelieveldt, I. I. Mandoiu, J. C. Marioni, T. Marschall, F. Mölder, A. Niknejad, L. Raczkowski, M. Reinders, J. D. Ridder, A. Emmanuel. Saliba, A. Somarakis, O. Stegle, F. J. Theis, H. Yang, A. Zelikovsky, A. C. McHardy, B. J. Raphael, S. P. Shah, and A. Schönhuth. Eleven grand challenges in single-cell data science. *Genome Biol.*, 21:31, February 2020.

[84] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, May 2015.

[85] D. Risso, F. Perraudeau, S. Gribkova, S. Dudoit, and J.-P. Vert. A general and flexible method for signal extraction from single-cell RNA-seq data. *Nat. Commun.*, 9(1):284, 2018.

[86] A. Verma and B. E. Engelhardt. A robust nonlinear low-dimensional manifold for single cell RNA-seq data. *BMC Bioinf.*, 21:324, July 2020.

[87] M. Huang, J. Wang, E. Torre, H. Dueck, S. Shaffer, R. Bonasio, J. I. Murray, A. Raj, M. Li, and N. R. Zhang. SAVER: gene expression recovery for single-cell rna sequencing. *Nat. Methods*, 15(7):539–542, 2018.

[88] C. H. Grønbech, M. F. Vording, P. N. Timshel, C. K. Sønderby, T. H. Pers, and O. Winther. scVAE: Variational auto-encoders for single-cell gene expression data. *Bioinformatics*, 36(16):4415–4422, 2020.

[89] A. Klimovskaia, D. Lopez-Paz, L. Bottou, and M. Nickel. Poincaré maps for analyzing complex hierarchies in single-cell data. *Nat. Commun.*, 11(1):1–9, 2020.

[90] J. Ding and A. Regev. Deep generative model embedding of single-cell RNA-Seq profiles on hyperspheres and hyperbolic spaces. *bioRxiv*, 2019.

[91] D. Kobak and P. Berens. The art of using t-SNE for single-cell transcriptomics. *Nat. Commun.*, 10(1):5416, 2019.

[92] E. Becht, L. McInnes, J. Healy, C. A. Dutertre, I. W. H. Kwok, L. G. Ng, F. Ginhoux, and E. W. Newell. Dimensionality reduction for visualizing single-cell data using UMAP. *Nat. Biotechnol.*, 37(1):38–44, 2019.

[93] D. A. Cusanovich, J. P. Reddington, D. A. Garfield, R. M. Daza, D. Aghamirzaie, R. Marco-Ferreres, H. A. Pliner, L. Christiansen, X. Qiu, F. J. Steemers, et al. The cis-regulatory dynamics of embryonic development at single-cell resolution. *Nature*, 555(7697):538–542, 2018.

[94] P. Prompsy, P. Kirchmeier, J. Marsolier, M. Deloger, N. Servant, and C. Vallot. Interactive analysis of single-cell epigenomic landscapes with ChromSCape. *Nat. Commun.*, 11(1):1–9, 2020.

[95] C. B. González-Blas, L. Minnoye, D. Papasokrati, S. Aibar, G. Hulselmans, V. Christiaens, K. Davie, J. Wouters, and S. Aerts. cisTopic: cis-regulatory topic modeling on single-cell ATAC-seq data. *Nat. Methods*, 16(5):397–400, 2019.

[96] Huidong Chen, Caleb Lareau, Tommaso Andreani, Michael E Vinyard, Sara P Garcia, Kendell Clement, Miguel A Andrade-Navarro, Jason D Buenrostro, and Luca Pinello. Assessment of computational methods for the analysis of single-cell ATAC-seq data. *Genome Biology*, 20(1):1–25, 2019.

[97] R. Erbe, M. D. Kessler, A. V. Favorov, H. Easwaran, and E. J. Gaykalova, D. A .and Fertig. Matrix factorization and transfer learning uncover regulatory biology across multiple single-cell ATAC-seq data sets. *Nucleic Acids Res.*, 48(12):e68–e68, 2020.

[98] G. L. Stein-O'Brien, B. S. Clark, T. Sherman, C. Zibetti, Q. Hu, R. Sealfon, S. Liu, J. Qian, C. Colantuoni, S. Blackshaw, L. A. Goff, and E. J. Fertig. Decomposing cell identity for transfer learning across cellular measurements, platforms, tissues, and species. *Cell Syst.*, 8(5):395–411, 2019.

[99] G. Sharma, C. Colantuoni, L. A. Goff, E. J. Fertig, and G. Stein-O'Brien. projectR: an R/Bioconductor package for transfer learning via PCA, NMF, correlation and clustering. *Bioinformatics*, 36(11):3592–3593, 2020.

[100] B. Mieth, J. R. F. Hockley, N. Görnitz, M. M. C. Vidovic, K. R. Müller, A. Gutteridge, and D. Ziemek. Using transfer learning from prior reference knowledge to improve the clustering of single-cell RNA-seq data. *Sci. Rep.*, 9(1):20353, 2019.

[101] C. Lin, S. Jain, H. Kim, and Z. Bar-Joseph. Using neural networks for reducing the dimensions of single-cell RNA-seq data. *Nucleic Acids Res.*, 45(17):e156–e156, 2017.

[102] J. Wang, D. Agarwal, M. Huang, G. Hu, Z. Zhou, C. Ye, and N. R. Zhang. Data denoising with transfer learning in single-cell transcriptomics. *Nat. Methods*, 16(9):875–878, 2019.

[103] M. B. Badsha, R. Li, B. Liu, Y. I. Li, M. Xian, N. E. Banovich, and A. Q. Fu. Imputation of single-cell gene expression with an autoencoder neural network. *Quantitative Biology*, 8(1):78–94, 2020.

[104] J. Hu, X. Li, G. Hu, Y. Lyu, K. Susztak, and M. Li. Iterative transfer learning with neural network for clustering and cell type classification in single-cell RNA-seq analysis. *Nat. Mach. Intell.*, 2(10):607–618, 2020.

[105] Z. Zhou, C. Ye, J. Wang, and N. R. Zhang. Surface protein imputation from single cell transcriptomes by deep neural networks. *Nat. Commun.*, 11(1):651, 2020.

[106] R. Petegrosso, Z. Li, and R. Kuang. Machine learning and statistical methods for clustering single-cell RNA-sequencing data. *Briefings Bioinf.*, 21:1209–1223, July 2020.

[107] M. Krzak, Y. Raykov, A. Boukouvalas, L. Cutillo, and C. Angelini. Benchmark and parameter sensitivity analysis of single-cell RNA sequencing clustering methods. *Front. Genet.*, 10:1253, 2019.

[108] Beate Vieth, Swati Parekh, Christoph Ziegenhain, Wolfgang Enard, and Ines Hellmann. A systematic evaluation of single cell rna-seq analysis pipelines. *bioRxiv*, page 583013, 2019.

[109] Z. J. Cao, L. Wei, S. Lu, D. C. Yang, and G. Gao. Searching large-scale scRNA-seq databases via unbiased cell embedding with Cell BLAST. *Nat. Commun.*, 11(1):3458, 2020.

[110] V. Y. Kiselev, A. Yiu, and M. Hemberg. scmap: projection of single-cell RNA-seq data across data sets. *Nat. Methods*, 15(5):359–362, 2018.

[111] A. Alavi, M. Ruffalo, A. Parvangada, Z. Huang, and Z. Bar-Joseph. A web server for comparative analysis of single-cell RNA-seq data. *Nat. Commun.*, 9(1):4768, 2018.

[112] K. Sato, K. Tsuyuzaki, K. Shimizu, and I. Nikaido. CellFishing.jl: an ultrafast and scalable cell search method for single-cell RNA sequencing. *Genome Biol.*, 20(1):31, 2019.

[113] Hoa Thi Nhu Tran, Kok Siong Ang, Marion Chevrier, Xiaomeng Zhang, Nicole Yee Shin Lee, Michelle Goh, and Jinmiao Chen. A benchmark of batch-effect correction methods for single-cell RNA sequencing data. *Genome Biol.*, 21(1):1–32, 2020.

[114] M. Amodio, D. van Dijk, K. Srinivasan, W. SS Chen, H. Mohsen, K. R. Moon, A. Campbell, Y. Zhao, X. Wang, M. Venkataswamy, A. Desai, V. Ravi, P. Kumar, R. Montgomery, G. Wolf, and S. Krishnaswamy. Exploring single-cell data with deep multitasking neural networks. *Nat. Methods*, 16:1139–1145, November 2019.

[115] S. Ge, H. Wang, A. Alavi, E. Xing, and Z. Bar-Joseph. Supervised adversarial alignment of single-cell RNA-seq data. *J. Comput. Biol.*, 2021.

[116] L. Zhang and Q. Nie. scMC learns biological variation through the alignment of multiple single-cell genomics datasets. *Genome Biol.*, 22(1):1–28, 2021.

[117] I. Korsunsky, N. Millard, J. Fan, K. Slowikowski, F. Zhang, K. Wei, Y. Baglaenko, M. Brenner, P. Loh, and S. Raychaudhuri. Fast, sensitive and accurate integration of single-cell data with Harmony. *Nat. Methods*, 16(12):1289–1296, 2019.

[118] Y. Yang, G. Li, H. Qian, K. C. Wilhelmsen, Y. Shen, and Y. Li. SMNN: batch effect correction for single-cell RNA-seq data via supervised mutual nearest neighbor detection. *Brief. Bioinform.*, 2020.

[119] Y. Wang, J. Hoinka, and T. M. Przytycka. Subpopulation detection and their comparative analysis across single-cell experiments with scPopCorn. *Cell Systems*, 8(6):506–513, 2019.

[120] N. Barkas, V. Petukhov, D. Nikolaeva, Y. Lozinsky, S. Demharter, K. Khodosevich, and P.V. Kharchenko. Joint analysis of heterogeneous single-cell RNA-seq dataset collections. *Nat. Methods*, 16(8):695–698, 2019.

[121] N. Johansen and G. Quon. scAlign: a tool for alignment, integration, and rare cell identification from scRNA-seq data. *Genome Biol.*, 20(1):1–21, 2019.

[122] T. Wang, T. S. Johnson, W. Shao, Z. Lu, B. R. Helm, J. Zhang, and K. Huang. BERMUDA: a novel deep transfer learning method for single-cell RNA sequencing batch correction reveals hidden high-resolution cellular subtypes. *Genome Biol.*, 20(1):165, 2019.

[123] M. Lotfollahi, F. A. Wolf, and F. J. Theis. scGen predicts single-cell perturbation responses. *Nat. Methods*, 16(8):715–721, 2019.

[124] M. Lotfollahi, M. Naghipourfar, F. J. Theis, and F. A. Wolf. Conditional out-of-distribution generation for unpaired data using transfer VAE. *Bioinformatics*, 36(Supplement_2):i610–i617, 2020.

[125] Y. Lin, S. Ghazanfar, K. Y. X. Wang, J. A. Gagnon-Bartsch, K. K. Lo, X. Su, Z.-G. Han, J. T. Ormerod, T. P. Speed, P. Yang, and J. Y. Hwa Yang. scMerge leverages factor analysis, stable expression, and pseudoreplication to merge multiple single-cell RNA-seq datasets. *Proc. Natl. Acad. Sci.*, 116(20):9775–9784, 2019.

[126] B. Hie, B. Bryson, and B. Berger. Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nat. Biotechnol.*, 37(6):685–691, 2019.

[127] T. Stuart, A. Butler, P. Hoffman, C. Hafemeister, E. Papelexi, W. M. Mauck III, Y. Hao, M. Stoeckius, P. Smibert, and R. Satija. Comprehensive integration of single-cell data. *Cell*, 177(7):1888–1902, 2019.

[128] C. Trapnell, D. Cacchiarelli, J. Grimsby, P. Pokharel, S. Li, M. Morse, N. J. Lennon, K. J. Livak, T. S. Mikkelsen, and J. L. Rinn. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nat. Biotechnol.*, 32(4):381–6, 2014.

[129] S. Ahmed, M. Rattray, and A. Boukouvalas. GrandPrix: scaling up the Bayesian GPLVM for single-cell data. *Bioinformatics*, 35:47–54, January 2019.

[130] H. Chen, L. Albergante, J. Y. Hsu, C. A. Lareau, G. Lo Bosco, J. Guan, S. Zhou, A. N. Gorban, D. E. Bauer, M. J. Aryee, D. M. Langenau, A. Zinovyev, J. D. Buenrostro, G.-C. Yuan, and L. Pinello. Single-cell trajectories reconstruction, exploration and mapping of omics data with STREAM. *Nat. Commun.*, 10:1903, April 2019.

[131] F. A. Wolf, F. K. Hamey, M. Plass, J. Solana, J. S Dahlin, B. Göttgens, N. Rajewsky, L. Simon, and F. J. Theis. PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biol.*, 20:59, March 2019.

[132] H. Todorov, R. Cannoodt, W. Saelens, and Y. Saeys. TinGa: fast and flexible trajectory inference with growing neural gas. *Bioinformatics*, 36:i66–i74, July 2020.

[133] J. Cao, M. Spielmann, X. Qiu, X. Huang, D. M. Ibrahim, A. J. Hill, F. Zhang, S. Mundlos, L. Christiansen, F. J Steemers, C. Trapnell, and J. Shendure. The single-cell transcriptional landscape of mammalian organogenesis. *Nature*, 566:496–502, February 2019.

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

[135] Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943, 2019.

[136] G. Peyré and M. Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[137] K. D. Yang, K. Damodaran, S. Venkatachalapathy, A. C. Soylemezoglu, G. V. Shivashankar, and C. Uhler. Predicting cell lineages using autoencoders and optimal transport. *PLoS Comput. Biol.*, 16:e1007828, April 2020.

[138] G. La Manno, R. Soldatov, A. Zeisel, E. Braun, H. Hochgerner, V. Petukhov, K. Lidschreiber, M. E. Kastriti, P. Lönerberg, A. Furlan, J. Fan, L. E. Borm, Z. Liu, D. van Bruggen, J. Guo, X. He, R. Barker, E. Sundström, G. Castelo-Branco, P. Cramer, I. Adameyko, S. Linnarsson, and P. V. Kharchenko. RNA velocity of single cells. *Nature*, 560:494–498, August 2018.

[139] V. Bergen, M. Lange, S. Peidli, F. A. Wolf, and F. J. Theis. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nat. Biotechnol.*, 38:1408–1414, December 2020.

[140] P.-C. Aubin-Frankowski and J.-P. Vert. Gene regulation inference from single-cell RNA-seq data with linear differential equations and velocity inference. *Bioinformatics*, 36:4774–4780, 2020.

[141] Y. Yuan and Z. Bar-Joseph. Deep learning for inferring gene relationships from single-cell expression data. *Proc. Natl. Acad. Sci. U.S.A.*, 116(52):27151–27158, December 2019.

[142] M. Efremova and S. A. Teichmann. Computational methods for single-cell omics across modalities. *Nat. Methods*, 17(1):14–17, 2020.

[143] A. Ma, A. McDermaid, J. Xu, Y. Chang, and Q. Ma. Integrative methods and practical challenges for single-cell multi-omics. *Trends Biotechnol.*, 38:1007–1022, 2020.

[144] P. Demetci, R. Santorella, B. Sandstede, W. S. Noble, and R. Singh. Gromov-Wasserstein optimal transport to align single-cell multi-omics data. *ICML 2020 Workshop on Computational Biology (WCB)*, 2020.

[145] C. Zuo and L. Chen. Deep-joint-learning analysis model of single cell transcriptome and open chromatin accessibility data. *Briefings Bioinf.*, 2020. bbaa287.

[146] K. R. Campbell, A. Steif, E. Laks, M. Zahn, D. Lai, A. McPherson, M. Farahani, F. Kabeer, C. O'Flanagan, J. Biele, J. Brimhall, B. Wang, P. Walters, IMAXT Consortium, A. Bouchard-Côté, S. Aparicio, and S. P. Shah. clonealign: Statistical integration of independent single-cell RNA and DNA sequencing data from human cancers. *Genome Biol.*, 20(1):54, 2019.

[147] S. G. Stark, J. Ficek, F. Locatello, X. Bonilla, S. Chevrier, F. Singer, Tumor Profiler Consortium, G. Rätsch, and K.-V. Lehmann. SCIM: Universal single-cell matching with unpaired feature sets. *Bioinformatics*, 36:i919–i927, 12 2020.

[148] J. Liu, Y. Huang, R. Singh, J.-P. Vert, and W. S. Noble. Jointly embedding multiple single-cell omics measurements. In *19th International Workshop on Algorithms in Bioinformatics (WABI 2019)*, volume 143 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:13, 2019.

[149] K. Cao, X. Bai, Y. Hong, and L. Wan. Unsupervised topological alignment for single-cell multi-omics integration. *Bioinformatics*, 36:i48–i56, 2020.

[150] J. D. Welch, V. Kozareva, A. Ferreira, C. Vanderburg, C. Martin, and E. Z. Macosko. Single-cell multi-omic integration compares and contrasts features of brain cell identity. *Cell*, 177(7):1873–1887, 2019.

