3-21-2023 2:00 PM

A Novel Method for Assessment of Batch Effect on single cell RNA sequencing data

Behnam Jabbarizadeh, *The University of Western Ontario*

Supervisor: Shooshtari, Parisa, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

Abstract

In recent years, technological developments have enabled the comprehensive transcriptional profiling of thousands of single cells in a single experiment. However, there is still much to be gained from the integration of datasets from different donors, studies, and technological platforms. One major challenge in this regard is the technical variability introduced by handling different batches, known as batch effects, which can obscure biological variations. Assessing batch effects within a dataset has been the focus of various studies seeking to establish reliable criteria for selecting a batch effect removal method. However, these methods do not always perform reliably.

This study provides a comprehensive review of both batch effect removal and assessment methods and introduces a novel method for batch effect removal assessment.

The performance of the proposed method is evaluated by comparing it to four other batch effect assessment methods using eleven test datasets. The results showed that the proposed method consistently outperformed the other methods, successfully passing all challenges while the other methods failed at least one test. The proposed method was applied to three biological integrated datasets to evaluate its performance on real-world data. The results of the evaluation showed that the proposed method demonstrated the highest correlation with the expert's assessment of the datasets, indicating that it was able to accurately identify batch effects in the data.

Keywords: Batch Effect Removal, Batch Effect Assessment, Integrated datasets, Dimension reduction, scRNA-seq, Single-cell RNA sequencing

Summary of Lay Audience

In recent years, scientists have been able to study the genetic activity of individual cells in unprecedented detail. However, when combining data from different studies, researchers often encounter a problem known as ”batch effects.” These are differences in the way that samples were processed or analyzed that can obscure real biological differences between cells. In this thesis, we review existing methods for identifying and removing batch effects and propose a new method for assessing the effectiveness of batch effect removal techniques.

To test our new method, we applied it to twelve different synthetic datasets and compared its performance to four other methods. Our method consistently outperformed the others, successfully identifying and removing batch effects in all cases. We also applied our method to three real-world datasets and found that it accurately identified batch effects that were missed by other methods.

Overall, our research provides a promising solution for addressing batch effects in large-scale studies of genetic activity. By improving our ability to combine data from different sources, we can gain a more comprehensive understanding of how genes are regulated in different cell types and under different conditions. This could ultimately lead to new insights into the causes of diseases and the development of more effective treatments.

Acknowlegements

I'd like to express my sincerest gratitude and appreciation to my supervisor, Dr. Parisa Shooshtari, for her unwavering support, patient guidance, and providing invaluable direction throughout my Master's program.

I also want to thank my beloved family for their endless encouragement and wise counsel. Without them, the completion of these studies would have been impossible.

Contents

List of Figures

List of Tables

List of Appendices

Chapter 1

Introduction

1.1 RNA sequencing

RNA sequencing, also known as RNA-seq, is a powerful technique for studying gene expression at the molecular level. Gene expression is the process by which the information in DNA is used to synthesize proteins and other molecules that carry out the functions of the cell. RNA sequencing allows researchers to identify the specific genes that are being expressed in a particular cell or tissue, and to quantify the relative abundance of each gene. By analyzing the RNA content of a cell, researchers can gain insight into the functions and activities of the genes in that cell and how they are regulated [1, 2].

RNA sequencing involves several steps, including the isolation and purification of

RNA from a sample, the conversion of RNA to complementary DNA (cDNA), and the sequencing of the cDNA using high-throughput DNA sequencing technology. To obtain a count matrix from the raw sequencing data (FASTQ format), one must first align the sequenced reads to a reference genome or transcriptome. This alignment process allows for the identification of the genomic locations of the reads. Next, the aligned reads are counted and assigned to their corresponding genes, generating a table that contains the number of reads that map to each gene in the sample. This table is the count matrix. The dimensions of this matrix are equal to the number of cells in the sample and the number of genes that are included in the study. This matrix can be sparse, which means that a large proportion of its elements have the value of zero, reflecting the fact that not all genes are expressed in every cell or that many genes have low or no expression in a given sample. RNA sequencing can be performed on a variety of sample types, including tissues, cells, and even individual cells. [3, 4, 5].

RNA sequencing has a wide range of applications in basic research, drug discovery, and clinical medicine. In basic research, RNA sequencing can be used to study gene expression patterns in different cell types or tissues, to understand how genes are regulated, and to identify genes that are associated with specific diseases or conditions. In drug discovery, RNA sequencing can be used to identify potential targets for new drugs, to understand how drugs work at the molecular level, and to identify biomarkers for drug

efficacy and toxicity. In clinical medicine, RNA sequencing can be used to diagnose and monitor diseases, identify potential therapeutic targets, and guide personalized medicine approaches [6].

1.1.1 Single cell RNA sequencing

Single-cell RNA sequencing (scRNA-Seq) is a variation of RNA sequencing that allows researchers to analyze the gene expression of individual cells within a sample [7]. Traditional RNA sequencing methods analyze the RNA content of a sample as a whole, which can mask the differences in gene expression between different cell types or even between individual cells within the same cell type. scRNA-Seq overcomes this limitation by allowing researchers to isolate and analyze the RNA content of individual cells. This is typically done by physically separating the cells using techniques such as microfluidics or laser-capture microdissection, and then performing RNA-Seq on each individual cell [8]. The resulting data is used to identify and quantify the gene expression of each cell, providing a detailed picture of the gene expression landscape of the sample.

ScRNA-Seq has a wide range of applications in basic research, drug discovery, and clinical medicine. In basic research, scRNA-Seq can be used to study the gene expression patterns of different cell types or tissues at the single cell level, to understand the heterogeneity within a cell population, and to identify rare cell types or subpopulations. In drug

discovery, scRNA-Seq can be used to identify potential targets for new drugs, to understand how drugs work at the molecular level, and to identify biomarkers for drug efficacy and toxicity. In clinical medicine, scRNA-Seq can be used to diagnose and monitor diseases, identify potential therapeutic targets, and guide personalized medicine approaches [9, 10]. Overall, scRNA-Seq provides a powerful tool for studying gene expression at the single-cell level and can provide valuable insights into the underlying biology of a sample.

1.2 Data integration

Data integration refers to the process of combining data from multiple sources into a single, unified view. It is a key aspect of data management and is often used to support a variety of business and research objectives, such as improving decision-making, identifying trends and patterns, and enabling data-driven insights. For example, in social sciences, data integration can involve merging datasets from different countries to study cross-national trends. In the field of genomics, data integration can involve combining single-cell RNA sequencing (scRNA-seq) data from multiple experiments or studies to create a more comprehensive atlas of cell types and gene expression patterns, such as the Human Cell Atlas project. Data integration is important in scRNA-seq because it allows researchers to gain a more comprehensive and accurate understanding of the cells and gene expression patterns within a tissue or organism. By integrating data from multiple sources,

researchers can identify trends, patterns, and relationships that would otherwise be missed if the data were analyzed in isolation [11, 12].

1.3 Batch effect

In general, batch effect refers to systematic variations in data that are caused by factors unrelated to the variables being studied. Batch effects can occur in a variety of contexts, including experiments, observational studies, and data analyses. They can be caused by a variety of factors, such as differences in the equipment or conditions used to collect the data, differences in the batch of reagents or samples used, or differences in the processing or analysis of the data [13].

Batch effects can have a significant impact on the validity and reliability of scientific research, as they can introduce false patterns and differences in the data. They can also make it difficult to compare results across studies or experiments, as any observed differences may be due to batch effects rather than the variables being studied.

In single-cell RNA sequencing (scRNA-Seq), batch effects can arise from a variety of sources. For example, batch effects can be introduced by differences in the sample preparation or library construction protocols used, differences in the sequencing platform or chemistry used, or differences in the data analysis methods applied [14].

To address batch effects in scRNA-Seq data, it is important to carefully control for

any potential sources of unwanted variability. This may involve standardizing the sample preparation and library construction protocols, using the same sequencing platform and chemistry for all samples, and using consistent data analysis methods. It is also important to carefully consider the design of the experiment and to ensure that any observed differences between samples or conditions are due to the variables being studied rather than batch effects.

1.4 Content of thesis

The goal of this thesis is to develop and evaluate a new method for assessing batch effect in single-cell RNA sequencing (scRNA-seq) data. To solve this problem, we propose a novel method for assessing batch effect in scRNA-seq data. We first tested our method using simulated datasets and compared it to several popular assessment methods. We then applied our method to three real-world datasets that showed evidence of batch effect and used it to evaluate the performance of seven different batch effect removal methods. Our results demonstrate that our method is more accurate and reliable than existing approaches and has the potential to improve the quality of scRNA-seq data.

By addressing the issue of batch effect, we can improve the reliability of scRNA-seq data, leading to more robust scientific conclusions and potentially advancing our understanding of gene expression and its role in health and disease. This thesis is organized as

follows:

- In Chapter 2, we provide a background on single-cell RNA sequencing, the problem of batch effect, batch effect removal methods, batch assessment methods, and the R programming language.
- In Chapter 3, we demonstrate that batch effect is not limited to biology but is also a problem in other fields such as biological imaging, molecular biology, and social sciences.
- In Chapter 4, we describe our novel method for assessing batch effect in single-cell RNA sequencing data and provide the results of our method applied to simulated test cases.
- In Chapter 5, we present the results of our method applied to real-world datasets and compare them with other batch effect assessment methods.
- In Chapter 6, we discuss the implications of our study and suggest directions for future research.

Chapter 2

Background

In this chapter, we will introduce the programming tools that were used and then discuss dimension reduction techniques for single-cell RNA-sequencing (scRNA-seq) data, which are essential tools for visualizing and analyzing this high-dimensional, noisy, and sparse type of data. We will also discuss the importance of addressing batch effects in scRNA-seq data and introduce various batch effect removal methods that can be used to ensure unbiased and accurate data analysis.

2.1 R programming language

R is a programming language and software environment for statistical computing and graphics. It was developed in the early 1990s by statisticians Ross Ihaka and Robert

Gentleman at the University of Auckland, New Zealand, as an open-source alternative to proprietary statistical software packages [15]. R is an interpreted, dynamically typing, and vector based language that enables distributed computing with strong graphical capabilities. R has become a popular choice for data analysis and statistical computing, particularly in the fields of statistics, economics, and computer science. It is widely used in research, education, and industry, and has a large and active user community that has contributed a wealth of libraries and packages for a wide range of applications. In the field of bioinformatics, R is widely used for the analysis of biological data, particularly in the areas of genomics, proteomics, and transcriptomics. It is particularly useful for the analysis of RNA-seq data, as it has a number of specialized packages and functions for analyzing RNA-seq data. R is also used for the visualization and exploration of biological data, including the creation of plots and graphs to visualize gene expression patterns, pathways, and networks. It is also used for the development of statistical models and algorithms to identify patterns and relationships in biological data. Overall, R is a powerful and widely used programming language that has become an essential tool for bioinformaticians and researchers working with RNA data and other types of biological data [16].

2.1.1 Seurat Package

Seurat is an R package for single-cell data analysis and visualization, developed by the Satija Lab[1] at the New York Genome Center. It is a widely used tool for the analysis of single-cell RNA-seq data and includes a range of functions and features for tasks such as data preprocessing, quality control, normalization, clustering, and visualization.

Seurat is designed to be flexible and user-friendly and includes functions for a wide range of single-cell analysis tasks, including:

- Quality control: Seurat includes functions for filtering and removing low-quality cells and features, as well as for identifying and correcting batch effects and other technical biases.
- Normalization: Seurat includes functions for normalizing and scaling the data to correct for differences in library size and sequencing depth.
- Clustering: Seurat includes functions for clustering cells into groups based on their gene expression patterns, using techniques such as k-means clustering, spectral clustering, and density-based clustering. The default method for finding clusters in Seurat package is the Louvain algorithm. This algorithm iteratively groups cells into clusters by optimizing a modularity score, which measures the density of connections within clusters compared to the density of connections between clusters in a

[1] https://satijalab.org/

shared nearest neighbor (SNN) graph constructed from the scRNA-seq data [17].

- Differential expression analysis: Seurat includes functions for identifying differentially expressed genes between different groups of cells. The default method for differential expression analysis is the non-parametric Wilcoxon rank-sum test (also known as the Mann-Whitney U test). This test is used to compare gene expression levels between two groups of cells (e.g., different clusters) and identify differentially expressed genes.
- Visualization: Seurat includes functions for visualizing the data, including t-SNE plots, heatmaps, and violin plots, as well as functions for integrating data from multiple sources, such as gene ontology and pathway information.

Overall, Seurat is a powerful and widely used tool for the analysis and visualization of single-cell RNA-seq data and is an essential tool for many researchers working in this field [18].

2.2 High-Performance Computing Resources

High-Performance Computing (HPC) refers to the use of supercomputers, clusters, and other specialized hardware and software systems to perform computationally intensive tasks. HPC systems are designed to provide high levels of performance and scalability and

are often used to solve complex problems that require large amounts of computing power and resources, such as simulations, data analysis, and machine learning. HPC systems are typically characterized by their high-speed processors, large amounts of memory, as high as 256 GB per node, and adequate temporary storage, which allow them to perform a large number of calculations in parallel [19].

2.2.1 Compute Canada

Compute Canada[2] is a national organization that provides high-performance computing (HPC) resources and services to researchers and scientists in Canada. Its mission is to enable Canadian researchers to perform advanced computing research and development, leading to new scientific discoveries, innovations, and economic benefits for Canada. Compute Canada operates a network of HPC centers across the country, providing access to a range of computing resources, including supercomputers, cloud computing platforms, and storage systems. Researchers and scientists can apply for access to these resources through Compute Canada's allocation process. Compute Canada also provides support and training to help researchers and scientists effectively use its HPC resources.

[2]https://ccdb.computecanada.ca

2.3 scRNA-seq data dimension reduction

Single-cell RNA sequencing (scRNA-seq) is a technique used to analyze gene expression at the single-cell level. scRNA-seq data is characterized by having a high number of dimensions, noise, and sparsity. In scRNA-seq data, in order to represent the gene expression levels of cells, a matrix is provided that is known as the count matrix. this matrix contains the counts for all samples, with the genes in rows and the samples in columns. In the count matrix, each cell is represented by a high number of dimensions or genes, and the expression levels of these genes can be noisy and sparse and comprised of mostly zero values. In order to better understand and analyze this type of data, it is often necessary to reduce its dimensionality. This is because scRNA-seq data is typically too complex for most modelling algorithms to process directly, and because many biological systems have lower intrinsic dimensionality. For example, a differentiating hematopoietic cell can be represented by just two or more dimensions, its progress in differentiation towards a particular cell type and its current cell cycle stage. This means that although the dimension reduction methods don't necessarily find new features that have biological meanings, it is a valid effort to reduce the dimension without losing information. Dimensionality reduction techniques can be used to project the high-dimensional scRNA-seq data into a lower-dimensional space, enabling the visualization of cluster structures and the inference of development trajectories [20]. There are many dimension reduction methods that can

be applied to scRNA-seq datasets but only PCA [21], UMAP [22], and t-SNE [23] are used in this thesis and we only explain them in this section.

2.3.1 PCA

Principal component analysis (PCA) that was introduced in 1901 by Karl Pearson, is a widely used technique for dimensionality reduction in scRNA-seq data. It transforms a large set of variables into a smaller one that still contains most of the information in the large set[21]. It works by identifying the patterns in the data that account for the highest variances and projecting the data onto a lower-dimensional space using these patterns. The first principal component is the pattern that accounts for the highest variance in the data, the second principal component is the pattern that accounts for the second highest variance, and so on.

PCA is calculated in several steps. The first step is standardization, where the range of the continuous initial variables is standardized. The second step is the computation of the covariance matrix to understand how the variables of the input data set are related to each other. The third step involves computing the eigenvectors and eigenvalues of the covariance matrix[24]. PCA is a useful tool for scRNA-seq data because it can help to identify the most important patterns in the data and reduce the complexity of the data, making it easier to visualize and analyze. However, PCA has some limitations, such as

its reliance on linear relationships between variables, which can make it less effective for data that is highly non-linear or has a complex structure.

2.3.2 t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE) is another popular dimensionality reduction technique for scRNA-seq data [23]. It was developed by Laurens van der Maaten and Geoffrey Hinton in 2008 and has become a widely used tool for visualizing high-dimensional data. This method works by constructing a low-dimensional representation of the data in which similar points are close together and dissimilar points are farther apart. It does this by constructing a probability distribution over pairs of points in the high-dimensional space. These probabilites represent similarities between neighbours. Then the algorithm minimizes the Kullback-Leibler divergence between the distribution and a low-dimensional version of the distribution. K-L divergance is a type of statistical distance that measures how one probability distribution is different from a second. t-SNE is particularly effective at preserving local structure in the data and can be used to identify patterns and clusters in the data. However, it can be sensitive to the choice of hyperparameters and can be slow for large datasets.

2.3.3 UMAP

Uniform Manifold Approximation and Projection (UMAP) is a dimensionality reduction technique that is specifically designed for scRNA-seq data. It was developed by Leland McInnes and John Healy in 2018 as an alternative to t-SNE. UMAP works by constructing a low-dimensional representation of the data that preserves the global structure of the data as well as the local structure. It does this by constructing a graph of the data and then using this graph to identify the underlying structure of the data. In this graph, the nodes represent the data points and the edges represent the connections between them. These connections are established based on the degree of similarity or proximity between the data points. The more similar or proximal two data points are in the high-dimensional space, the higher the weight of the edge between them. UMAP has been shown to be more effective than t-SNE at preserving the global structure, and it is also faster and more scalable, making it a good choice for large datasets [22].

2.4 Batch effect removal methods

As explained in section 1.3, a batch effect is a systematic difference between two or more groups of samples that are not due to the biological variables being studied and effective batch-effect removal is essential for unbiased and accurate data analysis. To convey the

idea of the batch effect, we generated a mock dataset and plotted it in figure 2.1. This figure illustrates the presence of systematic differences between two groups of samples that are not due to the variables of interest, but rather to some other source of variation. Batch effect removal methods aim to adjust for batch effects while batch effects can be complex and nonlinear, making it challenging to accurately align different datasets while maintaining the biological variations within the sample. To address this challenge, several tools have been proposed. In the following, some of these tools that were used throughout my thesis, are briefly explained.

2.4.1 Harmony

Harmony is one of the previously suggested methods for removing batch effects in single-cell sequencing data [25]. The harmony algorithm requires a PCA embedding of cells and their batch assignment. Using these two inputs, their designed algorithm clusters the cells with maximum diversity. More specifically, the clustering algorithm tries to adjust the cell population of each cluster. To achieve this goal, first, the algorithm tries to put similar cells in the same cluster and simultaneously tries to increase the diversity of each batch in each cluster as well so that each cluster is comprised of similar cells while having cells from every batch possible. This algorithm returns a clustering assignment and this assignment is used by the correction algorithm to remove batch effects. This algorithm calculates a

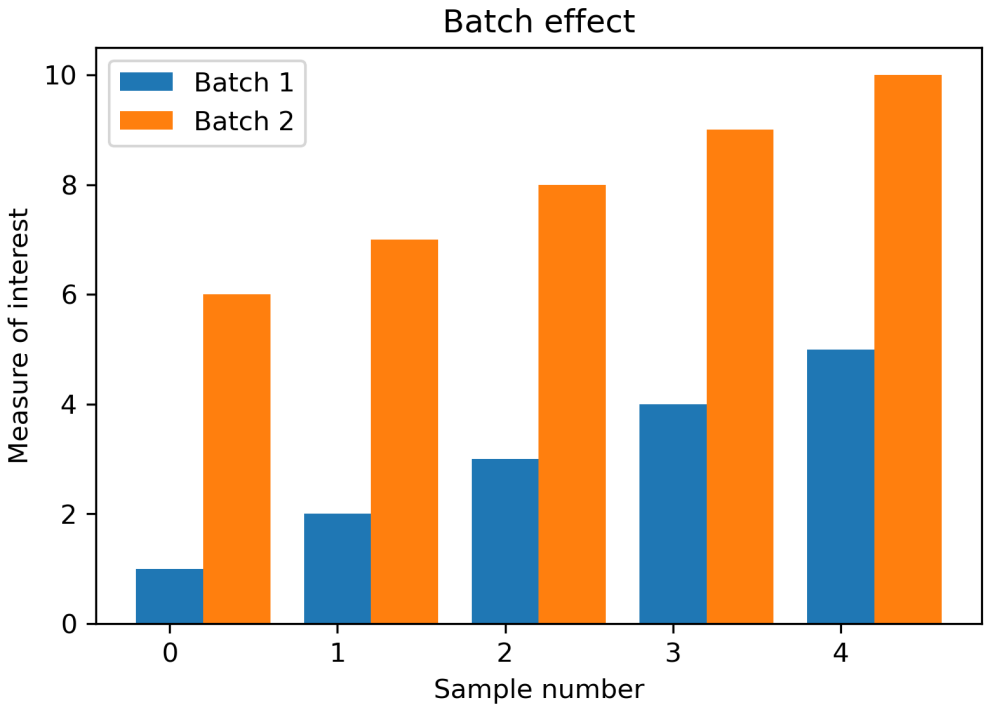


Figure 2.1: The data for this plot is generated just to demonstrate the concept of batch effect. The data comes from two different batches of samples, collected at different times and processed using different methods and a measure of interest has been obtained. The plot shows that the measure of interest is significantly higher in Batch 1 than in Batch 2, indicating the presence of a batch effect.

factor indicating the batch effect correction needed to be applied to each cell and passes that corrected space to the clustering algorithm. This process repeats until the output of the clustering algorithm is the same as the previous step and the algorithm converges. The equation 2.1 demonstrates the recurring steps of the Harmony algorithm while Z is the input embedding, to be corrected in Harmony, $\hat{Z}$ is the integrated embedding, output by Harmony, R is the soft cluster assignment matrix of cells to clusters, and ϕ is the assgnment matrix of cells to batches.

$$\begin{aligned} R &\leftarrow CLUSTER(\hat{Z}, \phi) \\ \hat{Z} &\leftarrow CORRECT(Z, R, \phi) \end{aligned} \tag{2.1}$$

2.4.2 Limma

The Limma method was originally developed to analyze microarray data [26]. It is based on linear modelling and uses empirical Bayes methods to shrink the standard errors of the estimated coefficients, which can improve the accuracy of the estimates. By using emprical Bayes procedure, the prior probability distribution is estimated from data. The algorithm of this method starts with normalizing data to unify the mean and variance of expressions. Then a linear model is fit to the data. The linear model has a blocking term to estimate the batch effects in the data. Blocking term is used to account for known sources of variability in the data and by including a blocking term in the model, we can control for

the effect of these sources of variability on the response variable. This estimation is used to correct the batch effect in the data.

2.4.3 MNN

Matching Mutual Nearest Neighbor (MNN) for batch correction is based on the idea that cells of similar types should have almost the same neighbours [27]. In the MNN method, the definition of neighbors is based on the distance metric used to compare cells. Typically, the Euclidean distance or cosine distance is used to calculate the distance between cells. Cells that are close in the high-dimensional space are considered neighbors. Mutual nearest neighbors are defined as pairs of cells from different samples that are each other's nearest neighbors. For example, if cell A from sample 1 is the nearest neighbor of cell B from sample 2, and cell B is the nearest neighbor of cell A from sample 1, then cells A and B are mutual nearest neighbors. Each neighbour is weighted by distance from the current cell. This idea is implemented by trying to put cells with similar mutual neighbours near each other [28]. This method first calculates the mutual nearest neighbours for each sample and then matches the samples based on their mutual nearest neighbours. After the samples have been matched, the data is adjusted to remove the batch effects.

FastMNN

Fast Mutual Nearest Neighbor (FastMNN) is a variant of the MNN method for correcting batch effects in high-dimensional data, such as gene expression data. Like the MNN method, FastMNN is based on the idea of matching samples based on their mutual nearest neighbours, which are samples that are most similarity in their mutual neighbours to each other in terms of their gene expression levels. The main difference between MNN and FastMNN is that FastMNN is designed to be faster and more efficient than MNN. It uses a fast nearest neighbour search algorithm and an efficient optimization method to speed up the calculation of the mutual nearest neighbours and the matching of the samples [29].

2.4.4 Liger

Linked Inference of Genomic Experimental Relationships (Liger) is a method for correcting batch effects in gene expression data [30]. It is based on the idea of using latent variables to capture the sources of batch effects and remove them from the data. This method first obtains lower dimensional space of the data using integrative non-negative matrix factorization (iNMF). This factorization is based on generating non-negative matrices to improve the inspection of factors [31]. Then shared factor space is used to create a shared factor neighbourhood graph instead of just finding the maximum factor in the space. The factor space provides a general coordinate system to describe the data from

different perspectives. With community detection performed on the graph, joint clusters are identified. Then one of the datasets, usually the biggest one, is chosen and the clusters of other datasets according to the joint clusters found by the community detection, would be joined to the bigger dataset.

2.4.5 Seurat 3 CCA

Seurat 3 is an updated version of Seurat 2 for integrating datasets that also uses canonical correlation analysis (CCA) for dimensionality reduction [32].Canonical-correlation analysis (CCA) is a statistical method used to find relationships between two sets of variables (X and Y) by identifying linear combinations of these variables that have the highest correlation with each other. It is commonly used when there are correlations among the variables and can be helpful in identifying underlying patterns and relationships [33]. It does this by identifying ”anchors” that represent similar cell states across different batches. These anchors are identified using mutual nearest neighbours in the normalized CCA subspace. To ensure that the anchors accurately represent similar cell states, Seurat 3 uses shared nearest neighbour graphs to assess the similarity between cell types. Once the anchors have been identified, Seurat 3 computes a correction vector using the difference in expression profiles between cells and uses this vector to transform the data in order to remove the batch effects.

2.4.6 Conos

The Clustering On Network Of Samples (Conos) is a method that tries to connect large scRNA-seq datasets together. This method in the first step, filters and normalizes each dataset individually. Then compares all pairs of datasets and creates an error-prone mapping between the cells from different datasets. Finally, it creates a joint graph and uses that graph for community detection and propagating labels [34].

2.4.7 ComBat

The ComBat method was also originally designed for microarray gene expression data [35], but its usage is extendable to RNA-seq data as well [36]. It was developed by Johnson, Li, and Rabinovic in 2007. ComBat uses the following steps to remove batch effects from gene expression data. It first standardizes the expression data by ensuring that all genes have similar means and variances. Then it uses a Bayesian approach to fit the standardized data to Gaussian distributions in order to estimate the batch effects present in the data. This method parameterizes the expression with background level, changes caused by biological condition and mean and variance batch effect. These parameters are estimated, and used for adjusting batch effect. Lastly, it utilizes batch effect estimators to correct the original expression matrix. This involves adjusting the data to remove the batch effects, resulting in a batch effect corrected dataset [36].

In this section, we have discussed various methods for batch effect removal and data integration in scRNA-seq analysis. Some methods, such as ComBat and Limma, correct the underlying gene expression, while others, including Conos, Seurat CCA, FastMNN, Liger, and Harmony, correct the PCA space or other low-dimensional spaces. Table 2.1 shows the summary of batch effect removal and data integration methods in scRNA-seq analysis based on their correction types.

Method	**Correction Type**
ComBat	Gene Expression
Limma	Gene Expression
Conos	PCA/low-dimensional space
Seurat CCA	PCA/low-dimensional space
FastMNN	PCA/low-dimensional space
Liger	PCA/low-dimensional space
Harmony	PCA/low-dimensional space

Table 2.1: Summary of batch effect removal and data integration methods in scRNA-seq analysis

2.5 Batch effect assessment methods

Batch effect assessment methods are tools that allow researchers to identify and quantify the extent of batch effects in their scRNA-seq data. Silhouette [37], Mixing metric [18], K-nearest neighbour batch effect test (kBET) [38], and adjusted Rand index (ARI) [39] are methods that are used in this thesis. In the following, brief explanations of these methods are provided.

2.5.1 Silhouette

The silhouette method is a technique used to evaluate the quality of clustering in a dataset [37]. It is based on the idea that points within a cluster should be similar to each other, while points in different clusters should be dissimilar. The silhouette value of a point is a measure of how well it is assigned to its cluster, with higher values indicating a better fit. To calculate the silhouette value of a point, the average distance between that point and all other points in the same cluster is first calculated. This is known as the intra-cluster distance. Then, the average distance between the point and all points in the nearest cluster is calculated. This is known as the nearest-cluster distance. The silhouette value of the point is then calculated as the difference between the nearest-cluster distance and the intra-cluster distance, divided by the maximum of the two distances. The formulation for

silhouette value is as follows:

$$s_i = \frac{d_{nearest} - d_{intra}}{\max(d_{nearest}, d_{intra})}$$

Where:

- s_i is the silhouette value of point i.
- $d_{nearest}$ is the average distance between point i and all points in the nearest cluster.
- d_{intra} is the average distance between point i and all other points in the same cluster.

The silhouette value is a measure of how well point i is assigned to its cluster, with higher values indicating a better fit. It is calculated by taking the difference between the nearest-cluster distance and the intra-cluster distance and dividing it by the maximum of the two distances. This normalizes the silhouette value to a range of -1 to 1, with values closer to 1 indicating a better fit and values closer to -1 indicating a poor fit.

In the context of assessing batch effects in scRNA-seq data, the Silhouette Index can be used to evaluate the impact of batch effect correction on clustering quality. If batch effects are present, cells from different batches but of the same cell type may form separate clusters, leading to a lower average silhouette value. After batch effect correction, these cells should be more similar to each other and form a single cluster, resulting in a higher average silhouette value. Thus, an improvement in the silhouette values after batch effect correction could be an indication of successful removal of batch effects.

2.5.2 kBET

K-nearest neighbour batch effect test (kBET) is a method for assessing the effectiveness of batch correction in scRNA-seq data [38]. To use kBET, the full gene expression dataset (D) is first preprocessed to remove low-quality or uninformative cells. Then, the k-NN distance is calculated for pairs of cells in the dataset. The k-NN distance is calculated using the cover-tree algorithm, which is a data structure that efficiently computes nearest neighbors in high-dimensional spaces by hierarchically partitioning the data points into nested sets. This algorithm allows for faster k-NN distance calculations compared to traditional methods [40]. To perform dimensionality reduction, the first 50 eigenvectors corresponding to the largest eigenvalues are calculated using the singular value decomposition (SVD) function. SVD is a linear algebra technique that decomposes a matrix into three matrices, where the middle matrix contains the singular values in descending order. The eigenvectors and eigenvalues are derived from the SVD, with the eigenvectors representing the principal components of the data and the eigenvalues indicating the amount of variance explained by each principal component. Next, the k-NN distances are compared between cells from the same batch and cells from different batches, and the distribution of the number of cells in each batch is compared to the distribution under the null hypothesis (i.e., the absence of a batch effect). The comparison is performed using the chi-squared test, a non-parametric test used to determine if there is a significant difference between

the observed frequencies in a categorical dataset and the expected frequencies under the null hypothesis [41]. The results of this comparison are then summarized by computing the average rejection rate (S) over all tests, which is a test statistic for the whole dataset. If S exceeds the chosen significance level (alpha), the null hypothesis can be rejected for the whole dataset, indicating the presence of a batch effect. Commonly used significance levels are 0.05 (5%) or 0.01 (1%), but the choice of the significance level should be made based on the specific research context and the desired balance between the risk of false positives (Type I error) and false negatives (Type II error).

2.5.3 ARI

The Adjusted Rand Index (ARI) is a measure of the similarity between two clustering results [39]. It was introduced by Hubert and Arabie in 1985 as a way to evaluate the quality of clustering algorithms and assess the similarity between different clustering results. The ARI measures the proportion of pairs of elements that are either assigned to the same cluster or to different clusters in both clusterings. It takes into account both the number of pairs that are assigned to the same cluster in both clusterings (true positives) and the number of pairs that are assigned to different clusters in both clusterings (true negatives). The ARI is calculated as follows:

$$ARI = \frac{true\ positives + true\ negatives - expected\ value}{maximum\ value - expected\ value}$$

where the expected value is calculated as:

$$expected\ value = \frac{(true\ positives + false\ positives) * (true\ positives + false\ negatives)}{total\ pairs}$$

and the maximum value is calculated as:

$$maximum\ value = \frac{total\ pairs - 1}{2}$$

So the ARI value increases as there are more cells that were assigned to the same clusters and belonged to the same cluster (Ture positive) or the cells that don't belong to the same clustering and weren't assigned to the same cluster correctly (True negatives).

The ARI can be used to evaluate the impact of batch effect correction on the similarity between clustering results. If batch effects are present, clustering results based on the original data and the batch-corrected data may be quite different, leading to a low ARI value. After successful batch effect correction, the clustering results should be more similar, resulting in a higher ARI value. In this case, a ground truth clustering (e.g., based on known cell types) is needed to compare the original and batch-corrected data, allowing for the assessment of batch effect removal.

2.5.4 Mixing Metric

The Mixing Metric is a feature of the Seurat package, as explained in 2.1.1, for analyzing scRNA-seq data. The Mixing Metric is a measure of the degree of batch effect correction

in a Seurat object, which is a data structure used to store and manipulate scRNA-seq data in Seurat. This metric is calculated by examining the local neighborhood of each cell in the dataset, looking at a maximum of k nearest neighbors for each cell. In this context, a ”group” refers to a set of cells originating from a specific batch. The Seurat mixing metric determines the k nearest neighbors of each cell within its own group (i.e., within the same batch) and the rank of those neighbors in the overall neighborhood, which includes cells from different batches. The median of these values is then calculated across all groups (batches), resulting in a mixing metric value for each cell in the dataset. A lower value in the output of this algorithm indicates better mixing and, therefore, more effective batch effect correction. In order to make this algorithm consistant with other assessment methods, the output is subtracted from a constant value. This constant value by the suggestion of the authors of the Seurat package, is chosen as the max number of clusters.

This section demonstrates that each method is founded on robust mathematical and biological principles. When applied to a dataset, these methods can yield quantifiable results regarding the presence of batch effects. However, the true value of these methods becomes evident when they are used to compare the performance of different batch effect removal techniques. By doing so, scientists can make informed decisions about which method to use for the remainder of their research.

Chapter 3

Examining the Role of Batch Effect in Other Fields

ScRNA-seq studies are particularly prone to batch effects because these studies often involve the use of multiple samples, which may be collected and processed at different times or in different locations. This can make it more difficult to control potential sources of batch effects. However, batch effects can occur in any field of research or study where samples or measurements are taken over time or from multiple sources. Some examples of fields where batch effects may be a concern are discussed in the rest of this chapter.