[151] S. Jin, L. Zhang, and Q. Nie. scAI: an unsupervised approach for the integrative analysis of parallel single-cell transcriptomic and epigenomic profiles. *Genome Biol.*, 21(1):1–19, 2020.

[152] B. Duan, C. Zhou, C. Zhu, Y. Yu, G. Li, S. Zhang, C. Zhang, X. Ye, H. Ma, S. Qu, Z. Zhang, P. Wang, S. Sun, and Q. Liu. Model-based understanding of single-cell CRISPR screening. *Nat. Commun.*, 10(1):1–11, 2019.

[153] L. Yang, Y. Zhu, H. Yu, X. Cheng, S. Chen, Y. Chu, H. Huang, J. Zhang, and W. Li. scMAGeCK links genotypes with multiple phenotypes in single-cell CRISPR screens. *Genome Biol.*, 21(1):1–14, 2020.

[154] A. Gayoso, Z. Steier, R. Lopez, J. Regier, K. L. Nazor, A. Streets, and N. Yosef. Joint probabilistic modeling of single-cell multi-omic data with totalVI. *Nat. Methods*, 18(3):272–282, 2021.

[155] R. Argelaguet, D. Arnol, D. Bredikhin, Y. Deloro, B. Velten, J. C. Marioni, and Stegle O. MOFA+: a probabilistic framework for comprehensive integration of structured single-cell data. *Genome Biol.*, 21:111, 2020.

[156] Y. Hao, S. Hao, E. Andersen-Nissen, W. M. Mauck, S. Zheng, A. Butler, M. J. Lee, A. J. Wilk, C. Darby, M. Zagar, P. Hoffman, M. Stoeckius, E. Papalexi, E. P. Mimitou, J. Jain, A. Srivastava, T. Stuart, L. B. Fleming, B. Yeung, A. J. Rogers, J. M. McElrath, C. A. Blish, R. Gottardo, P. Smibert, and R. Satija. Integrated analysis of multimodal single-cell data. *bioRxiv*, 2020.

[157] W. Zeng, X. Chen, Z. Duren, Y. Wang, R. Jiang, and W. H. Wong. DC3 is a method for deconvolution and coupled clustering from bulk and single-cell genomics data. *Nat. Commun.*, 10(1):1–11, 2019.

[158] R. Singh, P. Demetci, G. Bonora, V. Ramani, C. Lee, H. Fang, Z. Duan, X. Deng, J. Shendure, C. Disteche, and W. S. Noble. Unsupervised manifold alignment for single-cell multi-omics data. In *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 1–10, 2020.

[159] A. A. Kolodziejczyk, J. K. Kim, V. Svensson, J. C Marioni, and S. A. Teichmann. The technology and biology of single-cell RNA sequencing. *Molecular Cell*, 58(4):610–620, 2015.

[160] Amit Zeisel, Ana B Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, et al. Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq. *Science*, 347(6226):1138–1142, 2015.

[161] B. Tasic, V. Menon, T. N. Nguyen, T. K. Kim, T. Jarsky, Z. Yao, B. Levi, L. T. Gray, S. A. Sorensen, T. Dolbeare, D. Bertagnolli, J. Goldy, N. Shapovalova, S. Parry, C. Lee, K. Smith, A. Bernard, L. Madisen, S. M. Sunkin, M. Hawrylycz, C. Koch, and H. Zeng. Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. *Nat. Neurosci.*, 19(2):335–346, Feb 2016.

[162] P. V. Kharchenko, L. Silberstein, and D. T. Scadden. Bayesian approach to single-cell differential expression analysis. *Nat. Methods*, 11(7):740–742, Jul 2014.

[163] Stephanie C Hicks, F. William Townes, Mingxiang Teng, and Rafael A Irizarry. Missing data and technical variability in single-cell RNA-sequencing experiments. *Biostatistics*, 19(4):562–578, 2017.

[164] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. Single-cell RNA sequencing technologies and bioinformatics pipelines. *Experimental & Molecular Medicine*, 50(8):96, 2018.

[165] Shiquan Sun, Jiaqiang Zhu, Ying Ma, and Xiang Zhou. Accuracy, robustness and scalability of dimensionality reduction methods for single cell rnaseq analysis. *bioRxiv*, page 641142, 2019.

[166] Angelo Duò, Mark D Robinson, and Charlotte Soneson. A systematic performance evaluation of clustering methods for single-cell rna-seq data. *F1000Research*, 7, 2018.

[167] Luyi Tian, Xueyi Dong, Saskia Freytag, Kim-Anh Lê Cao, Shian Su, Abolfazl JalalAbadi, Daniela Amann-Zalcenstein, Tom S Weber, Azadeh Seidi, Jafar S Jabbari, et al. Benchmarking single cell rna-sequencing analysis pipelines using mixture control experiments. *Nature methods*, page 1, 2019.

[168] Davis J McCarthy, Kieran R Campbell, Aaron TL Lun, and Quin F Wills. Scater: preprocessing, quality control, normalization and visualization of single-cell rna-seq data in r. *Bioinformatics*, 33(8):1179–1186, 2017.

[169] Davide Risso, Fanny Perraudeau, Svetlana Gribkova, Sandrine Dudoit, and Jean-Philippe Vert. A general and flexible method for signal extraction from single-cell RNA-seq data. *Nature communications*, 9(1):284, 2018.

[170] Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8:14049, 2017.

[171] Nicholas Schaum, Jim Karkanias, Norma F Neff, Andrew P May, Stephen R Quake, Tony Wyss-Coray, Spyros Darmanis, Joshua Batson, Olga Botvinnik, Michelle B Chen, et al. Single-cell transcriptomics of 20 mouse organs creates a tabula muris: The tabula muris consortium. *Nature*, 562(7727):367, 2018.

[172] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau,

M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[173] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(Oct):2837–2854, 2010.

[174] Sébastien Lê, Julie Josse, François Husson, et al. Factominer: an r package for multivariate analysis. *Journal of statistical software*, 25(1):1–18, 2008.

[175] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, pages 1–12, 2020.

[176] Marina K. Ayrapetov, Ozge Gursoy-Yuzugullu, Chang Xu, Ye Xu, and Brendan D. Price. DNA double-strand breaks promote methylation of histone H3 on lysine 9 and transient formation of repressive chromatin. *Proceedings of the National Academy of Sciences*, 111(25):9169–9174, 2014.

[177] Dario Nicetto, Greg Donahue, Tanya Jain, Tao Peng, Simone Sidoli, Lihong Sheng, Thomas Montavon, Justin S. Becker, Jessica M. Grindheim, Kimberly Blahnik, Benjamin A. Garcia, Kai Tan, Roberto Bonasio, Thomas Jenuwein, and Kenneth S. Zaret. H3K9me3-heterochromatin loss at protein-coding genes enables developmental lineage specification. *Science*, 363(6424):294–297, 2019.

[178] Joel C. Eissenberg and Ali Shilatifard. Histone H3 lysine 4 (H3K4) methylation in development and differentiation. *Developmental Biology*, 339(2):240–249, 2010.

[179] James E. Audia and Robert M. Campbell. Histone modifications and cancer. *Cold Spring Harbor perspectives in biology*, 8(4):a019521, 2016.

[180] Bingjie Zhang, Avi Srivastava, Eleni Mimitou, Tim Stuart, Ivan Raimondi, Yuhan Hao, Peter Smibert, and Rahul Satija. Characterizing cellular heterogeneity in chromatin state with scCUT&Tag-pro. *Nature Biotechnology*, 2022.

[181] Ian Dunham, Anshul Kundaje, Shelley F. Aldred, Patrick J. Collins, Carrie A. Davis, Francis Doyle, Charles B. Epstein, Seth Frietze, Jennifer Harrow, Rajinder Kaul, Jainab Khatun, Bryan R. Lajoie, Stephen G. Landt, Bum-Kyu Lee, Florencia Pauli, Kate R. Rosenbloom, Peter Sabo, Alexias Safi, Amartya Sanyal, Noam Shoresh, Jeremy M. Simon, Lingyun Song, Nathan D. Trinklein, Robert C. Altshuler, Ewan Birney, James B. Brown, Chao Cheng, Sarah Djebali, Xianjun Dong, Ian Dunham, Jason Ernst, Terrence S. Furey, Mark Gerstein, Belinda Giardine, Melissa Greven, Ross C. Hardison, Robert S. Harris, Javier Herrero, Michael M. Hoffman, Sowmya Iyer, Manolis Kellis, Jainab Khatun, Pouya Kheradpour, Anshul Kundaje, Timo Lassmann, Qunhua Li, Xinying Lin, Georgi K. Marinov, Angelika Merkel, Ali Mortazavi, Stephen C. J. Parker, Timothy E. Reddy, Joel Rozowsky, Felix Schlesinger, Robert E. Thurman, Jie Wang, Lucas D. Ward, Troy W.

Whitfield, Steven P. Wilder, Weisheng Wu, Hualin S. Xi, Kevin Y. Yip, Jiali Zhuang, Bradley E. Bernstein, Ewan Birney, Ian Dunham, Eric D. Green, Chris Gunter, Michael Snyder, Michael J. Pazin, Rebecca F. Lowdon, Laura A. L. Dillon, Leslie B. Adams, Caroline J. Kelly, Julia Zhang, Judith R. Wexler, Eric D. Green, Peter J. Good, Elise A. Feingold, Bradley E. Bernstein, Ewan Birney, Gregory E. Crawford, Job Dekker, Laura Elnitski, Peggy J. Farnham, Mark Gerstein, Morgan C. Giddings, Thomas R. Gingeras, Eric D. Green, Roderic Guigó, Ross C. Hardison, Timothy J. Hubbard, Manolis Kellis, W. James Kent, Jason D. Lieb, Elliott H. Margulies, Richard M. Myers, Michael Snyder, John A. Stamatoyannopoulos, Scott A. Tenenbaum, Zhiping Weng, Kevin P. White, Barbara Wold, Jainab Khatun, Yanbao Yu, John Wrobel, Brian A. Risk, Harsha P. Gunawardena, Heather C. Kuiper, Christopher W. Maier, Ling Xie, Xian Chen, Morgan C. Giddings, Bradley E. Bernstein, Charles B. Epstein, Noam Shoresh, Jason Ernst, Pouya Kheradpour, Tarjei S. Mikkelsen, Shawn Gillespie, Alon Goren, Oren Ram, Xiaolan Zhang, Li Wang, Robbyn Issner, Michael J. Coyne, Timothy Durham, Manching Ku, Thanh Truong, Lucas D. Ward, Robert C. Altshuler, Matthew L. Eaton, Manolis Kellis, Sarah Djebali, Carrie A. Davis, Angelika Merkel, Alex Dobin, Timo Lassmann, Ali Mortazavi, Andrea Tanzer, Julien Lagarde, Wei Lin, Felix Schlesinger, Chenghai Xue, Georgi K. Marinov, Jainab Khatun, Brian A. Williams, Chris Zaleski, Joel Rozowsky, Maik Röder, Felix Kokocinski, Rehab F. Abdelhamid, Tyler Alioto, Igor Antoshechkin, Michael T. Baer, Philippe Batut, Ian Bell, Kimberly Bell, Sudipto Chakrabortty, Xian Chen, Jacqueline Chrast, Joao Curado, Thomas Derrien, Jorg Drenkow, Erica Dumais, Jackie Dumais, Radha Duttagupta, Megan Fastuca, Kata Fejes-Toth, Pedro Ferreira, Sylvain Foissac, Melissa J. Fullwood, Hui Gao, David Gonzalez, Assaf Gordon, Harsha P. Gunawardena, Cédric Howald, Sonali Jha, Rory Johnson, Philipp Kapranov, Brandon King, Colin Kingswood, Guoliang Li, Oscar J. Luo, Eddie Park, Jonathan B. Preall, Kimberly Presaud, Paolo Ribeca, Brian A. Risk, Daniel Robyr, Xiaoan Ruan, Michael Sammeth, Kuljeet Singh Sandhu, Lorain Schaeffer, Lei-Hoon See, Atif Shahab, Jorgen Skancke, Ana Maria Suzuki, Hazuki Takahashi, Hagen Tilgner, Diane Trout, Nathalie Walters, Huaien Wang, John Wrobel, Yanbao Yu, Yoshihide Hayashizaki, Jennifer Harrow, Mark Gerstein, Timothy J. Hubbard, Alexandre Reymond, Stylianos E. Antonarakis, Gregory J. Hannon, Morgan C. Giddings, Yijun Ruan, Barbara Wold, Piero Carninci, Roderic Guigó, Thomas R. Gingeras, Kate R. Rosenbloom, Cricket A. Sloan, Katrina Learned, Venkat S. Malladi, Matthew C. Wong, Galt P. Barber, Melissa S. Cline, Timothy R. Dreszer, Steven G. Heitner, Donna Karolchik, W. James Kent, Vanessa M. Kirkup, Laurence R. Meyer, Jeffrey C. Long, Morgan Maddren, Brian J. Raney, Terrence S. Furey, Lingyun Song, Linda L. Grasfeder, Paul G. Giresi, Bum-Kyu Lee, Anna Battenhouse, Nathan C. Sheffield, Jeremy M. Simon, Kimberly A. Showers, Alexias Safi, Darin London, Akshay A. Bhinge, Christopher Shestak, Matthew R. Schaner, Seul Ki Kim, Zhuzhu Z. Zhang, Piotr A. Mieczkowski, Joanna O. Mieczkowska, Zheng Liu, Ryan M. McDaniell, Yunyun Ni, Naim U. Rashid, Min Jae Kim, Sheera Adar, Zhancheng Zhang, Tianyuan Wang, Deborah Winter, Damian Keefe, Ewan Birney, Vishwanath R. Iyer, Jason D. Lieb, Gregory E. Crawford, Guoliang Li, Kuljeet Singh Sandhu, Meizhen Zheng, Ping Wang, Oscar J. Luo, Atif Shahab, Melissa J. Fullwood, Xiaoan Ruan, Yijun Ruan, Richard M. Myers, Florencia Pauli, Brian A. Williams, Jason Gertz, Georgi K. Marinov, Timothy E.

Reddy, Jost Vielmetter, E. Partridge, Diane Trout, Katherine E. Varley, Clarke Gasper, The E. N. C. O. D. E. Project Consortium, Overall coordination (data analysis coordination), Data production leads (data production), Lead analysts (data analysis), Writing group, N. H. G. R. I. project management (scientific management), Principal investigators (steering committee), Boise State University, University of North Carolina at Chapel Hill Proteomics groups (data production, analysis), Broad Institute Group (data production, analysis), Center for Genomic Regulation Barcelona R. I. K. E. N. Sanger Institute University of Lausanne Genome Institute of Singapore group (data production Cold Spring Harbor, University of Geneva, analysis), Data coordination center at UC Santa Cruz (production data coordination), University of Texas Austin University of North Carolina-Chapel Hill group (data production Duke University, EBI, analysis), Genome Institute of Singapore group (data production, analysis), U. C. Irvine Stanford group (data production HudsonAlpha Institute, Caltech, and analysis). An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489(7414):57–74, 2012.

[182] Yuhan Hao, Stephanie Hao, Erica Andersen-Nissen, William M. Mauck III, Shiwei Zheng, Andrew Butler, Maddie J. Lee, Aaron J. Wilk, Charlotte Darby, Michael Zagar, Paul Hoffman, Marlon Stoeckius, Efthymia Papalexi, Eleni P. Mimitou, Jaison Jain, Avi Srivastava, Tim Stuart, Lamar B. Fleming, Bertrand Yeung, Angela J. Rogers, Juliana M. McElrath, Catherine A. Blish, Raphael Gottardo, Peter Smibert, and Rahul Satija. Integrated analysis of multimodal single-cell data. *Cell*, 2021.

[183] Tim Stuart, Avi Srivastava, Shaista Madad, Caleb A Lareau, and Rahul Satija. Single-cell chromatin state analysis with Signac. *Nature Methods*, 18(11):1333–1341, 2021.

[184] Rongxin Fang, Sebastian Preissl, Yang Li, Xiaomeng Hou, Jacinta Lucero, Xinxin Wang, Amir Motamedi, Andrew K Shiau, Xinzhu Zhou, Fangming Xie, et al. Comprehensive analysis of single cell ATAC-seq data with SnapATAC. *Nature Communications*, 12(1):1–15, 2021.

[185] Darren A Cusanovich, Riza Daza, Andrew Adey, Hannah A Pliner, Lena Christiansen, Kevin L Gunderson, Frank J Steemers, Cole Trapnell, and Jay Shendure. Multiplex single-cell profiling of chromatin accessibility by combinatorial cellular indexing. *Science*, 348(6237):910–914, 2015.

[186] Darren A. Cusanovich, Andrew J. Hill, Delasa Aghamirzaie, Riza M. Daza, Hannah A. Pliner, Joel B. Berletch, Galina N. Filippova, Xingfan Huang, Lena Christiansen, William S. DeWitt, Choli Lee, Samuel G. Regalado, David F. Read, Frank J. Steemers, Christine M. Disteche, Cole Trapnell, and Jay Shendure. A single-cell atlas of in vivo mammalian chromatin accessibility. *Cell*, 174(5):1309–1324.e18, aug 2018.

[187] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):1–5, 2018.

[188] Adam Gayoso, Romain Lopez, Galen Xing, Pierre Boyeau, Valeh Valiollah Pour Amiri, Justin Hong, Katherine Wu, Michael Jayasuriya, Edouard Mehlman, Maxime Langevin, Yining Liu, Jules Samaran, Gabriel Misrachi, Achille Nazaret, Oscar Clivio, Chenling

Xu, Tal Ashuach, Mariano Gabitto, Mohammad Lotfollahi, Valentine Svensson, Eduardo da Veiga Beltrame, Vitalii Kleshchevnikov, Carlos Talavera-López, Lior Pachter, Fabian J. Theis, Aaron Streets, Michael I. Jordan, Jeffrey Regier, and Nir Yosef. A python library for probabilistic analysis of single-cell omics data. *Nature Biotechnology*, 40(2):163–166, 2022.

[189] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[190] James M Dickey. Multiple hypergeometric functions: Probabilistic interpretations and statistical uses. *Journal of the American Statistical Association*, 78(383):628–637, 1983.

[191] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

[192] Pol Cuscó and Guillaume J Filion. Zerone: a chip-seq discretizer for multiple replicates with built-in quality control. *Bioinformatics*, 32(19):2896–2902, 2016.

[193] Qunhua Li, James B Brown, Haiyan Huang, Peter J Bickel, et al. Measuring reproducibility of high-throughput experiments. *The annals of applied statistics*, 5(3):1752–1779, 2011.

[194] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[195] Pierre Boyeau, Romain Lopez, Jeffrey Regier, Adam Gayoso, Michael I Jordan, and Nir Yosef. Deep generative models for detecting differential expression in single cells. *bioRxiv*, page 794289, 2019.

[196] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[197] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[198] Avantika Lal, Zachary D Chiang, Nikolai Yakovenko, Fabiana M Duarte, Johnny Israeli, and Jason D Buenrostro. Deep learning-based enhancement of epigenomics data with atacworks. *Nature communications*, 12(1):1–11, 2021.

[199] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[200] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[201] David Lähnemann, Johannes Köster, Ewa Szczurek, Davis J McCarthy, Stephanie C Hicks, Mark D Robinson, Catalina A Vallejos, Kieran R Campbell, Niko Beerenwinkel, Ahmed Mahfouz, et al. Eleven grand challenges in single-cell data science. *Genome biology*, 21(1):1–35, 2020.

[202] Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: simulation of single-cell rna sequencing data. *Genome biology*, 18(1):174, 2017.

[203] Beate Vieth, Christoph Ziegenhain, Swati Parekh, Wolfgang Enard, and Ines Hellmann. powsimr: power analysis for bulk and single cell rna-seq experiments. *Bioinformatics*, 33(21):3486–3488, 2017.

[204] Xiuwei Zhang, Chenling Xu, and Nir Yosef. Symsim: simulating multi-faceted variability in single cell rna sequencing. *bioRxiv*, page 378646, 2018.

[205] Aaron Lun, Davide Risso, and K Korthauer. Singlecellexperiment: S4 classes for single cell data. *R package version*, 1(0), 2018.

[206] Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M Mauck III, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *Cell*, 177(7):1888–1902, 2019.

# Appendix A

# Supplementary figures

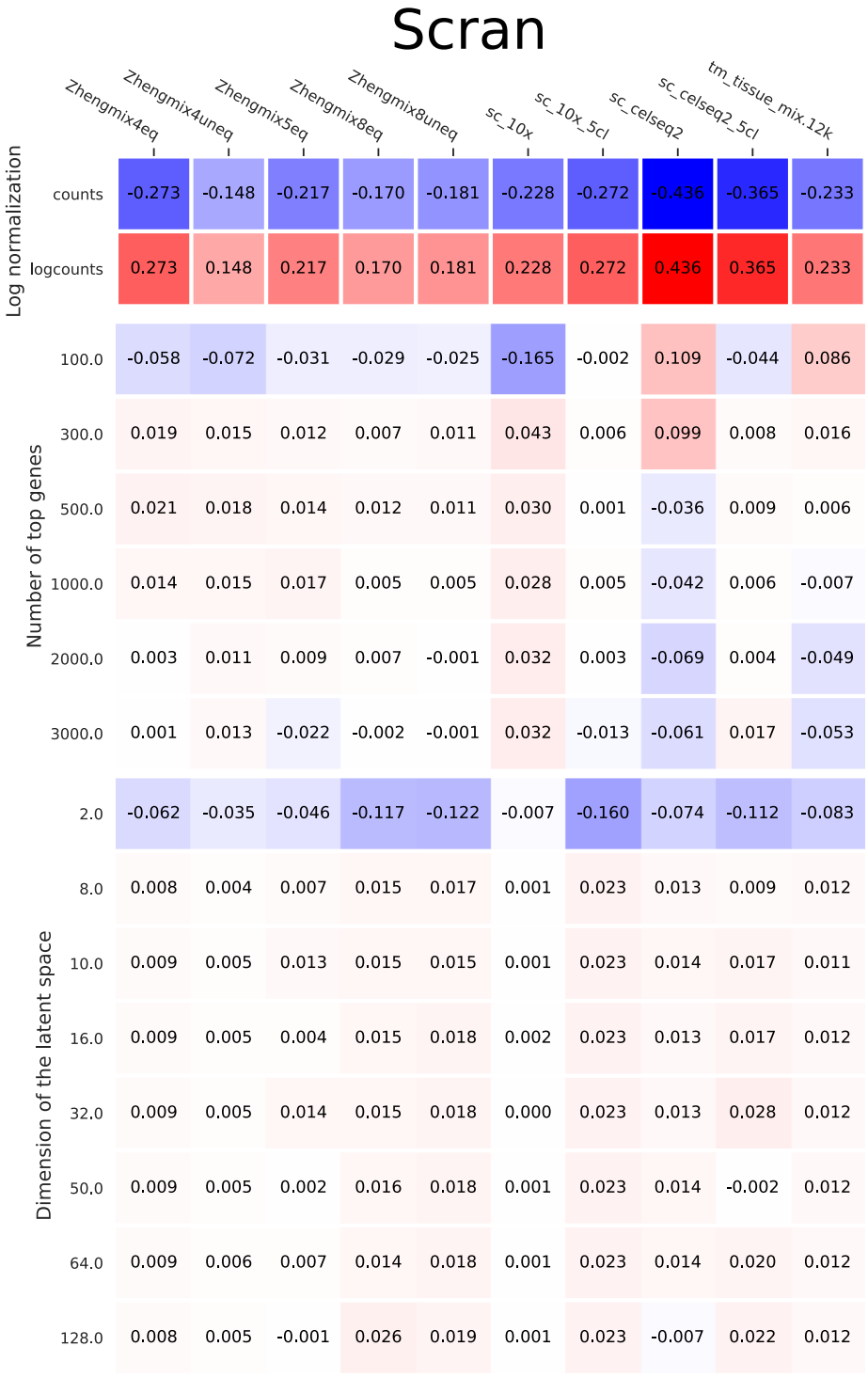


Figure S1: Heatmap visualization of the mean effect of each parameter value of scran on its AMI. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the AMI compared to the mean AMI for scran on that dataset.
These effects come from a factorial ANOVA.

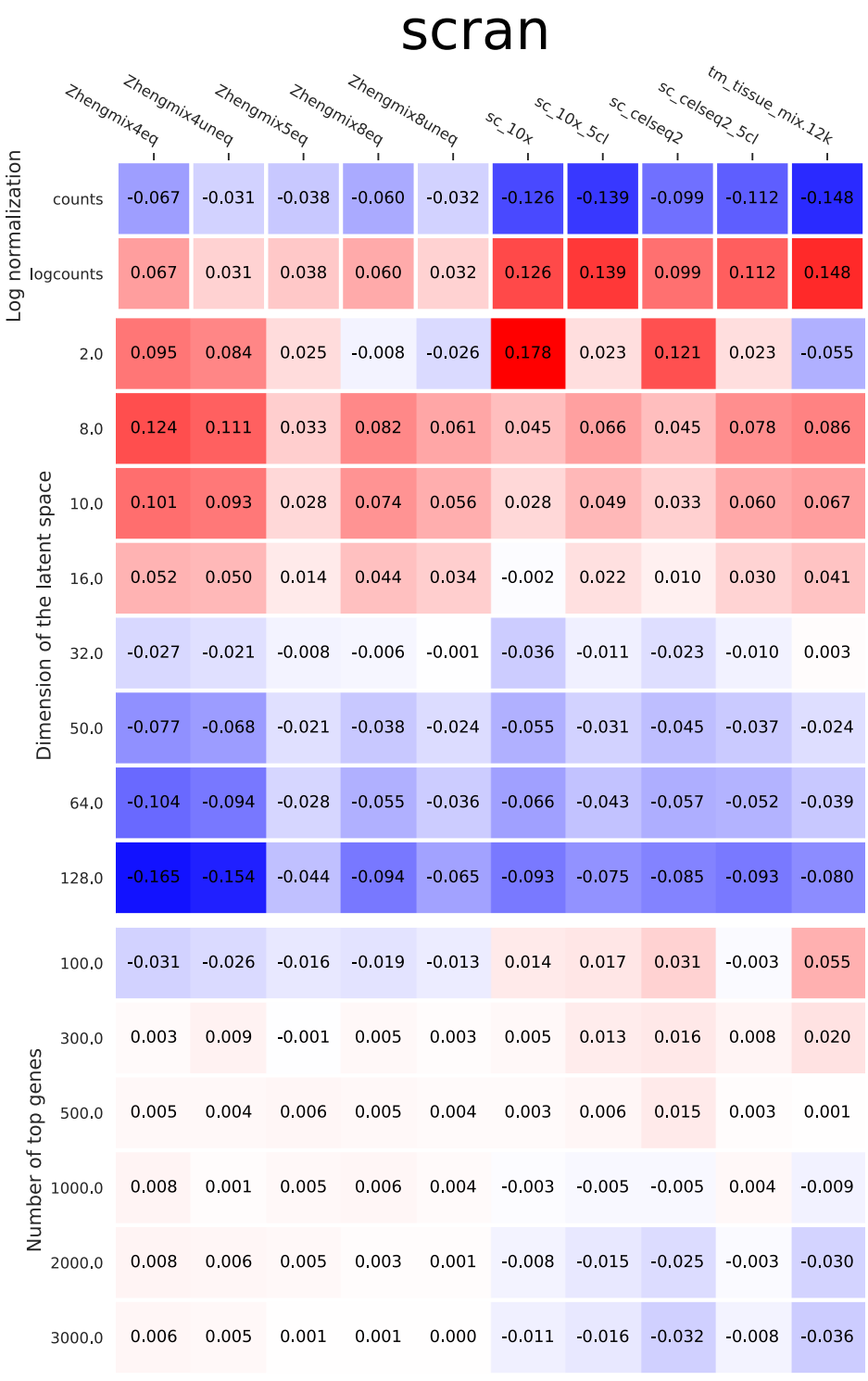


Figure S2: Heatmap visualization of the mean effect of each parameter value of scran on its silhouette. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the silhouette compared to the mean silhouette for scran on that dataset.
These effects come from a factorial ANOVA.

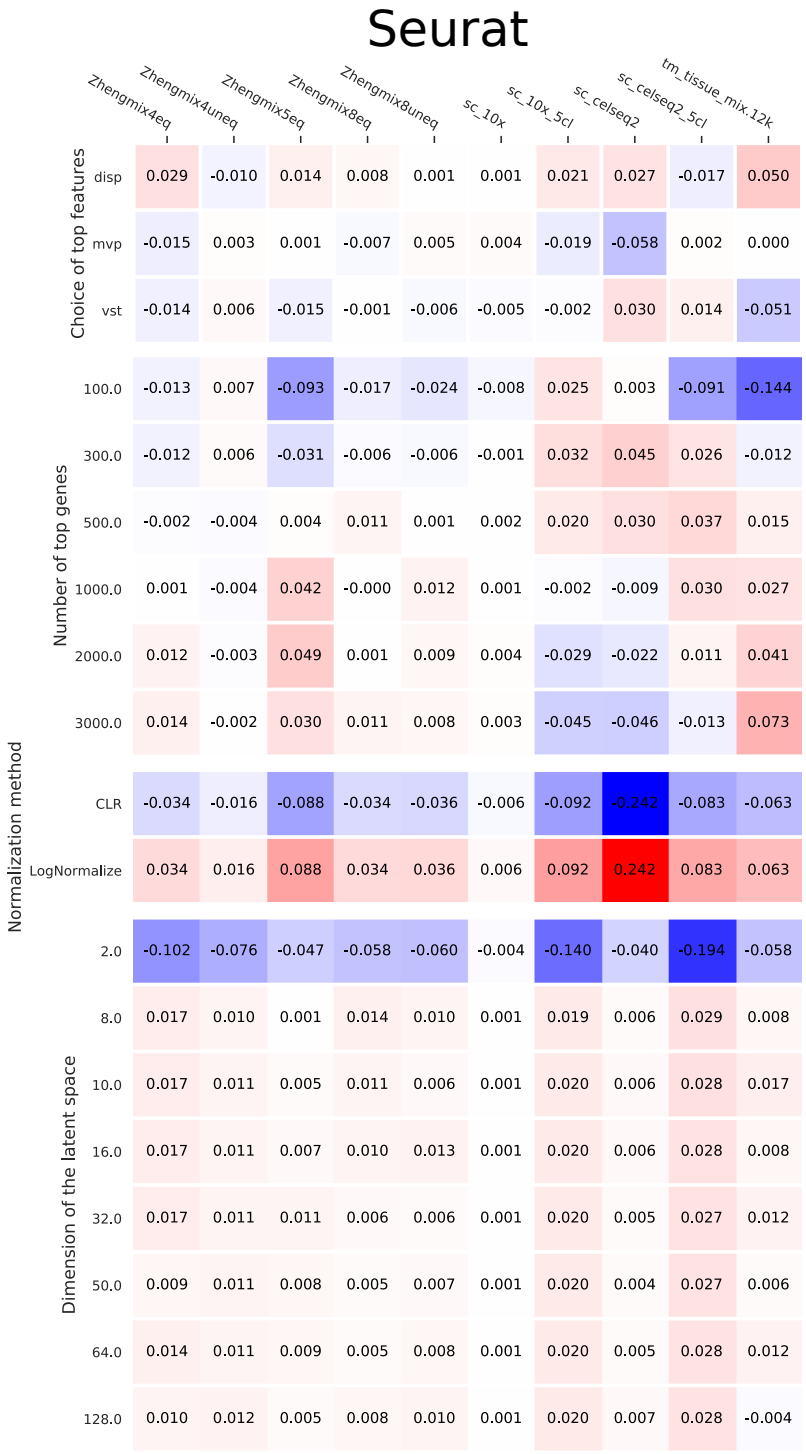


Figure S3: Heatmap visualization of the mean effect of each parameter value of Seurat on its AMI. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the AMI compared to the mean AMI for Seurat on that dataset.
These effects come from a factorial ANOVA.