3.1 Biological imaging

In biological imaging, batch effects can occur when images are acquired or processed at different times or in different locations; for example, no two MRI machines capture the same image even if they scan the exact same person. One common source of batch effects in biological imaging is the use of different imaging instruments or software to acquire or process images. For example, if images are acquired using different microscopes or cameras, differences in the sensitivity or resolution of these instruments could introduce batch effects in the data. Similarly, if images are processed using different software or settings, variations in the algorithms or parameters used could also introduce batch effects [42].

3.2 Molecular biology

In molecular biology, batch effects can be introduced through variations in the quality or stability of reagents or other materials used in the experimental process. For example, if different batches of a particular enzyme are used to digest DNA samples, variations in the activity of the enzyme could introduce batch effects in the data. Similarly, variations in the conditions under which samples are stored or handled, such as temperature or humidity, can also introduce batch effects. To minimize batch effects in molecular biology studies,

it is important to carefully control and standardize all aspects of the experimental process, including the use of reagents and materials, and the handling and storage of samples [43].

3.3 Social sciences

Batch effects can occur in social science studies when there are variations in the way that data is collected or processed, or when there are differences in the characteristics of the study population. To address these issues, researchers can carefully standardize the data collection process, using consistent methods and instruments across all study sites. Statistical methods can also be used to identify and adjust for batch effects in the data analysis. Here are a few examples of how batch effects have been identified and addressed in social science research:

- Survey data: In survey research, batch effects can occur when data is collected by different teams of researchers at different times, using different methods or instruments. Standardizing the data collection process and using consistent methods and instruments can help minimize these effects [44].

- Longitudinal studies: In longitudinal studies, batch effects can occur when there are variations in the way that data is collected or processed over time. Ensuring the use of consistent methods and instruments across all study sites and time points can help

minimize these effects [45].

- Meta-analyses: In meta-analyses, batch effects can occur when there are variations in the way that data is collected or processed in the studies being analyzed. Standardizing the data collection process and using consistent methods and instruments across all studies can help address these issues [46].

3.4 Impact of batch effect on results

According to [47], batch effects can have significant impacts on the results of experiments, particularly when data from multiple studies or batches is combined. Batch effects are systematic differences in measurements that are introduced by the batch or sample preparation rather than by the natural variability being studied. They can occur when samples are processed in different batches over time or under different external conditions and can introduce significant variability into the data, making it difficult to accurately compare measurements between samples or to draw meaningful conclusions from the data. It is important to carefully control for batch effects in experimental design and analysis to ensure that the results of the study are reliable and accurate. In situations where data from multiple studies or batches must be combined, it is important to consider the potential for batch effects and take steps to minimize their impact on the results of the analysis. This

can be done through careful experimental design and the use of statistical methods to correct batch effects. Failing to adequately control for batch effects can lead to incorrect or biased results, which can compromise the validity and usefulness of the study.

Chapter 4

Batch Finder

4.1 Introduction

Technological advances in recent years have enabled the generation of single-cell transcriptional profiling of thousands of cells in one experiment [48], yet there is much more to reveal by the integration of datasets across donors, studies, and technological platforms. The technical variation that originates from handling different batches, which is known as the batch effect, is an obstacle in the way of recognizing biological variations. In [13], Tung et al. have shown that there can be a substantial variation between technical replicates. By decreasing the cost and time of sequencing and increasing the number of researches done on scRNA-seq data, it is also becoming more of a problem in the age of

big data [49]. Figure 4.1 hypothetically demonstrates the batch effect in the integration of biological or technical replicates of a sample. In part a, the same sample has been through two different procedures to generate data and in part b, there are two samples that are biological replicates but processed simultaneously. Biological replicates are samples that are taken in the same biological situation; For example, a sample of a mouse brain taken on the same day. Removing batch effect from integrated data is a crucial step in the pre-processing workflow of data analysis [50].

4.1.1 Batch Effect Removal Methods

Various studies have proposed methods to reduce or remove the batch effects [25, 26, 29, 30, 32, 34, 36]. Although it seems simple to remove the batch effect, it becomes challenging when the biological variability, which is valuable information within the datasets, must be preserved and not removed. These studies utilize different techniques and models to perform this task. The batch effect removal methods vary from newer methods like the projection of mutual nearest neighbours (MNNs) [27] to the ones that have been available for longer such as ComBat [35], which is a linear regression-based method. Some recent studies [51, 52, 53] focus on benchmarking batch effect removal methods. In [54], an in-depth benchmark study has been conducted on several batch effect removal methods, in order to find the best method in terms of their computational runtime, the ability to handle

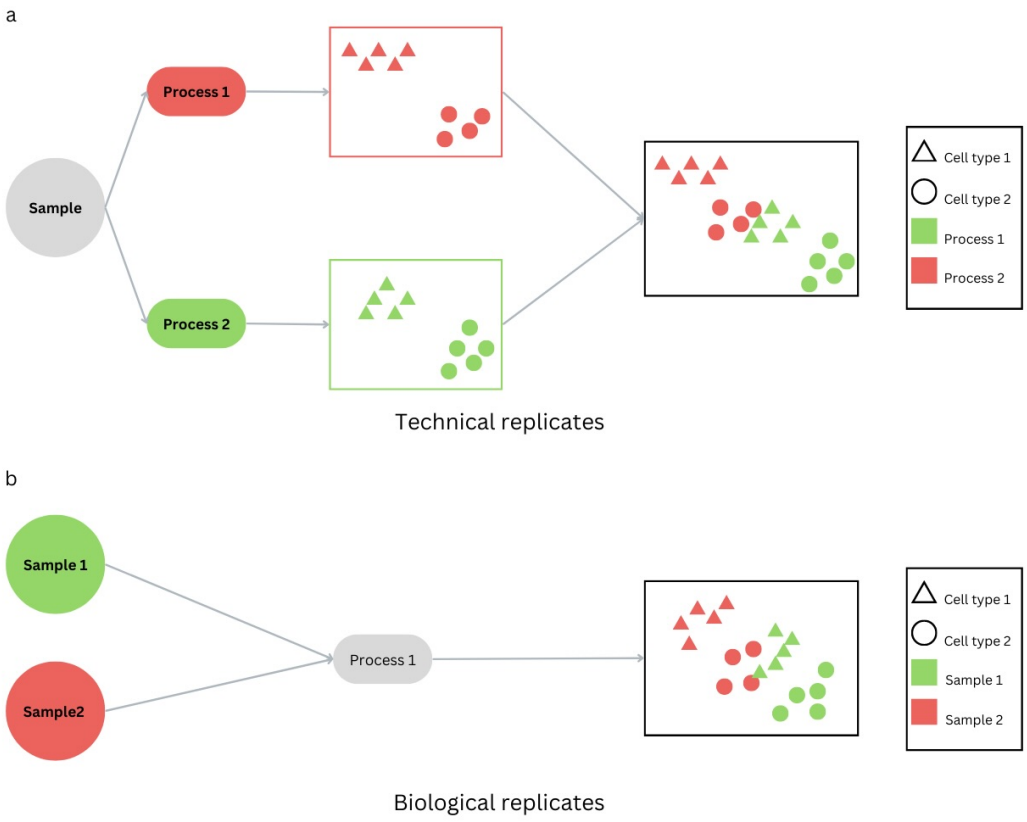


Figure 4.1: Batch effect in sc-RNA seq data. **a** Technical replicates, The batch effect is caused because the same cells have gone through the same technical setup but not simultaneously and through two processes. **b** Biological replicates, The batch effect is caused due to having biological replicates and going through the same process and technical setup.

large datasets, and effectiveness.

4.1.2 Batch Effect Assessment Methods

One of the most common ways of finding the best batch effect removal method in terms of their effectiveness is by employing UMAP and t-SNE visualizations. As these methods are explained in 2.3.2 and 2.3.3, they construct a low-dimensional representation of the data in which similar points are close together and dissimilar points are farther apart. While these low-dimensional embeddings play a crucial role in providing insight into the data, they cannot represent the full information within the dataset and the results are subjective. Also, as the dataset grows in the number of cells, it gets more complicated to determine the best batch removal method. The effort to find a quantitative parameter representing the batch effect within the dataset resulted in different assessment methods for the batch effect. Some methods are common procedures for evaluating clustering algorithms, such as Silhouette [55] and ARI [39] and some, like kBET [38], only focus on single-cell RNA-seq batch effect assessment. While these methods have been proven effective, they can't be trusted completely, One of the reasons is that these methods do not incorporate biological insight into their assessment.

4.1.3 Scope of Research

In this study, a new batch assessment method is proposed, proper simulation data is designed and tested, and the performance of this new method is benchmarked against the performance of 3 previously developed batch assessment methods, which are kBET[38], Silhouette [37], and ARI [39]. Then mixing metric assessment [18] is added to these methods to be tested on 3 public single-cell RNA-seq datasets as this method is a built-in Seurat function and only tested on biological datasets. These datasets are chosen so that each demonstrates a different challenge in batch effect assessment. Afterwards, all 5 batch effect assessment methods' results including the suggested method are calculated and the expert opinion on the performance of each batch effect removal method using visualization methods is provided. Finally, the quantitative results are compared with the expert batch effect assessment.

4.2 Proposed Batch Effect Assessment Method

I propose Batch Finder as a method to produce a quantitative measure for representing batch effect within scRNA-seq data. In the following the intuitive idea behind Batch Finder is explained, then the algorithm and workflow are explained, and finally, the implementation and challenges are expounded.

4.2.1 Concept

As explained in section 1.3, the batch effect is due to the biased emplacement of cells in the multi-dimensional space of gene expressions or the acquired dimension reduction after combining two or more different sets of data. As the cells from the same cell type have similar functions, it is expected that in general, they are in closer proximity in the feature space than those from other cell types. This basic rule is the core that Batch Finder has been shaped around.

After mixing multiple datasets, it is most desirable to have the same cell types well mixed and close to each other and have different cell types apart from each other. It is the process that happens when an expert is trying to determine batch effect presence in lower-dimension visualizations. If the same cell types from different datasets are well mixed and simultaneously away from the other cell types, it's a good and acceptable mix for continuing the data analysis.

Batch Finder is a method that extracts the relationship between cell types in the dataset. Although the main purpose of this method is for batch effect recognition, it also provides valuable information about the dataset as well. In Batch Finder, batch effect measurement is based on a normalized Euclidean distance between the cells of the same cell types and different cell types. As it is shown in figure 4.2, the distance between cells of the same type is correlated with an increase in batch effect, while the distance between cells of different

types is correlated with a decrease in batch effect. One advantage of using this method to measure batch effect is that it can be applied to datasets whose cell types don't have complete overlap and there are cell types that are only available in one dataset, even if the datasets do not have any cell types in common. This allows for accurate measurement of a batch effect in a mixture of two datasets, even if in the worst case, the datasets do not have any cell types in common. Batch Finder is designed specifically so that it considers biological concepts and the direct thought process of experts in the field during the mathematical calculations of batch effect.

4.2.2 Algorithm and Implementation

The algorithm for implementing the concept explained in section 4.2.1 is demonstrated in the flowchart in figure 4.3 and figure 4.5.

First, a data frame containing all distances between pairs of cells is generated. It uses Euclidean distance as it is mentioned in section 4.2.1. So for each cell, we have their distance calculated as in formula 4.1. In equation 4.1, C is the set of all cells and for each cell c, Pos_c is the vector of all assigned values in the input data to that cell. This data can be a count matrix or a dimension reduction matrix. I is the length of the Pos vector, which

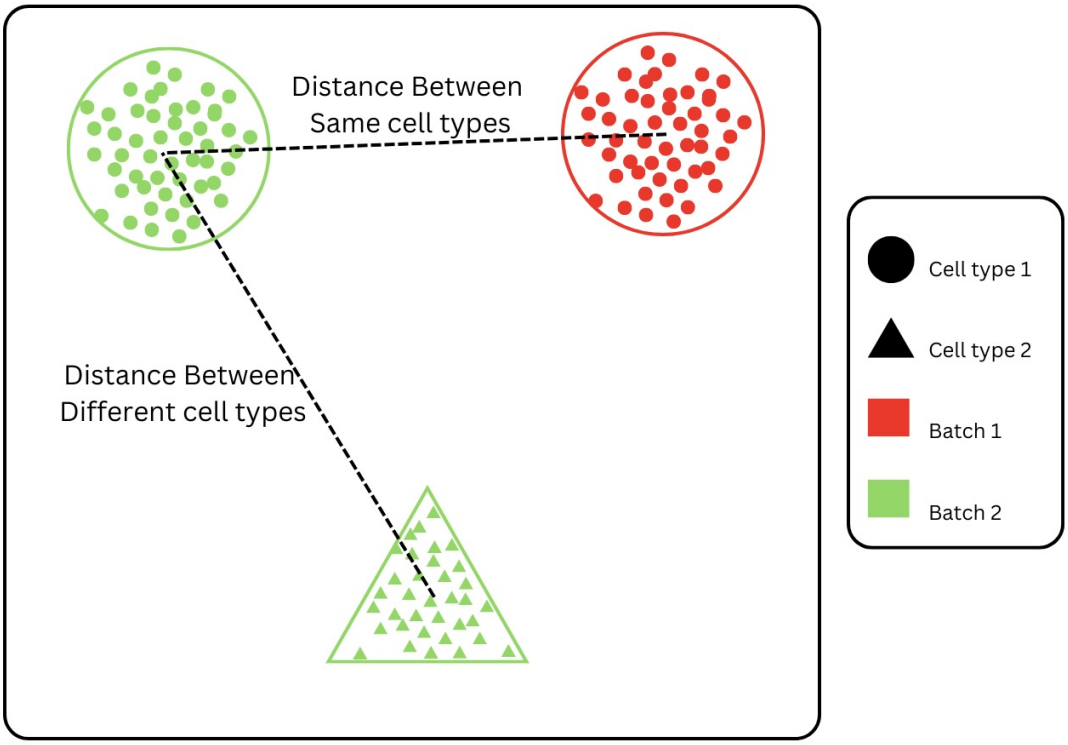


Figure 4.2: Demonstration of distances between different cell types. The distance between cell type 1 in batch 1 and batch 2 is desired to be less and the distance between cell type 1 and cell type 2 in batch 1 needs to increase so that cell type 1 and cell type 2 remain separable.

has the same length for all cells.

$$\begin{aligned} &\forall c_1, \forall c_2 \in C, \\ &dist_{1,2} = \|\overrightarrow{Pos_{c_1} - Pos_{c_2}}\| = \sqrt{\sum_{i=1}^{I} |Pos_{c_1,i} - Pos_{c_2,i}|^2} \end{aligned} \tag{4.1}$$

Now that the distance is obtained, each cell type should be analyzed solely. A loop iterates over all cell types, performing a series of calculations that collectively generate the final result. First, the distances between all cells of a specified cell type and all other cell types are normalized. These normalized distances are then summarized by calculating the mean and variance for each combination of batch and cell type. Additionally, the number of values used in each summary calculation is recorded for future reference. This number of values is equal to the product of the number of samples for each cell type. It is important to note that cells from different batches are treated differently, as this is the primary source of the batch effect. Ignoring this distinction would compromise the integrity of the analysis. For this step's implementation, we first normalize and then calculate the variables.

This step of normalization occurs at the level of distances between cells of a specific cell type and all other cells. Equation 4.2 formulates this step, where T is the set of all cell types and $dist_{1,2}$ is the result value in 4.1. I is the number of cells in t_1 and J is the number

of cells in C. $d_{1,2}$ is the normalized distance.

$$\begin{aligned} &\forall t_1 \in T, \\ &\forall c_1 \in t_1, \forall c_2 \in C, \\ &d_{1,2} = \frac{dist_{1,2} \times I \times J}{\sum \sum_{i=1,j=1}^{I,J} dist_{i,j}} \end{aligned} \tag{4.2}$$

Equation 4.3 demonstrates the formulation of finding the variables of the second step after normalizing distances. In this equation, B is the set of all batches present in the datasets. $I_{1,1}$ is the number of cells that fits in the conditions for c_1 which is the cell type t_1 and batch b_1 and $J_{2,2}$ is the same number for c_2.

$$\begin{aligned} &\forall t_1 \in T, \forall t_2 \in T, \forall b_1 \in B, \forall b_2 \in B, \\ &\forall c_1 s.t. c_1 \in t_1 \wedge c_1 \in b_1, \\ &\forall c_2 s.t. c_2 \in t_2 \wedge c_2 \in b_2, \\ &m_{1,1-2,2} = \frac{\sum \sum_{i=1,j=1}^{I_{1,1},J_{2,2}} d_{i,j}}{I_{1,1} \times J_{2,2}} \end{aligned} \tag{4.3}$$

To clarify the notation for $m_{1,1-2,2}$, it's the mean of normalized distances from cell type 1 of batch 1 to cell type 2 of batch 2. Also, let's define a new variable derived from equation 4.3 as the number of samples as in equation 4.4.

$$s_{1,1-2,2} = I_{1,1} \times J_{2,2} \tag{4.4}$$

The other parameters such as variance and standard deviation can be obtained in the same

step but as for the current implementation of the method it's not needed, it has been neglected. Note that after this step the dimension of distances is reduced from the square of the number of cells in the dataset to the square of the summation of the number of cell types within each batch. This means that if there is a cell type A both in batch 1 and batch 2, that cell type must be counted twice.

In the following step, another normalization is performed at the level of cell types. This normalization aims to make the distance results comparable across cell types. To achieve this, the distance between cells of the same type within the same batch is used as a reference to normalize the distances for that cell type. Equation 4.5 shows the mathematical procedure for calculating the normalization term, where $m_{1,1-1,1}$ is obtained from equation 4.3, $s_{1,1-1,1}$ from equation 4.4, I is number of batches in B

$$\begin{aligned} &\forall t_1 \in T, \\ &n_1 = \frac{\sum_{i=1}^{I} m_{1,i-1,i} \times s_{1,i-1,i}}{\sum_{i=1}^{I} s_{1,i-1,i}} \end{aligned} \tag{4.5}$$

And now all mean distances can be normalized as in equation 4.6

$$\begin{aligned} &\forall t_1 \in T, \forall t_2 \in T, \forall b_1 \in B, \forall b_2 \in B, \\ &norm\_mean_{1,1-2,2} = \frac{m_{1,1-2,2}}{n_1} \end{aligned} \tag{4.6}$$

The term "distance between cell types," as used in Section 4.2.1, refers to the normalized mean distance calculated in equation 4.6. This normalized mean distance is used as a measure of the distance between cell types.

In order to help experts to have a better understanding of the relation between different components of the dataset, this result is also provided in the final output of the suggested method. There are several reasons why this information may be useful for experts:

- The relation between cell types is a key metric in determining the batch effect. This method successfully summarizes all the information in the count matrix or dimension reduction matrix into much smaller yet informative new data.

- Since datasets may contain outliers, calculating means can be a useful method for mitigating the impact of these outliers without completely discarding their influence. As such, this data not only provides valuable insights but also filters out unnecessary information, making it more reliable and robust.

- The size of each cell type is taken into account, eliminating the need for additional processing. As a result of this approach, cell types with a small number of cells do not have as much influence as cell types with a large number of cells. This ensures that the analysis is balanced and fair and that the results are not unduly influenced by cell types with a disproportionate number of cells.

- This method generates the final data from the original dimensions of the dataset, ensuring that all of the information present in the original data is retained. As a result, the final data is a complete and comprehensive representation of the original

dataset.

- As mentioned, although this method is designed for batch effect detection, it is also a cell-type-based summarization. Using its descriptive abilities, it helps the best for finding unidentified cell types based on their relationship with other cell types; For example, if an unidentified cell type has the closest relation with immune cell types, it most probably has the same functionality.

As the final goal of the method is to provide a final quantitative measure for batch effect, cell type distances need more steps to take so that a final decision about batch effect inside the dataset would be achieved. In order to do that we need two more parameters which are two weights.

- Same cell type weight: It is a weight for showing how important is to have the same cell types closer to each other.

- Different cell type weight: This weight is for the importance of having cells with different cell types apart.

It is usually that the same cell types being close together have more value than different cell types. In fact, it is the ratio of these weights that matters, but having two values prevents extra complications. These weights can be adjusted and determined by the scientist using the method but they have a recommended method as the default value. The default ratio

is one to two for same cell types to different cell types which means that not mixing different cell types is twice as important as getting the same cell types together. It is because preserving the biological variation is more important than removing batch effect as the research can not proceed after removing biological variation.

Equation 4.7 shows how the final *batch_number* is created. *batch_number* is a batch effect acceptance measure for the dataset, that shows how acceptable is the dataset in terms of batch effect. $norm\_mean_{1,1-2,2}$ is obtained from equation 4.6 and is initially 0. This equation decreases *batch_number* as the distance between the same cell types among all batches increases.

$$\begin{aligned} &\forall t_1 \in T, \forall b_1 \in B, \forall b_2 \in B, \\ &batch\_number = batch\_number - w_{same} \times norm\_mean_{1,1-1,2} \end{aligned} \tag{4.7}$$

And then n value gets updated by the distance between different cell types, from the same batch, so that no batch is squeezed to make the same cell types close. Equation 4.8 demonstrates this update step.

$$\begin{aligned} &\forall t_1 \in T, \forall t_2 \in T, \forall b_1 \in B, \\ &batch\_number = batch\_number + w_{different} \times norm\_mean_{1,1-2,1} \end{aligned} \tag{4.8}$$

Parameter	Description
c	Lowercase letter c is used to refer to a cell within the dataset
C	Uppercase letter C is the set of all cells available in the dataset
$\overrightarrow{Pos_c}$	Vector for cell c that has the gene expression values or dimension reduction values
$dist_{1,2}$	Euclidean distance between c_1 and c_2
t	Lowercase letter t is used to refer to a cell type within the dataset
T	Uppercase letter T is the set of all cell types available in the dataset
$d_{1,2}$	normalized distance between c_1 and c_2
b	Lowercase letter b is used to refer to a batch within the dataset
B	Uppercase letter B is the set of all batches available in the dataset
$I_{1,2}$ or $J_{1,2}$	number of cells that are from cell type t_1 and batch b_2
$m_{1,2-3,4}$	Mean of normalized distances between cells that are from cell type t_1 and batch b_2 to cells from cell type t_3 and batch b_4
$s_{1,2-3,4}$	number of pairs that can be made between cells that are from cell type t_1 and batch b_2 to cells from cell type t_3 and batch b_4
n_1	normalization coefficient for cell type 1
$norm\_mean_{1,2-3,4}$	normalized value of $m_{1,2-3,4}$ by normalization coefficient of first cell type ,n_1
$batch\_number$	Final batch indicator for the dataset
w_{same}	Determined weight for the normalized mean distance between same cell types
$w_{different}$	Determined weight for the normalized mean distance between different cell types

Table 4.1: A list of parameters that were used during section 4.2.2 with a brief description.

4.2.3 Explaining Decisions in Algorithm

Here we discuss the reason behind certain steps and calculations in the algorithm and show how they are useful and meaningful for the batch effect assessment.

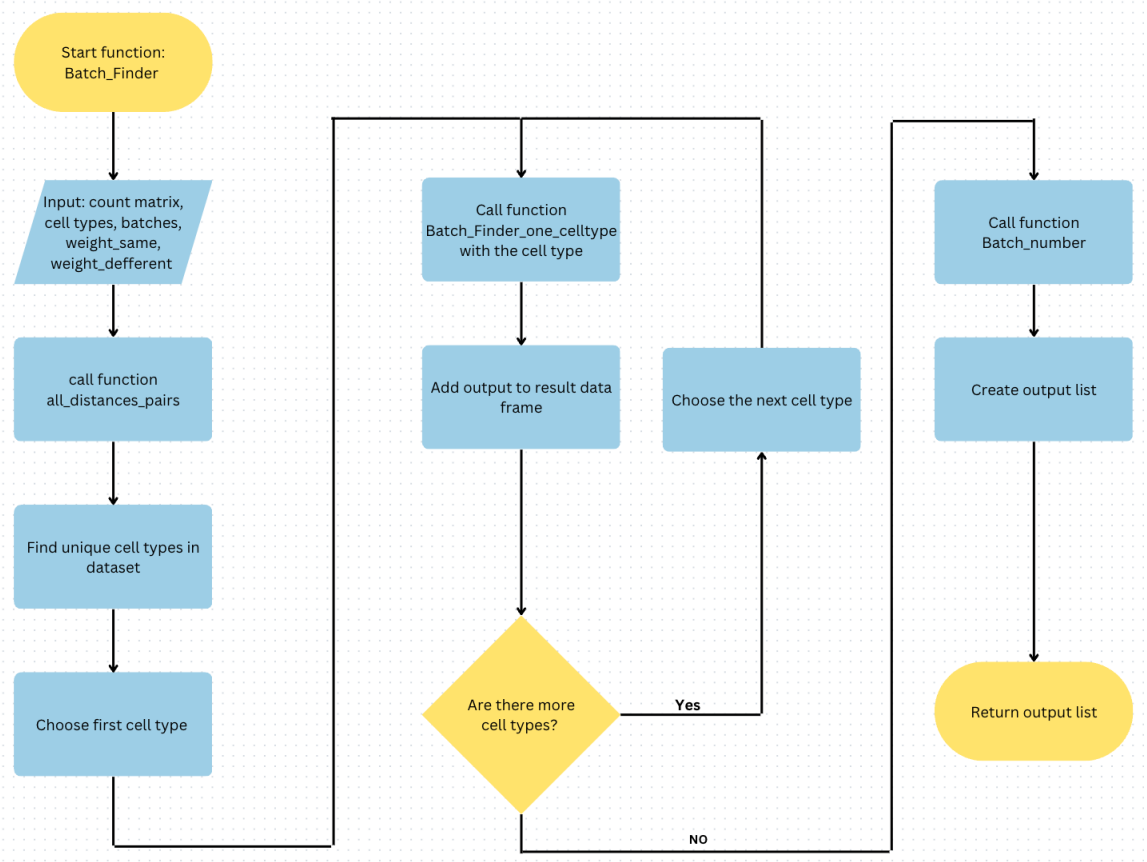


Figure 4.3: Flowchart of batch_finder, the main function for implementing Batch Finder method. The algorithm starts by getting the input and performing the explained algorithm step by step. In the first step, the function all_distances_pairs is called. The algorithm of this function is provided in figure 4.5 a. Then the unique cell types are detected and By selecting the first cell type the Batch_finder_one_celltype function is called. This algorithm flowchart is available in figure 4.5 b. The output of this algorithm is saved into a data frame and this process repeats until all cell types are analyzed and then the function Batch_number which is explained in figure 4.4, is called and the final output is generated.

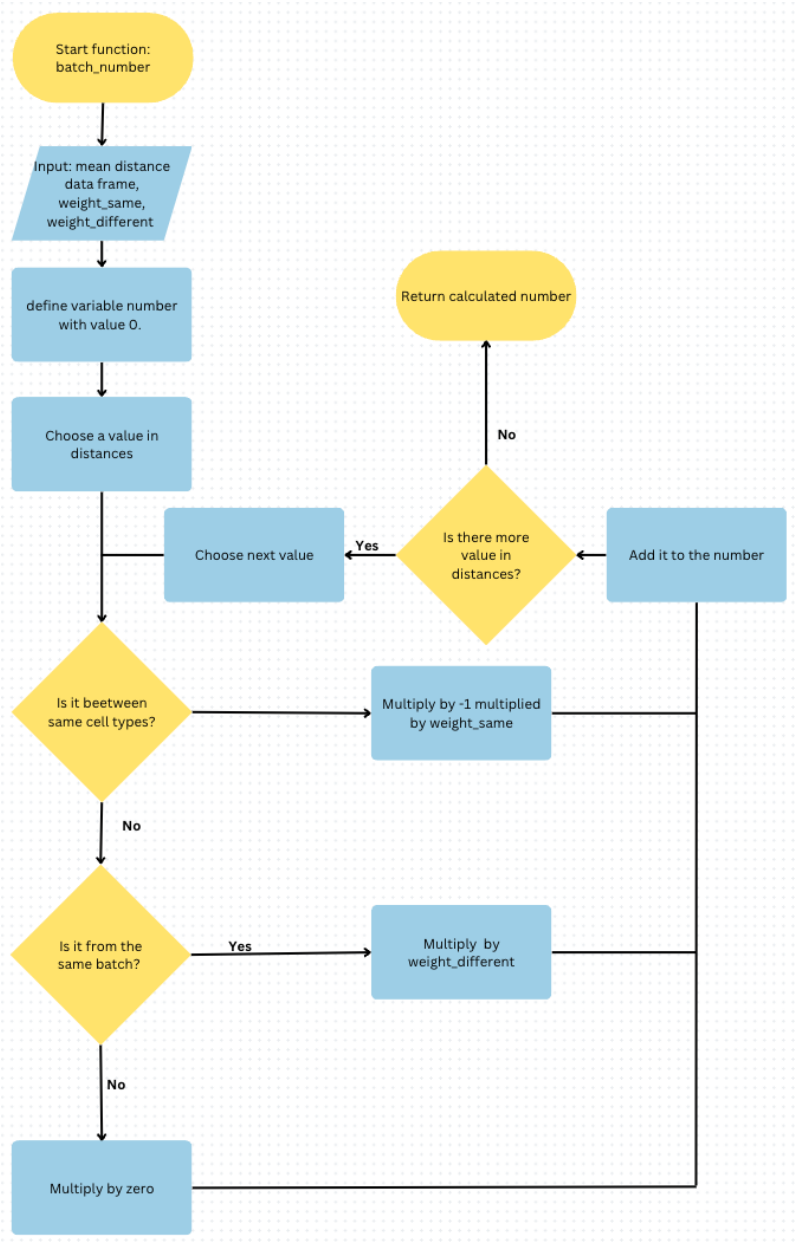


Figure 4.4: Flow chart of batch_number function. This function is called in the final step and calculates a metric for batch effect by the given weights and the mean distances. This algorithm strictly follows the logic behind calculating batch effect. It increases the batch acceptance if different cell types from the same dataset are distant and decrease the acceptance if the same cell types are distant.

Figure 4.5: **a**, Flowchart of all_distances_pairs function. This function simply calculates distances between all cells in the dataset.**b**, Flowchart of Batch_finder_one_celltype function. This function generates the normalized distance data frame for each cell type.

- Normalization in equation 4.2: This step normalizes all distances from all cells in cell type to all other cells by its mean. In other words, it's balancing the distance from cells from a cell type to all other cells. This means that the distance from cell type a to b is relative to the proximity of the other cell types to cell type a while the value for the distance from cell type b to is is relative to the proximity of the other cell types to cell type b.

- Normalization term in equation 4.5: Here the normalization term is used for the mean distance between every pair of cell types and in order to make them comparable to each other, every mean distance is divided by the weighted mean of mean distances from the first cell type to all other cell types. This weighted mean indicates the mean distance of that cell type to all others and using this normalization if a normalized mean is more than one, it indicates that it is further than the majority of other cell types. Figure 4.6 demonstrates this concept.

- Using cell types from the same batch in equation 4.8: As in data integration, only a relation between the same cell types can be assumed and the relation between different cell types can't be expected in any way. The reason that distance between the same cell types is important to be considered is that cell types in the same batch shouldn't be compressed because the decrease between the same cell types is just a cause of compression not actually removing batch effect.

4.3 Test data

After designing the algorithm, a set of data has been simulated in order to test the performance of my batch effect assessment method alongside other assessment methods. The goal of designing these test cases is to minimize the number of unknown parameters and simplify the comparison of different test cases, in order to increase confidence in the results. This is done by reducing the number of variables. This series of test data is designed in lower dimensions so the data can be visualized without losing information and the assumption of batch effect is accurate. Having defined the test cases, five major tests are subsequently proposed that utilize these test cases to illustrate their respective arguments and evaluate the difficulty of assessing batch effects using various methods.

4.3.1 Defined test cases

I designed 12 test cases to evaluate the performance of batch effect assessment methods. In each test case, we constructed 2 or 3 hypothetical cell types, with each cell type being defined by two dimensions. The limits of these dimensions were set to constrain the cells belonging to a specific type. Subsequently, a number of cells were generated for each cell type, such that all the cells in a cell type are restricted to a desired area, and the locations of cells are randomly selected in that specific area using a uniform distribution. The number of cells in each cell type was fixed as 500, but only in the cases tat mentions an increase in

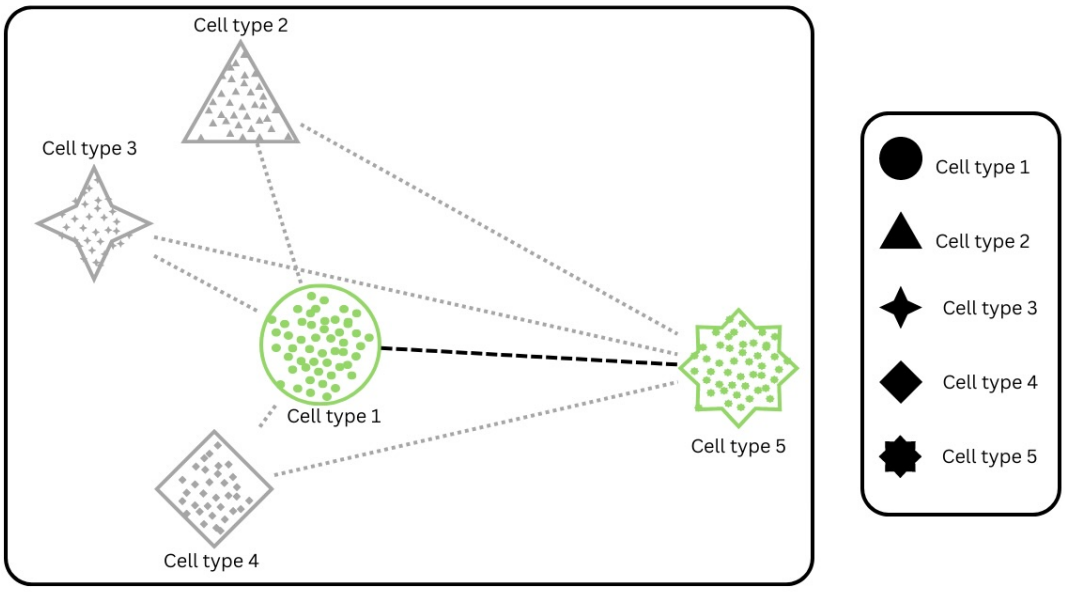


Figure 4.6: Necessity of normalizing distances between cell types in a data set. The distance between cell type 1 and cell type 5 is a determined value but in cell type 1 perspective, it is a relatively high distance as it has the greatest value among the distances of cell type 1 to other cell types, but it is a relatively low distance from cell type 5 point of view because it is the lowest among all distances between cell type 5 and all other cell types.

the cell frequency, the number of cells in each cell type was increased to 1000. Here I first explain all of the test cases and my purposes for using each test case.

Test case 1

Figure 4.7 demonstrates a sample distribution for Test case 1. This test case has well-separated different cell types inside a batch and well-mixed the same cell types. This case is designed to show a very good case data integration. All cell types are well apart and also batches are completely mixed.

Test case 2

Figure 4.8 demonstrates the sample distribution of Test Case 2. Test case 2 is very similar to test case 1. However, it has a big and challenging difference. Although as shown in figure 4.8-a, the same cell types a and b, both are close to each other, while by looking at figure 4.8-b, it can be seen that batches 1 and 2 are not well mixed and they are just side by side. In terms of clustering, it's considered a good cluster but from a biological perspective, these cells need to be mixed a little bit more.

Test case 3

Figure 4.9 shows how cells are distributed in 2-dimensional space for Test Case 3. In case 3, cell type b is almost separated from its equivalent in the other batch but is still closer to

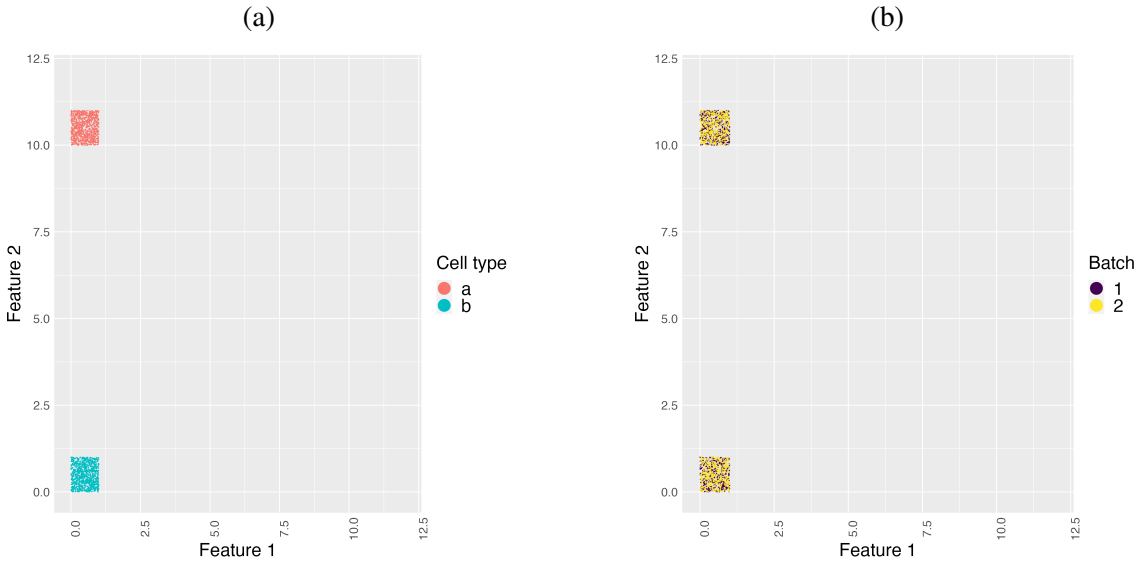


Figure 4.7: Test Case 1: **a**, Dataset visualization based on their cell type. Here cell types a and b are completely separated **b**, Dataset visualization based on their original batch. Batches 1 and 2 are very well mixed.

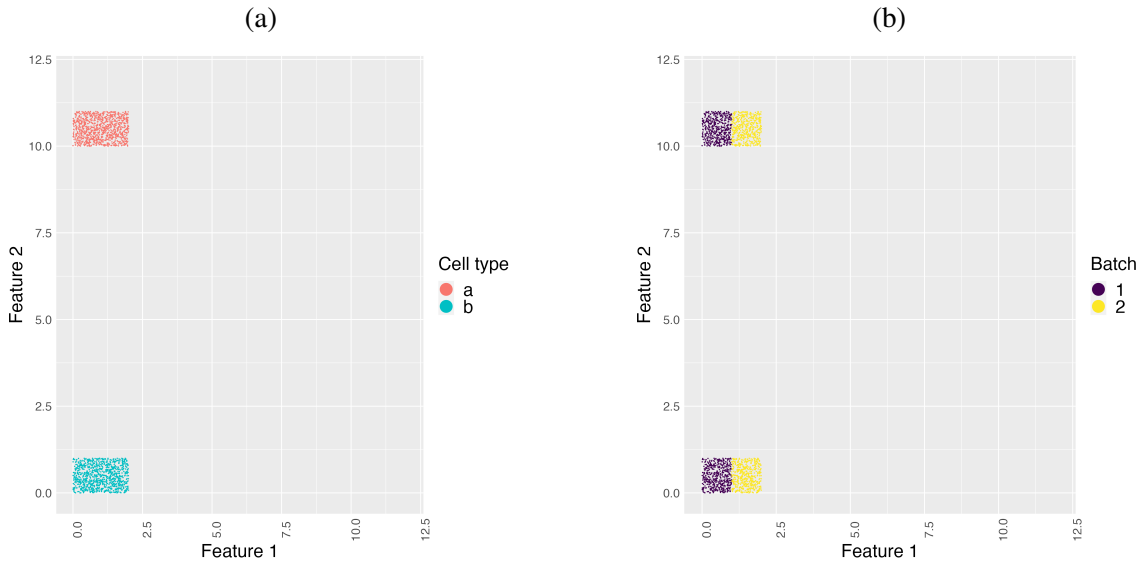


Figure 4.8: Test Case 2: **a**, Dataset visualization based on their cell type. **b**, Dataset visualization based on their original batch.

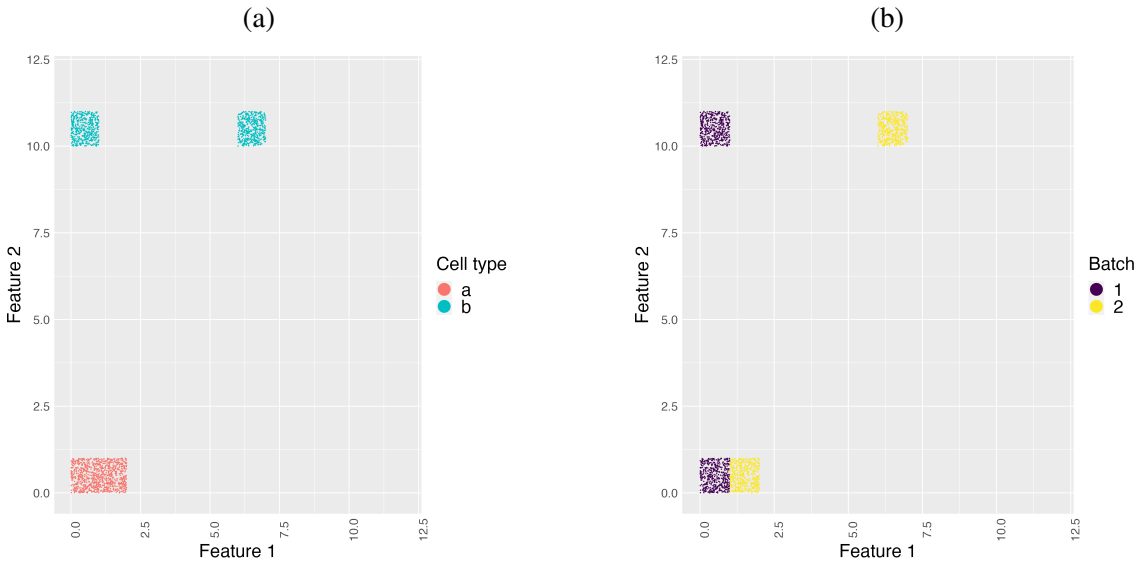


Figure 4.9: Test Case 3: **a**, Dataset visualization based on their cell type. Cells in cell type a are close to each other while cells in cell type b are separated.**b**, Dataset visualization based on their original batch. It can be seen that batch 2 is getting further away from batch 1.

the same cell type than the others. This case is a mid-step between cases 2 and 4 to check if the batch effect metric changes accordingly or not.

Test case 4

The purpose of this case is to smoothly change from case one to case seven. Here cell type a in batch 2 is completely apart from cell type a in batch 1 but cells in cell type b are still

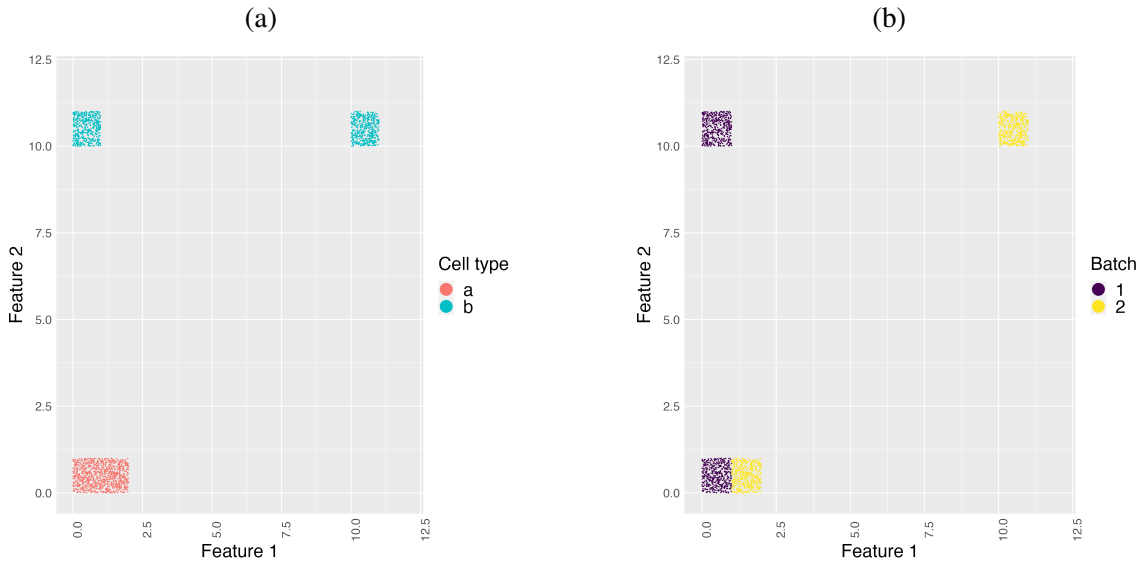


Figure 4.10: Test Case 4: **a**, Dataset visualization based on their cell type. Cells in cell type a are very close while cells in cell type b are apart.**b**, Dataset visualization based on their original batch. The batches are partly apart while and partly close to eachother.

together. Figure 4.10 illustrates this case.

Test case 5

Figure 4.11 represents test case 5. Case 5 represents a border between two main scenarios. In the first scenario, cells of the same type are closer to each other than cells from different batches, while in the second scenario, cells within the same batch are closer to each other than cells of the same type but from different batches.

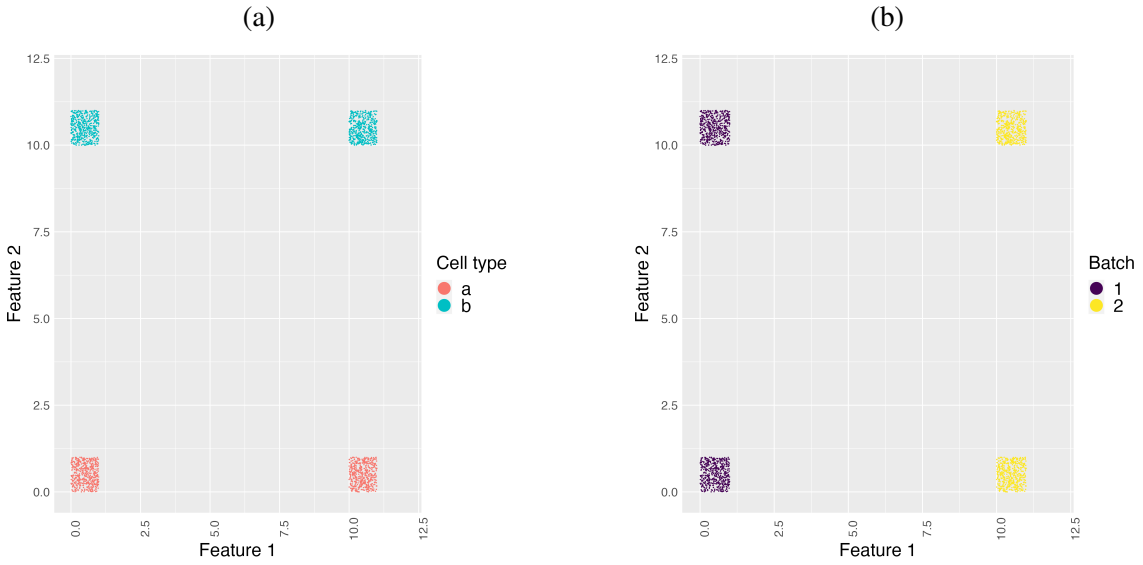


Figure 4.11: Test Case 5: **a**, Dataset visualization based on their cell type. Here call types are completely apart from each other. **b**, Dataset visualization based on their original batch. The batches in this case are also apart and also divided into two parts.

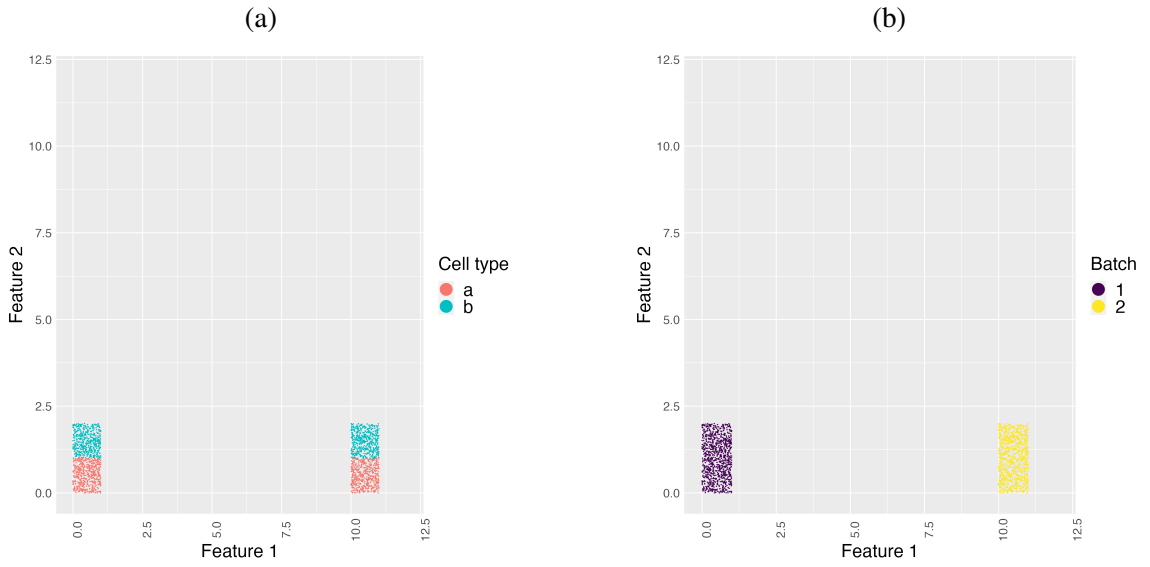


Figure 4.12: Test Case 6: **a**, Dataset visualization based on their cell type. cell types a and b are very close to each other in this case.**b**, Dataset visualization based on their original batch. Batches 1 and 2 are completely separated and indicate a high amount of batch effect.

Test case 6

Case 6 has a very high amount of batch effect, where two cell types in the same batch are close to each other and batches are completely apart. Figure 4.12 shows sample distribution in test case 6.

Test case 7

Case 7 has the most batch effect among all the cases. As shown in figure4.13-a two cell types are not separable at all while figure 4.13-b shows that batches 1 and 2 are completely apart. This is a naturally uncommon case. However, this situation can arise as a result of using a batch regression method, and it is important to investigate the performance of batch effect assessment methods in this case. This is because the inability to determine cell types can have significant consequences for the accuracy of the batch effect assessment.

Test case 8

Figure 4.14 shows sample distribution in test case 8. As in this test case, the number of samples is doubled without changing the data distribution of test case 1. This evaluates the sensitivity of the assessment method to the number of cells.

Test case 9

Case 9 has the same positioning as case 3, but the population of cell type ”a” has been increased. It is because cell type ”a” is the cell type that has a good distribution in terms of batch effect. Figure 4.15 shows sample distribution in test case 9.

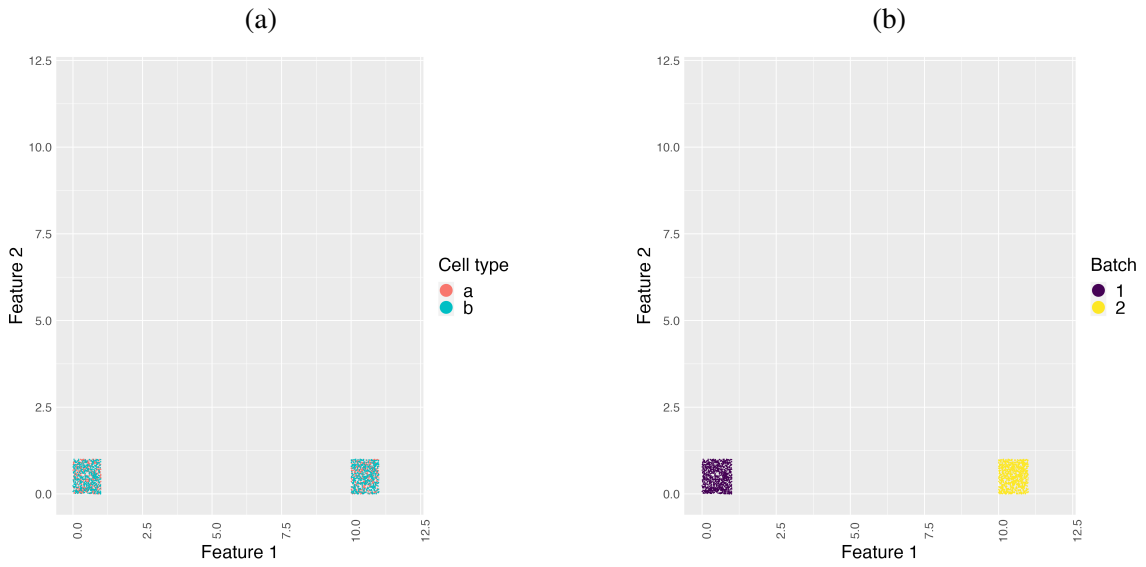


Figure 4.13: Test Case 7: **a**, Dataset visualization based on their cell type. Cell types are completely mixed here. **b**, Dataset visualization based on their original batch. batches 1 and 2 are completely separated from each other.

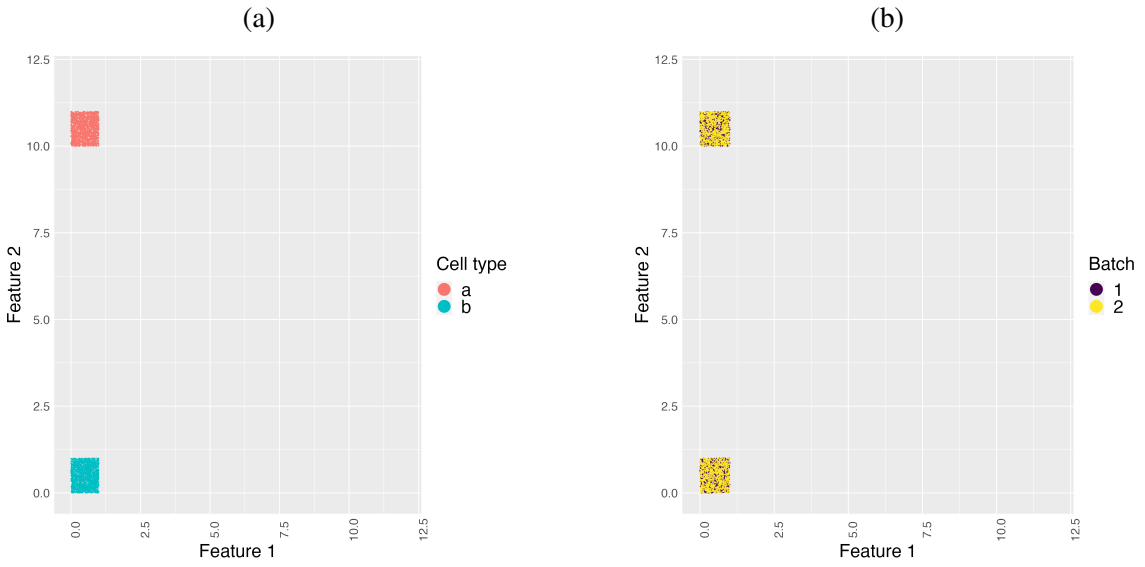


Figure 4.14: Test Case 8: **a**, Dataset visualization based on their cell type. **b**, Dataset visualization based on their original batch.

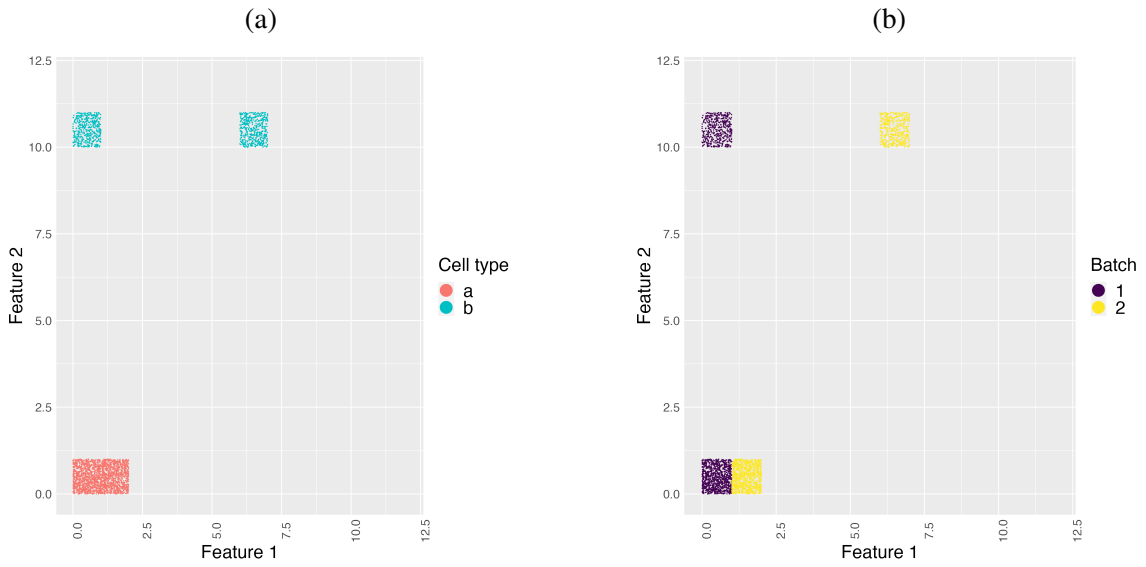


Figure 4.15: Test Case 9: **a**, Dataset visualization based on their cell type. The population of cell type a is more than cell type b. **b**, Dataset visualization based on their original batch. These batches are not well mixed but in the part where they are closer, there are more cells concentrated.

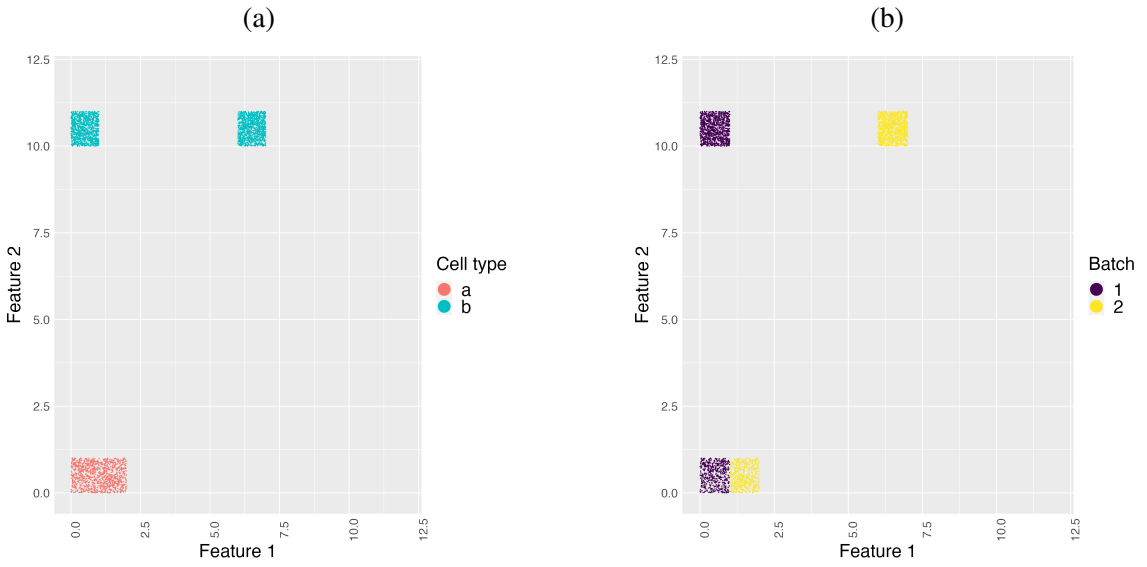


Figure 4.16: Test Case 10: **a**, Dataset visualization based on their cell type. It is visible that the population of cell type b is more than cell type a. **b**, Dataset visualization based on their original batch. These batches are not well mixed and in the part where they are further, there are more cells concentrated.

Test case 10

Case 10 also has the same positioning as case 3, but the population of cell type b has been increased. It is because cell type b is the cell type that has a bad distribution in terms of batch effect. Figure 4.16 shows sample distribution in test case 10.

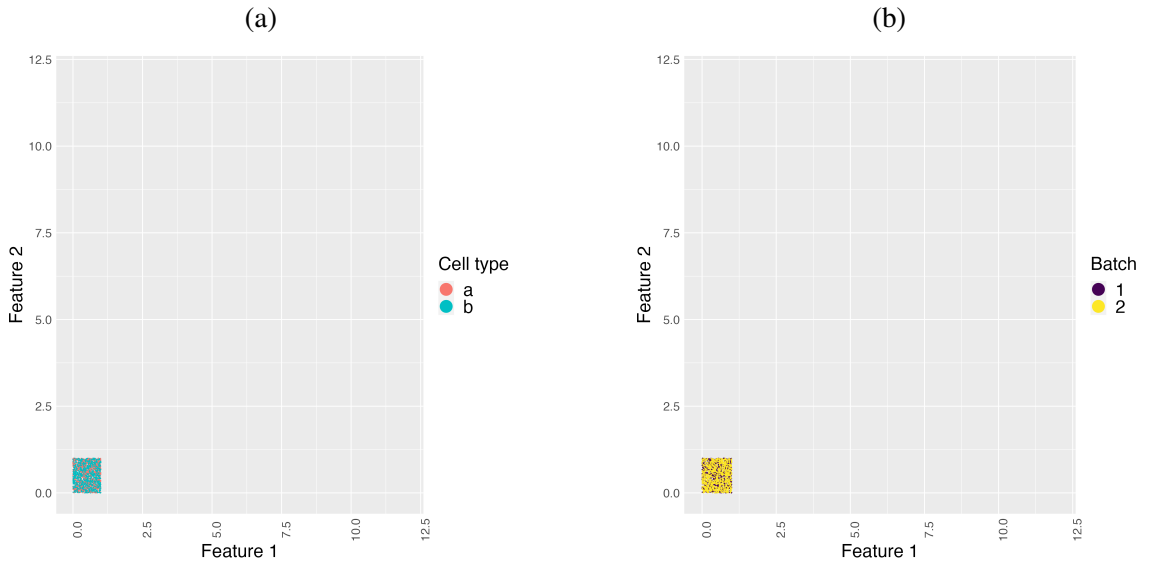


Figure 4.17: Test Case 11: **a**, Dataset visualization based on their cell type. All cell types are completely mixed and inseparable. **b**, Dataset visualization based on their original batch. Batches 1 and 2 are also completely mixed.

Test case 11

Case 11 is the case that all data is mixed together. This can be a result of the batch effect removal method that has moved the same cell types close but it also moved the different cell types close too. Figure 4.17 shows sample distribution in test case 11.

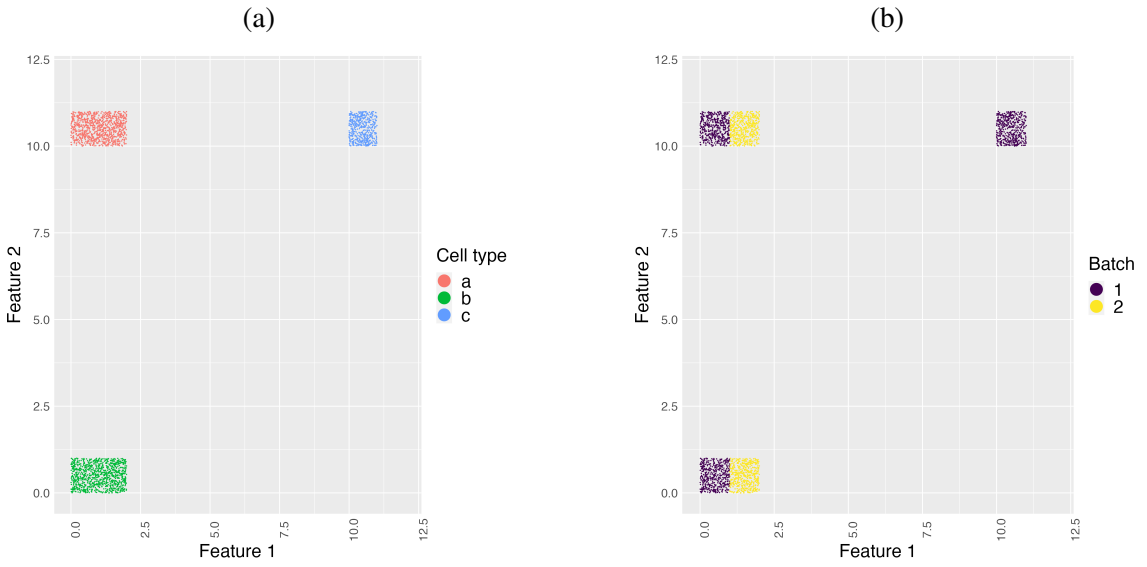


Figure 4.18: Test Case 12: **a**, Dataset visualization based on their cell type. Cell type c is new in this case. This cell type is separated from types a and b. **b**, Dataset visualization based on their original batch. This plot shows that cell type c only is available in batch 1 and batch 2 only has cell types a and b.

Test case 12

This case tries to demonstrate the performance of cell types when a new cell type is added to case 2. Figure 4.18 shows sample distribution in test case 12. In this test, the new cell type is only available in batch two and it is distant from other cell types to only evaluate the presence of this cell type on the results.

4.3.2 Results on Batch Finder

In this section, the batch effect within test cases has been evaluated by the proposed method, batch finder. The results have been explained from two perspectives:

- An expert who wants to look at the detailed results of cell type relationships. The detailed output of the method is used to satisfy this need.
- An expert who wants to use data for further process and needs to choose one of the test cases for their study. The final batch number is used to respond to this need.

The results are explained in groups that cases have a meaningful connection.

Test 1: Cases 1 to 7

For cases from 1 to 7, all of them have the same number of cell types and samples. Also in terms of batch effect, case 1 is the best case and cases gradually get worse until case 7 which is the worst case. As table 4.2 shows, as the cases are getting worst, the batch number is also reducing which means that cases are becoming less acceptable and the proposed method passes this test. Figure 4.19 shows the detailed result of the Batch finder. To demonstrate how this data should be studied, some of the cells are analyzed. For example, the distance from cell type a from batch 1 to cell type b in the same batch, a1-b1, is 1.9 in case 1 and 0.1 in case 7. It means that a and b have a proper distance in case 1.

Case	Batch Finder number
Case 1	4.565
Case 2	4.076
Case 3	1.897
Case 4	0.638
Case 5	-4.592
Case 6	-8.837
Case 7	-9.070

Table 4.2: Batch finder result on cases from 1 - 7

Also, the same relation for case 5 has a value of 1.16 while their positions are the same in the two cases. It is because from the perspective of cell type a batch 1 in case 1, other cell types are closer so this distance is considered as a higher value rather than in case 5 other cell types are also far apart from cell type a batch 1.

Test 2: Cases 1 and 8

Comparing cases 1 and 8 is a test for analyzing the effectiveness of the number of cells in the dataset. It is expected that a well-normalized method calculates almost the same value for both cases 1 and 8.