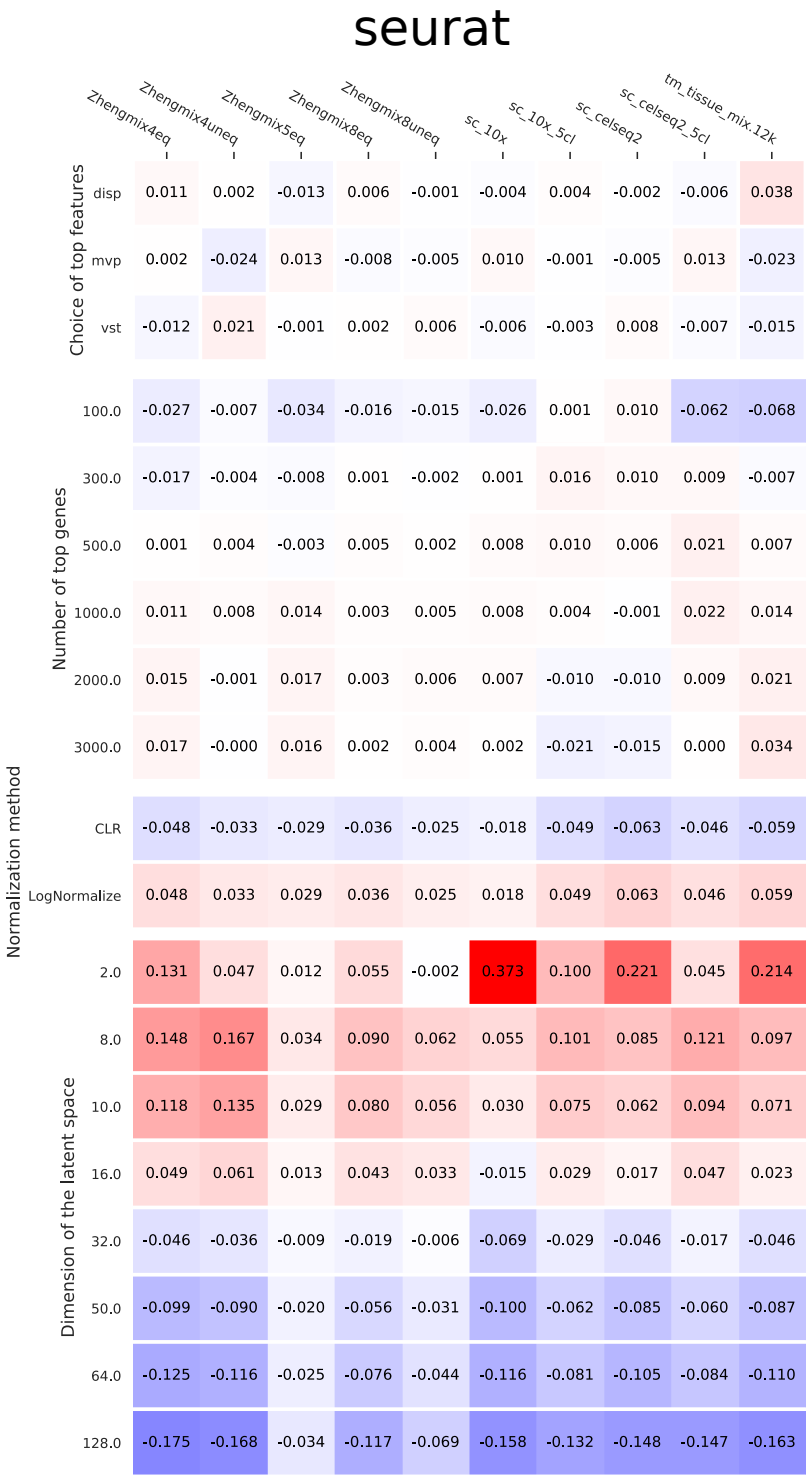


Figure S4: Heatmap visualization of the mean effect of each parameter value of Seurat on its silhouette. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the silhouette compared to the mean silhouette for Seurat on that dataset.
These effects come from a factorial ANOVA.

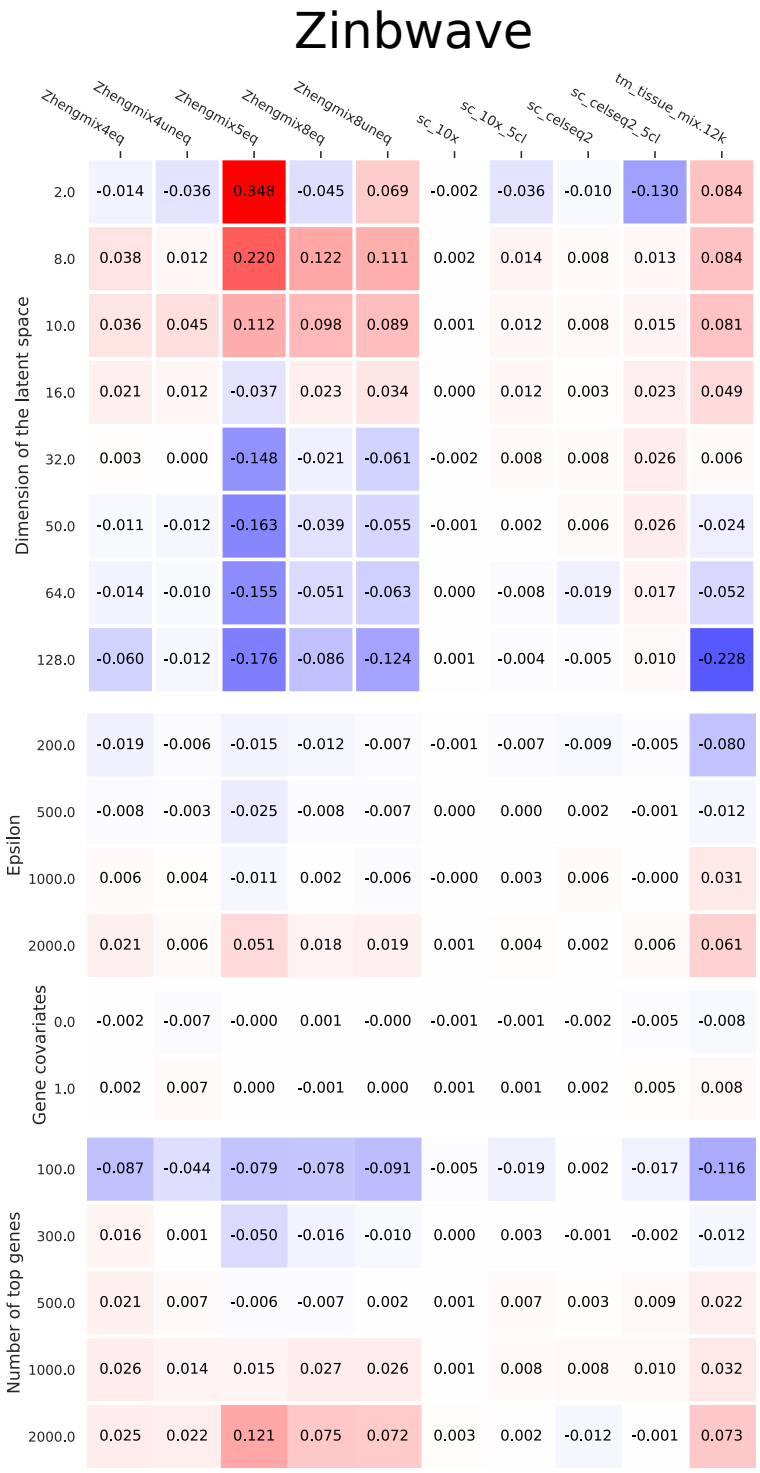


Figure S5: Heatmap visualization of the mean effect of each parameter value of ZinbWave on its AMI. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the AMI compared to the mean AMI for ZinbWave on that dataset.
These effects come from a factorial ANOVA.

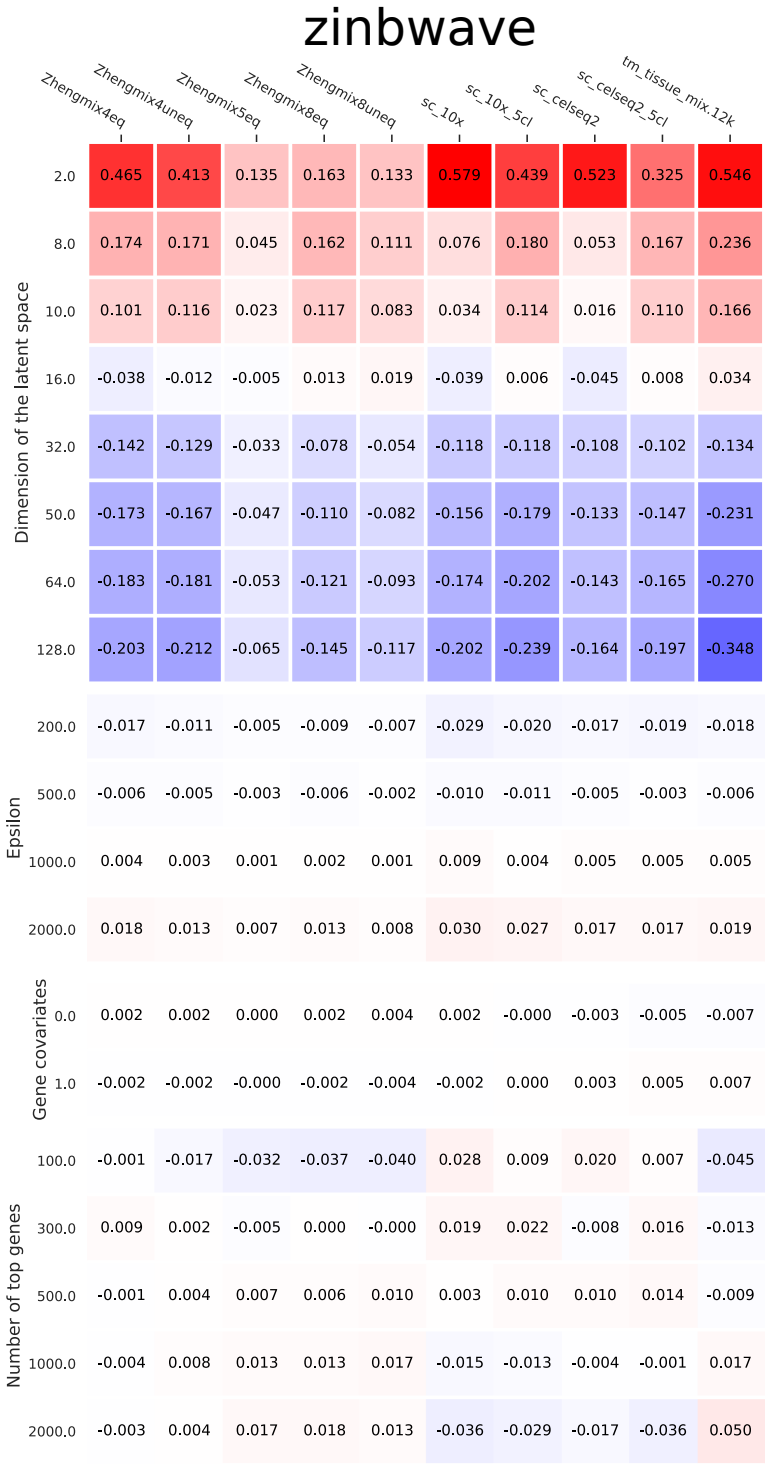


Figure S6: Heatmap visualization of the mean effect of each parameter value of ZinbWave on its silhouette Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the silhouette compared to the mean silhouette for ZinbWave on that dataset.
These effects come from a factorial ANOVA.

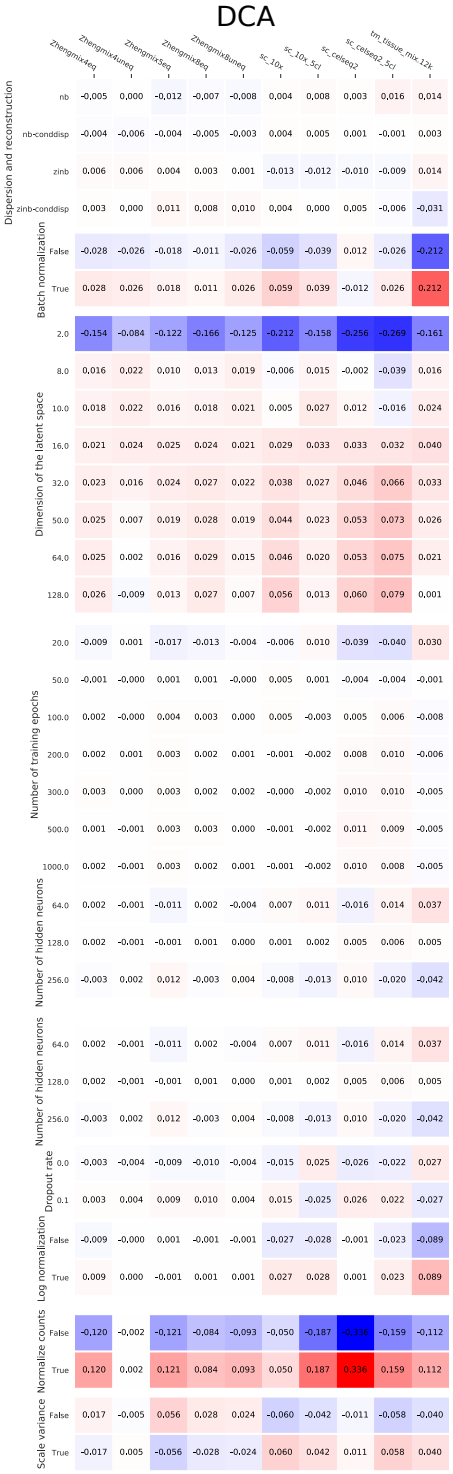


Figure S7: Heatmap visualization of the mean effect of each parameter value of DCA on its AMI. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the AMI compared to the mean AMI for DCA on that dataset.
These effects come from a factorial ANOVA.

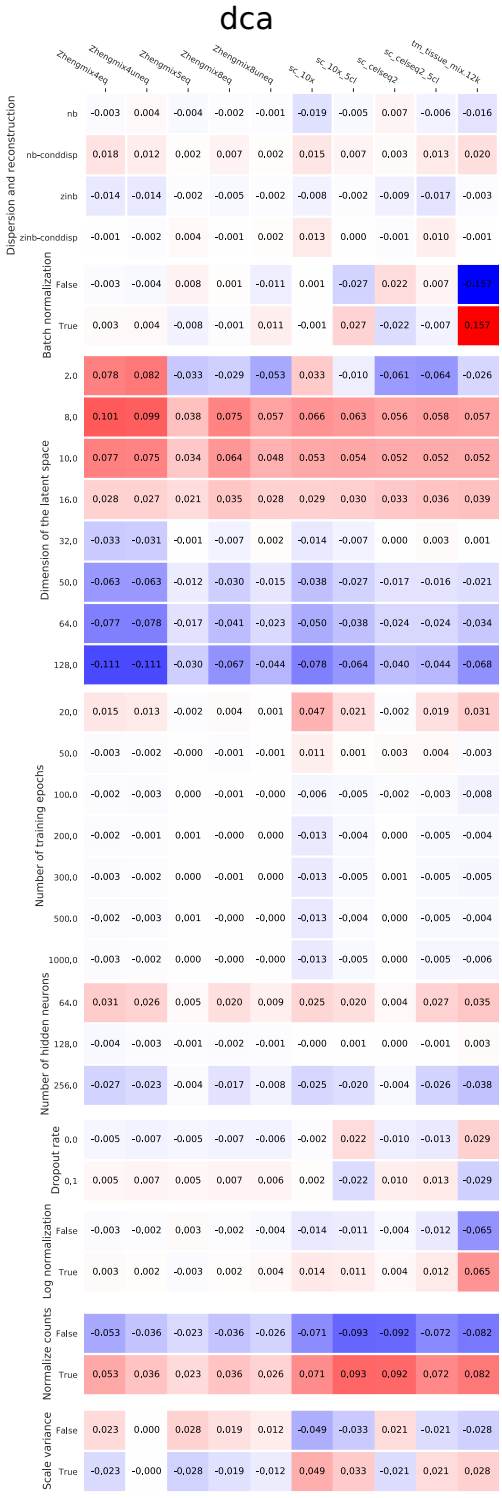


Figure S8: Heatmap visualization of the mean effect of each parameter value of DCA on its silhouette. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the silhouette compared to the mean silhouette for DCA on that dataset.
These effects come from a factorial ANOVA.

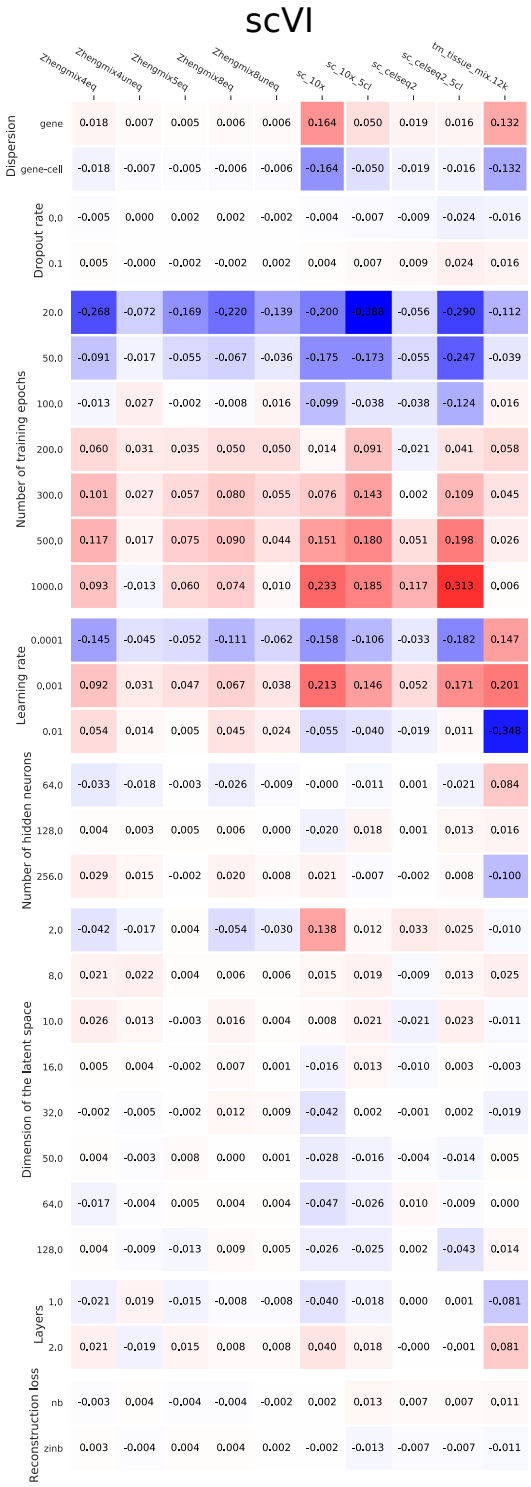


Figure S9: Heatmap visualization of the mean effect of each parameter value of scVI on its AMi. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the AMI compared to the mean AMI for scVI on that dataset.
These effects come from a factorial ANOVA.