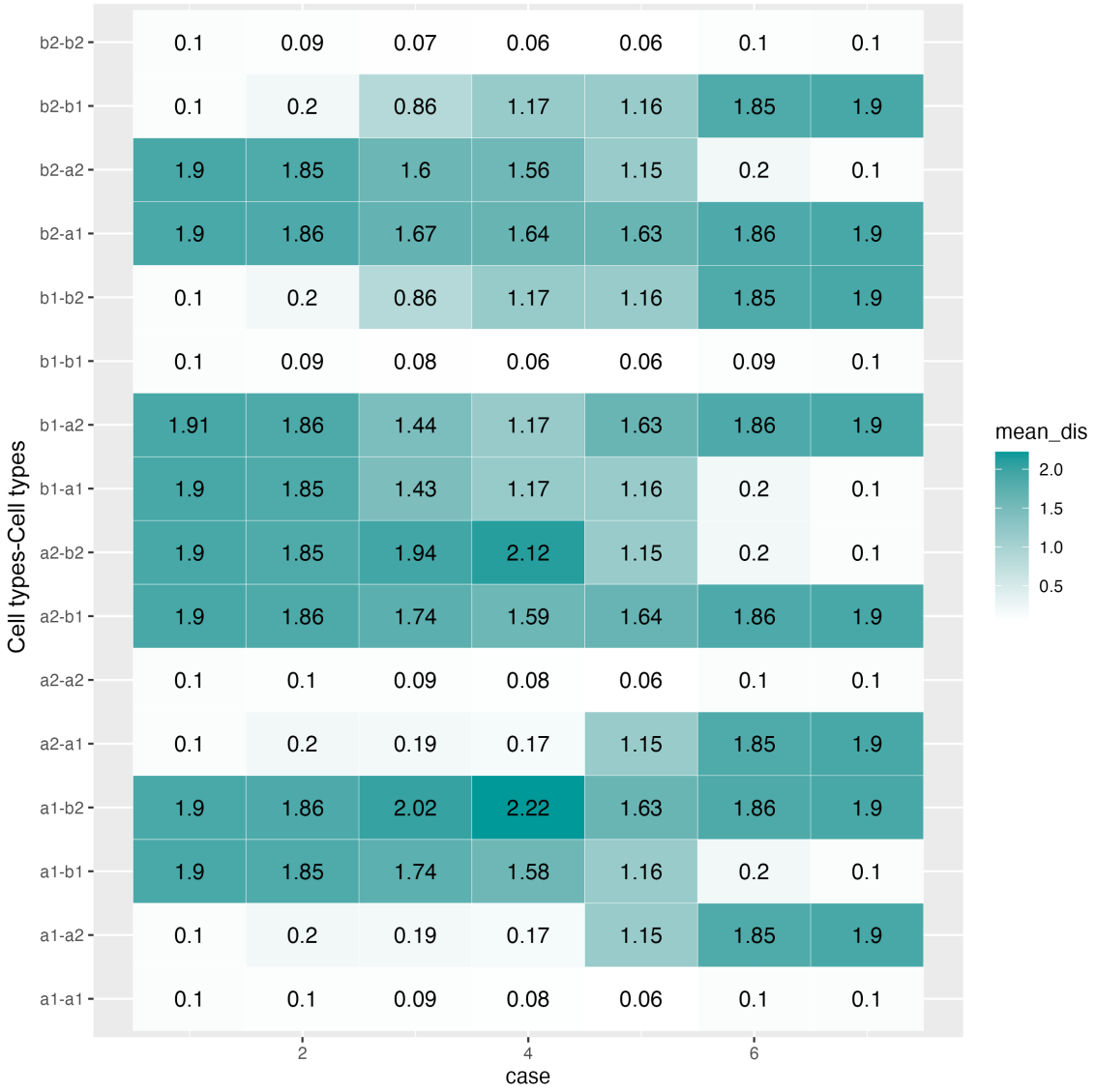


Figure 4.19: Detailed result of running Batch finder on cases 1-7: each row shows the distances between two cell types, either ”a” or ”b”, and two batches, either ”1” or ”2”. Each column shows all mean distances in each case.

The Batch number for case 1 is 4.565 and for case 8 it is 4.596. As the cells are not exactly the same, this variance is an acceptable value and this test is also passed.

Test 3: Cases 3, 9, and 10

Cases 9 and 10 are designed to confirm that the Batch finder method handles a change in the number of cells properly. compared to case 3, case 9 has more cells in type ”a”, which are close in batch ”1” and ”2”, and must have a higher acceptance value. On the other hand, case 10 has more cells in cell type ”b”, which are obviously separated by their batch. The batch number for these cases is presented in table 4.3 and it shows that case 9 is better than 3 and case 3 is better than case 10. So this test is also passed by the Batch finder.

Case	Batch Finder number
Case 3	1.897
Case 9	2.631
Case 10	-0.483

Table 4.3: batch finder result on cases 3, 9, and 10

Test 4: Case 11

This case is a special case just to measure to make sure that by just randomly putting all the cells together it won't result in a high acceptance rate. The result for this case is -0.0009, which is between case 4 and case 5. As case 5 is a very well-separated case, this value is acceptable and this test is passed.

Test 5: Cases 2, 12

This test aims to evaluate the effect of non-common cell types on the results. As another cell type is added to batch "1" in case 12, it is expected to have a higher value as this cell type is not mixed with the other two cell types. The result value for case 12 is 6.134, compared to 4.076 for case 2. It means that this test is passed and the method can correctly take non-common cell types into consideration.

4.3.3 Results on other assessments

kBET, Silhouette, and ARI are other assessment methods that can be used here to evaluate test cases. Mixing metric is not used in this section because it is a built-in method in Seuratv4 [18] and can only run on the Seurat objects.

Figure 4.20 shows the results. In order to show results better, the Silhouette method results have been negated and biased, because the Silhouette method returns an average dissimi-

larity and not an acceptance measure.

kBET

kBET method seems to be oversensitive. The results only in cases 1 and 8 are acceptable and it doesn't accept cases 5, 6, and 7. So in test 1, it doesn't fail although some bad cases are considered equal.

kBET passes test 2 as cases 8 and 1 have almost the same value.

In test 3, the value for case 9 is more than case 10 but it is less than case 3, which is a fail.

kBET fails test 4 completely and it considers case 11 as one of the best cases.

For test 5 also kBET results in the wrong direction and ranks case 2 higher than 12.

Silhouette

Silhouette method passes test 1 as all the values gradually decay from case 1 to 7. It passes tests 2 and 3 as well. This method fails in test 4 as it has completely accepted case 11. In test 5, it is not a complete fail but as the value for case 12 is less than case 2, it's considered a fail.

ARI

ARI resulted in the same value for many cases. It is because the ARI algorithm decides based on k-mean clustering and until case 4, all cell types are clustered completely cor-

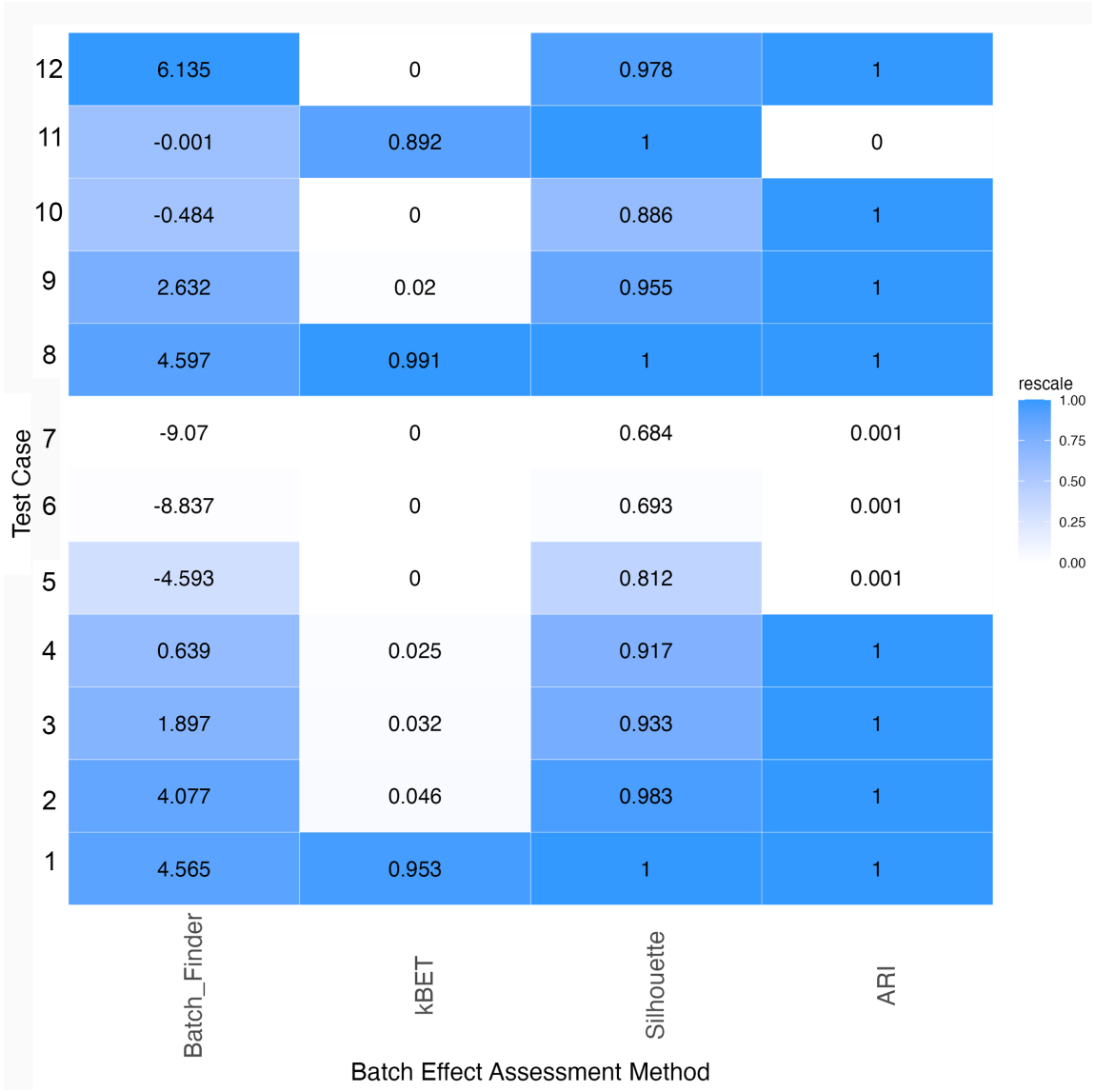


Figure 4.20: Result of batch assessment with 4 assessment methods. Every column is scaled as the results of different methods are not on the same scale.

rectly. The results are not contradictory with any expected result and test 4 passes as well. As powerful as ARI is, it may result in some misleading results in complex datasets.

4.3.4 Discussion

As the table 4.4 summarizes the test results with these methods, Batch finder passes all tests, kBET fails in the last three, Silhouette fails in the last two, and ARI fails in one test but as it doesn't differentiate the cases well enough, it may face problems in intricate datasets.

Test	Batch Finder	kBET	Silhouette	ARI
Test 1	Pass	Pass	Pass	Pass
Test 2	Pass	Pass	Pass	Pass
Test 3	Pass	Fail	Pass	Fail
Test 4	Pass	Fail	Fail	Pass
Test 5	Pass	Fail	Fail	Pass

Table 4.4: Summary of performance of batch assessment method on simulation data

4.4 Conclusion

In this chapter, Batch finder was introduced as a new method for batch effect assessment. Then 12 test datasets were defined and 5 different scenarios were evaluated by this simulation data. Batch Finder was compared to three other methods using these tests, and the results were analyzed and discussed. The results demonstrated that Batch Finder consistently outperforms the other methods and provides a reliable estimation of batch effects for each dataset. Overall, this research suggests that Batch Finder is a valuable tool for accurately assessing batch effects in a variety of contexts.

Chapter 5

Results on Biological Datasets

Batch finder was introduced in 4.2 and simulation data has been defined and tested. In this chapter, 3 biological datasets are going to be introduced and after preprocessing, 7 batch removal methods will be applied to them. The performance of these methods is going to be evaluated with batch finder and 4 other methods. Finally using an expert opinion on the batch effect within the datasets, all assessment methods are compared together. In order to evaluate batch assessment methods, proper datasets with specific characteristics are chosen. These datasets must have a systematic batch effect so that the effort to remove them is valid. Each dataset needed to be standardized to be coherent with other datasets and mix easily. To do that all gene IDs where converted to Ensembl IDs. Each cell type annotation has also been manually checked to be consistent with the other datasets. Then

ddatasets would be ready to merge together.

Before generating other assessment results, an expert opinion, also referred to as manual assessment, was carried out by the author. This assessment utilized all three visualization methods: PCA, UMAP, and t-SNE. Although the removal method labels were eliminated, the assessment cannot be considered entirely blinded, as the author initially generated these labels. The objective of the manual assessment was to apply the same criteria as those established for the algorithm, which aimed to estimate batch effect in the dataset. This process was considered the thought process for estimating batch effect and served as a foundation for the development and validation of the proposed algorithm

5.1 dataset1: Pancreas

The pancreas dataset consists of four different datasets for different experiments. These datasets are:

- Dataset 1: This dataset is available at GSE81076. In this research, CelSeq protocol was used on Illumina platform[56]. This dataset has 1004 cells.
- Dataset 2: This data is available at GSE85241 and uses CelSeq2 protocol on Illumina platform [57]. There are 2285 cells in this dataset.
- Dataset 3: This dataset which uses the Fluidigm C1 system, is available at GSE86469

[58]. This dataset contains 638 cells.

- Dataset 4: The last dataset is obtained using the SMART-Seq2 protocol and is available at E-MTAB-5061 [59]. This dataset has 2394 cells.

Integration of these datasets will create a dataset with 6321 cells and 34363 genes and 13 unique cell types. Although all these datasets are obtained from Illumina devices, they used different technologies to produce count matrices and their ”tech” is the origin of the batch effect. But before showing the presence of a batch effect, we need to pre-process every dataset and then merge these datasets together.

5.1.1 Pre-processing

After loading each dataset, they went through the standard steps of normalization, finding variable features, and scaling. Figure 5.2 shows these steps. In order to make a successful integration, all gene names should be from the same standard naming system so that the same genes have the same symbol in all datasets. It can be Ensembl, HUGO id, or any other standard. After Converting all gene names to a unified system , count matrices can be merged. As shown in figure 5.1, gene names are merged together and cells are binded together. The cell type information is provided in the original datasets.

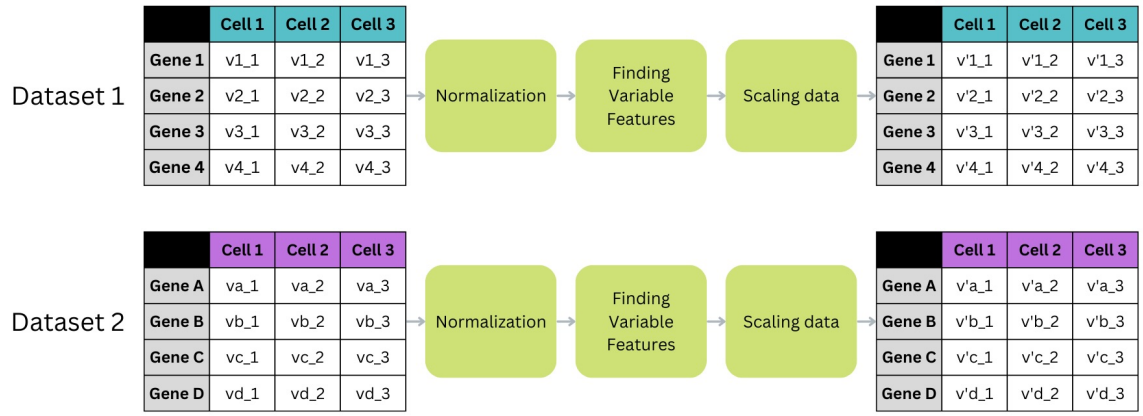


Figure 5.1: Preprocess of each dataset before merging together. Count matrices values of two datasets are subject to this step.

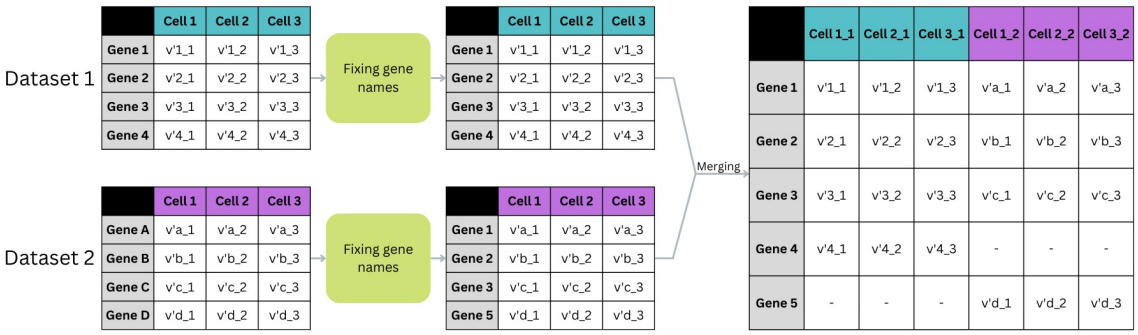


Figure 5.2: Merging two datasets.

5.1.2 Dataset Visualization

After mixing 4 datasets as explained in 5.1.1, for a better understanding of the dataset, UMAP has been plotted in figure 5.3. Also, the t-SNE visualization of the dataset is available in Appendix A. As it is visible in 5.3-b, batches are isolated and 5.3-a shows that the same cell types are apart, indicating batch effect is present within the integrated dataset.

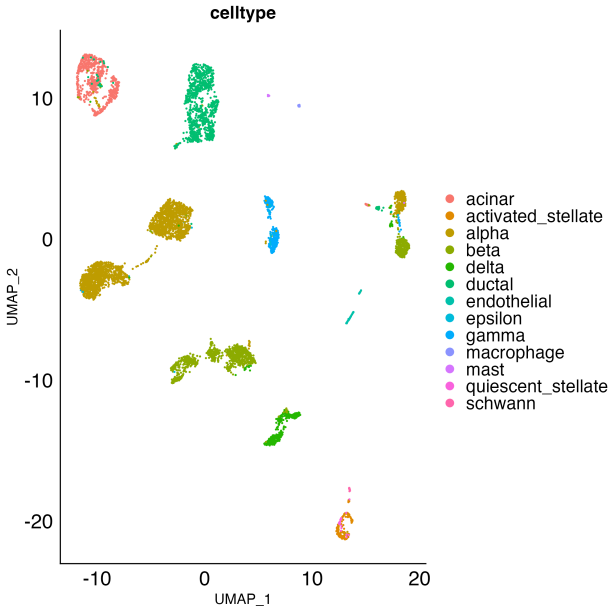
5.1.3 Performing Batch Effect Removal Methods

As shown in 5.1.2, a substantial batch effect exists in the integrated data. Here 7 batch effect removal methods, mentioned in 2.4, will be applied to the integrated dataset. These 7 methods are Harmony, Liger, Limma, Seurat CCA, FastMNN, Conos, and Combat. Although these methods use different techniques, finally all of them provide either a new count matrix or a new dimension reduction for the dataset as the result of their attempt to correct the batch effect.

Harmony

After running the standard workflow on the raw integrated dataset including normalization, finding variable features, scaling the data, and running PCA, the built-in function RunHarmony in the Seurat object is used to generate harmony dimension reduction data. Figure

(a)



(b)

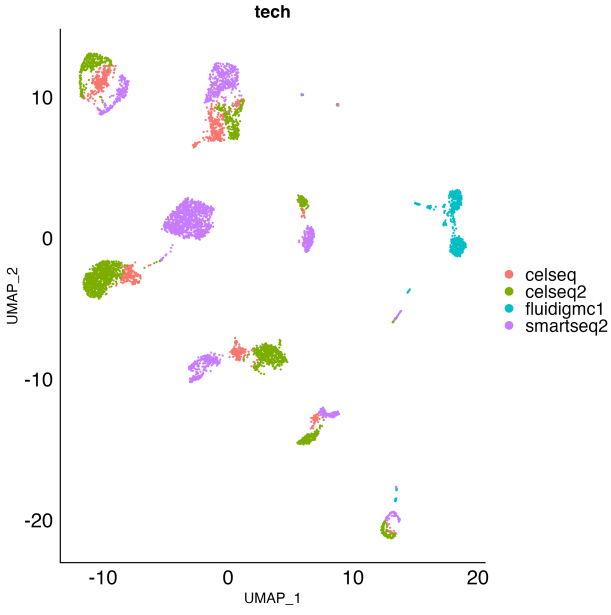


Figure 5.3: UMAP visualization of the pancreas integrated data before batch effect removal. Each point represents a cell in the dataset. **a**Each colour shows a different cell type in this figure. **b**, This figure specifies batches with different colours.

5.4 demonstrates the UMAP dimension reductions of the harmony batch effect removal method.

Limma

The Limma method uses the count matrix and batch factors to create a new count matrix. By generating a new Seurat object with this new count matrix and the original metadata from the dataset, the standard preparation workflow is applied to the corrected dataset. Figure 5.5 shows UMAP visualization on the dataset corrected with the limma batch effect removal method.

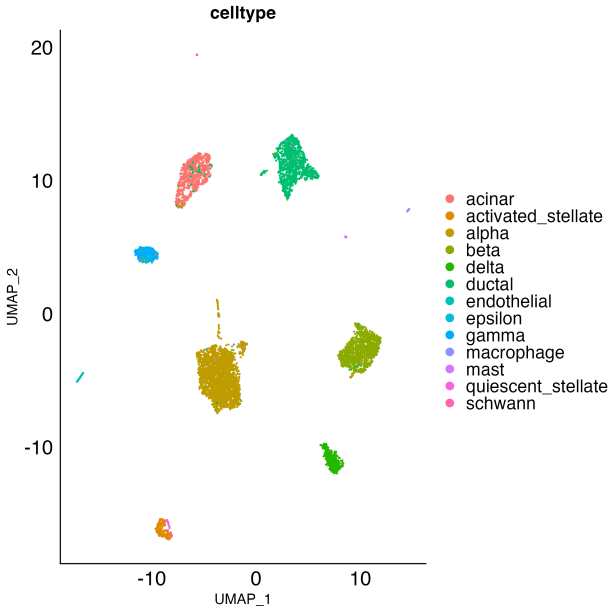
Liger

In order to implement the Liger method, first, the standard workflow must be run on the dataset and then RunOptimizeALS and RunQuantileNorm from the SeuratWrappers package which is provided for Seurat object [18] to generate a new dimension reduction named "iNMF". The UMAP of this batch effect correction method is provided in fig 5.6.

Seurat Canonical Correlation Analysi

After implementing Seurat CCA and finding the integration anchors, we obtained the integrated assay visualized in Figure 5.7.

(a)



(b)

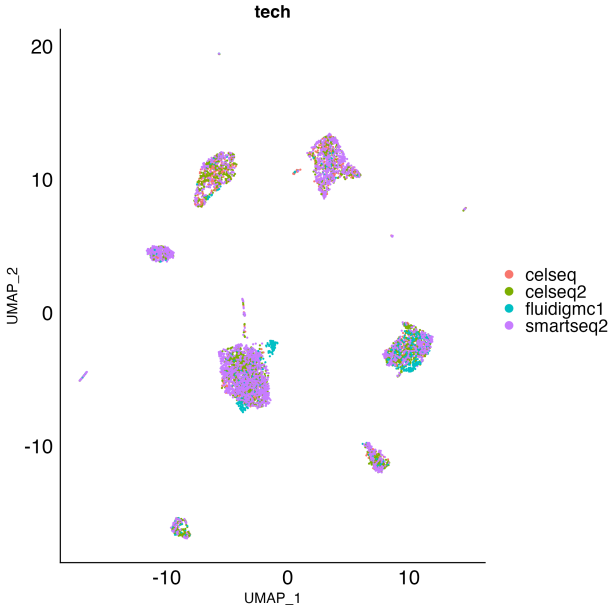
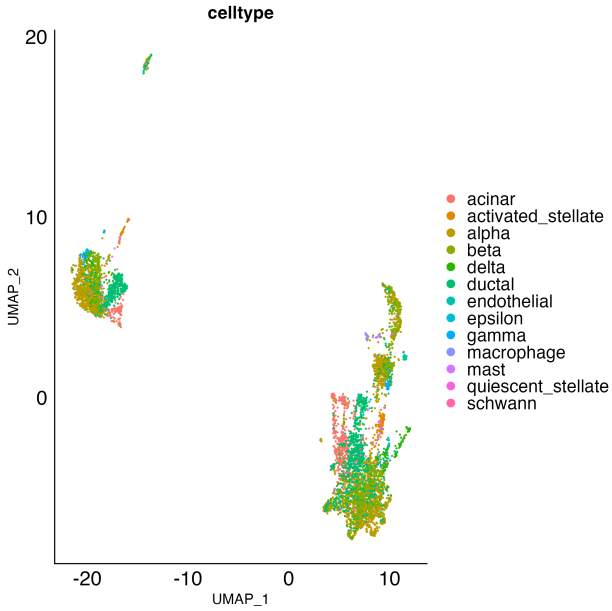


Figure 5.4: UMAP of the pancreas integrated data after harmony method. **a**, cell types. **b**, batches.

(a)



(b)

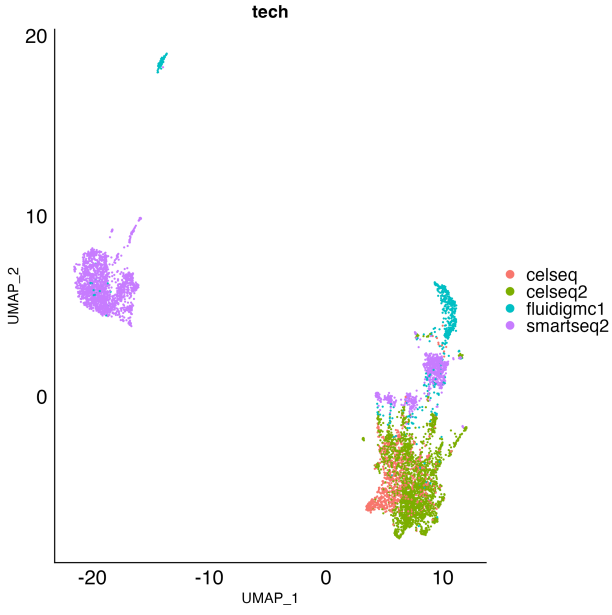
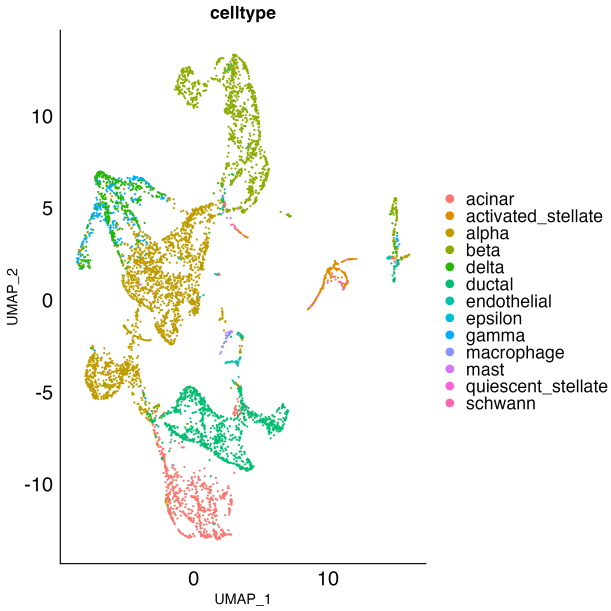


Figure 5.5: UMAP of the pancreas integrated data after limma method. **a**, cell types. **b**, batches.

(a)



(b)

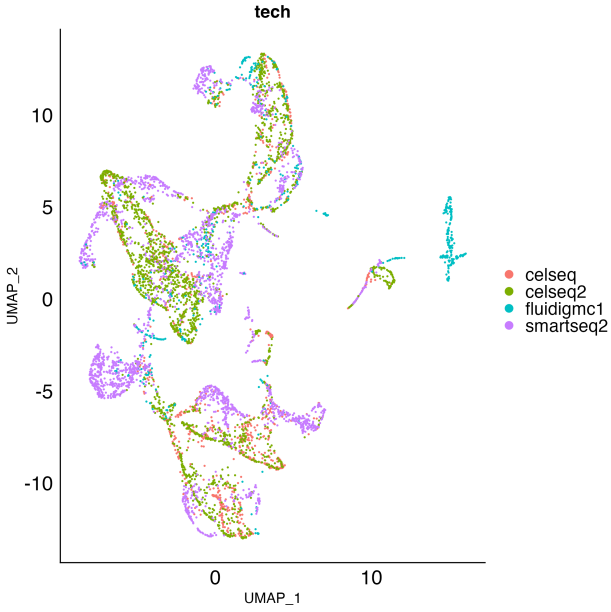
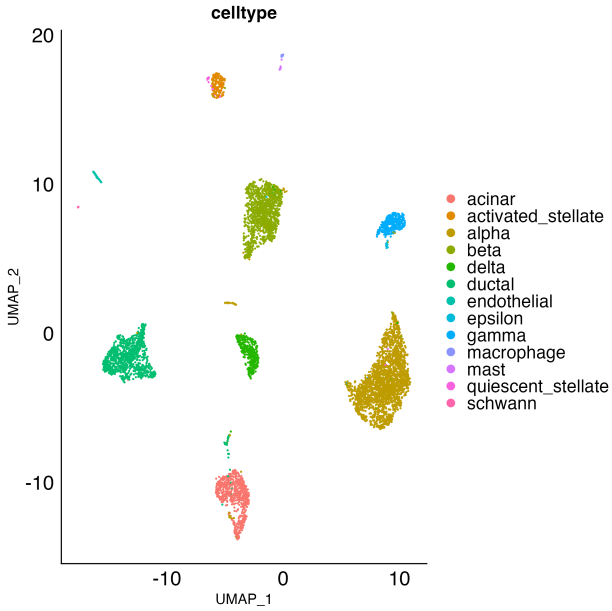


Figure 5.6: UMAP of the pancreas integrated data after liger method. **a**, cell types. **b**, batches.

(a)



(b)

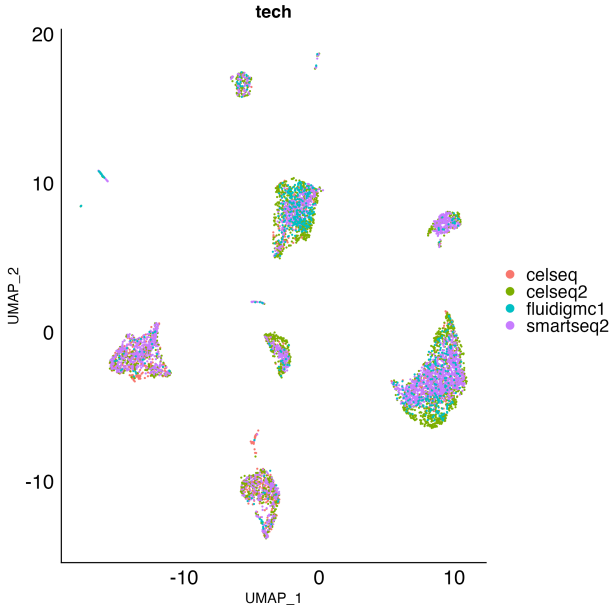


Figure 5.7: UMAP of the pancreas integrated data after Seurat CCA method. **a**, cell types. **b**, batches.

FastMNN

Using the built-in function in Seurat, we generated the UMAP visualization of the ”mnn” dimension reduction, as shown in Figure 5.8.

Conos

In the Conos package after building a graph and finding communities, a 2-D visualization for data is generated. This visualization is named ”largeVis” and is shown in figure 5.9.

Combat

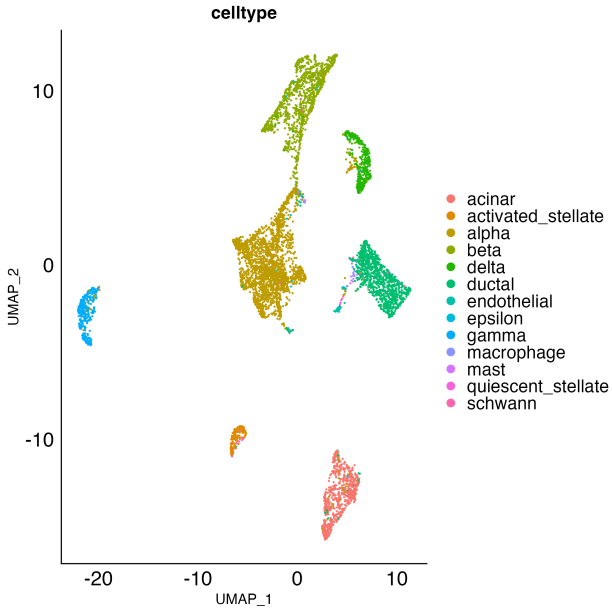
The Combat method performs its algorithm on the count matrix and generates a new count matrix that after normalization and finding variable features, has a UMAP visualization that is shown in figure 5.10.

5.1.4 Manual assessment of batch effect removal on pancreas dataset

In order to evaluate the performance of batch effect assessment methods, an expert opinion is needed to be used as a reference. In this section, a thorough analysis is performed on UMAP visualization data. The evaluation of batch effect removal methods is listed in order from the best batch effect removal method to the worst method:

1. CCA: In the UMAP of CCA demonstrated in figure 5.7, batches are well mixed.

(a)



(b)

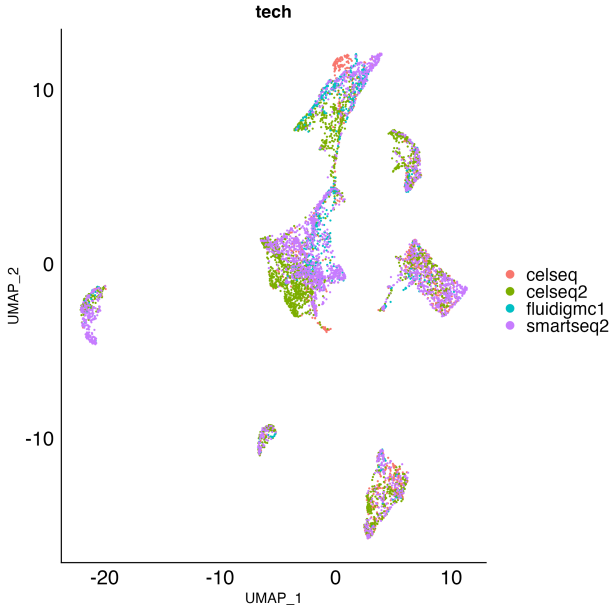
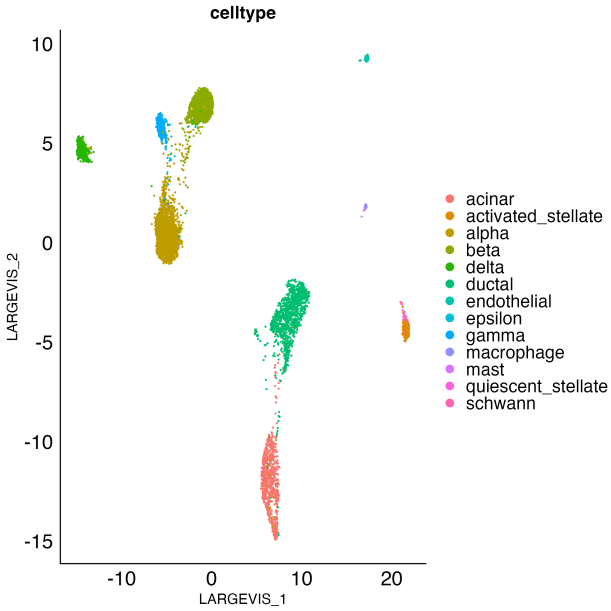


Figure 5.8: UMAP of the pancreas integrated data after FastMNN method. **a**, cell types. **b**, batches.

(a)



(b)

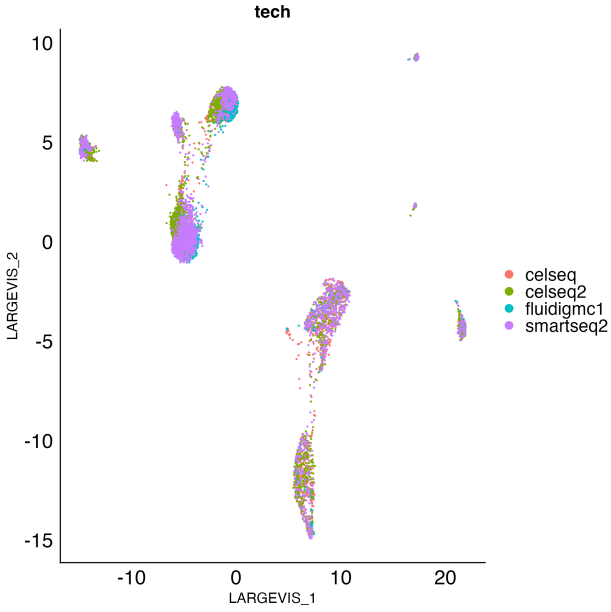
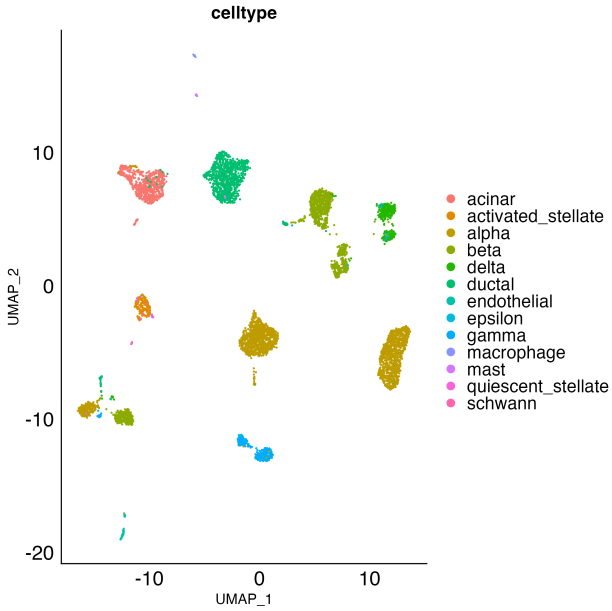


Figure 5.9: largeVis visualization of the pancreas integrated data after Conos method. **a**, cell types. **b**, batches.

(a)



(b)

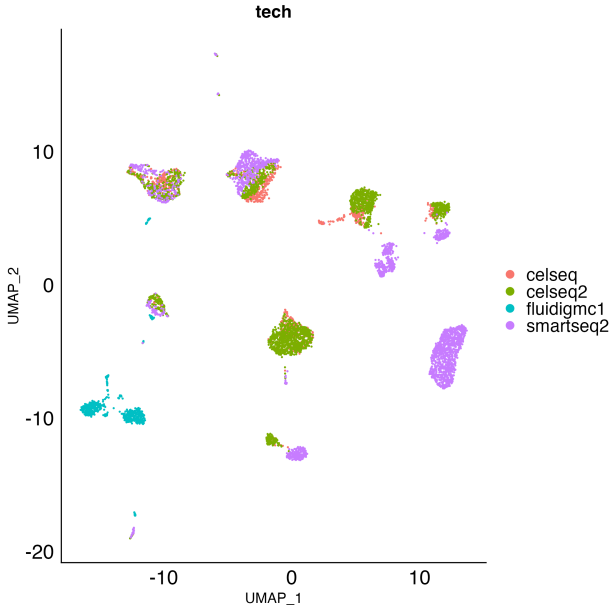


Figure 5.10: UMAP visualization of the pancreas integrated data after Combat method. **a**, cell types. **b**, batches.

The same cell types are also well mixed except for a few alpha cells and a few ductal cells. Epsilon and gamma cells are too close but as it is the same in each batch, they can be ignored. There is also a very small number of cells that are in the wrong cell types. In total, this is the best batch effect removal attempt among all others.

2. Conos: Referring to largeVis shown in figure 5.9, it's obvious that parts of batches around alpha and beta cell types are not well mixed. The number of miss placed cells is also small.

3. Harmony: In the harmony method, alpha cell types from the fluidigm c1 batch are separated by a distance from other alpha cells. Some misplacement of ductal cells in acinar cells is also visible.

4. FastMNN: Referring to figure 5.8, batches are well mixed except for some parts in alpha cell type and gamma cell type. In terms of organizing cell types, although they are separable, epsilon, endothelial, microphage, and mast cell types are too close to each other and ductal cell types while in each batch these cell types are not very close to each other. There are some alpha cells near delta cells and some cells can be seen around cells from different cell types.

5. Combat: This method result is almost the same as the raw data i.e. before batch

effect removal, provided in figure 5.10. Most parts of batches are still separable and the same cell types from different batches are apart. There have been only a few improvements on the batch mixture in acinar and ductal cell types.

6. Raw data: This is the original positioning of cells in the dataset after mixing batches together shown in figure 5.3. Fluidigm c1 is completely separated, but other batches are better mixed. yet in each cell type batches are not mixed at all.

7. Liger: The result of the liger method shown in figure 5.6is worse than the raw data. The reason is that here even separating cell types is more difficult and the fluidigm c1 batch is still away from the others.

8. Limma: In the UMAP of this method demonstrated in figure 5.5, batches are still unmixed and even cell types are not separable at all. They are just shuffled and mixed together and that is why this is the worst attempt in this dataset.

5.1.5 Batch assessment methods results

Each batch effect removal method has been evaluated by 5 batch assessment methods. These methods are Batch_Finder, ARI, kBET, Silhouette, and Mixing metric. Figure 5.11 provides the result of these methods. Each value is an acceptance value and a higher value means better batch effect removal performance. The heatmap is coloured based on

each column's values and not the whole table. As these numbers are in different ranges, for a better understanding of the results and to make them comparable, the rank of each batch effect removal method is calculated according to every assessment method. Figure 5.12 shows the rank table for the pancreas dataset. Also, the expert's opinion is also added to the table as "Manual" assessment. To understand the correlation results better, a boxplot is provided in figure 5.13, the ranks for each batch effect removal method are summarized in a boxplot and the colour of each point determines which method has given this rank to the removal method. The goal is to find out which assessment method is closer to the expert's opinion. Finding the correlation between rank table values can reveal the similarity between all assessment methods. Figure 5.14 demonstrates the correlation between all assessment methods. A higher value of correlation shows the similarity of assessment methods.

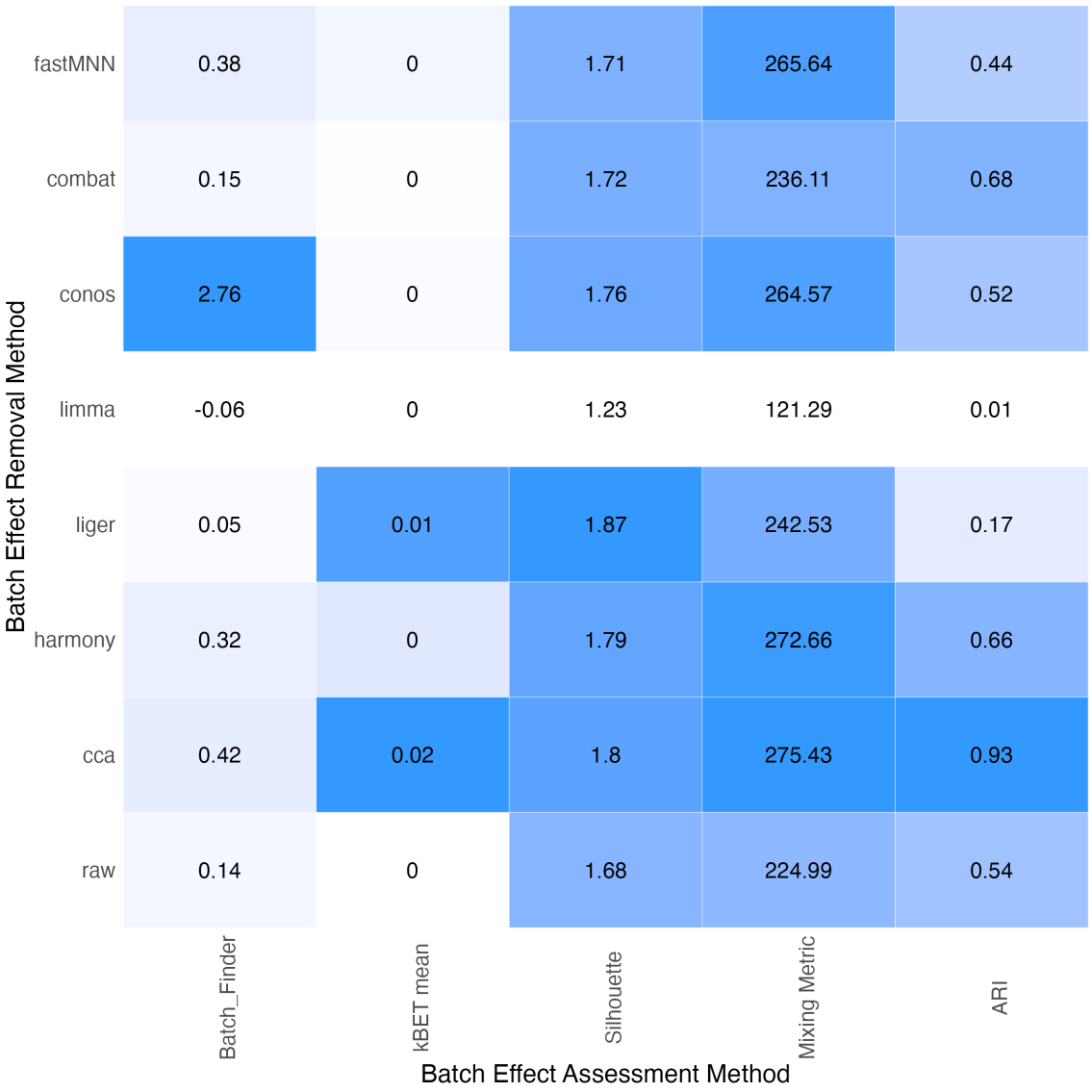


Figure 5.11: Batch effect assessment values on pancreas dataset. Each column of this figure is for one batch effect assessment method. Heatmap colours are based on each column's values.

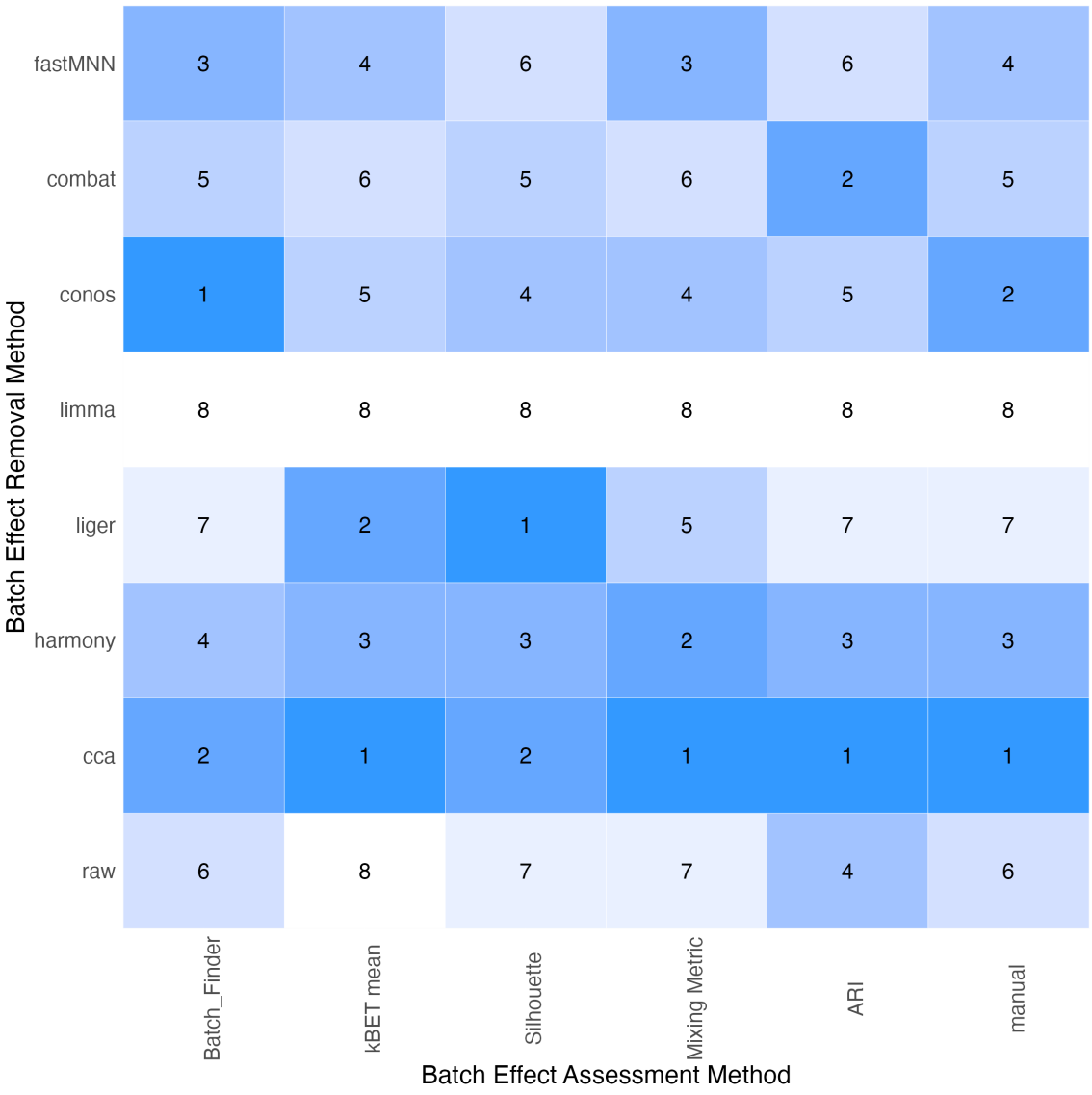


Figure 5.12: Batch effect assessment ranks on pancreas dataset. Each column of this figure is for one batch effect assessment method. The manual column is according to the expert's opinion.

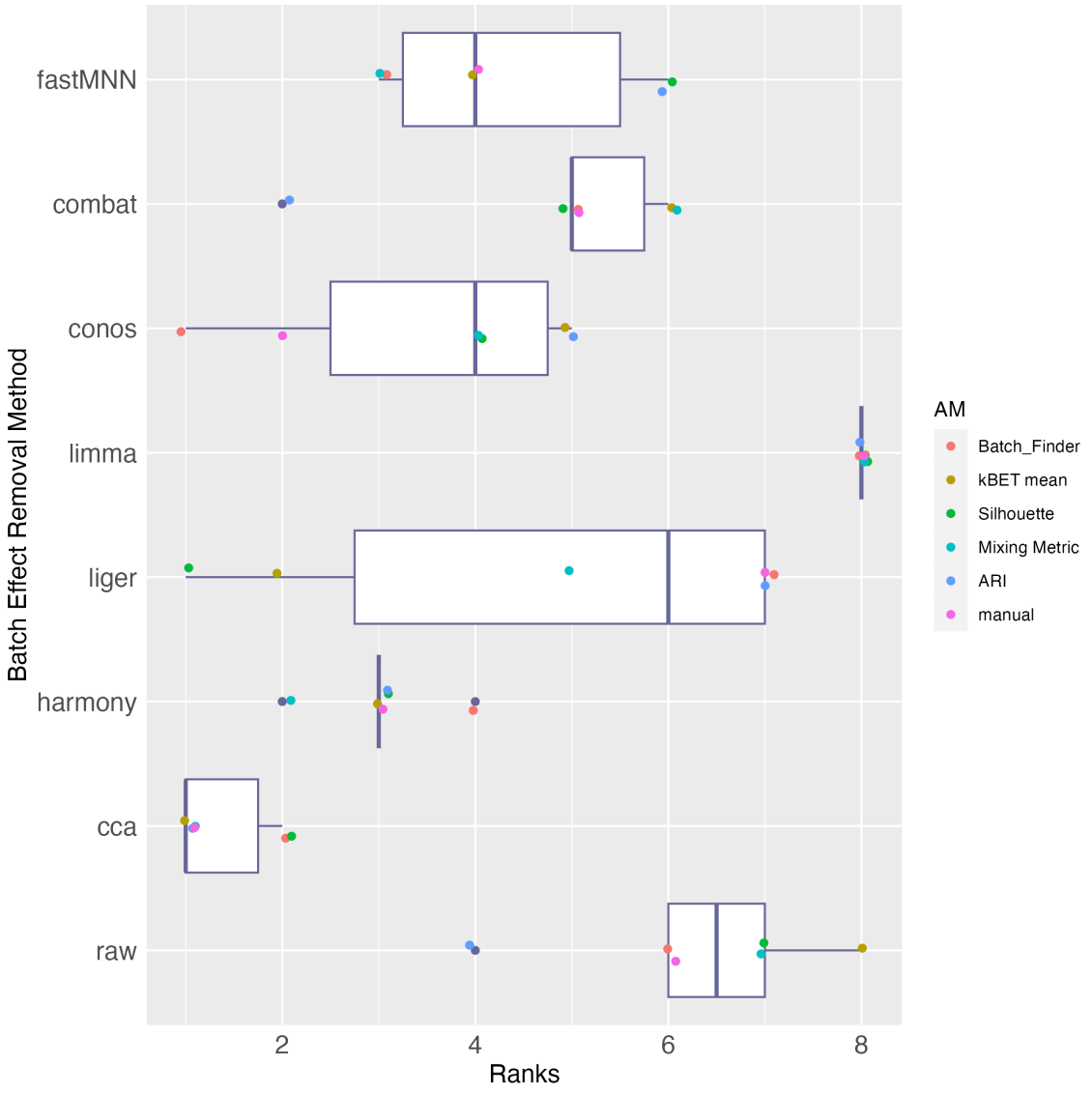


Figure 5.13: Batch effect assessment ranks boxplot on pancreas dataset. Each box shows the ranks given to each batch effect removal method.

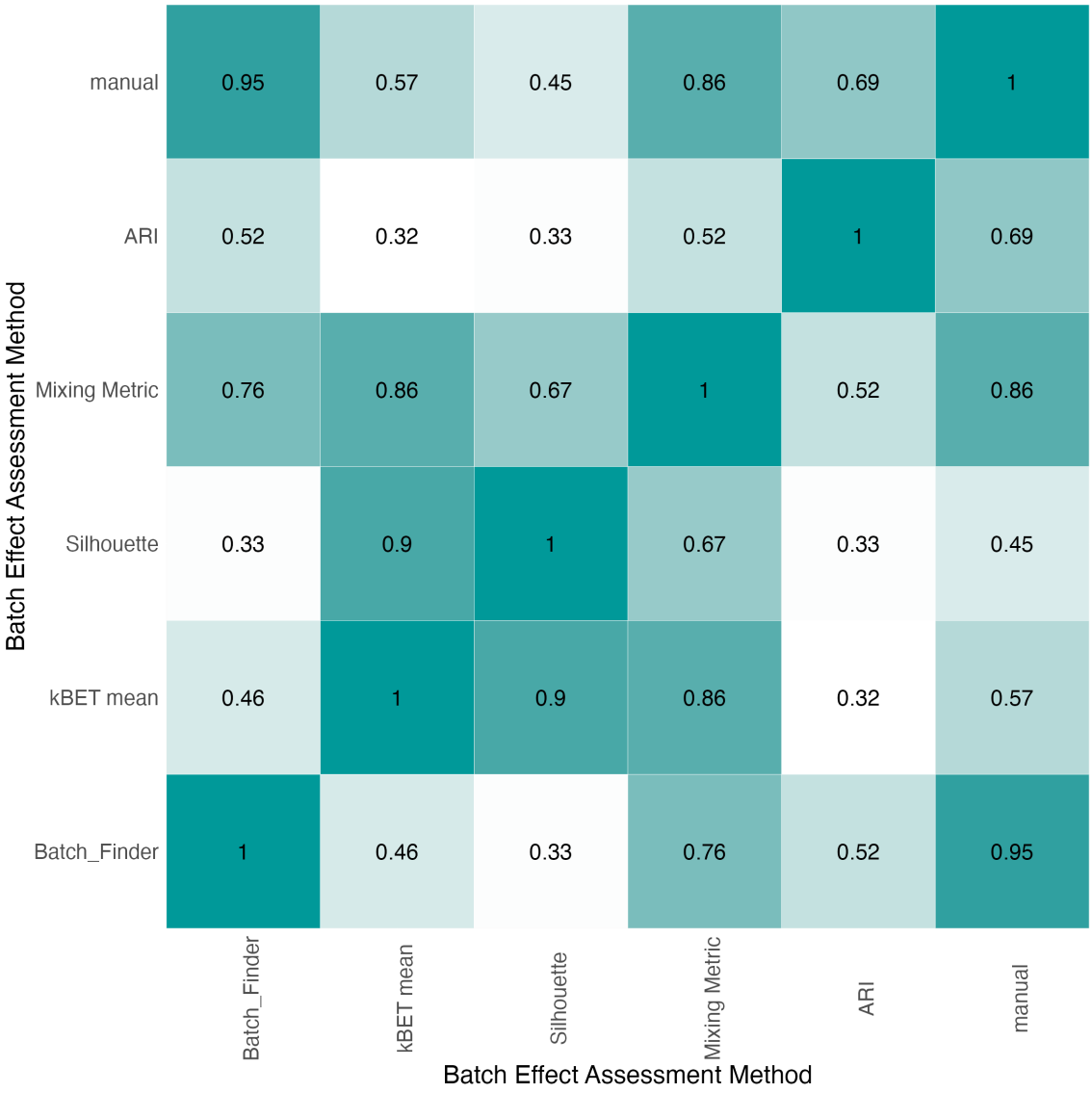


Figure 5.14: Batch effect assessment ranks correlation on pancreas dataset.

5.2 dataset2: Mouse brain

The mouse brain dataset includes 2 of 11 separate samples from research done on mouse cerebral cortex [60]. These samples are:

- Sample 1: This sample is from a mouse brain and belongs to the 18th embryonic day. It has 7137 cells.
- Sample 2: This data is also a sample on embryonic day 18. It has 13138 cells in it.

Integration of these samples created a dataset with 20275 cells and 19712 genes and 19 unique cell types. Although all these datasets are obtained by the same procedure and from the same lab, they used different technologies to produce count matrices and their ”donor” is the origin of the batch effect. These two samples have been chosen because both belong to the same embryonic day and almost similar brain development is expected.

5.2.1 Pre-processing

To merge two datasets together, a shorter set of steps needs to be taken because both experiments were done by the same lab so gene names and metadata available for them are coherent. Two datasets just need to be binded and then go through the standard workflow.

5.2.2 Dataset Visualization

After creating the integrated dataset, the UMAP visualization of the dataset is in figure 5.15. Batches are completely apart while in each batch cell types pattern is analogous, indicating batch effect is present within the integrated dataset.

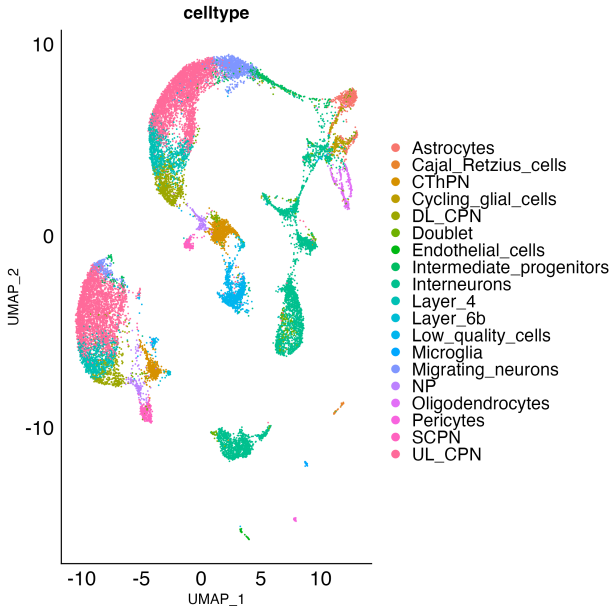
5.2.3 Performing Batch Effect Removal Methods

As a substantial batch effect exists in the integrated data, 7 batch effect removal methods, mentioned in 2.4, were applied to the integrated dataset. These methods are Harmony, Liger, Limma, Seurat CCA, FastMNN, Conos, and Combat. Although these methods use different techniques, finally all of them provide either a new count matrix or a new dimension reduction for the dataset as the result of their attempt to correct the batch effect.

Harmony

To process the integrated dataset, the standard workflow was followed, which included normalization, identifying variable features, scaling the data, and running PCA. After these steps, the RunHarmony function in the Seurat object was used to generate harmony dimension reduction data. Figure 5.16 demonstrates the UMAP dimension reductions of the harmony batch effect removal method.

(a)



(b)

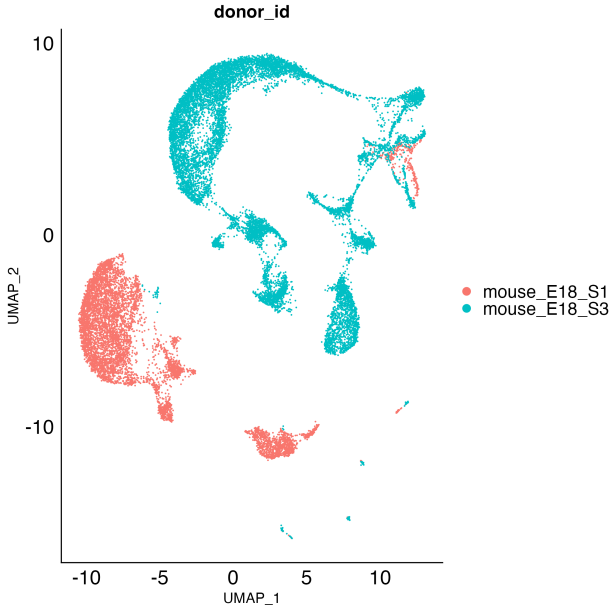
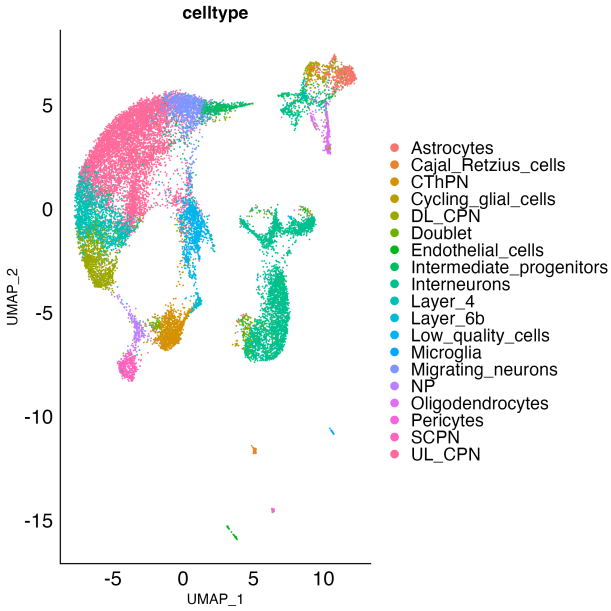


Figure 5.15: UMAP of the brain integrated data before batch effect removal. **a**, cell types. **b**, batches.

(a)



(b)

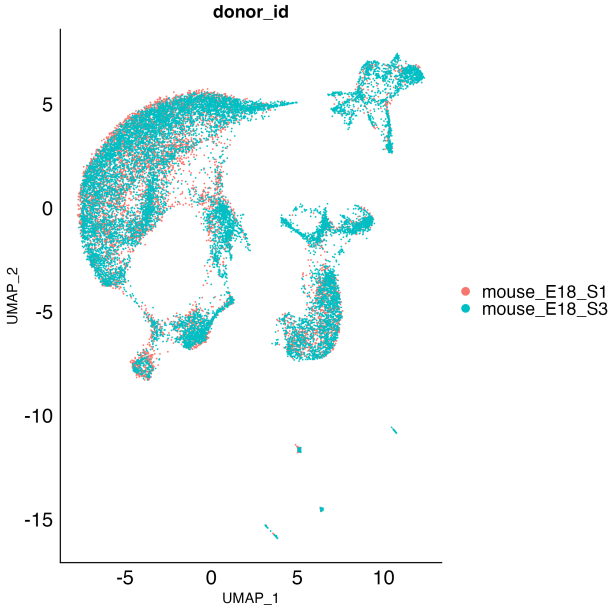


Figure 5.16: UMAP of the brain integrated data after harmony method. **a**, cell types. **b**, batches.

Limma

The Limma method uses the count matrix and batch factors to create a new count matrix. By generating a new Seurat object with this new count matrix and the original metadata from the dataset, the standard preparation workflow would be applied to the corrected dataset. Figure 5.17 shows UMAP visualization on the dataset corrected with the limma batch effect removal method.

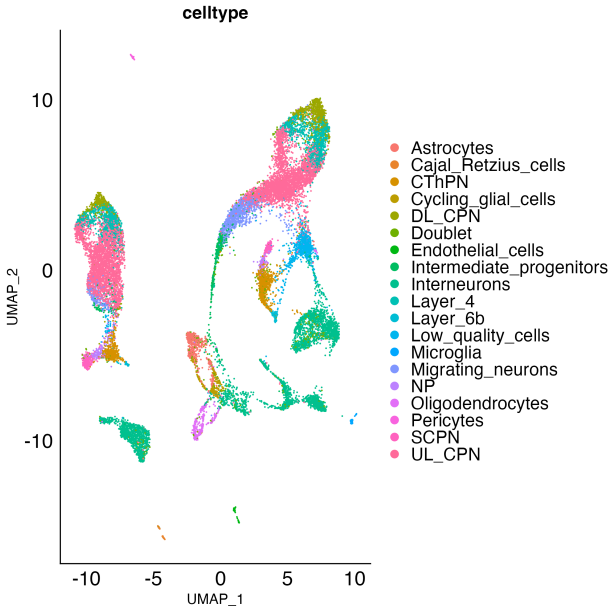
Liger

To implement the Liger method, the standard workflow must first be applied to the dataset. Then, the RunOptimizeALS and RunQuantileNorm functions from the SeuratWrappers package can be used to generate a new dimension reduction called ”iNMF”. The UMAP of this batch effect correction method is provided in fig 5.18.

Seurat CCA

After implementing Seurat CCA on the brain dataset by finding the integration anchors, the new integrated assay has been visualized by UMAP in figure 5.19.

(a)



(b)

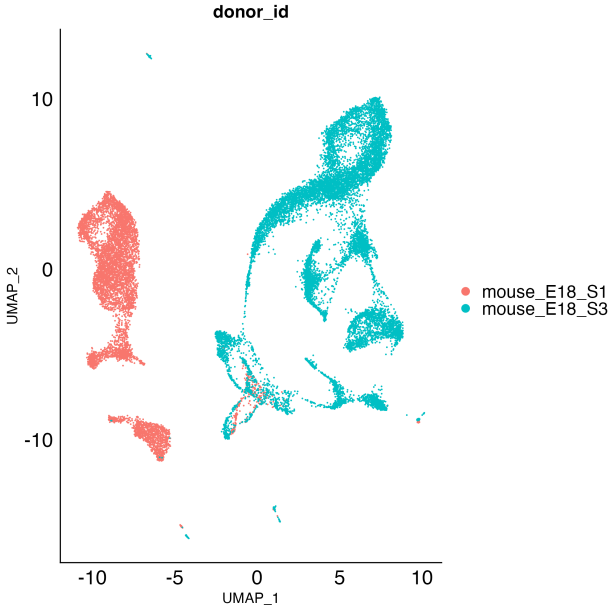
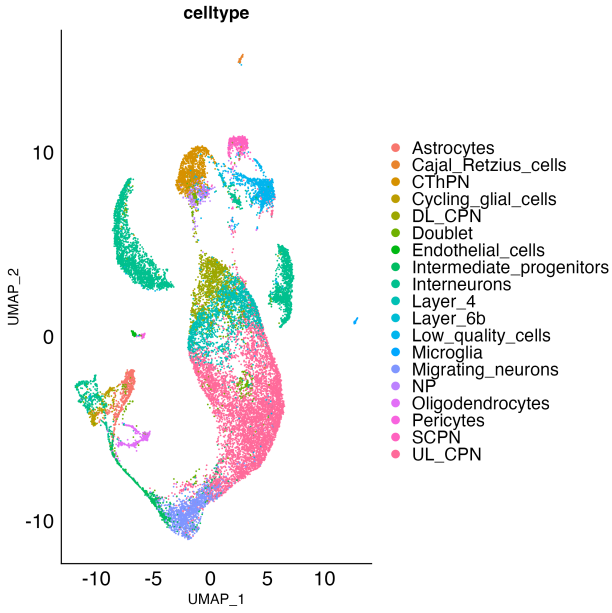


Figure 5.17: UMAP of the brain integrated data after limma method. **a**, cell types. **b**, batches.

(a)



(b)

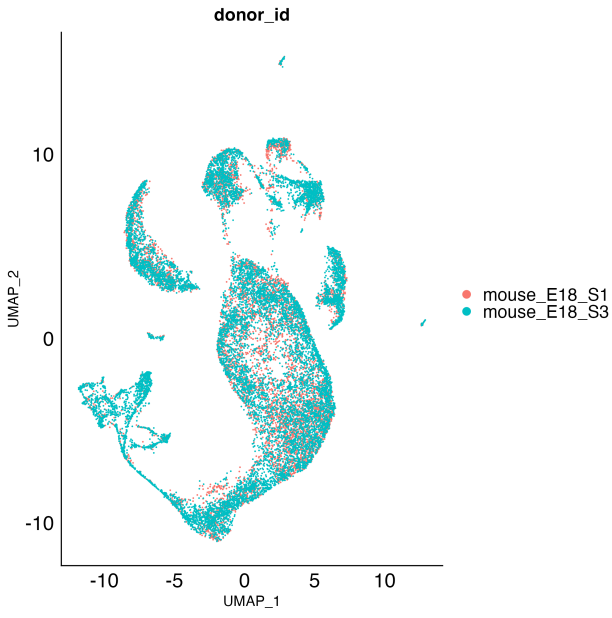
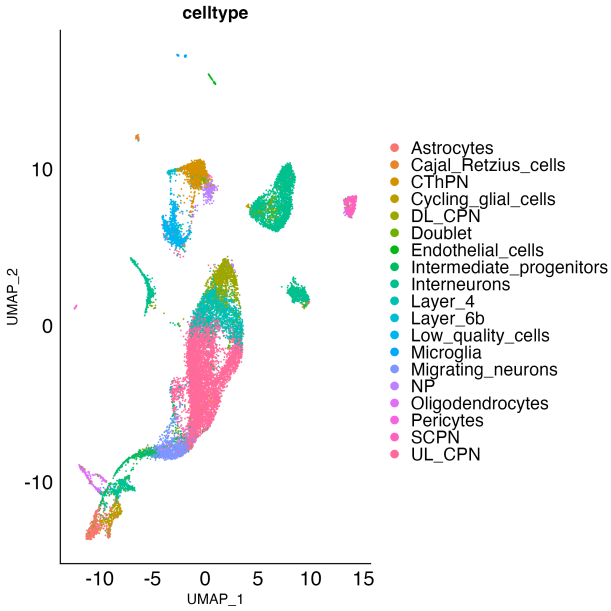


Figure 5.18: UMAP of the brain integrated data after liger method. **a**, cell types. **b**, batches.