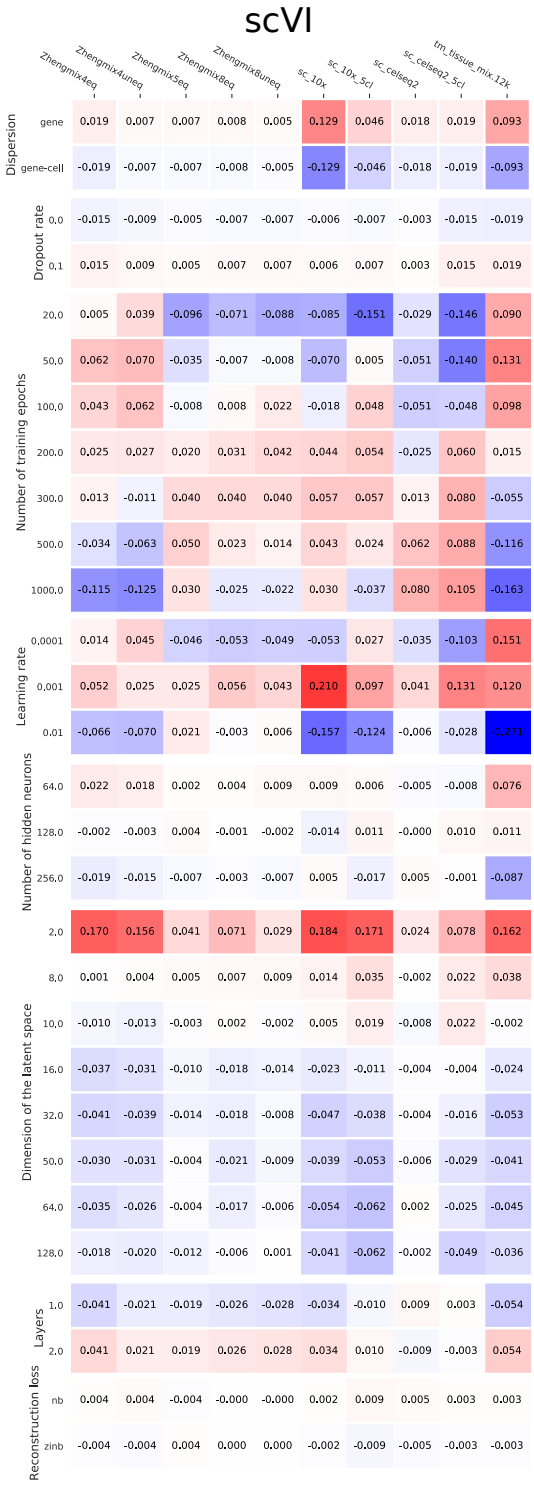


Figure S10: Heatmap visualization of the mean effect of each parameter value of scVI on its silhouette. Each columns corresponds to a dataset. The rows are split by parameter and their values, the numbers show the average effect of that parameter value on the silhouette compared to the mean silhouette for scVI on that dataset.
These effects come from a factorial ANOVA.

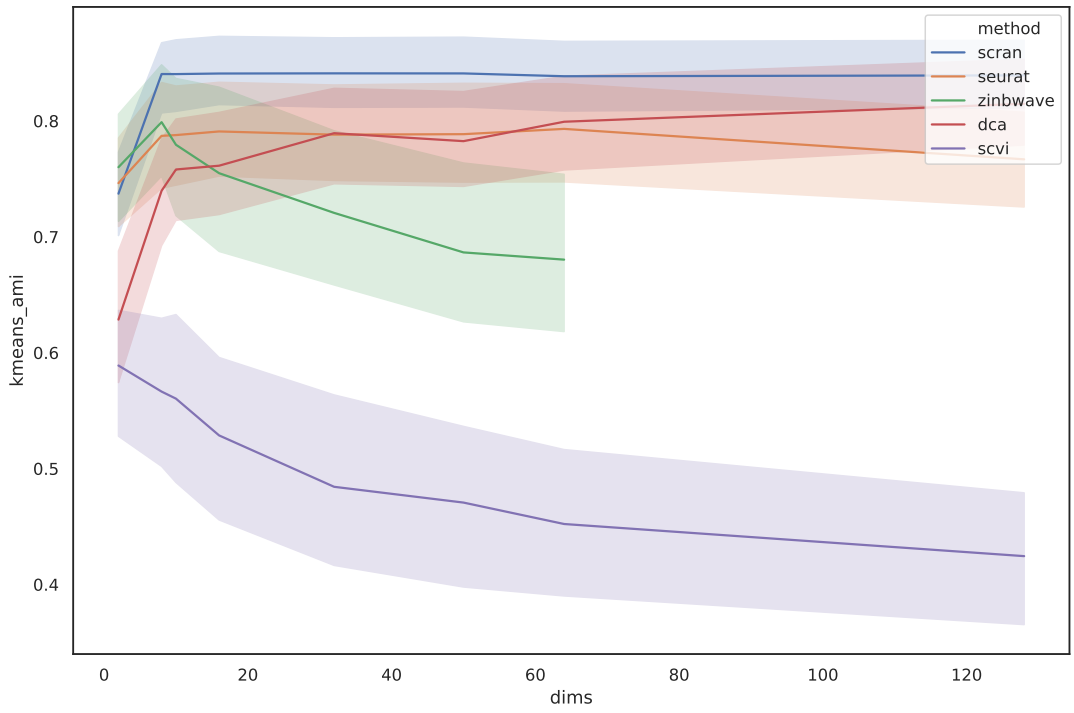


Figure S11: Mean AMI score across the 10 datasets (solid lines) for each method with default parameters. The transparent lines are the 95% confidence interval, which is large since we only have samples per point. The x axis is the dimension of the latent space, in order to observe its effect on the AMI.

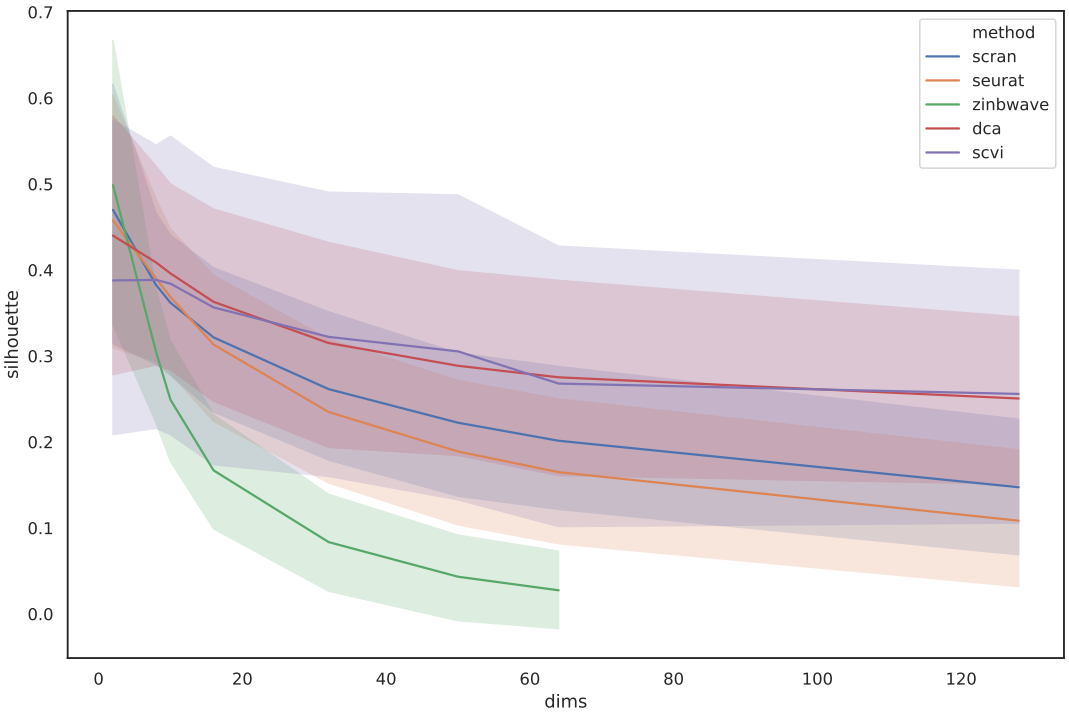


Figure S12: Mean silhouette score across the 10 datasets (solid lines) for each method with default parameters. The transparent lines are the 95% confidence interval, which is large since we only have samples per point. The x axis is the dimension of the latent space, in order to observe its effect on the silhouette.

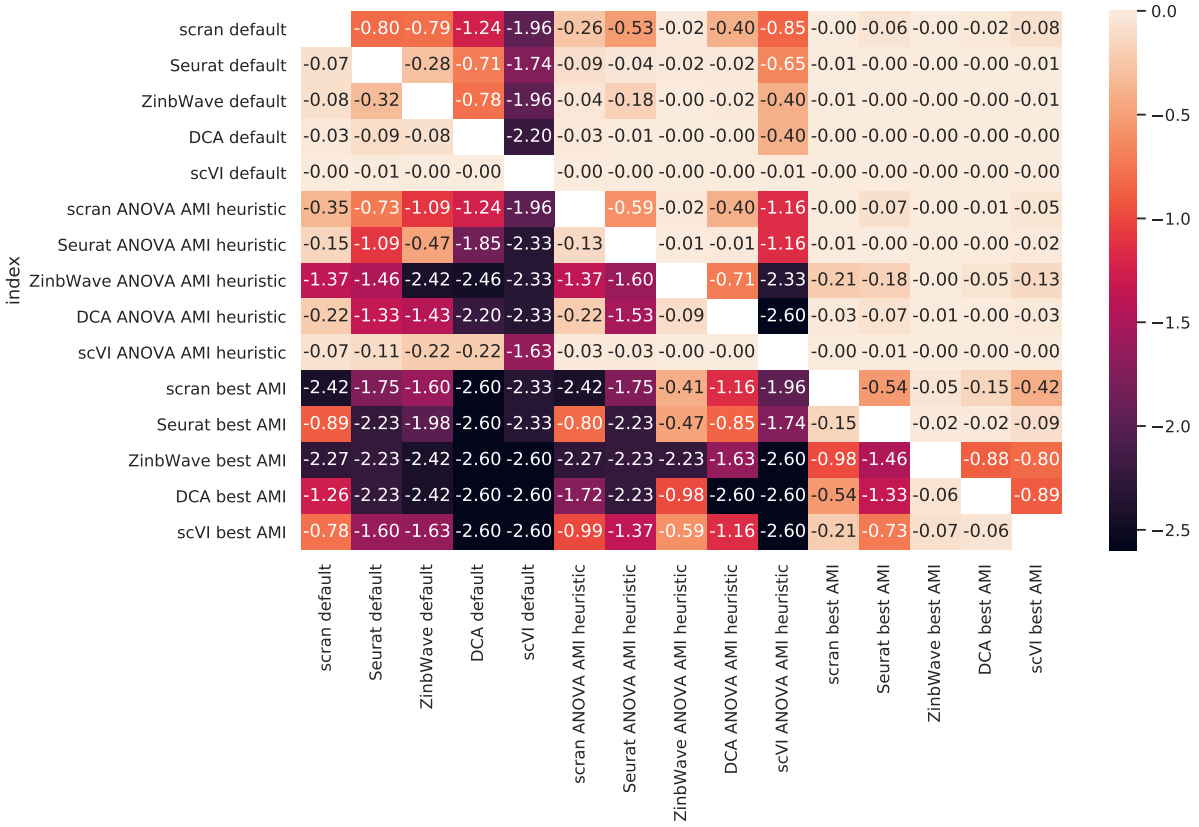


Figure S13: log base 10 p-values for the wilcoxon one-way test between the various methods and parameter configurations in AMI. A p-value of 0.05 coresponds to -1.3 in log base 10. The test is used to see if the method and parameters in the row achieve a higher AMI than the one in the column.

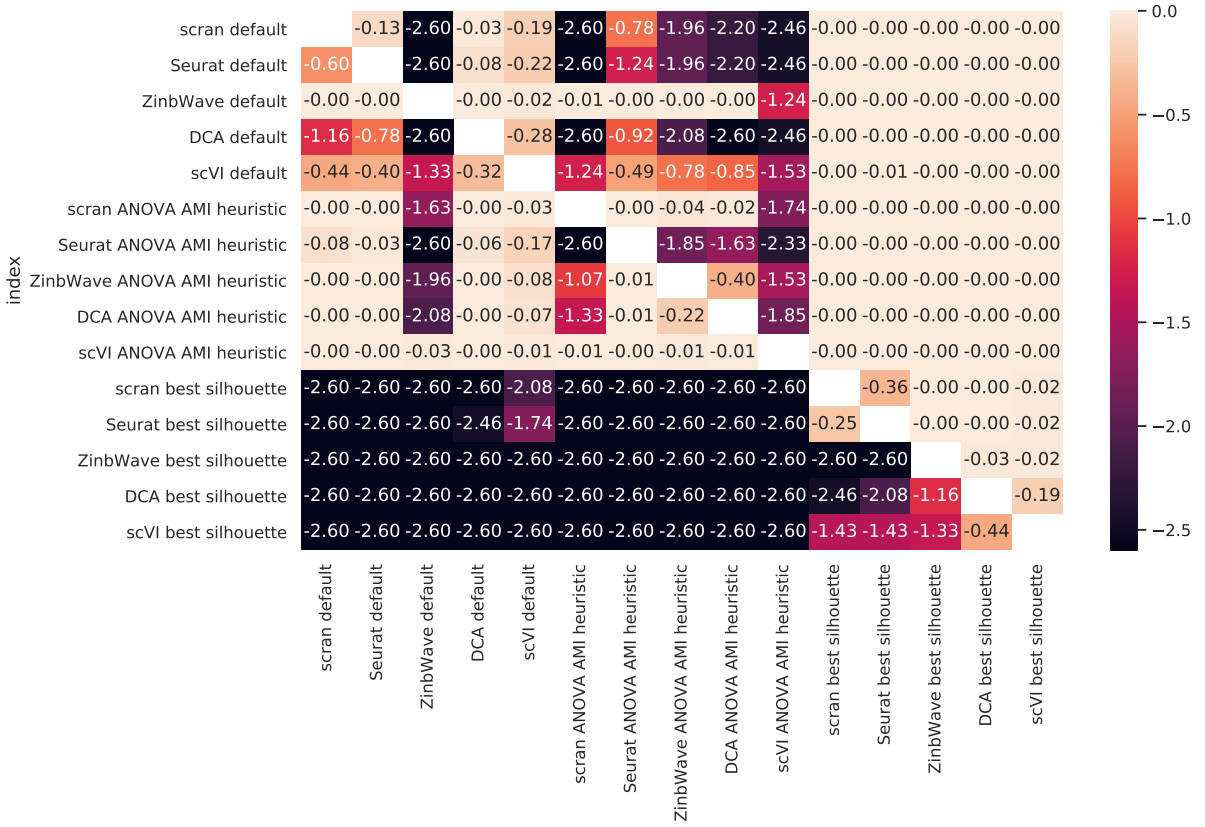


Figure S14: log base 10 p-values for the wilcoxon one-way test between the various methods and parameter configurations in silhouette. A p-value of 0.05 coresponds to -1.3 in log base 10. The test is used to see if the method and parameters in the row achieve a higher silhouette than the one in the column.

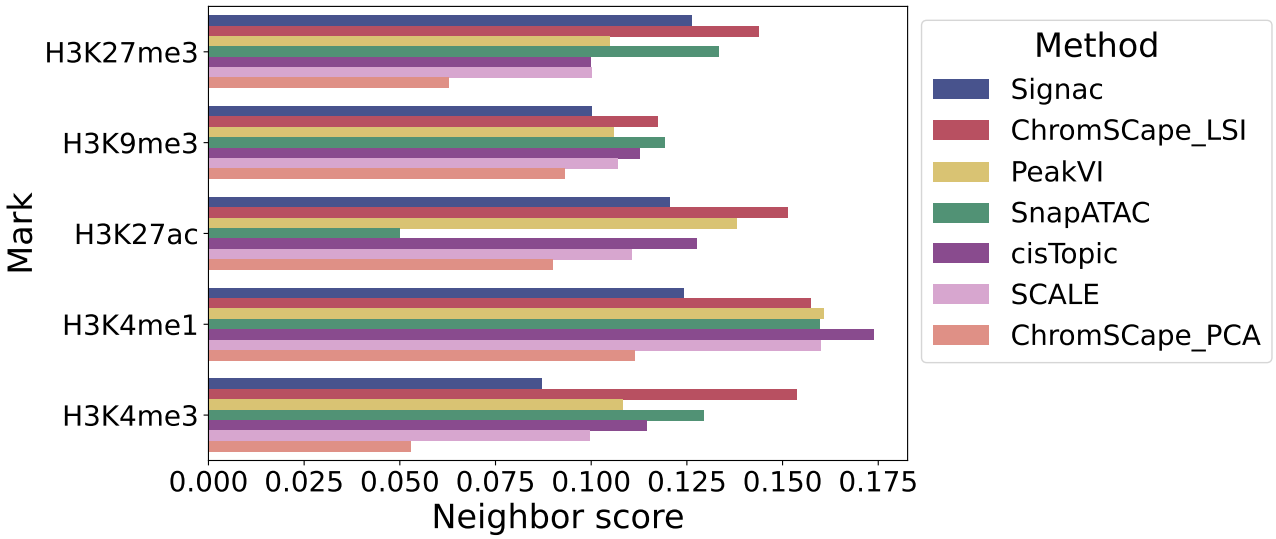


Figure S15: Best performances of the different representation methods on the human PBMC dataset.

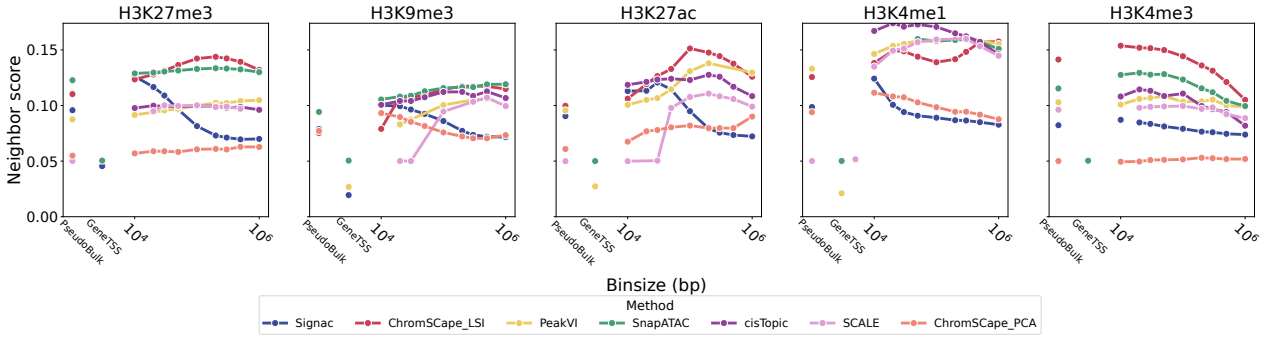


Figure S16: Performances of the 7 dimension reduction algorithms on the 5 marks in the human PBMC dataset, as a function of the matrix construction.

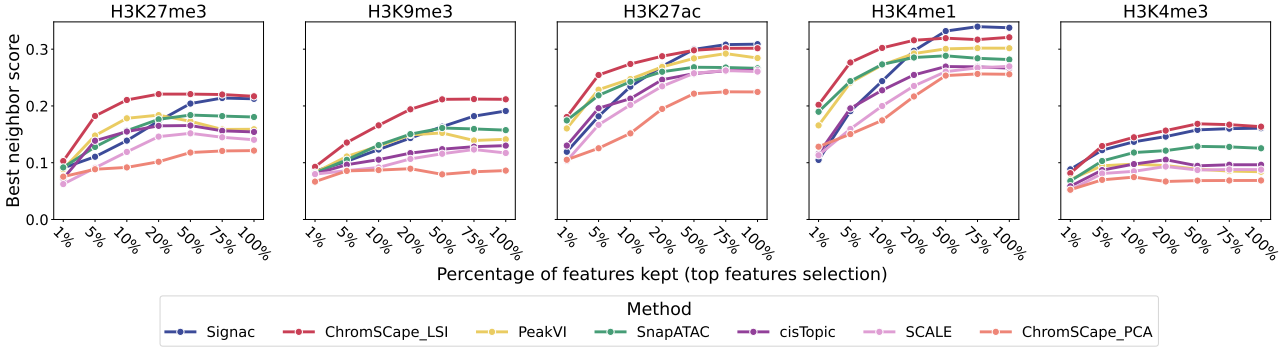


Figure S17: Role of feature selection, using the top features method used for scRNA-seq Each point corresponds to the best performance across matrix construction of a given method and a given percentage of features kept, for the 7 methods, 5 marks, and 7 features selection conditions

# Appendix B

# Supplementary tables

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| scran_n_tops | 0.45 | 5.00 | 0.09 | 35.11 | 0.0000 |
| dataset | 22.93 | 9.00 | 2.55 | 982.90 | 0.0000 |
| scran_sum_factor | 0.00 | 1.00 | 0.00 | 0.05 | 0.8268 |
| scran_ercc | 0.00 | 1.00 | 0.00 | 0.00 | 0.9968 |
| scran_assay | 135.85 | 1.00 | 135.85 | 52420.33 | 0.0000 |
| scran_n_pcs | 2.06 | 7.00 | 0.29 | 113.30 | 0.0000 |
| scran_n_tops:dataset | 3.74 | 45.00 | 0.08 | 32.04 | 0.0000 |
| dataset:scran_assay | 21.32 | 9.00 | 2.37 | 913.92 | 0.0000 |
| dataset:scran_n_pcs | 0.60 | 63.00 | 0.01 | 3.70 | 0.0000 |
| Residuals | 5.60 | 2162.00 | 0.00 | | |

Table S1: Summary result of the ANOVA for the influence of the parameters of scran on its AMI. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the AMI) to observe an F statistic this high. Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the AMI, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 19.50 | 9.00 | 2.17 | 591.87 | 0.0000 |
| scran_n_tops | 0.07 | 5.00 | 0.01 | 4.06 | 0.0011 |
| scran_sum_factor | 0.01 | 1.00 | 0.01 | 2.50 | 0.1138 |
| scran_ercc | 0.00 | 1.00 | 0.00 | 0.00 | 0.9974 |
| scran_assay | 15.45 | 1.00 | 15.45 | 4218.86 | 0.0000 |
| scran_n_pcs | 6.94 | 7.00 | 0.99 | 270.77 | 0.0000 |
| dataset:scran_n_tops | 0.42 | 45.00 | 0.01 | 2.57 | 0.0000 |
| dataset:scran_assay | 3.70 | 9.00 | 0.41 | 112.44 | 0.0000 |
| dataset:scran_n_pcs | 2.25 | 63.00 | 0.04 | 9.75 | 0.0000 |
| Residuals | 7.92 | 2162.00 | 0.00 | | |

Table S2: Summary result of the ANOVA for the influence of the parameters of scran on its silhouette. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the silhouette) to observe an F statistic this high.
Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the silhouette, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 103.26 | 9.00 | 11.47 | 2711.13 | 0.0000 |
| seurat_n_features | 0.75 | 5.00 | 0.15 | 35.53 | 0.0000 |
| seurat_n_pcs | 2.49 | 7.00 | 0.36 | 84.12 | 0.0000 |
| seurat_norm | 13.92 | 1.00 | 13.92 | 3289.15 | 0.0000 |
| seurat_find_variable | 0.23 | 2.00 | 0.12 | 27.52 | 0.0000 |
| dataset:seurat_n_features | 2.51 | 45.00 | 0.06 | 13.20 | 0.0000 |
| dataset:seurat_n_pcs | 1.12 | 63.00 | 0.02 | 4.20 | 0.0000 |
| dataset:seurat_norm | 11.95 | 9.00 | 1.33 | 313.81 | 0.0000 |
| dataset:seurat_find_variable | 1.05 | 18.00 | 0.06 | 13.84 | 0.0000 |
| Residuals | 11.51 | 2719.00 | 0.00 | | |

Table S3: Summary result of the ANOVA for the influence of the parameters of Seurat on its AMI. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the AMI) to observe an F statistic this high. Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the AMI, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 62.00 | 9.00 | 6.89 | 2855.27 | 0.0000 |
| seurat_n_features | 0.36 | 5.00 | 0.07 | 30.15 | 0.0000 |
| seurat_n_pcs | 21.87 | 7.00 | 3.12 | 1295.01 | 0.0000 |
| seurat_norm | 4.71 | 1.00 | 4.71 | 1953.86 | 0.0000 |
| seurat_find_variable | 0.02 | 2.00 | 0.01 | 3.92 | 0.0200 |
| dataset:seurat_n_features | 0.51 | 45.00 | 0.01 | 4.66 | 0.0000 |
| dataset:seurat_n_pcs | 7.03 | 63.00 | 0.11 | 46.28 | 0.0000 |
| dataset:seurat_norm | 0.57 | 9.00 | 0.06 | 26.12 | 0.0000 |
| dataset:seurat_find_variable | 0.42 | 18.00 | 0.02 | 9.56 | 0.0000 |
| Residuals | 6.56 | 2719.00 | 0.00 | | |

Table S4: Summary result of the ANOVA for the influence of the parameters of Seurat on its silhouette. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the silhouette) to observe an F statistic this high.
Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the silhouette, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 186.02 | 9.00 | 20.67 | 6293.87 | 0.0000 |
| zinbwave_dims | 5.10 | 7.00 | 0.73 | 221.82 | 0.0000 |
| zinbwave_epsilon | 0.51 | 3.00 | 0.17 | 51.78 | 0.0000 |
| zinbwave_gene_covariate | 0.02 | 1.00 | 0.02 | 6.49 | 0.0109 |
| zinbwave_keep_variance | 2.65 | 4.00 | 0.66 | 201.45 | 0.0000 |
| dataset:zinbwave_dims | 12.67 | 63.00 | 0.20 | 61.22 | 0.0000 |
| dataset:zinbwave_epsilon | 0.81 | 27.00 | 0.03 | 9.14 | 0.0000 |
| dataset:zinbwave_gene_covariate | 0.03 | 9.00 | 0.00 | 0.93 | 0.5013 |
| dataset:zinbwave_keep_variance | 2.18 | 36.00 | 0.06 | 18.45 | 0.0000 |
| Residuals | 9.56 | 2910.00 | 0.00 | | |

Table S5: Summary result of the ANOVA for the influence of the parameters of ZinbWave on its AMI. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the AMI) to observe an F statistic this high. Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the AMI, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 22.65 | 9.00 | 2.52 | 2978.44 | 0.0000 |
| zinbwave_dims | 96.94 | 7.00 | 13.85 | 16391.25 | 0.0000 |
| zinbwave_epsilon | 0.43 | 3.00 | 0.14 | 168.12 | 0.0000 |
| zinbwave_gene_covariate | 0.00 | 1.00 | 0.00 | 0.55 | 0.4577 |
| zinbwave_keep_variance | 0.10 | 4.00 | 0.03 | 30.21 | 0.0000 |
| dataset:zinbwave_dims | 17.16 | 63.00 | 0.27 | 322.33 | 0.0000 |
| dataset:zinbwave_epsilon | 0.08 | 27.00 | 0.00 | 3.65 | 0.0000 |
| dataset:zinbwave_gene_covariate | 0.04 | 9.00 | 0.00 | 4.82 | 0.0000 |
| dataset:zinbwave_keep_variance | 0.98 | 36.00 | 0.03 | 32.08 | 0.0000 |
| Residuals | 2.46 | 2910.00 | 0.00 | | |

Table S6: Summary result of the ANOVA for the influence of the parameters of ZinbWave on its silhouette. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the silhouette) to observe an F statistic this high.
Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the silhouette, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 6045.41 | 9.00 | 671.71 | 50549.34 | 0.0000 |
| ae_type | 0.21 | 3.00 | 0.07 | 5.28 | 0.0012 |
| batchnorm | 381.96 | 1.00 | 381.96 | 28744.52 | 0.0000 |
| dims | 864.32 | 7.00 | 123.47 | 9292.01 | 0.0000 |
| epochs | 2.66 | 6.00 | 0.44 | 33.39 | 0.0000 |
| normalize_per_cell | 3268.93 | 1.00 | 3268.93 | 246001.82 | 0.0000 |
| scale | 16.85 | 1.00 | 16.85 | 1268.15 | 0.0000 |
| log1p | 64.31 | 1.00 | 64.31 | 4839.39 | 0.0000 |
| hidden_dropout | 3.47 | 1.00 | 3.47 | 261.21 | 0.0000 |
| hidden | 3.95 | 2.00 | 1.97 | 148.48 | 0.0000 |
| dataset:ae_type | 15.73 | 27.00 | 0.58 | 43.84 | 0.0000 |
| dataset:batchnorm | 731.66 | 9.00 | 81.30 | 6117.89 | 0.0000 |
| dataset:dims | 135.71 | 63.00 | 2.15 | 162.11 | 0.0000 |
| dataset:epochs | 14.74 | 54.00 | 0.27 | 20.54 | 0.0000 |
| dataset:normalize_per_cell | 1432.67 | 9.00 | 159.19 | 11979.42 | 0.0000 |
| dataset:scale | 294.29 | 9.00 | 32.70 | 2460.72 | 0.0000 |
| dataset:log1p | 145.38 | 9.00 | 16.15 | 1215.58 | 0.0000 |
| dataset:hidden_dropout | 58.16 | 9.00 | 6.46 | 486.27 | 0.0000 |
| dataset:hidden | 29.01 | 18.00 | 1.61 | 121.30 | 0.0000 |
| Residuals | 2727.97 | 205292.00 | 0.01 | | |

Table S7: Summary result of the ANOVA for the influence of the parameters of DCA on its AMI. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the AMI) to observe an F statistic this high. Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the AMI, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset | 3542.22 | 9.00 | 393.58 | 46229.81 | 0.0000 |
| ae_type | 9.36 | 3.00 | 3.12 | 366.43 | 0.0000 |
| batchnorm | 53.66 | 1.00 | 53.66 | 6303.34 | 0.0000 |
| dims | 396.37 | 7.00 | 56.62 | 6651.01 | 0.0000 |
| epochs | 7.71 | 6.00 | 1.28 | 150.92 | 0.0000 |
| normalize_per_cell | 700.98 | 1.00 | 700.98 | 82336.34 | 0.0000 |
| scale | 1.56 | 1.00 | 1.56 | 183.13 | 0.0000 |
| log1p | 26.88 | 1.00 | 26.88 | 3157.50 | 0.0000 |
| hidden_dropout | 0.06 | 1.00 | 0.06 | 7.41 | 0.0065 |
| hidden | 52.00 | 2.00 | 26.00 | 3054.02 | 0.0000 |
| dataset:ae_type | 7.71 | 27.00 | 0.29 | 33.56 | 0.0000 |
| dataset:batchnorm | 494.51 | 9.00 | 54.95 | 6453.93 | 0.0000 |
| dataset:dims | 124.46 | 63.00 | 1.98 | 232.05 | 0.0000 |
| dataset:epochs | 8.94 | 54.00 | 0.17 | 19.46 | 0.0000 |
| dataset:normalize_per_cell | 131.01 | 9.00 | 14.56 | 1709.83 | 0.0000 |
| dataset:scale | 142.09 | 9.00 | 15.79 | 1854.47 | 0.0000 |
| dataset:log1p | 72.51 | 9.00 | 8.06 | 946.32 | 0.0000 |
| dataset:hidden_dropout | 37.63 | 9.00 | 4.18 | 491.17 | 0.0000 |
| dataset:hidden | 14.03 | 18.00 | 0.78 | 91.53 | 0.0000 |
| Residuals | 1747.77 | 205292.00 | 0.01 | | |

Table S8: Summary result of the ANOVA for the influence of the parameters of DCA on its silhouette. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the silhouette) to observe an F statistic this high.
Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the silhouette, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| n_latent | 4.07 | 7.00 | 0.58 | 12.56 | 0.0000 |
| dataset | 3304.57 | 9.00 | 367.17 | 7934.53 | 0.0000 |
| epochs | 836.88 | 6.00 | 139.48 | 3014.13 | 0.0000 |
| dispersion | 143.17 | 1.00 | 143.17 | 3093.80 | 0.0000 |
| n_layers | 23.07 | 1.00 | 23.07 | 498.51 | 0.0000 |
| n_hidden | 0.90 | 2.00 | 0.45 | 9.78 | 0.0001 |
| dropout_rate | 2.98 | 1.00 | 2.98 | 64.44 | 0.0000 |
| lr | 471.85 | 2.00 | 235.92 | 5098.25 | 0.0000 |
| reconstruction_loss | 0.76 | 1.00 | 0.76 | 16.35 | 0.0001 |
| n_latent:dataset | 39.76 | 63.00 | 0.63 | 13.64 | 0.0000 |
| dataset:epochs | 383.74 | 54.00 | 7.11 | 153.57 | 0.0000 |
| dataset:dispersion | 240.35 | 9.00 | 26.71 | 577.09 | 0.0000 |
| dataset:n_layers | 54.17 | 9.00 | 6.02 | 130.08 | 0.0000 |
| dataset:n_hidden | 61.33 | 18.00 | 3.41 | 73.63 | 0.0000 |
| dataset:dropout_rate | 5.08 | 9.00 | 0.56 | 12.20 | 0.0000 |
| dataset:lr | 657.45 | 18.00 | 36.53 | 789.30 | 0.0000 |
| dataset:reconstruction_loss | 2.88 | 9.00 | 0.32 | 6.91 | 0.0000 |
| Residuals | 3679.78 | 79519.00 | 0.05 | | |