(a)



(b)

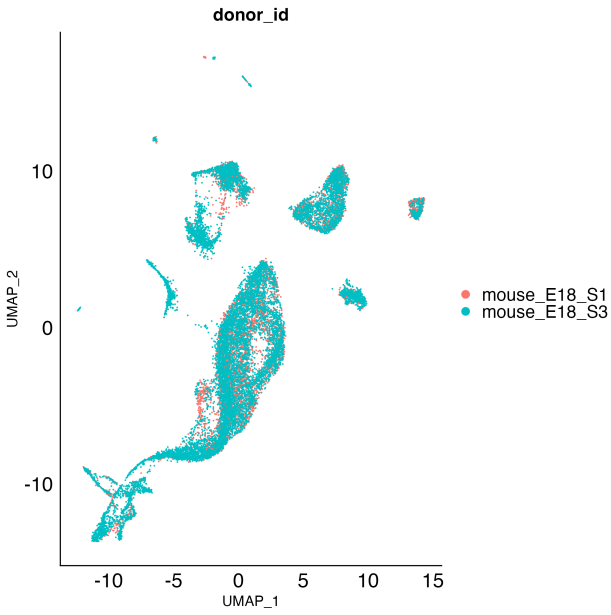


Figure 5.19: UMAP of the brain integrated data after Seurat CCA method. **a**, cell types. **b**, batches.

FastMNN

The FastMNN method has a built-in function in Seurat, and the UMAP visualization of the "mnn" dimension reduction produced by FastMNN is shown in figure 5.20.

Conos

After building a graph and identifying communities using the Conos package, a 2-dimensional visualization of the data is generated. This visualization is named "largeVis" and is shown in figure 5.21.

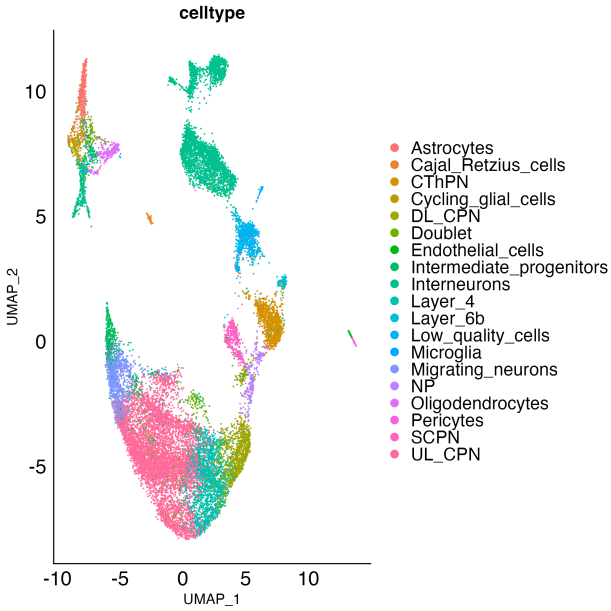
Combat

The Combat method applies its algorithm to the count matrix and produces a new count matrix. After normalizing the data and identifying variable features, a UMAP visualization of the data is generated, as shown in Figure 5.22.

5.2.4 Manual assessment on batch effect removal on mouse brain dataset

To evaluate the performance of batch effect assessment methods, we need to use expert opinion as a reference. In this section, we perform a thorough analysis of UMAP visualization data. The evaluation of batch effect removal methods is listed in order from the best method to the worst.

(a)



(b)

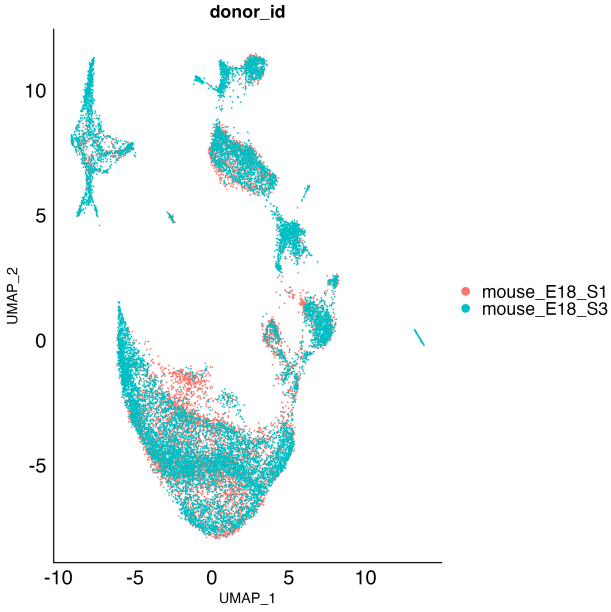
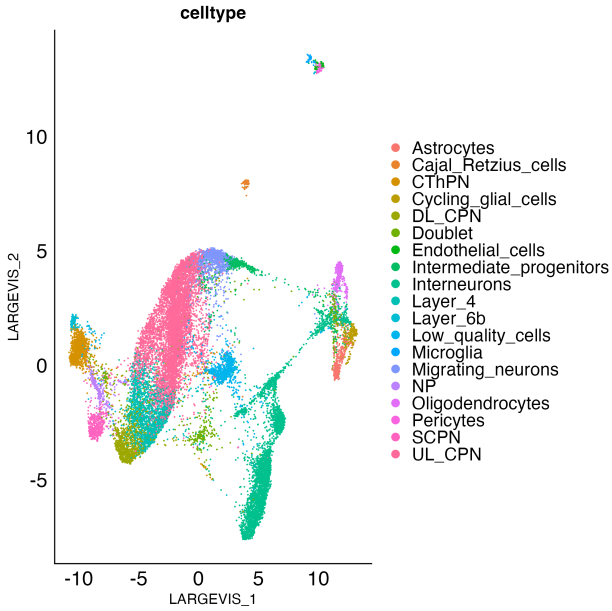


Figure 5.20: UMAP of the brain integrated data after FastMNN method. **a**, cell types. **b**, batches.

(a)



(b)

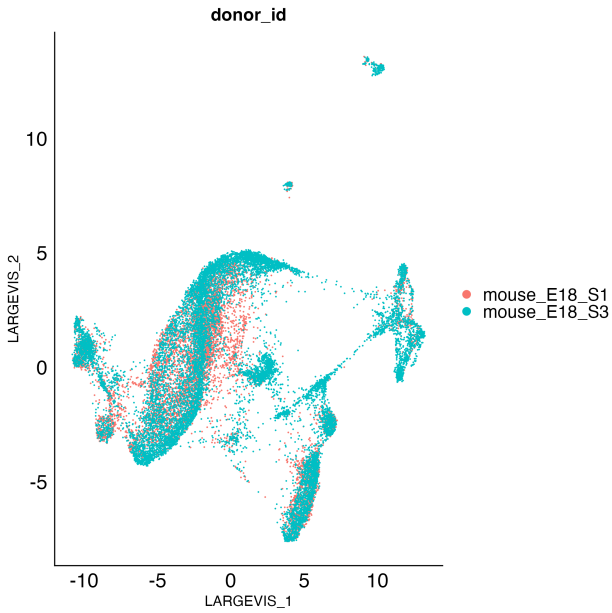
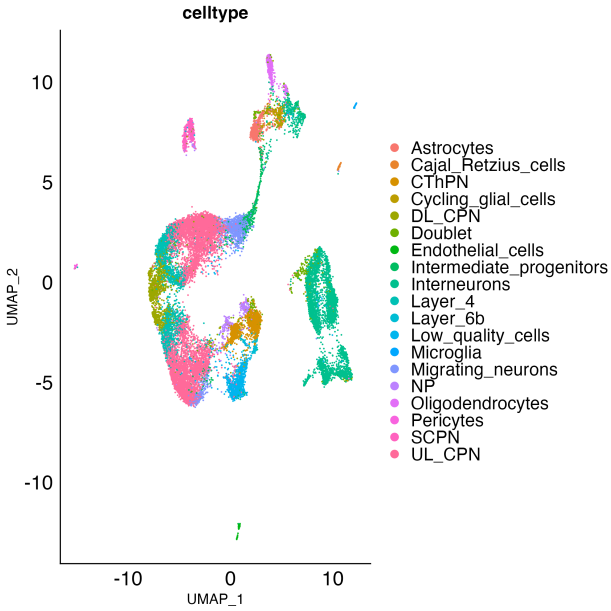


Figure 5.21: largeVis visualization of the brain integrated data after Conos method. **a**, cell types. **b**, batches.

(a)



(b)

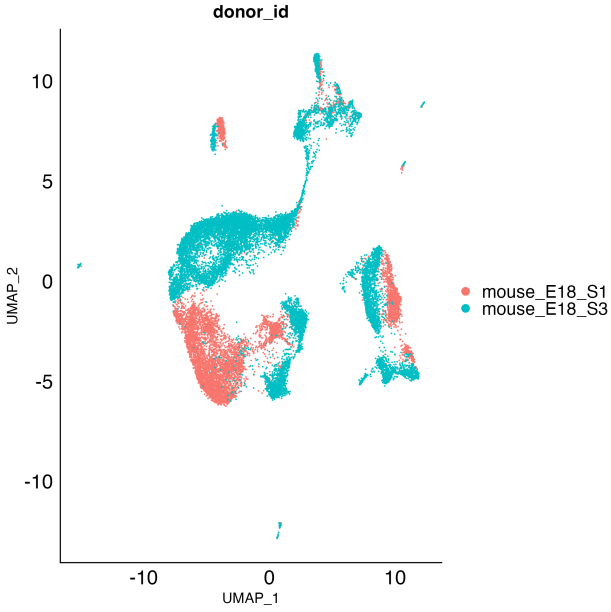


Figure 5.22: UMAP visualization of the brain integrated data after Combat method. **a**, cell types. **b**, batches.

1. Conos: In the largeVis dimension reduction demonstrated in figure 5.21, batches are well mixed. The same cell types are also well mixed except for a few interneuron cells that are not mixed with other cells but they are a little apart from the core of this cell type.

2. CCA: Here in the UMAP shown in figure 5.19, the batches are also well mixed and cell types are apart but some interneuron cells are apart and the difference with the Conos method is that they are far from the core of the cell type.

3. Liger: In the liger method shown in figure 5.18, some interneuron cells are apart and there are several cells that are misplaced.

4. Harmony: As it is visible in figure 5.16, although cell types are apart and separable but batches are not well mixed. As the number of cells in the first dataset is almost half of the second dataset, the second batch cells should circumvent the first batch cells.

5. Combat: In this method, batches are closer than raw data but it couldn't mix them well enough.

6. Raw data: This is the original positioning of cells in the dataset after mixing batches together shown in figure 5.15. Two batches are completely separated but in each batch cell types are recognizable.

7. Limma: The results are almost like raw data visualized in figure 5.17. Batches are not mixed and cell types are separable, but interneurons are more spread than the raw data.

8. FastMNN: The UMAP of the fastMNN method is shown in figure 5.20. Although the batches seem well mixed, there is a critical problem that makes it even worst than the raw dataset. Some cell types that were more separable are now not as good as before. Layer 4 dataset is more merged into the UL_CPN and from the other side more engaged with DL_CPN. This happened for migrating_neurons and UL_CPN. Misplacement of cells has also happened in multiple parts. This is overcorrecting the batch effect that mixes cell types that should not be mixed.

5.2.5 Batch assessment methods results

Five batch assessment methods have been used to evaluate the performance of different batch effect removal methods. These methods are Batch_Finder, ARI, kBET, Silhouette, and the Mixing metric. The results of the batch effect removal methods are shown in Figure 5.23. The acceptance values for each method are displayed, with higher values indicating better performance in removing batch effects. The heatmap is coloured based on the values in each column, rather than the overall table. To facilitate comparison and improve the interpretability of the results, the ranks of the batch effect removal methods are calculated

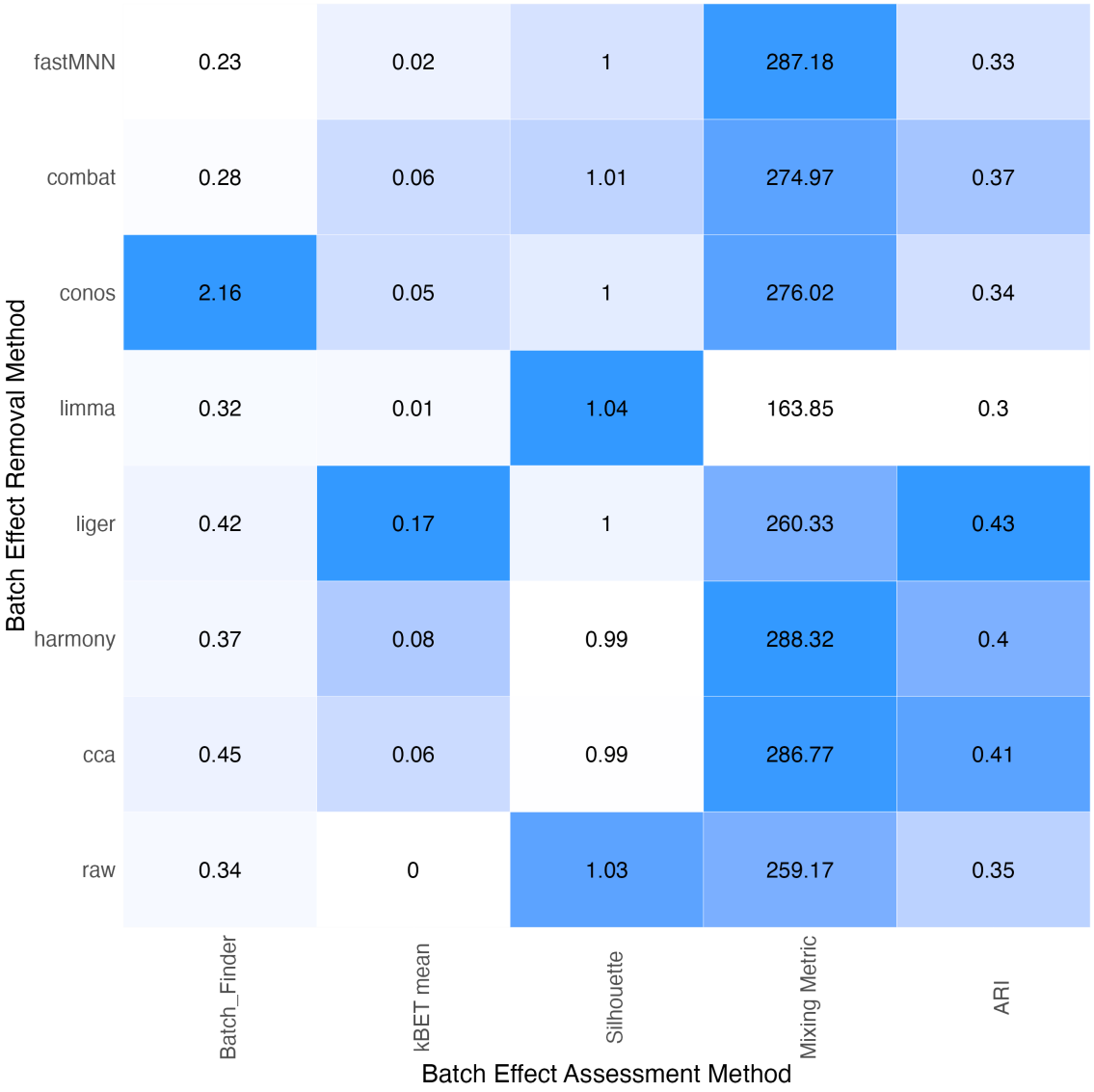


Figure 5.23: Batch effect assessment values on brain dataset. Each column of this figure is for one batch effect assessment method. Heatmap colours are based on each column's values.

for each assessment method. This allows for a better understanding of the performance of the different methods, despite the fact that the raw values are in different ranges. Figure 5.24 shows the rank table for the pancreas dataset. Also, the expert's opinion is also added to the table as "Manual" assessment. To better understand the correlation results, a boxplot is provided in Figure 5.25. The ranks of each batch effect removal method are summarized in the boxplot, and the colour of each point indicates which method assigned the rank to the removal method. The goal is to find out which assessment method is closer to the expert's opinion. Finding the correlation between rank table values can reveal the similarity between all assessment methods. Figure 5.26 demonstrates the correlation between all assessment methods. A higher value of correlation shows the similarity of assessment methods.

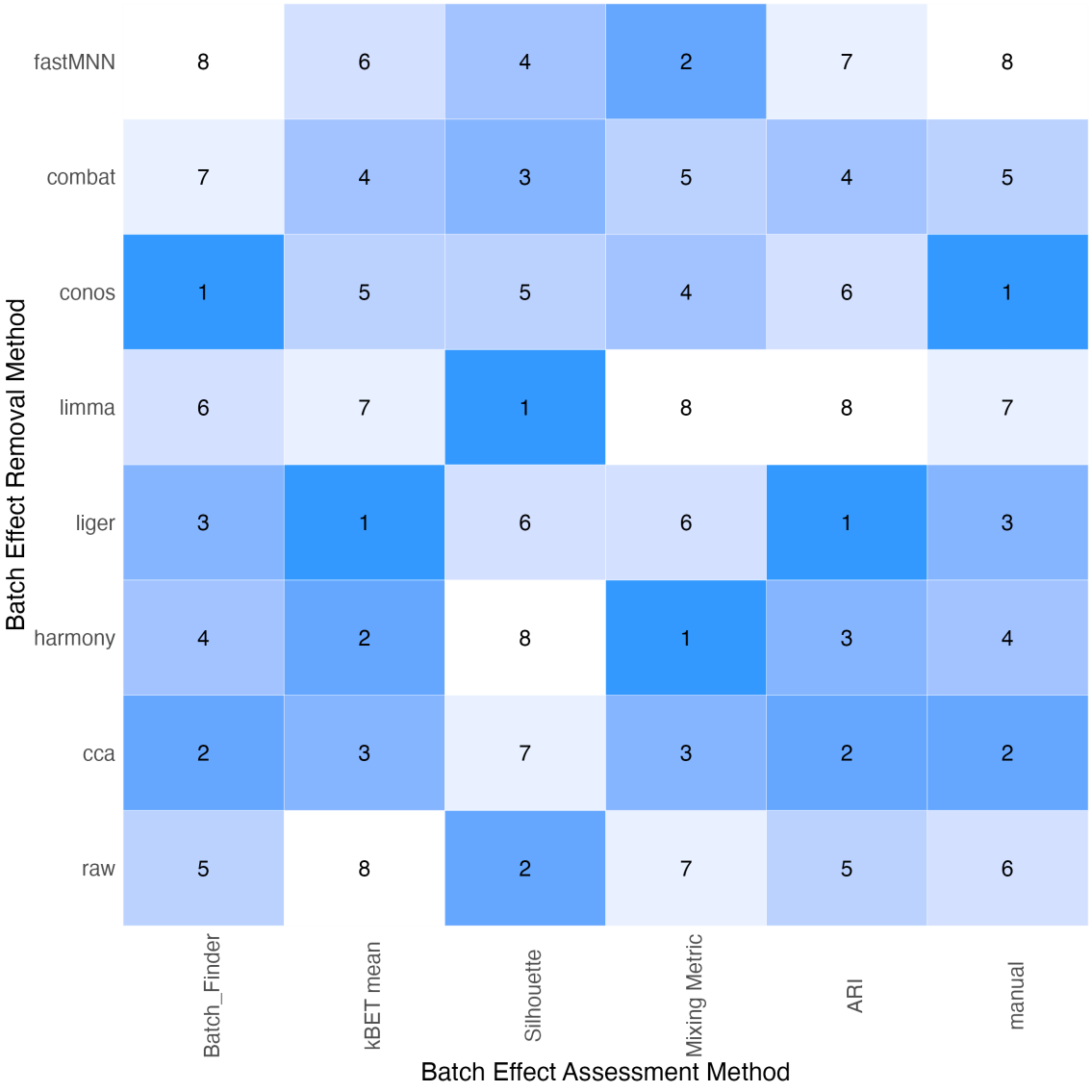


Figure 5.24: Batch effect assessment ranks on brain dataset. Each column of this figure is for one batch effect assessment method. The manual column is according to the expert's opinion.

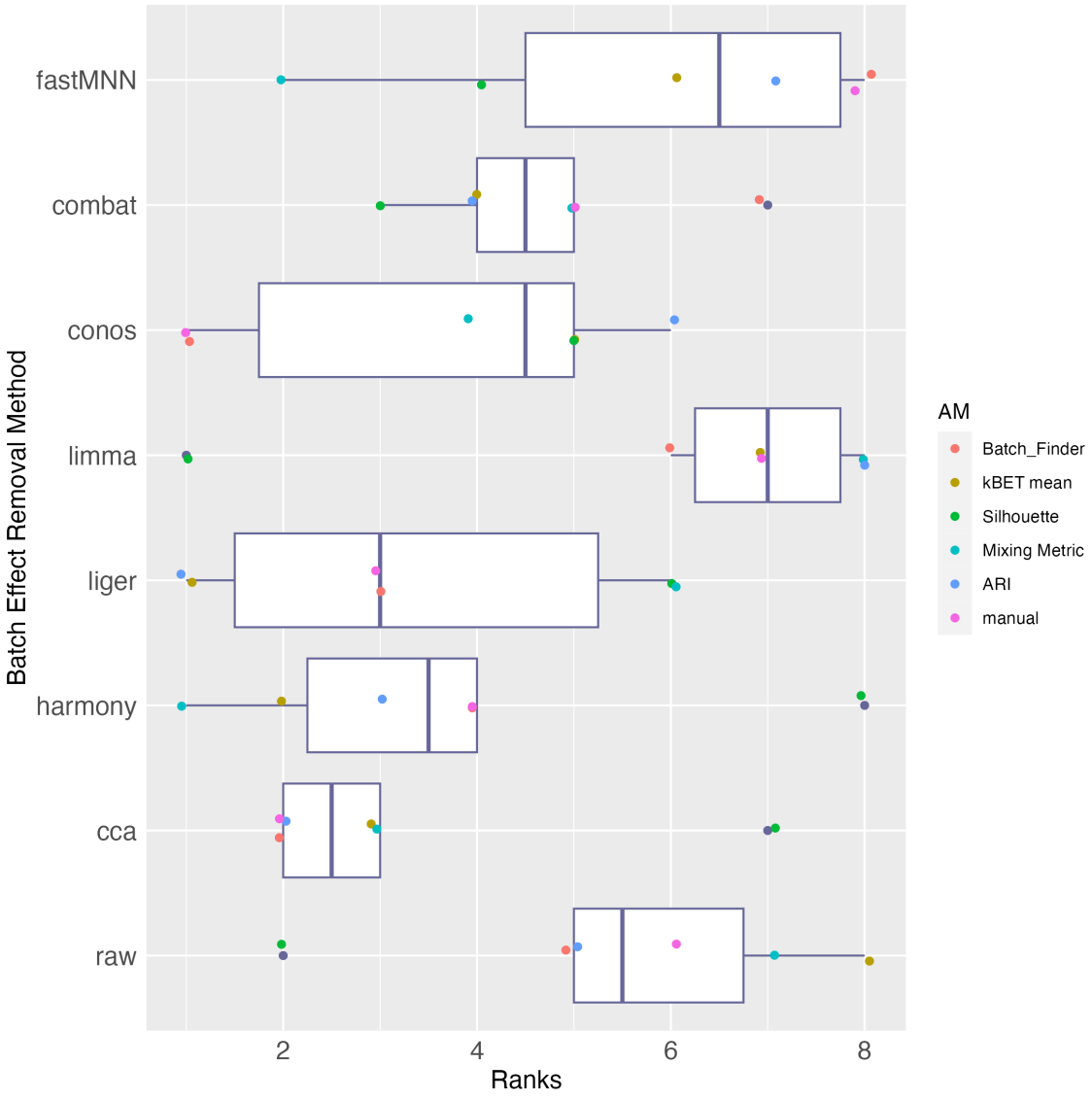


Figure 5.25: Batch effect assessment ranks boxplot on brain dataset. Each box shows the ranks given to each batch effect removal method.

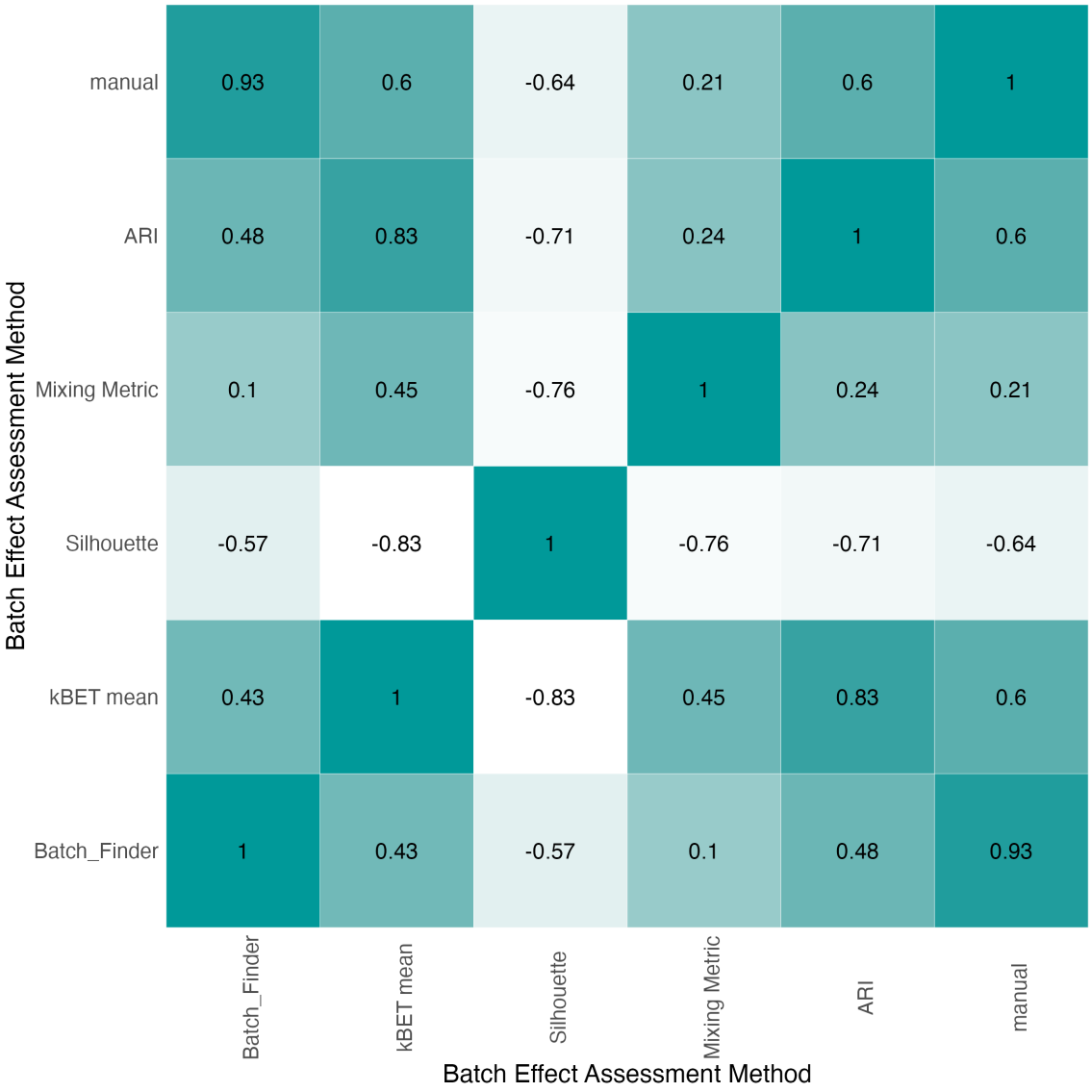


Figure 5.26: Batch effect assessment ranks correlation on brain dataset.

5.3 dataset3: PBMC

The Seurat PBMC dataset is a publicly available dataset of peripheral blood mononuclear cells (PBMCs) that has been widely used in the field of single-cell genomics. The data includes gene expression levels, cell type annotations, and other metadata for each cell[61]. In this dataset, 9 different methods have been gathered together during two experiments. These datasets are as below:

- Sample 1: 526 cells obtained from Smart-seq2 method.
- Sample 2: 526 cells obtained from CEL-Seq2 method.
- Sample 3: 3222 cells obtained from 10x Chromium (v2) A method.
- Sample 4: 3222 cells obtained from 10x Chromium (v2) B method.
- Sample 5: 3222 cells obtained from 10x Chromium (v3) method.
- Sample 6: 6584 cells obtained from Drop-seq method.
- Sample 7: 3773 cells obtained from Seq-Well method.
- Sample 8: 6584 cells obtained from inDrops method.
- Sample 9: 3362 cells obtained from 10x Chromium (v2) method.

Integration of these samples created a dataset with 31021 cells and 33694 genes and 10 unique cell types. The origin of the batch effect is considered the experimental method

that each dataset is generated with. This data has been gathered with two experiments, the first experiment has 19838 cells and the second experiment has 11183 cells.

5.3.1 Pre-processing

This dataset has been already processed through Seurat standard workflow and all samples are mixed in a single object.

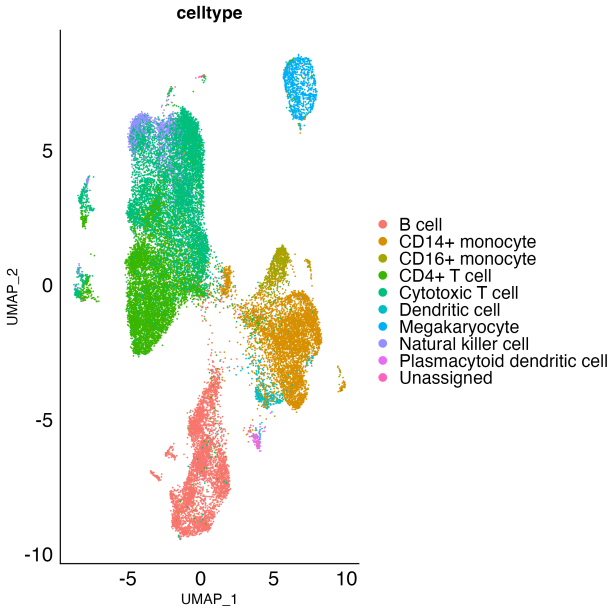
5.3.2 Dataset Visualization

The UMAP visualization of the dataset is in figure 5.27. batches are completely apart while each batch cell type pattern is analogous, indicating batch effect is present within the integrated dataset.

5.3.3 Performing Batch Effect Removal Methods

As batch effect exists in the integrated data between two experiments, 7 batch effect removal methods, mentioned in 2.4, were applied to the integrated dataset. These methods are Harmony, Liger, Limma, Seurat CCA, FastMNN, Conos, and Combat. Although these methods use different techniques, finally all of them provide either a new count matrix or a new dimension reduction for the dataset as the result of their attempt to correct the batch effect.

(a)



(b)

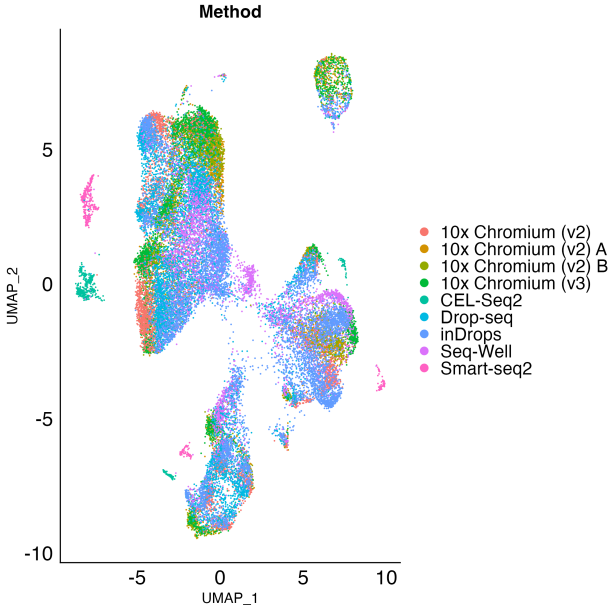


Figure 5.27: UMAP of the PBMC integrated data before batch effect removal. **a**, cell types. **b**, batches.

Harmony

To process the integrated dataset, the standard workflow was followed, which included normalization, identifying variable features, scaling the data, and running PCA. After these steps, the RunHarmony function in the Seurat object was used to generate harmony dimension reduction data. Figure 5.28 demonstrates the UMAP dimension reductions of the harmony batch effect removal method.

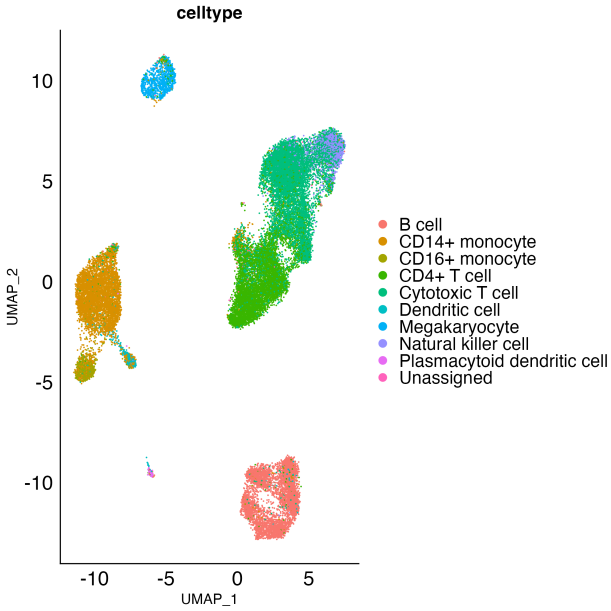
Limma

The Limma method uses the count matrix and batch factors to create a new count matrix. By generating a new Seurat object with this new count matrix and the original metadata from the dataset, the standard preparation workflow would be applied to the corrected dataset. Figure 5.29 shows UMAP visualization on the dataset corrected with the limma batch effect removal method.