Table S9: Summary result of the ANOVA for the influence of the parameters of scVI on its AMI. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the AMI) to observe an F statistic this high. Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the AMI, it is used to represent the inherent complexity of the data.

| | SS | df | MS | F value | Pr(>F) |
|---|---|---|---|---|---|
| n_latent | 151.33 | 7.00 | 21.62 | 618.87 | 0.0000 |
| dataset | 914.48 | 9.00 | 101.61 | 2908.80 | 0.0000 |
| epochs | 61.21 | 6.00 | 10.20 | 292.04 | 0.0000 |
| dispersion | 98.93 | 1.00 | 98.93 | 2832.20 | 0.0000 |
| n_layers | 39.34 | 1.00 | 39.34 | 1126.30 | 0.0000 |
| n_hidden | 10.57 | 2.00 | 5.28 | 151.29 | 0.0000 |
| dropout_rate | 6.65 | 1.00 | 6.65 | 190.50 | 0.0000 |
| lr | 302.15 | 2.00 | 151.07 | 4324.90 | 0.0000 |
| reconstruction_loss | 0.53 | 1.00 | 0.53 | 15.27 | 0.0001 |
| n_latent:dataset | 56.63 | 63.00 | 0.90 | 25.73 | 0.0000 |
| dataset:epochs | 293.43 | 54.00 | 5.43 | 155.56 | 0.0000 |
| dataset:dispersion | 130.38 | 9.00 | 14.49 | 414.73 | 0.0000 |
| dataset:n_layers | 27.01 | 9.00 | 3.00 | 85.91 | 0.0000 |
| dataset:n_hidden | 32.82 | 18.00 | 1.82 | 52.19 | 0.0000 |
| dataset:dropout_rate | 1.94 | 9.00 | 0.22 | 6.18 | 0.0000 |
| dataset:lr | 410.50 | 18.00 | 22.81 | 652.87 | 0.0000 |
| dataset:reconstruction_loss | 0.91 | 9.00 | 0.10 | 2.88 | 0.0021 |
| Residuals | 2777.70 | 79519.00 | 0.03 | | |

Table S10: Summary result of the ANOVA for the influence of the parameters of scVI on its silhouette. "SS" corresponds to the variance explained by a parameter, "df" its number of degrees of freedom, "MS" is "SS" divided by "df", i.e. the mean variance explained by each degree of freedom, "F value", is the observed F statistic, and "Pr(>F)" is the probability under the null hypothesis (this parameter has no influence on the silhouette) to observe an F statistic this high.
Each row corresponds to a factor, when they follow the format "parameter" it is the effect that this parameter has on average, when they follow the format "dataset:parameter" it is the effect of the parameter on each specific dataset (the interaction factors), "dataset" is a special one that represents the effect of the dataset on the silhouette, it is used to represent the inherent complexity of the data.

| Method | Parameter | AMI effect | AMI best | AMI worst | AMI distance |
|---|---|---|---|---|---|
| scran | Log normalization | 0.504701 | logcounts | counts | 0.000000 |
| | Dimension of the latent space | 0.095888 | 32 | 2 | 0.011168 |
| | Number of top genes | 0.046545 | 300 | 100 | 0.070318 |
| | Sum factors normalization | 0.000485 | 0 | 1 | |
| | ERCC factor normalization | 0.000013 | 0 | 1 | |
| Seurat | Choice of top features | 0.020766 | disp | mvp | 0.031045 |
| | Number of top genes | 0.046823 | 500 | 100 | 0.058046 |
| | Dimension of the latent space | 0.090035 | 10 | 2 | 0.006698 |
| | Normalization method | 0.139065 | LogNormalize | CLR | 0.000000 |
| ZinbWave | Dimension of the latent space | 0.130836 | 8.0 | 128.0 | 0.127330 |
| | Epsilon | 0.034987 | 2000.0 | 200.0 | 0.003970 |
| | Gene covariates | 0.005273 | 1.0 | 0.0 | 0.001516 |
| | Number of top genes | 0.091435 | 2000.0 | 100.0 | 0.020083 |
| DCA | Dispersion and reconstruction | 0.002516 | nb | zinb | 0.021833 |
| | Batch normalization | 0.086200 | True | False | 0.023997 |
| | Dimension of the latent space | 0.202636 | 32 | 2 | 0.017934 |
| | Number of training epochs | 0.010789 | 300 | 20 | 0.035311 |
| | Number of hidden neurons | 0.010530 | 64 | 256 | 0.026189 |
| | Dropout rate | 0.008436 | 0.1 | 0 | 0.053644 |
| | Log normalization | 0.035889 | True | False | 0.001784 |
| | Normalize counts | 0.252914 | True | False | 0.000000 |
| | Scale variance | 0.018196 | True | False | 0.112139 |
| scVI | Dispersion | 0.084756 | gene | gene-cell | 0.000000 |
| | Dropout rate | 0.012287 | 0.1 | 0 | 0.005222 |
| | Number of training epochs | 0.299515 | 1000 | 20 | 0.052207 |
| | Learning rate | 0.180853 | 0.001 | 0.0001 | 0.000000 |
| | Number of hidden neurons | 0.008113 | 128 | 64 | 0.067728 |
| | Dimension of the latent space | 0.019958 | 8 | 64 | 0.123639 |
| | Layers | 0.033922 | 2 | 1 | 0.038795 |
| | Reconstruction loss | 0.006086 | nb | zinb | 0.008399 |

Table S11: Summary of the parameter influence on the AMI. The column "AMI effect" is the maximum difference between the mean effect of the parameters on the AMI As explained in Section 3.2.4. The column "AMI best" is the parameter value with the best mean effect, and are the ones used in the "ANOVA AMI heuristic". The column "AMI worst" is the parameter value with the worst mean effect. The column "AMI distance" is the maximum distance between the parameter values with the best effect on a dataset specific way, and the "AMI best" effect on a dataset specific way.

| Method | Parameter | silhouette effect | silhouette best | silhouette worst | silhouette distance |
|---|---|---|---|---|---|
| scran | Log normalization | 0.170177 | logcounts | counts | 0.000000 |
| | Dimension of the latent space | 0.167868 | 8 | 128 | 0.132922 |
| | Number of top genes | 0.017043 | 300 | 3000 | 0.035144 |
| | Sum factors normalization | 0.004166 | 0 | 1 | |
| | ERCC factor normalization | 0.000000 | 0 | 0 | |
| Seurat | Choice of top features | 0.006136 | disp | mvp | 0.026131 |
| | Number of top genes | 0.033238 | 1000 | 100 | 0.020503 |
| | Dimension of the latent space | 0.250641 | 2 | 128 | 0.119350 |
| | Normalization method | 0.080933 | LogNormalize | CLR | 0.000000 |
| ZinbWave | Dimension of the latent space | 0.561425 | 2.0 | 128.0 | 0.000000 |
| | Epsilon | 0.031739 | 2000.0 | 200.0 | 0.000000 |
| | Gene covariates | 0.000780 | 1.0 | 0.0 | 0.007692 |
| | Number of top genes | 0.016251 | 500.0 | 100.0 | 0.059128 |
| DCA | Dispersion and reconstruction | 0.017509 | nb-conddisp | zinb | 0.003257 |
| | Batch normalization | 0.032509 | True | False | 0.043511 |
| | Dimension of the latent space | 0.132412 | 8 | 128 | 0.000000 |
| | Number of training epochs | 0.018071 | 20 | 1000 | 0.004797 |
| | Number of hidden neurons | 0.039340 | 64 | 256 | 0.000000 |
| | Dropout rate | 0.001184 | 0.1 | 0 | 0.057457 |
| | Log normalization | 0.023018 | True | False | 0.005908 |
| | Normalize counts | 0.117338 | True | False | 0.000000 |
| | Scale variance | 0.005635 | True | False | 0.055650 |
| scVI | Dispersion | 0.070433 | gene | gene-cell | 0.000000 |
| | Dropout rate | 0.018295 | 0.1 | 0 | 0.000000 |
| | Number of training epochs | 0.082262 | 200 | 20 | 0.115803 |
| | Learning rate | 0.149796 | 0.001 | 0.01 | 0.031726 |
| | Number of hidden neurons | 0.028093 | 64 | 256 | 0.017822 |
| | Dimension of the latent space | 0.136278 | 2 | 32 | 0.000000 |
| | Layers | 0.044539 | 2 | 1 | 0.018577 |
| | Reconstruction loss | 0.005252 | nb | zinb | 0.007876 |

Table S12: Summary of the parameter influence on the silhouette. The column "silhouette effect" is the maximum difference between the mean effect of the parameters on the silhouette As explained in Section 3.2.4. The column "silhouette best" is the parameter value with the best mean effect, and are the ones used in the "ANOVA silhouette heuristic". The column "silhouette worst" is the parameter value with the worst mean effect. The column "silhouette distance" is the maximum distance between the parameter values with the best effect on a dataset specific way, and the "silhouette best" effect on a dataset specific way.

| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
|---|---|---|---|---|---|
| Signac | 88.2% | 88.5% | 90.3% | 93.6% | 84.8% |
| ChromSCape_LSI | 80.0% | 80.3% | 78.8% | 93.9% | 78.2% |
| PeakVI | 86.1% | 87.9% | 85.2% | 84.5% | 77.3% |
| SnapATAC | 88.2% | 89.1% | 90.6% | 94.8% | 84.8% |
| cisTopic | 88.2% | 88.5% | 90.3% | 93.6% | 84.8% |
| SCALE | 86.1% | 87.3% | 90.6% | 92.1% | 85.2% |
| ChromSCape_PCA | 88.2% | 88.5% | 90.3% | 93.6% | 84.8% |

Table S13: Percentage of successful runs on the mouse brain data.

| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
|---|---|---|---|---|---|
| Signac | 91.7% | 91.7% | 91.7% | 91.7% | 83.3% |
| ChromSCape_LSI | 83.3% | 83.3% | 83.3% | 83.3% | 83.3% |
| PeakVI | 83.3% | 50.0% | 75.0% | 83.3% | 91.7% |
| SnapATAC | 91.7% | 91.7% | 8.3% | 58.3% | 91.7% |
| cisTopic | 75.0% | 75.0% | 75.0% | 75.0% | 75.0% |
| SCALE | 75.0% | 50.0% | 83.3% | 83.3% | 83.3% |
| ChromSCape_PCA | 83.3% | 83.3% | 83.3% | 83.3% | 83.3% |

Table S14: Percentage of successful runs on the human PBMC data.

| Method | neighbor score | | | | |
|---|---|---|---|---|---|
| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
| Signac | 0.213 | 0.191 | **0.309** | **0.338** | 0.161 |
| ChromSCape_LSI | **0.217** | **0.212** | 0.302 | 0.321 | **0.164** |
| PeakVI | 0.159 | 0.141 | 0.284 | 0.302 | 0.084 |
| SnapATAC | 0.180 | 0.157 | 0.266 | 0.282 | 0.125 |
| cisTopic | 0.154 | 0.130 | 0.263 | 0.267 | 0.096 |
| SCALE | 0.140 | 0.117 | 0.261 | 0.270 | 0.088 |
| ChromSCape_PCA | 0.121 | 0.086 | 0.225 | 0.256 | 0.069 |

Table S15: Best performance of each method across feature engineering methods on the mouse brain dataset, the best performing method for each mark is bolded.

| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
|---|---|---|---|---|---|
| H3K27me3 | | **0.0078** | 1.0000 | 1.0000 | **0.0078** |
| H3K9me3 | 1.0000 | | 1.0000 | 1.0000 | **0.0078** |
| H3K27ac | **0.0078** | **0.0078** | | 1.0000 | **0.0078** |
| H3K4me1 | **0.0078** | **0.0078** | **0.0078** | | **0.0078** |
| H3K4me3 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | |

Table S16: p-values for paired Wilcoxon one-sided (line greater than column) test between the different marks on the mouse brain data.

| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
|---|---|---|---|---|---|
| H3K27me3 | | 0.4688 | 0.8516 | 0.9922 | 0.4688 |
| H3K9me3 | 0.5938 | | 0.8516 | 1.0000 | 0.4688 |
| H3K27ac | 0.1875 | 0.1875 | | 1.0000 | 0.1875 |
| H3K4me1 | **0.0156** | **0.0078** | **0.0078** | | **0.0078** |
| H3K4me3 | 0.5938 | 0.5938 | 0.8516 | 1.0000 | |

Table S17: p-values for paired Wilcoxon one-sided (line greater than column) test between the different marks on the human PBMC data.

| | Signac | ChromSCape_LSI | PeakVI | SnapATAC | cisTopic | SCALE | Chro |
|---|---|---|---|---|---|---|---|
| Signac | | 0.9863 | 0.0801 | 0.1377 | 0.0801 | **0.0322** | |
| ChromSCape_LSI | **0.0186** | | **0.0020** | **0.0049** | **0.0029** | **0.0020** | |
| PeakVI | 0.9346 | 0.9990 | | 0.7842 | 0.3125 | **0.0098** | |
| SnapATAC | 0.8838 | 0.9971 | 0.2461 | | 0.0801 | 0.0527 | |
| cisTopic | 0.9346 | 0.9980 | 0.7217 | 0.9346 | | **0.0068** | |
| SCALE | 0.9756 | 0.9990 | 0.9932 | 0.9580 | 0.9951 | | |
| ChromSCape_PCA | 1.0000 | 1.0000 | 1.0000 | 0.9971 | 1.0000 | 1.0000 | |

Table S18: p-values for paired Wilcoxon one-sided (line greater than column) test between the different methods accross both the human PBMC and mouse brain data.

| | Signac | ChromSCape_LSI | PeakVI | SnapATAC | cisTopic | SCALE | Chron |
|---|---|---|---|---|---|---|---|
| Signac | | 0.5938 | **0.0312** | **0.0312** | **0.0312** | **0.0312** | |
| ChromSCape_LSI | 0.5000 | | **0.0312** | **0.0312** | **0.0312** | **0.0312** | |
| PeakVI | 1.0000 | 1.0000 | | 0.7812 | 0.1562 | 0.0625 | |
| SnapATAC | 1.0000 | 1.0000 | 0.3125 | | **0.0312** | **0.0312** | |
| cisTopic | 1.0000 | 1.0000 | 0.9062 | 1.0000 | | 0.0938 | |
| SCALE | 1.0000 | 1.0000 | 0.9688 | 1.0000 | 0.9375 | | |
| ChromSCape_PCA | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | |

Table S19: p-values for paired Wilcoxon one-sided (line greater than column) test between the different methods on the mouse brain data.

| Method | ratio |
|---|---|
| Chromscape_LSI | 1.886 |
| Chromscape_PCA | 1.855 |
| PeakVI | 1.793 |
| SCALE | 1.449 |
| Signac | 1.796 |
| SnapATAC | 1.445 |
| cisTopic | 1.399 |

Table S20: Ratio between the best and worst performances for each method across matrix construction, with no preprocessing, averaged over the mouse brain dataset marks.

| Mark | ratio |
|---|---|
| H3K27ac | 1.517 |
| H3K27me3 | 2.022 |
| H3K4me1 | 1.550 |
| H3K4me3 | 1.606 |
| H3K9me3 | 1.608 |

Table S21: Ratio between the best and worst performances for each mark of the mouse brain dataset across matrix construction, with no preprocessing, averaged over the 7 methods.

| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
|---|---|---|---|---|---|
| Signac | 1.96 | 1.96 | 1.65 | 1.79 | 1.62 |
| ChromSCape_LSI | 2.54 | 1.95 | 1.37 | 1.22 | 2.35 |
| PeakVI | 2.83 | 1.57 | 1.65 | 1.36 | 1.57 |
| SnapATAC | 1.58 | 1.61 | 1.28 | 1.37 | 1.38 |
| cisTopic | 1.43 | 1.35 | 1.35 | 1.54 | 1.33 |
| SCALE | 1.80 | 1.33 | 1.38 | 1.43 | 1.30 |
| ChromSCape_PCA | 2.01 | 1.49 | 1.94 | 2.14 | 1.70 |

Table S22: Ratio between the best and worst performances across matrix construction on the raw data (no feature or cell selection applied) for each mark and method combination on the mouse brain dataset.