Liger

To implement the Liger method, the standard workflow must first be applied to the dataset. Then, the RunOptimizeALS and RunQuantileNorm functions from the SeuratWrappers package can be used to generate a new dimension reduction called ”iNMF”. The UMAP of this batch effect correction method is provided in fig 5.30.

(a)



(b)

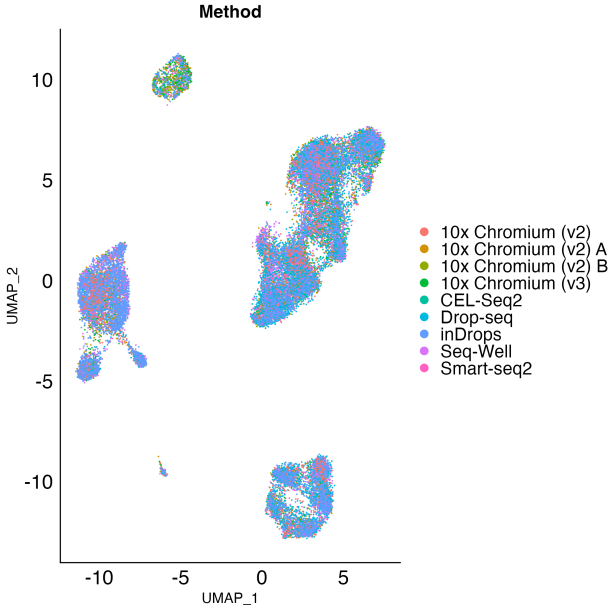
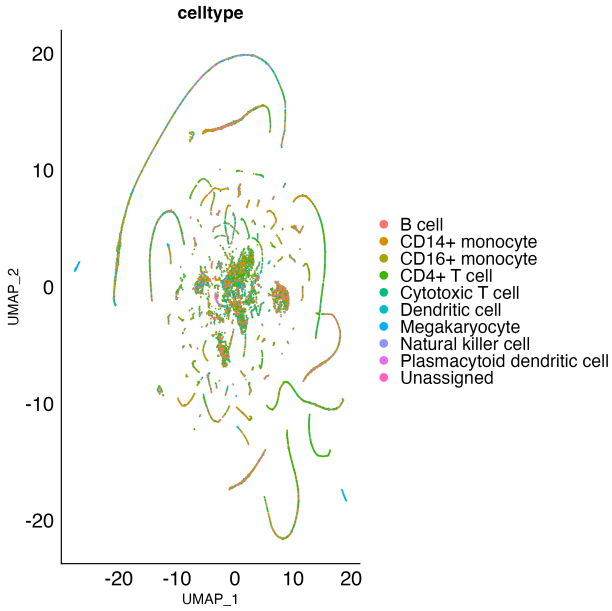


Figure 5.28: UMAP of the PBMC integrated data after harmony method. **a**, cell types. **b**, batches.

(a)



(b)

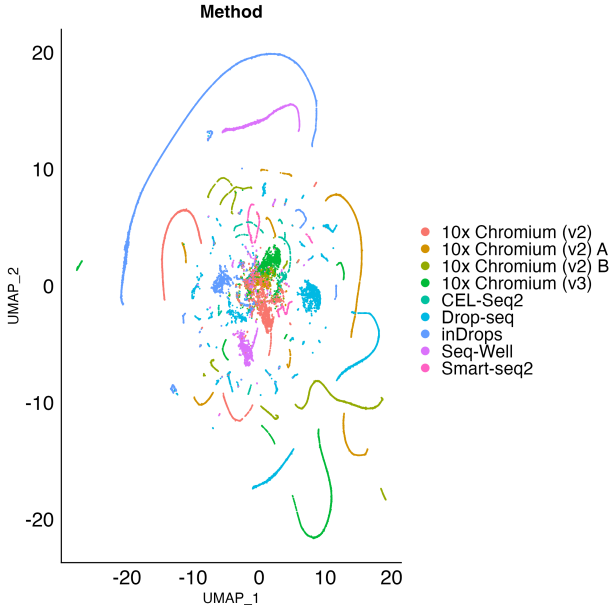
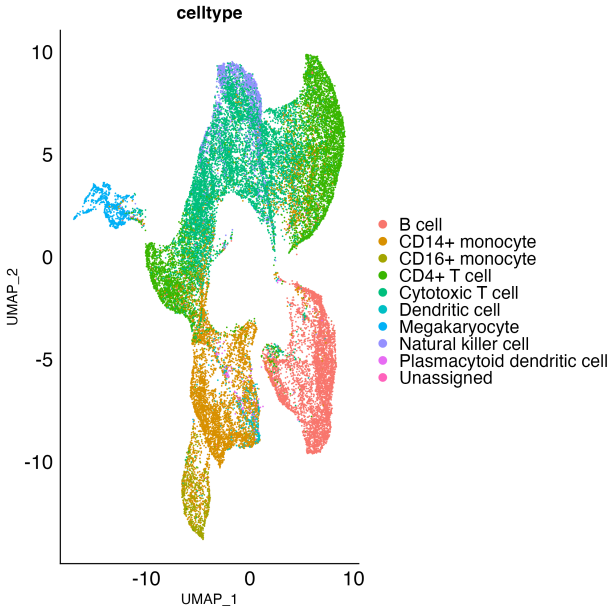


Figure 5.29: UMAP of the PBMC integrated data after limma method. **a**, cell types. **b**, batches.

(a)



(b)

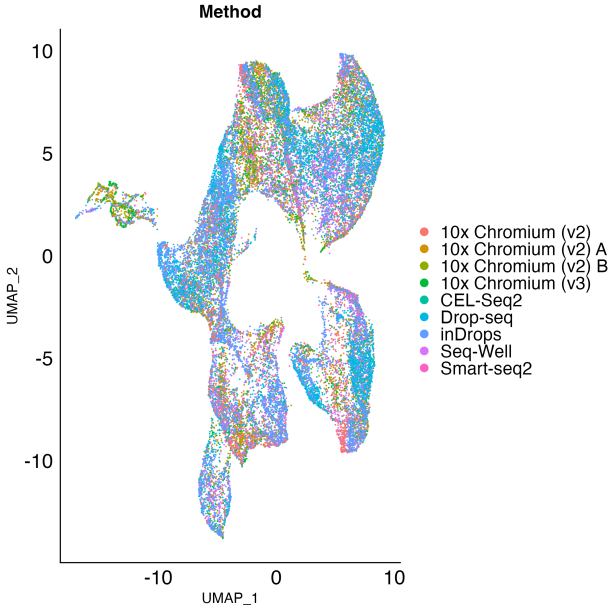


Figure 5.30: UMAP of the PBMC integrated data after liger method. **a**, cell types. **b**, batches.

Seurat CCA

After implementing Seurat CCA on the PBMC dataset by finding the integration anchors, the new integrated assay has been visualized by UMAP in figure 5.31.

FastMNN

The FastMNN method has a built-in function in Seurat, and the UMAP visualization of the ”mnn” dimension reduction produced by FastMNN is shown in figure 5.32.

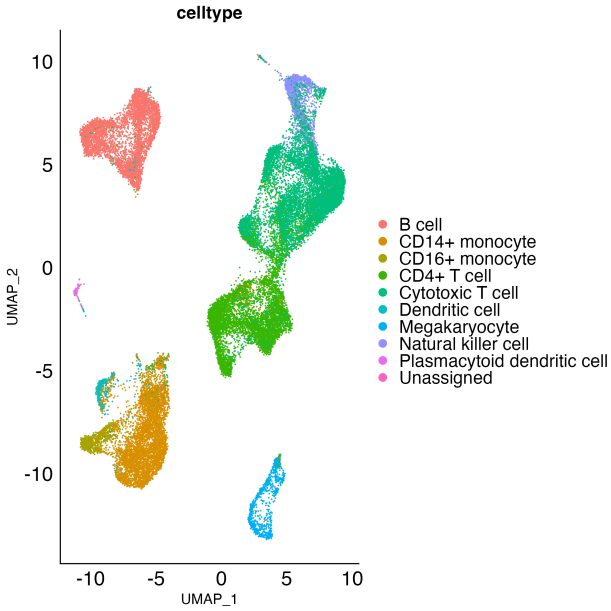
Conos

After building a graph and identifying communities using the Conos package, a 2-dimensional visualization of the data is generated. This visualization is named ”largeVis” and is shown in figure 5.33.

Combat

The Combat method applies its algorithm to the count matrix and produces a new count matrix. After normalizing the data and identifying variable features, a UMAP visualization of the data is generated, as shown in Figure 5.34.

(a)



(b)

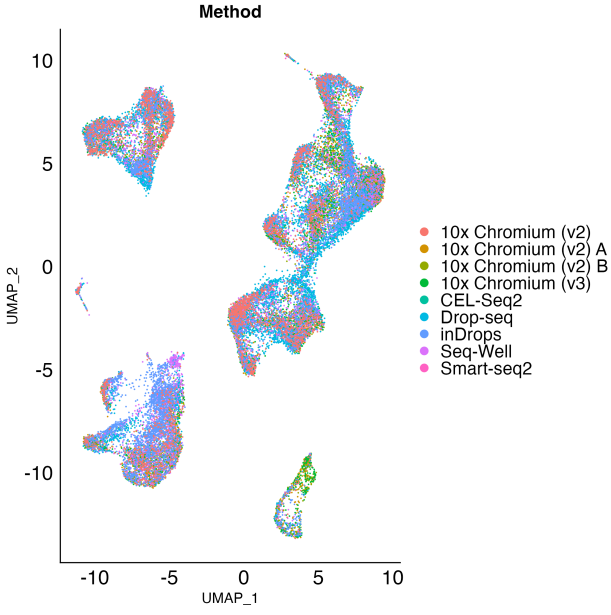
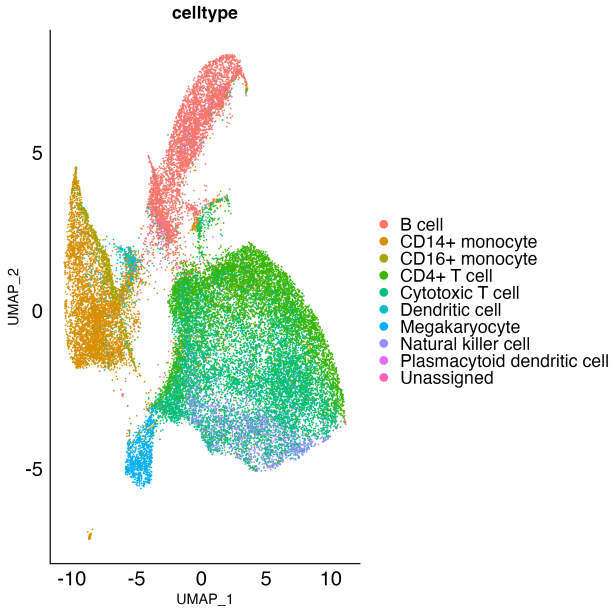


Figure 5.31: UMAP of the PBMC integrated data after Seurat CCA method. **a**, cell types. **b**, batches.

(a)



(b)

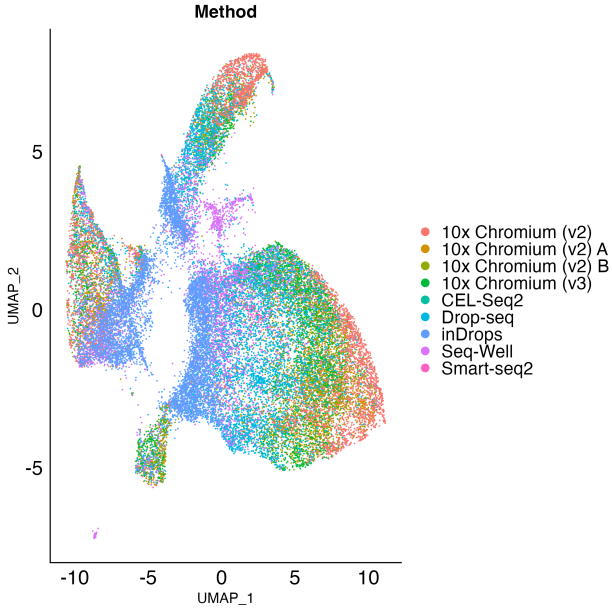
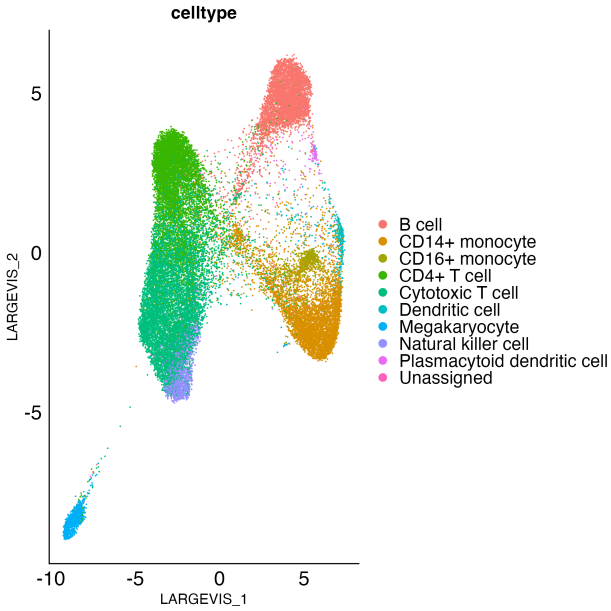


Figure 5.32: UMAP of the PBMC integrated data after FastMNN method. **a**, cell types. **b**, batches.

(a)



(b)

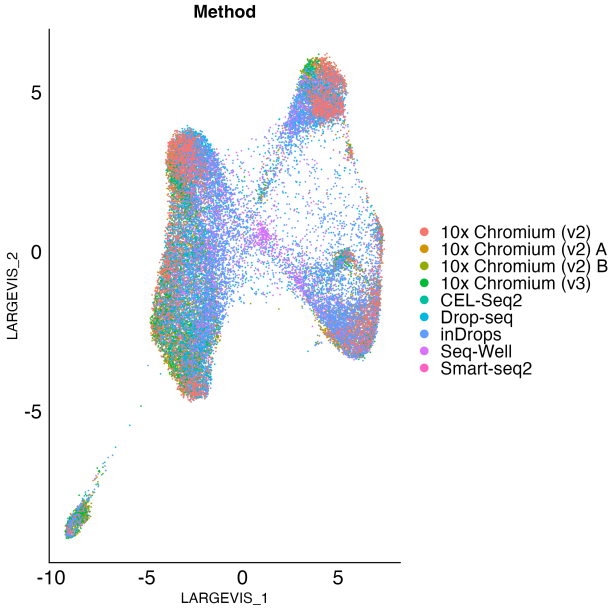
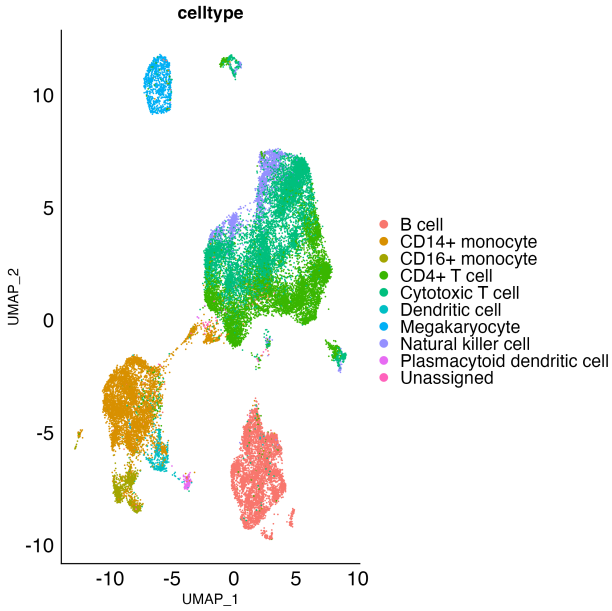


Figure 5.33: largeVis visualization of the PBMC integrated data after Conos method. **a**, cell types. **b**, batches.

(a)



(b)

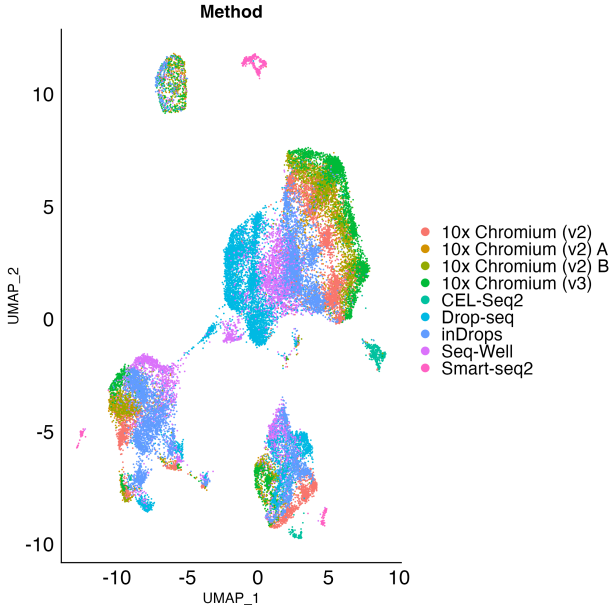


Figure 5.34: UMAP visualization of the PBMC integrated data after Combat method. **a**, cell types. **b**, batches.

5.3.4 Manual assessment on batch effect removal on PBMC dataset

To evaluate the performance of batch effect assessment methods, we use expert opinion as a reference. In this section, we conduct a thorough analysis of UMAP visualization data. The evaluation of batch effect removal methods is listed in order from the best method to the worst based on this analysis. According to the dataset size, providing an expert opinion was harder and less accurate.

1. CCA: In this method shown in figure 5.31, cell types are separated and batches are well mixed.

2. Harmony: Here the batches are also well mixed and cell types are apart but CD4+ T cells are more mixed with Cytotoxic T cells as shown in figure 5.28.

3. Conos: In the visualization of this method provided in figure 5.33, although batches are mixed better, some cells are scattered in the middle of the visualization.

4. Liger: The batches are well mixed but CD4+ T cells are not all together demonstrated in figure 5.30.

5. Raw data: This is the original positioning of cells in the dataset shown in figure 5.27. Batches are not well mixed, but cell types are almost apart. Some cell types are overlapping.

6. Combat: In this method batches are more separated than raw data while the cell types still overlap visualized in figure 5.34.

7. FastMNN: The results are almost like the Combat method shown in figure 5.32. batches are separated, but cell types are unnecessarily closer.

8. Limma: With the parameters used in the experiment, the Limma method failed to perform a good batch effect removal on the dataset. The reason that the UMAP shown in figure 5.29 looks like lines rather than the more typical, continuous spread of points that is expected, is the Limma result is not well-suited for UMAP and doesn't have a clear, continuous structure, such as a smooth, low-dimensional manifold.

5.3.5 Batch assessment methods results

Five batch assessment methods were applied to evaluate the effectiveness of various batch effect removal techniques: Batch_Finder, ARI, kBET, Silhouette, and the Mixing metric. The results of these techniques are illustrated in Figure 5.35, where higher values indicate better performance in removing batch effects. The heatmap is colour-coded based on the values in each column, rather than the overall table. To facilitate comparison and improve the interpretability of the results, the ranks of the batch effect removal methods are calculated for each assessment method. This allows for a better understanding of the per-

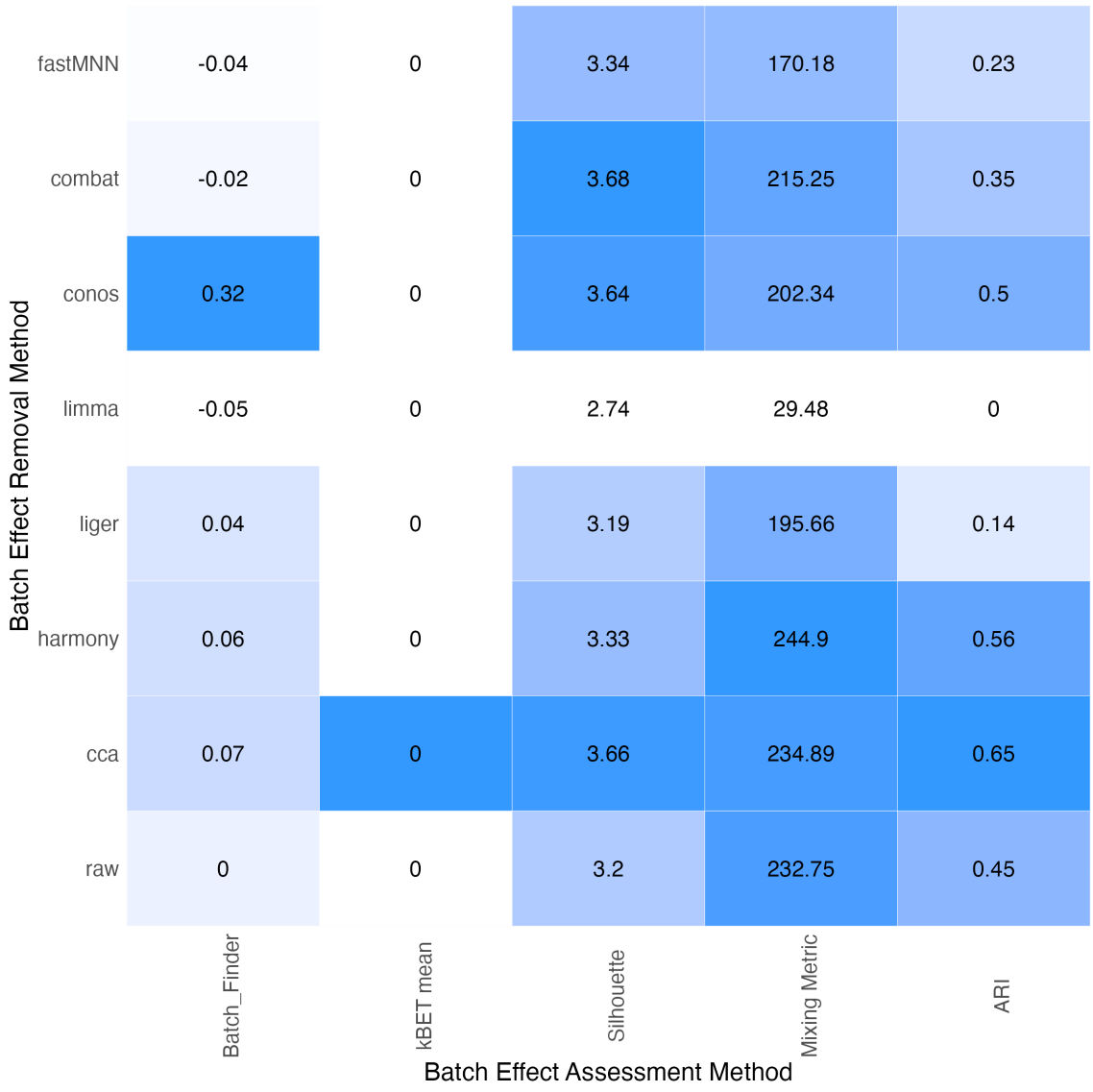


Figure 5.35: Batch effect assessment values on pbmc dataset. Each column of this figure is for one batch effect assessment method. Heatmap colours are based on each column's values.

formance of the different methods, even though the raw values are in different ranges. The rank table for the pancreas dataset is shown in Figure 5.36, and includes the expert's opinion as the "Manual" assessment. To better understand the correlation results, a boxplot is provided in Figure 5.37. The ranks of each batch effect removal method are summarized in the boxplot, and the colour of each point indicates which method assigned the rank to the removal method. The goal is to find out which assessment method is closer to the expert's opinion. Finding the correlation between rank table values can reveal the similarity between all assessment methods. Figure 5.38 demonstrates the correlation between all assessment methods. A higher value of correlation shows the similarity of assessment methods.

5.4 Time and Memory Consumption analysis

The time taken for running each assessment method on each batch effect removal method of the datasets has been measured and recorded. Also, the memory used by each method is also recorded. To measure memory, the garbage collector has been used and the summation of all memory used by each method is calculated and it's not the maximum memory that the system should have. It's a measure that shows how efficiently a method uses system memory. Also to effectively process the large datasets obtained from integrating multiple sources, it was necessary to utilize resources beyond the capacity of personal

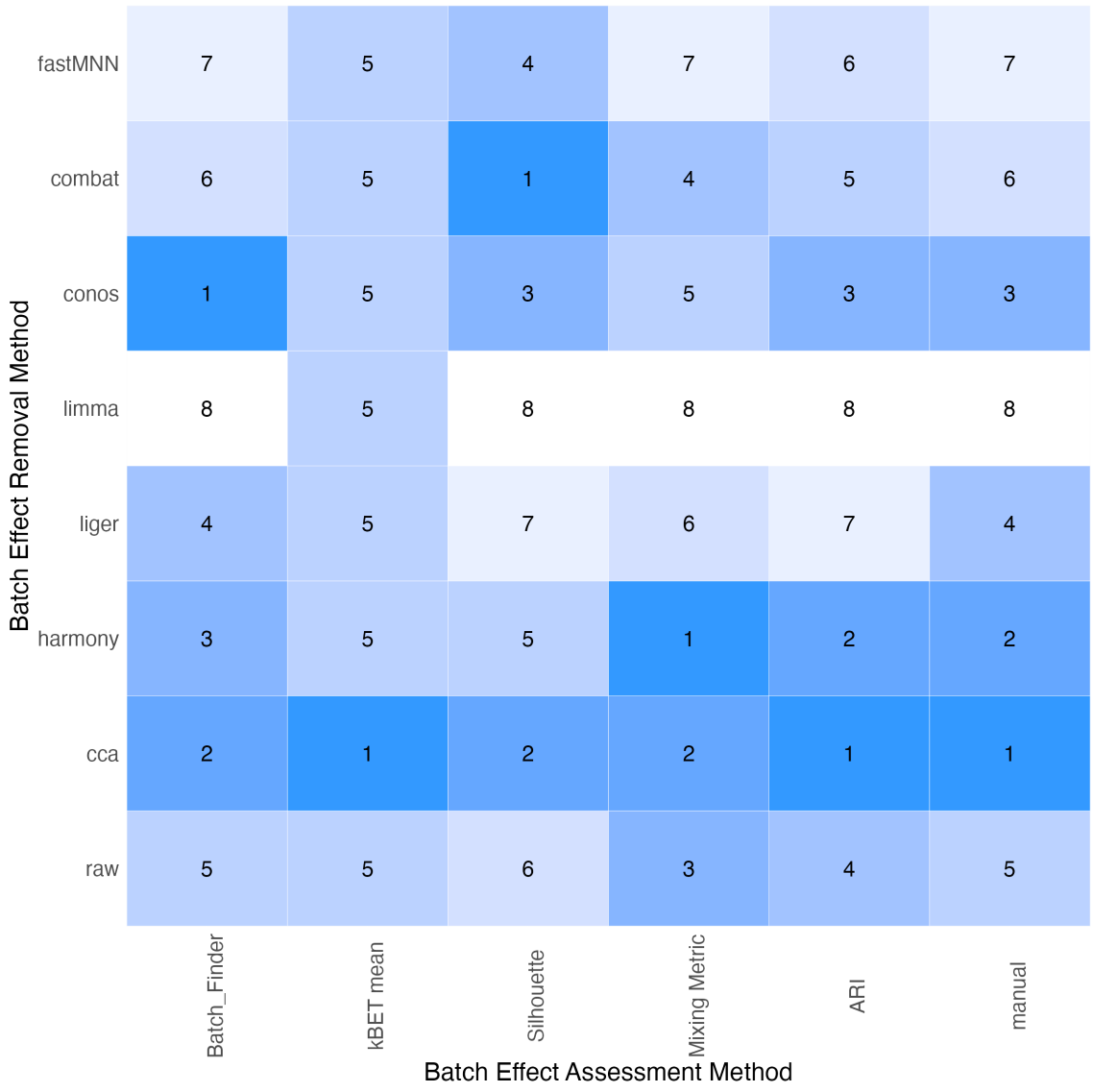


Figure 5.36: Batch effect assessment ranks on PBMC dataset. Each column of this figure is for one batch effect assessment method. The manual column is according to the expert's opinion.

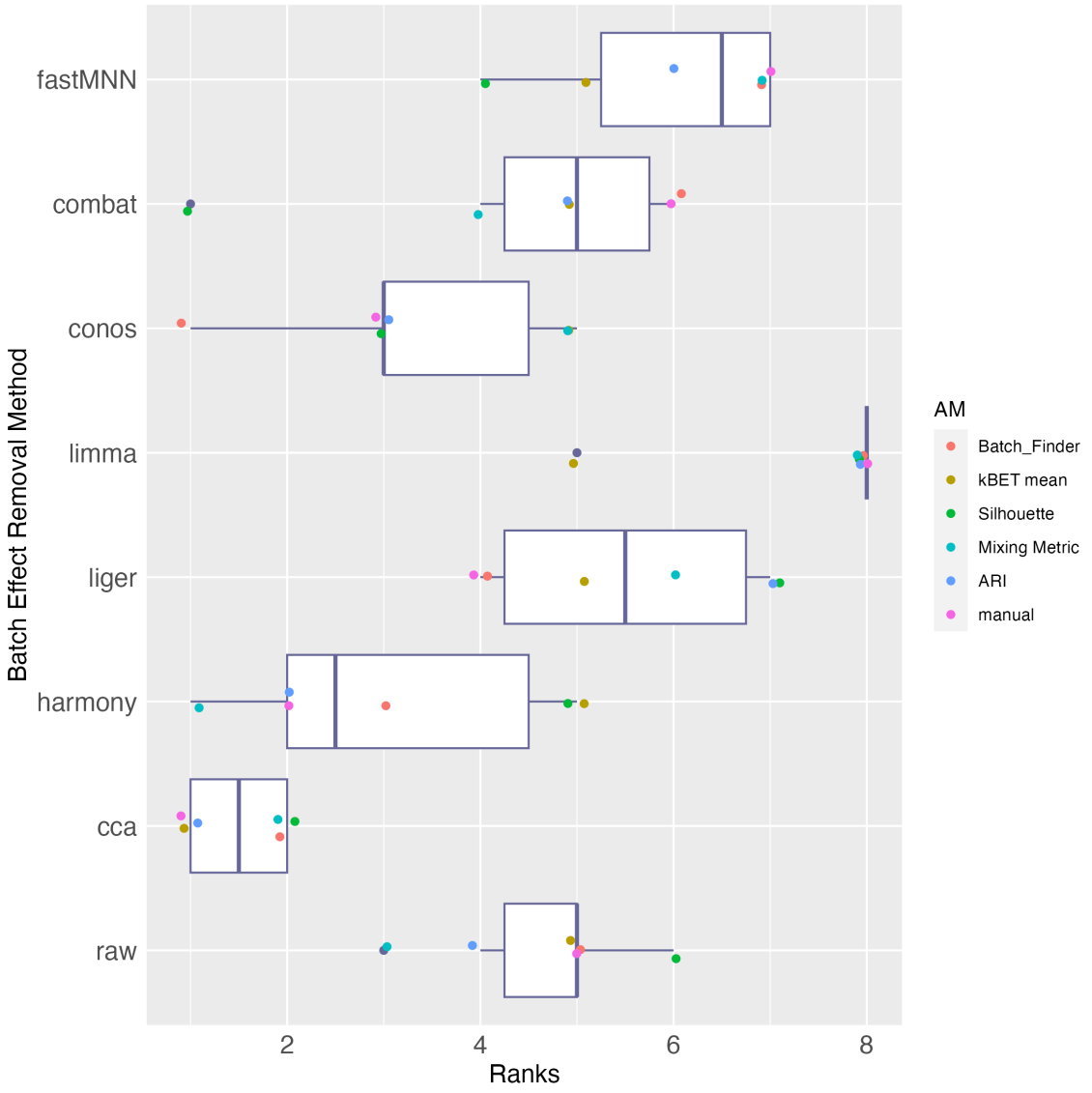


Figure 5.37: Batch effect assessment ranks boxplot on PBMC dataset. Each box shows the ranks given to each batch effect removal method.

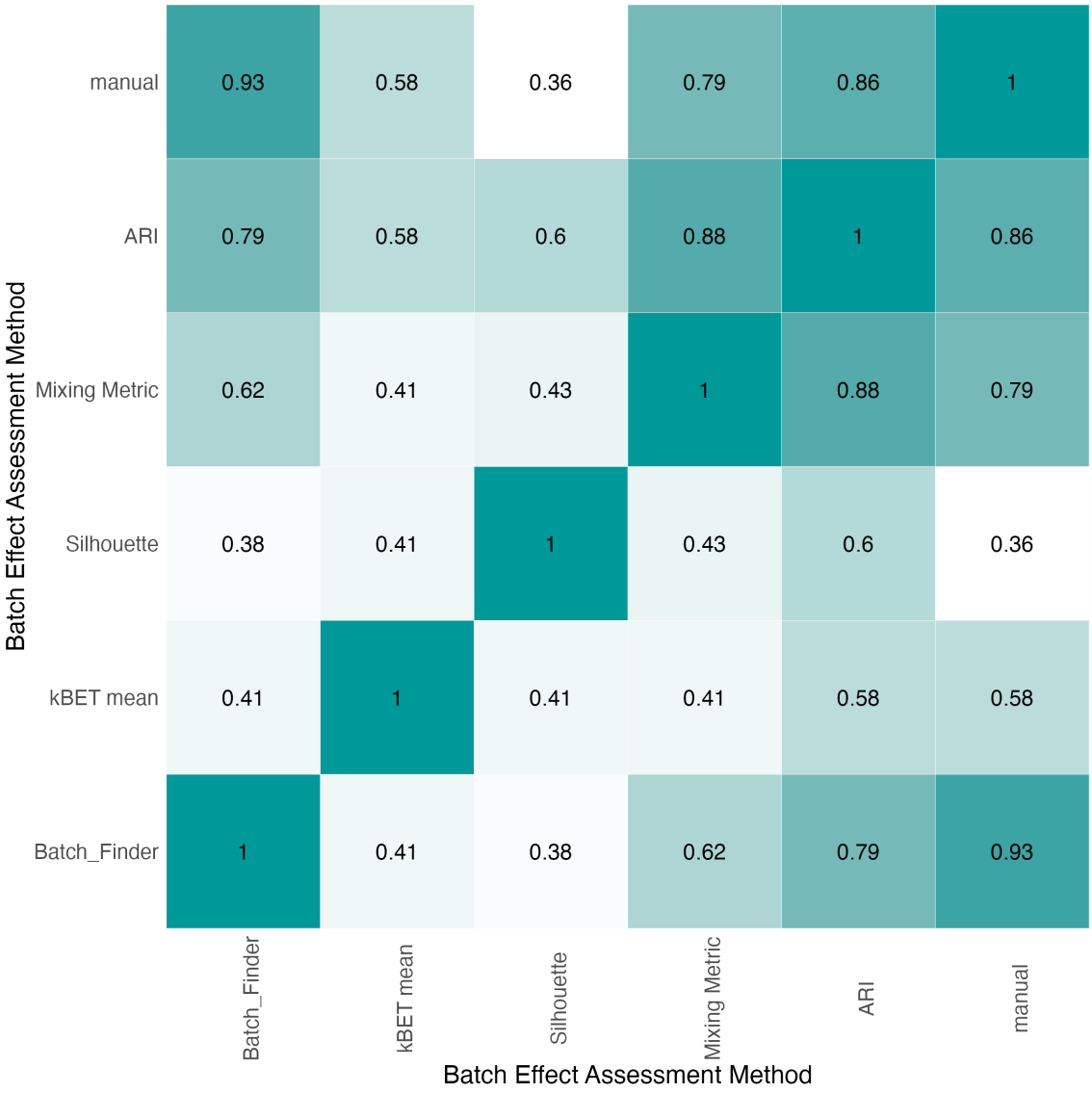


Figure 5.38: Batch effect assessment ranks correlation on PBMC dataset.

computers and laptops. As such, all computer programs for this research were run using Compute Canada resources and all these results are based on single CPU usage.