| | H3K27me3 | H3K9me3 | H3K27ac | H3K4me1 | H3K4me3 |
|---|---|---|---|---|---|
| Signac | 2.78 | 5.16 | 4.41 | 2.47 | 1.18 |
| ChromSCape_LSI | 1.30 | 1.56 | 1.52 | 1.25 | 1.46 |
| PeakVI | 1.20 | 3.95 | 5.07 | 7.64 | 1.10 |
| SnapATAC | 2.65 | 2.36 | 1.00 | 3.19 | 2.57 |
| cisTopic | 1.04 | 1.12 | 1.18 | 1.19 | 1.40 |
| SCALE | 2.00 | 2.14 | 2.22 | 3.19 | 1.12 |
| ChromSCape_PCA | 1.14 | 1.32 | 1.48 | 1.27 | 1.07 |

Table S23: Ratio between the best and worst performances across matrix construction on the raw data (no feature or cell selection applied) for each mark and method combination on the human PBMC dataset.

| Mark | best increase Signac | best increase ChromSCape_LSI |
|---|---|---|
| H3K27me3 | 1.104 | 1.071 |
| H3K9me3 | 1.073 | 1.043 |
| H3K27ac | 1.110 | 1.095 |
| H3K4me1 | 1.149 | 1.133 |
| H3K4me3 | 1.008 | 1.082 |

Table S24: Ratio of the performances between the best coverage threshold and the worst one for each mark on the mouse brain dataset.

| Method | best increase |
|---|---|
| Chromscape_LSI | 1.085 |
| Chromscape_PCA | 1.409 |
| PeakVI | 1.076 |
| SCALE | 1.211 |
| Signac | 1.089 |
| SnapATAC | 1.098 |
| cisTopic | 1.082 |

Table S25: Ratio of the performances between between the best coverage threshold and the worst one, averaged by method on the mouse brain dataset.

| | Signac | | ChromSCape_LSI | |
|---|---|---|---|---|
| Mark | From 40% | From 60% | From 40% | From 60% |
| H3K27ac | 1.091 | 1.047 | 1.088 | 1.028 |
| H3K27me3 | 1.155 | 1.061 | 1.118 | 1.084 |
| H3K9me3 | 1.143 | 1.089 | 1.096 | 1.017 |
| H3K4me1 | 1.057 | 1.013 | 1.057 | 1.025 |
| H3K4me3 | 1.172 | 1.081 | 1.185 | 1.075 |

Table S26: Ratio of the performances, averaged by mark, between having all the cells present and having either only 40% or 60% of themin the mouse brain dataset.

| Method | From 40% | From 60% |
|---|---|---|
| Chromscape_LSI | 1.109 | 1.046 |
| Chromscape_PCA | 1.188 | 1.092 |
| PeakVI | 1.342 | 1.180 |
| SCALE | 1.290 | 1.159 |
| Signac | 1.124 | 1.058 |
| SnapATAC | 1.084 | 1.038 |
| cisTopic | 1.210 | 1.108 |

Table S27: Ratio of the performances, averaged by method, between having all the cells present and having either only 40% or 60% of themin the mouse brain dataset.

| | Signac | | ChromSCape_LSI | |
|---|---|---|---|---|
| Mark | large increase | baseline | large increase | baseline |
| H3K27ac | 1.708 | 1.150 | 1.666 | 1.119 |
| H3K27me3 | 1.711 | 1.216 | 1.464 | 1.170 |
| H3K9me3 | 1.555 | 1.223 | 1.355 | 1.140 |
| H3K4me1 | 1.688 | 1.193 | 1.754 | 1.204 |
| H3K4me3 | 1.682 | 1.087 | 2.067 | 1.220 |

Table S28: Ratio of the performances between having high coverage (q50_100 condition) in the mouse brain dataset, and either low coverage or baseline coverage, averaged by mark. The "large increase" column is the increase in performance observed against q0_50 where we select the cells with the lowest coverage. The "baseline" column is the increase against no selection.

| Method | large increase | baseline |
|---|---|---|
| Chromscape_LSI | 1.661 | 1.170 |
| Chromscape_PCA | 1.791 | 1.502 |
| PeakVI | 1.561 | 1.152 |
| SCALE | 1.504 | 1.467 |
| Signac | 1.669 | 1.174 |
| SnapATAC | 1.421 | 1.143 |
| cisTopic | 1.361 | 1.174 |

Table S29: Ratio of the performances between having high coverage (q50_100 condition) in the mouse brain dataset, and either low coverage or baseline coverage, averaged by method. The "large increase" column is the increase in performance observed against q0_50 where we select the cells with the lowest coverage. The "baseline" column is the increase against no selection.

# Appendix C

# Supplementary text

## C.1 scRNA-seq Immune cell mixtures generator

### C.1.1 Introduction

Single-cell RNA sequencing (scRNA-seq) is becoming a standard assay to understand the biological heterogeneity of tissues and tumors. recent years have seen an explosion in the number of computational methods to analyze such data, as well as the kind of questions that it can answer. This has lead the community to become interested in having solid benchmarks of these methods [201], however one of the limiting factors in our ability to benchmarks these data is the current small number of datasets annotated with ground truth data (i.e. that does not come from another computational method).
One solution to this limitation is to simulate data with tools such as Splatter [202], powsimR [203], or SymSim [204]; however all these tools simulate data based on their own modelling and may not be representative of actual data. Another solution is to manually mix in silico purified cell populations whose cell type is known in advance, in this application note we provide a software to automatically generate data following that principle, based on the immune cell populations published in [170] as first done in [166]. Extension to other cell populations is also straightforward.

### C.1.2 Library

In [166], Duo et al. introduced 3 datasets (Zhengmix4eq, Zhengmix4uneq, and Zhengmix8eq) that were created using an in-silico mixture of the FACS-purified immune cell populations from [170]. These dataset were made available through the DuoClustering Bioconductor package, and have since been used in most benchmarks as well as standard evaluations when developing new methods. However these three datasets alone cannot answer all biological questions, in particular they cannot answer questions about the influence of the number of cells or the sensitivity of the methods to small populations.

In this project, we want to extend these three datasets by allowing users to specify the kind of mixtures that they want, as well as the kind of quality control (QC) steps they want to apply (the three previous datasets do not do any QC on the amount of mitochondrial RNA).

We are providing this with a package that can be used "off the shelf" while staying modular enough that it can be used by other practioners to build datasets specifically made to challenge their methods on some specific questions. Users can generate the original data from 10x with a simple bash script, specify their mixtures as well as the QC steps with a simple R script, and can generate the data in common formats such as SingleCellExperiment [205], Seurat object [206], CSV, Loompy object or AnnData [187] (the last two being generated from the CSV with a Python script we provide). By defaults the seed for the subsampling is the same as in [166], but it can also be changed in order to have multiple mixes of the same sizes. This way the method can be run on multiple very similar datasets in order to have confidence intervals in a fashion similar to bootstrap.

The command to regenerate the Zhengmix4eq data in SingleCellExperiment format will be:

Listing C.1: Generating Zhengmix4eq in SingleCellExperiment

```
svn export https://github.com/google-research/google-research/trunk/cell_m
cd cell_mixer
bash fetch_10x.sh --data_path=data
Rscript cell_mixer.R \
--data_path=data \
--format=SingleCellExperiment \
--name=Zhengmix4eq \
--qc_count_mad_lower=3 \
--qc_feature_count_mad_lower=3 \
--qc_mito_mad_upper=-1 \
--b_cells=1000 \
--naive_cytotoxic=1000 \
--cd14_monocytes=1000 \
--regulatory_t=1000
# Creates Zhengmix4eq.rds
```

The same for AnnData would look like:

Listing C.2: Generating Zhengmix4eq in Anndata

```
svn export https://github.com/google-research/google-research/trunk/cell_m
cd cell_mixer
bash fetch_10x.sh --data_path=data
Rscript cell_mixer.R \
--data_path=data \
--format=csv \
--name=Zhengmix4eq \
--qc_count_mad_lower=3 \
--qc_feature_count_mad_lower=3 \
--qc_mito_mad_upper=-1 \
--b_cells=1000 \
--naive_cytotoxic=1000 \
--cd14_monocytes=1000 \
--regulatory_t=1000
```

```
# Creates Zhengmix4eq.counts.csv, Zhengmix4eq.metadata.csv,
# and Zhengmix4eq.featuredata.csv
python3 converter.py --input_csv_prefix=Zhengmix4eq \
--format=anndata
# Creates Zhengmix4eq.h5a
```

The cell_mixer script will read the data, apply the sub-sampling to get the specified amount of cells, and then apply the QC steps. It supports 3 QC steps: filtering cells by number of reads measured, number of genes expressed, and the percentage of mitochondrial reads. It also supports the 8 cell types shown in Table S1.

| Cell type | Number of cells | Flag |
|---|---|---|
| CD19+ B cells | 10085 | –b_cells |
| CD8+/CD45RA+ Naive Cytotoxic T Cells | 11953 | –naive_cytotoxic |
| CD14+ monocytes | 2612 | –cd14_monocytes |
| CD4+/CD25+ Regulatory T Cells | 10263 | –regulatory_t |
| CD56+ natural killer cells | 8385 | –cd56_nk |
| CD4+ helper T cells | 11213 | –cd4_t_helper |
| CD4+/CD45RO+ Memory T Cells | 10224 | –memory_t |
| CD4+/CD45RA+/CD25- Naive T cells | 10479 | –naive_t |

Table S1: Description of the 8 cell types

We can see in Figure S1 an example of the kinds of datasets that could be build. Here these four datasets all contain the same amount of cells before QC and are a mix of two to five cell types in equal proportions, this can be used to measure the influence of the number of cell types on the performances of a scRNA-seq analysis method.

### C.1.3 Conclusion

In this application note, we have provided methods developers with an easy to use set of script to generate in-silico cell mixes in most of the commonly used formats. They can also easily be modified to work with cells outside of [170]. We believe that this will allow both methods developers and benchmark writers to have better data to analyse their methods, as they can now specify the kind of data that they want.

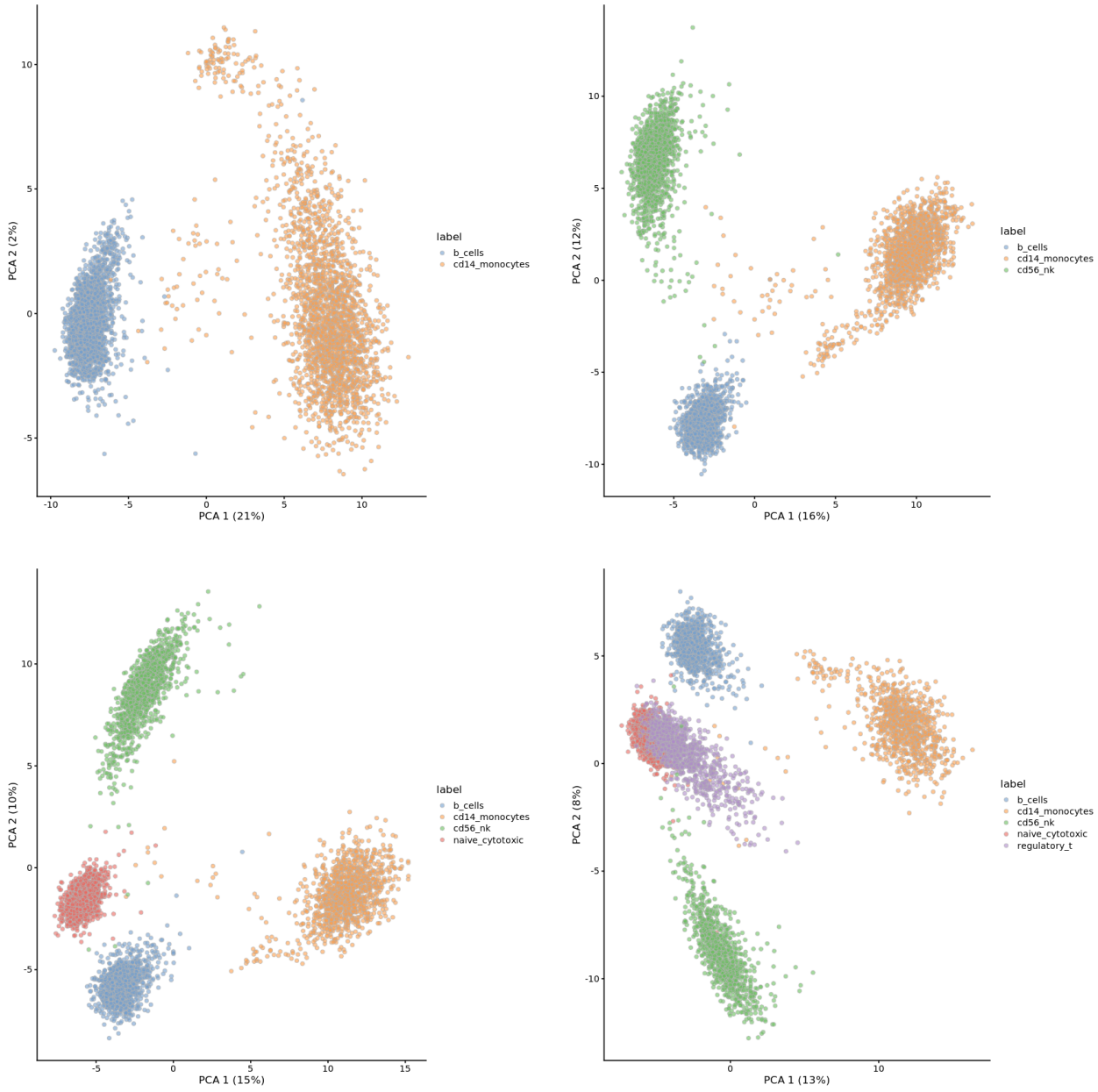


Figure S1: First two principal components of the PCA of the datasets after processing with scran with default parameters, colored by cell type. All datasets contain 4.800 cells and are an equal mix of the different cell populations, before QC.

## RÉSUMÉ

Ces dernières années, la transcriptomique et l'épigénomique en cellule unique ont permis aux biologistes d'observer les tissus à une nouvelle résolution. Grâce à ces protocoles, nous sommes maintenant en mesure d'observer l'ensemble de la distribution des états cellulaires dans un tissu, au lieu de simplement leur agrégat. Avec ces nouveaux types de mesures, est apparu le besoin de nouvelles méthodes statistiques pour les analyser. En effet, la génération précédente d'outils d'analyse était conçue pour un régime de peu d'échantillons de haute qualité, alors que ces nouvelles mesures sont beaucoup plus importantes en quantité, mais de qualité nettement inférieure. Ce problème de faible qualité est encore plus prononcé pour les protocoles d'épigénomique en cellule unique, du fait que les cellules ne possèdent que deux copies du génome, par rapport aux centaines de milliers de molécules d'ARN présentes dans la cellule. Le profil transcriptomique et épigénomique des cellules étant mesuré en grande dimension, la communauté scientifique s'est beaucoup intéressée aux méthodes permettant de réduire la dimension des données.

Cette explosion d'intérêt a conduit à de nombreux nouveaux algorithmes et à une communauté florissante de développeurs de méthodes. Leurs travaux n'ont cependant pas encore été adoptés par les bioinformaticiens, soit parce qu'ils n'étaient pas jugés suffisamment fiables, soit parce qu'ils ne répondaient pas correctement aux questions biologiques. Dans cette thèse, nous avons tenté de mesurer la fiabilité de ces nouvelles méthodes, ainsi que la façon dont elles sont affectées par les étapes qui les précèdent. Nous avons en outre tenté de développer un nouvel algorithme pour apprendre des représentations de mesures épigénétiques de bout en bout, apprenant ainsi à la fois la représentation des cellules, ainsi qu'une annotation du génome.

## MOTS CLÉS

Apprentissage statistique, single-cell, cancer, transcriptomique, epigenetique, bioinformatique.

## ABSTRACT

In recent years, single-cell transcriptomics and epigenomics have allowed biologist to observe tissues at a new resolution. Using these protocols, we are now able to observe the whole distribution of cell states within a tissue, instead of just measuring an aggregate cell state. With these new types of measurements has come the need for new statistical methods to analyze them. Indeed the previous generation of analysis tools were designed for a regime of few high quality samples, while these new measurements are much higher in quantity, but of significantly lower quality. This problem of low quality is even more pronounced for single-cell epigenomics protocols, due to cells only having two copies of the genome, compared to the hundreds of thousands of RNA molecules present in the cell. Since epigenomics and transcriptomics profiles are evaluated across a high number of variables, there has been a great interest in methods for reducing the dimension of the data.

This explosion of interest has led to numerous new algorithms and a thriving community of methods developers. Their work has however not yet been fully adopted by practicing bioinformaticians, either because they were not deemed reliable enough, or because they failed to properly answer biological questions. In this thesis, we measured how reliable these new methods are, as well as how they are affected by the steps preceding them. We found that the recent deep learning methods fail to outperform linear methods on current datasets, for most modalities. We further found, for epigenetic assays, that the feature engineering steps were more important than the dimension reduction algorithm, in order to obtain good representation of cells. We further attempted to develop a novel algorithm to learn embeddings of epigenomic measurements in an end-to-end fashion, learning at once both the low-dimension representation of the cells, as well as the epigenomic annotation.

## KEYWORDS

Statistical learning, single-cell, cancer, transcriptomics, epigenetics, bioinformatics.