5.4.1 Time measurements

Figures 5.39, 5.40, and 5.41 Show the duration of each assessment method on a logarithmic scale.

5.4.2 Memory Consumption measurement

Figures 5.42, 5.43, and 5.44 Show the memory consumption of each assessment method in megabytes on a logarithmic scale.

5.5 Discussion

5.5.1 Datasets

Datasets used in this chapter are chosen in order to represent different sources of batch effect and propose different challenges to the study. Table 5.1 shows a summary of these datasets. The pancreas dataset is used to demonstrate the technical batch effect in data integration. The size of batches is not big so the expert analysis is more accurate. Also, there are some cell types that are only available in one dataset to propose more challenges

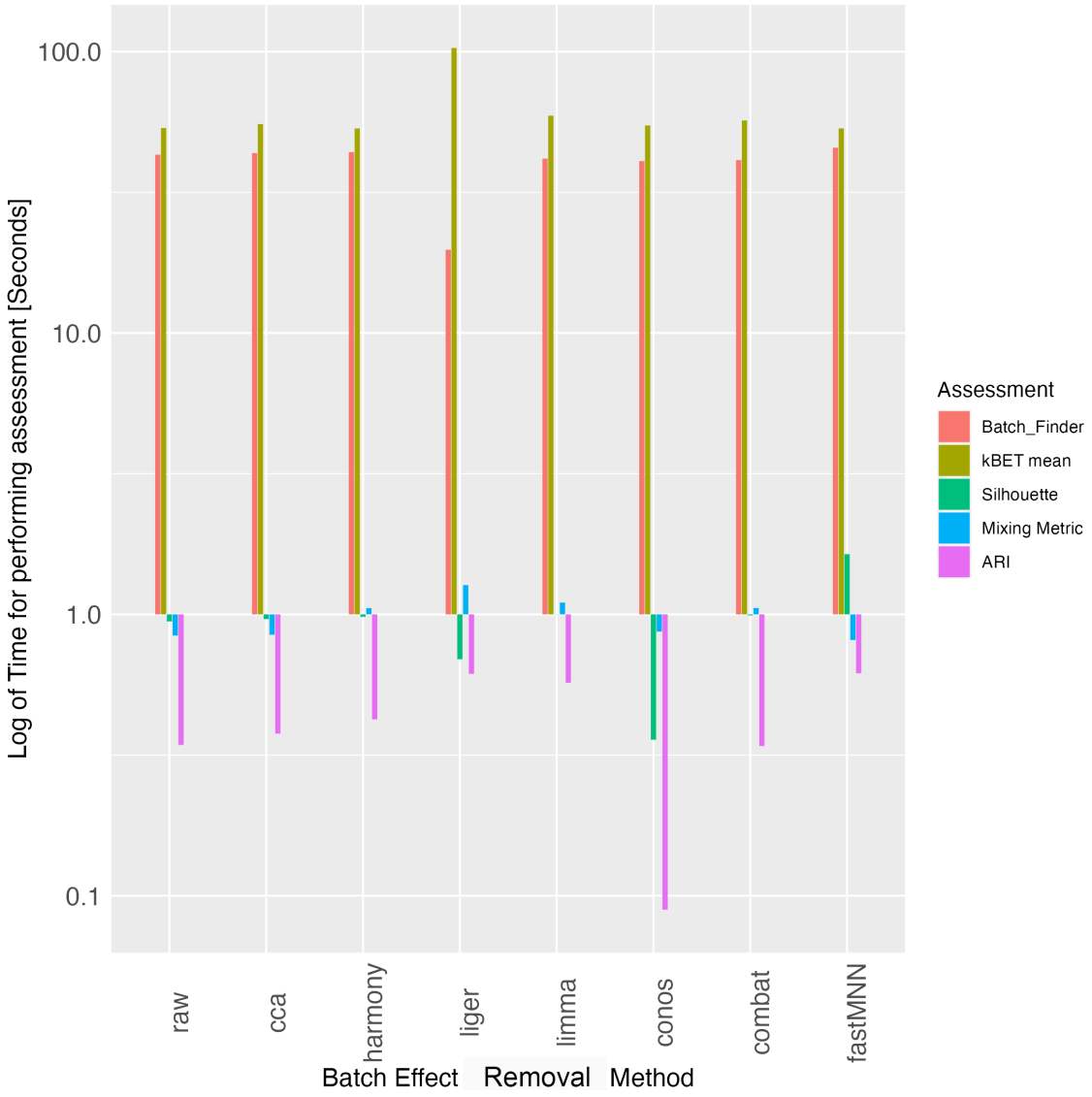


Figure 5.39: The time taken for running batch effect assessment methods on Pancreas dataset.

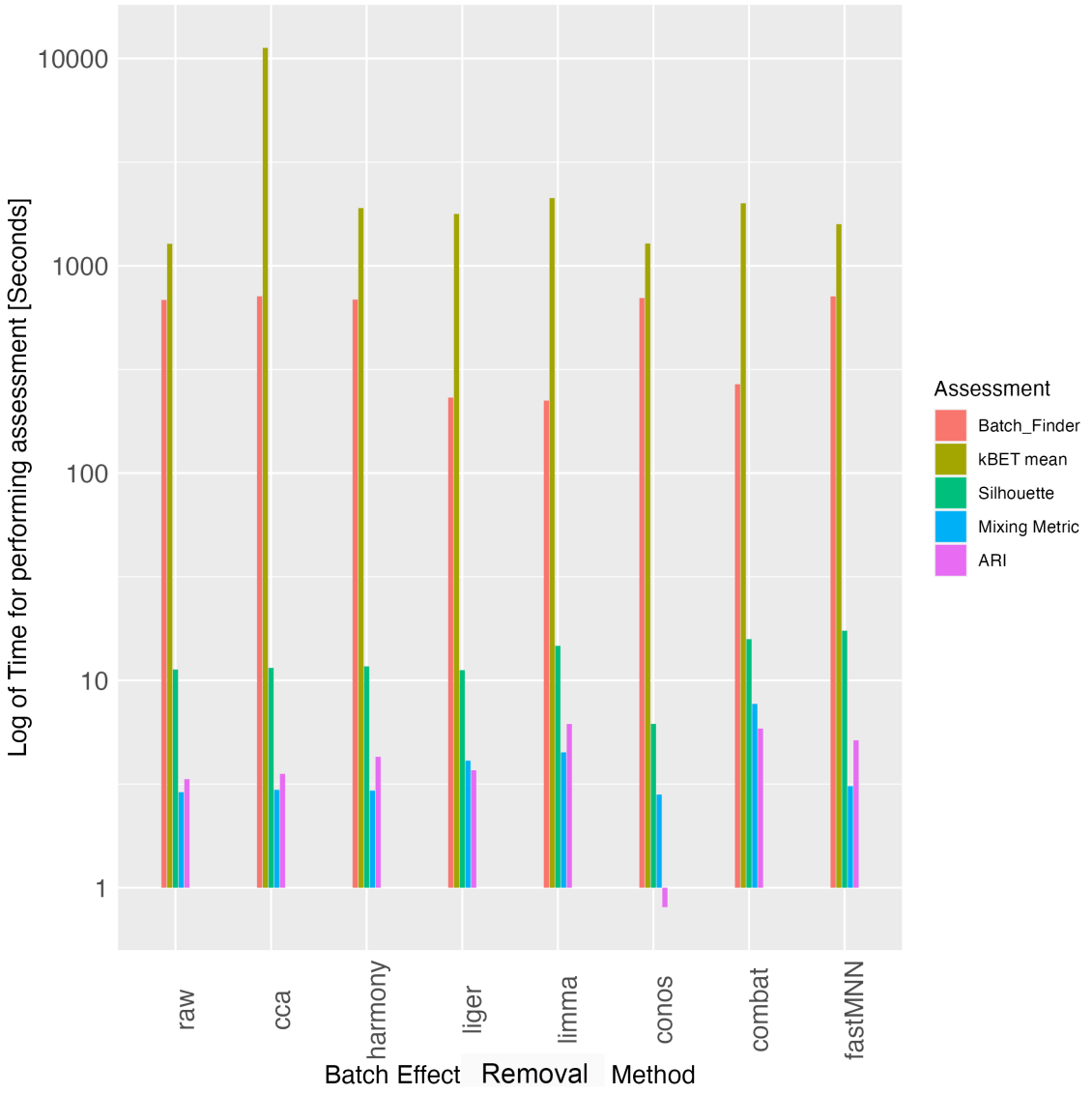


Figure 5.40: The time taken for running batch effect assessment methods on Brain dataset.

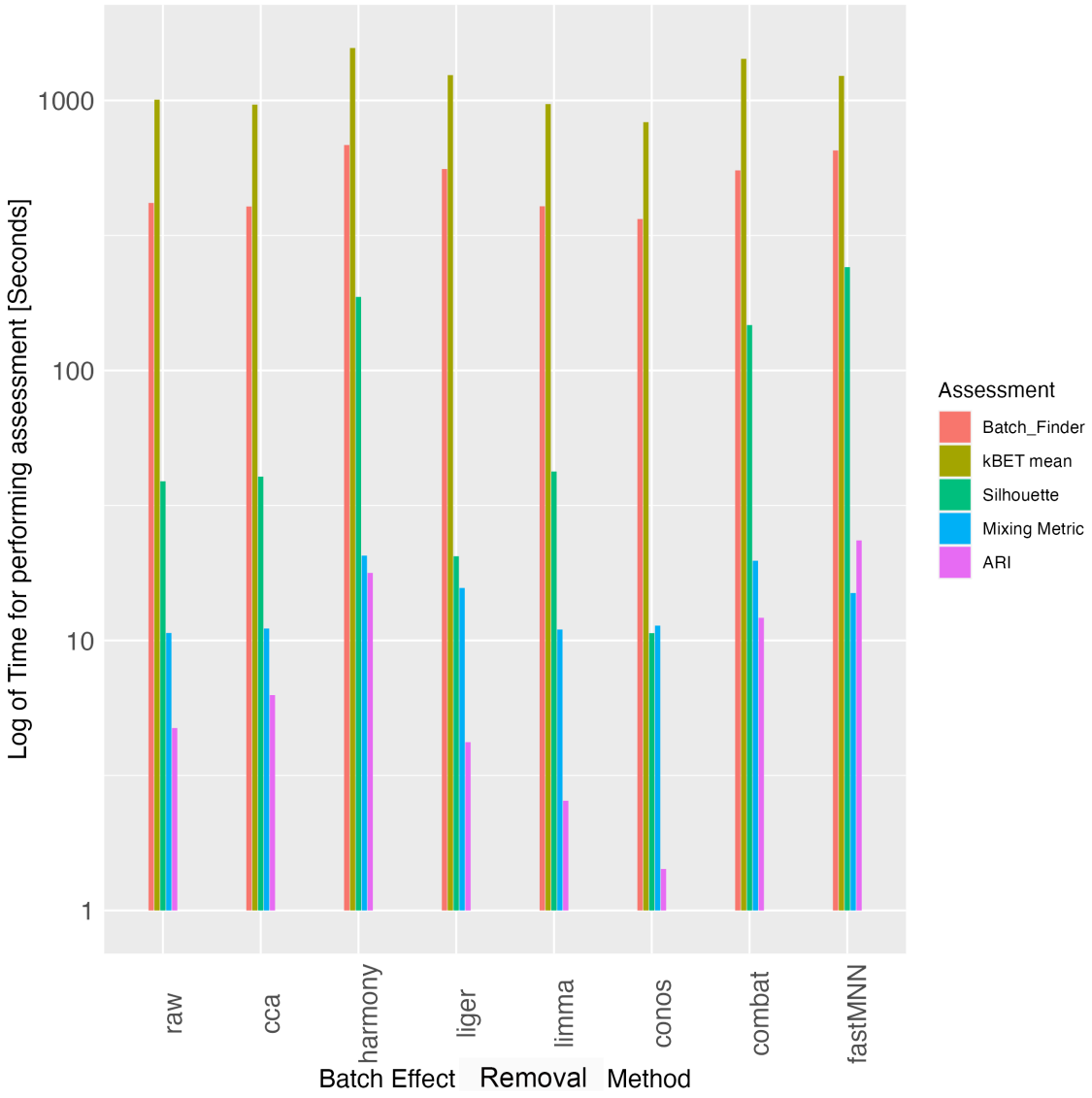


Figure 5.41: The time taken for running batch effect assessment methods on PBMC dataset.

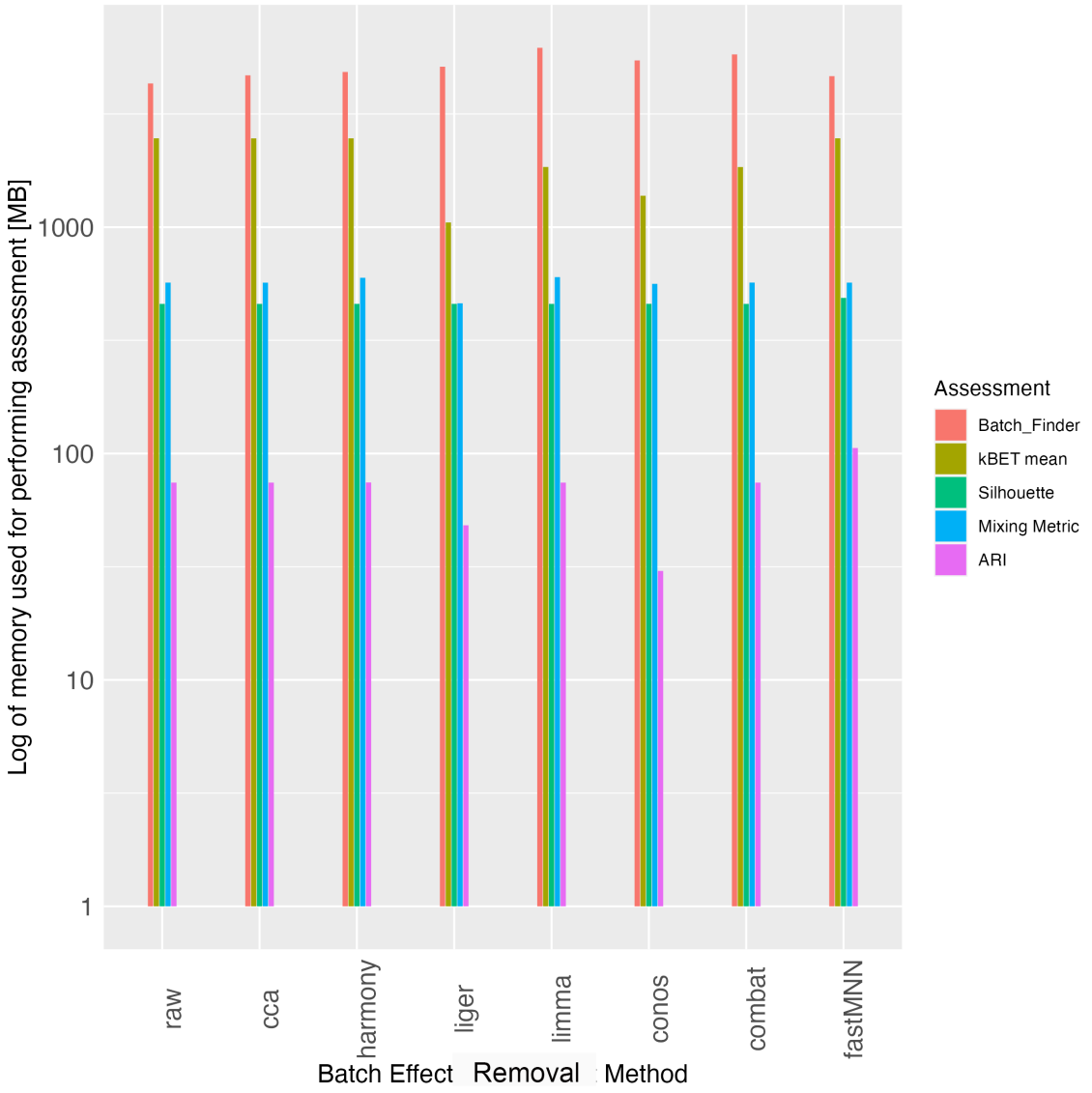


Figure 5.42: The memory usage for running batch effect assessment methods on Pancreas dataset.

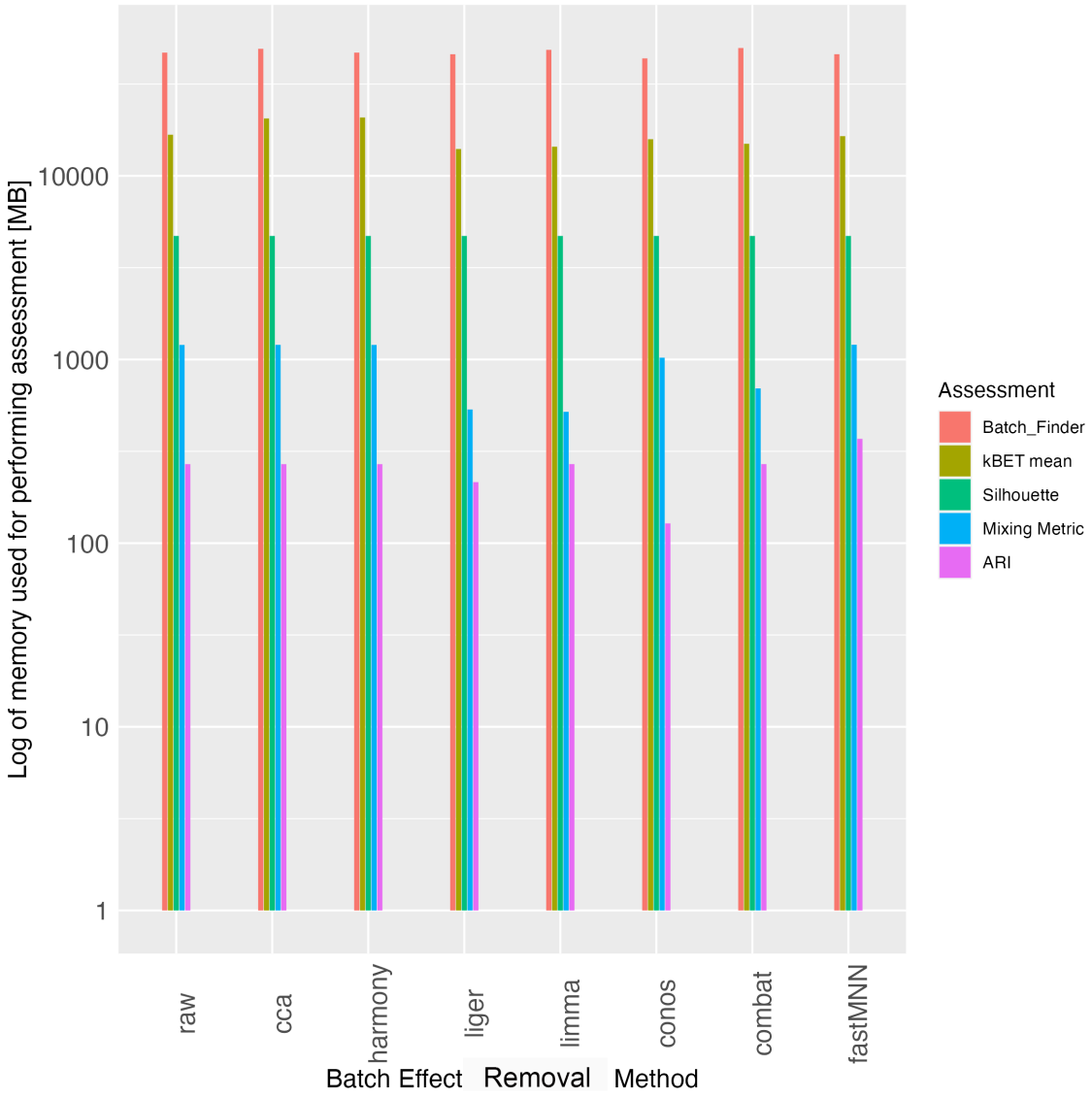


Figure 5.43: The memory usage for running batch effect assessment methods on Brain dataset.

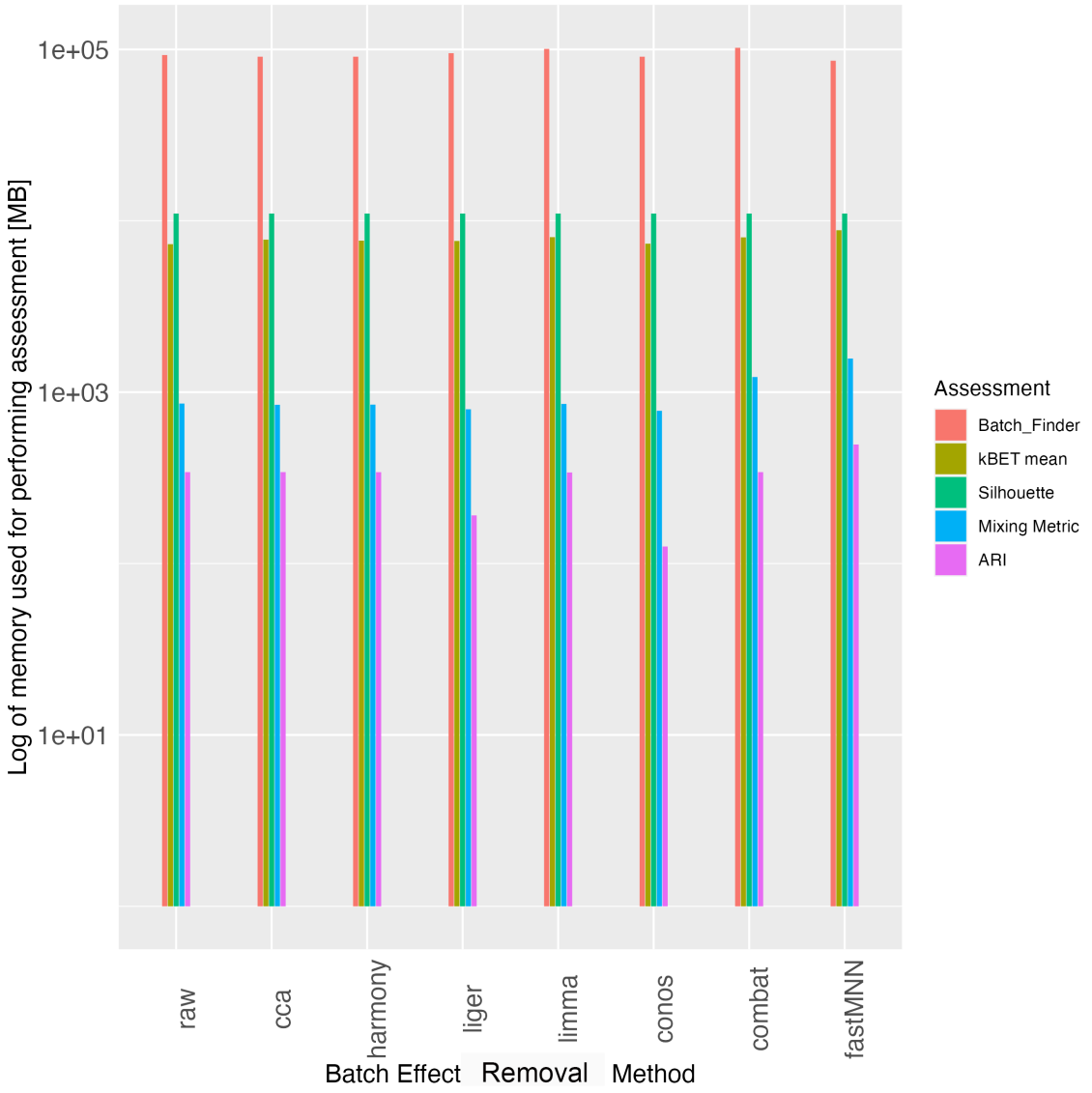


Figure 5.44: The memory usage for running batch effect assessment methods on PBMC dataset.

for batch effect assessment methods. The mouse brain dataset shows the batch effect originated from biological bases. The number of cells is more than in the pancreas dataset and the number of cell types is more but the cell types are all common between the two samples. Finally, the PBMC dataset is the biggest dataset in terms of the number of cell types. The batch effect in this dataset is also originated from technical variation. There are also cell types that are not common between all samples.

Dataset	Dataset 1	Dataset 2	Dataset 3
Organism	Human	Mouse	Human
Organ	Pancreas	Brain	PBMC
Number of samples	4	2	9
Number of cells	6321	20275	31021
Number of cell types	13	19	10
Number of genes	34363	19712	33694
Batch effect origin	Technology	Donor	Method
Year of publish	2016-2017	2019	2019

Table 5.1: Summary of datasets

5.5.2 Results analysis

According to figure 5.14, Batch_Finder has the most correlation with the manual assessment with the value of 0.95, Mixing metric also has provided a good assessment with a correlation of 0.85 with the manual method. By looking at the ranks figure 5.12, it's revealed that No method failed to recognize Limma as the worst method and they gave the CCA almost the best ranking. boxplot in figure 5.13 provides a better insight into understanding the variety of choices for each method. Decisions on CCA, Limma, Harmony, and Conos methods have almost been consistent resulting in smaller boxes. For Conos, Liger method, and Raw data the manual result is on the border of the box or even out of it, which means most of the assessment methods failed to assess the batch effect correctly in those cases. For the mouse brain dataset also the Btach_Finder has the most similar performance according to manual batch effect assessment shown in figure 5.26. ARI and kBET are the second-best methods with a correlation of 0.6 which is not an acceptable value. In figure 5.25, boxes are mostly big and there hasn't been any consensus on any removal method. As in the PBMC dataset, figure 5.38 demonstrates that Batch_Finder is the most correlated with the manual assessment with the value of 0.93 and then ARI has an acceptable correlation of 0.86. In figure 5.37 boxes are mostly big, but almost all methods agree that the limma removal method performed the worst and they all categorized the CCA as one of the best.

In order to analyze the performance of all methods together, the correlations between manual assessment and all other assessments in all datasets have been gathered in figure 5.45. As is shown, The Batch_Finder is the most correlated with the manual method in all three datasets. The ARI also had a good performance and kBET was around 0.60 in all datasets. The silhouette method seems to have problems with the brain dataset. It may be because of a specific structure in the dataset or cells' positions. As a matter of runtime, shown in figures 5.39, 5.40, and 5.41, The ARI method is the fastest overall and then mixing metric is the fastest method. While kBET is the slowest method and Batch_finder is the second slow assessment method. As Batch_Finder attempts to consider every cell distance and also uses the cell types to increase the accuracy, it has become slower than methods that perform acceptably but not as well as Batch_Finder.

The memory usage by Batch_Finder demonstrated in figures 5.42, 5.43, and 5.44, is the highest amount and it should be considered that it is in a logarithmic scale.

By observing the performance of Batch_Finder in all datasets, it seems biased towards the batch effect removals with lower dimension output. This problem originates from the calculation of euclidean distance and the more dimensions mean the higher distance. To handle this problem a proper normalization should be done on euclidean distances so that they become irrelevant of the dimension of data.

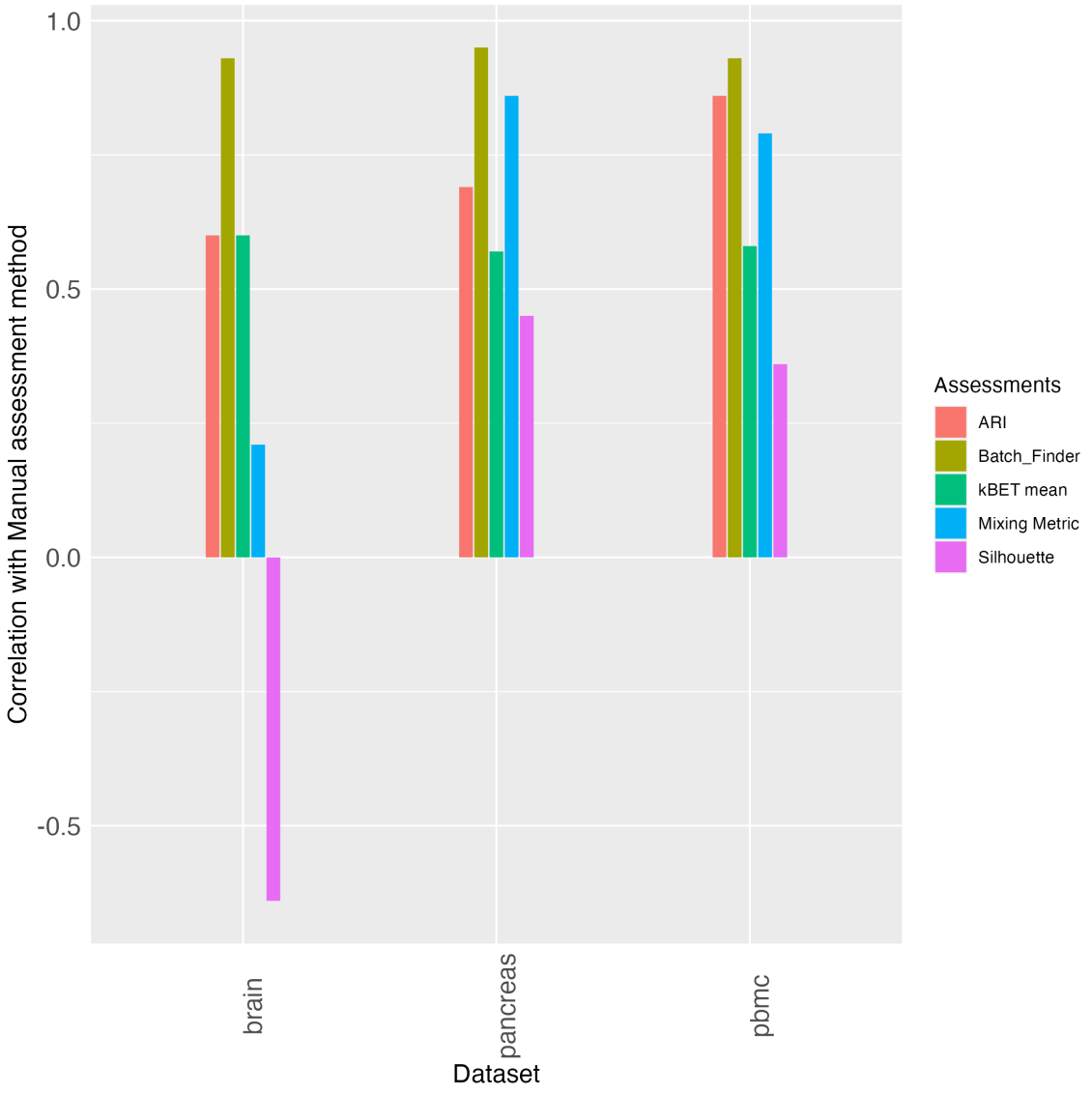


Figure 5.45: Correlation of all assessment methods with manual assessment for all 3 datasets.

5.5.3 Conclusion

In this chapter the proposed batch effect assessment method, Batch_Finder, has been tested on 3 real biological datasets. These three datasets are integrated from multiple datasets to provide a sufficient amount of batch effect. Then 7 batch effect removal methods were implemented on each of them and the result accompanying the raw data have been evaluated by 5 batch assessment methods. The results of these assessments have been compared in terms of performance, runtime, and memory usage. It has been shown that the proposed method strictly outperforms the other tested batch effect assessment methods in terms of performance but its time and memory efficiency needs to be improved.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this study, we developed a novel batch effect assessment method, Batch_Finder, to address the challenges posed by large and complex datasets. Our method helps to improve the precision of detecting batch effects and assists in selecting the most appropriate method for removing them, making it particularly useful for datasets that have been merged from multiple sources. Through a thorough evaluation using both synthetic and real-world data, we demonstrated that Batch_Finder consistently outperforms existing methods in detecting batch effects and has a strong foundation in mathematical and logical principles. Additionally, Batch_Finder provides a summarized view of the dataset that is more manageable and

less overwhelming, enabling experts to gain a deeper understanding of the relationships between cell types.

In chapter 4, we introduced and formulated Batch_Finder. We then conducted a thorough evaluation of Batch Finder by comparing it to four other batch effect assessment methods using eleven test datasets. The results showed that Batch Finder consistently outperformed the other methods, successfully passing all challenges while the other methods failed at least one test.

In chapter 5, we applied Batch_Finder, our novel batch effect assessment method, to three biological integrated datasets to evaluate its performance on real-world data. These datasets were selected to present at least one challenge for batch assessment methods and were chosen for their relevance to our research question. These datasets have been subject to seven batch effect removal methods. The results of our evaluation showed that Batch_Finder demonstrated the highest correlation with the expert's assessment of the datasets, indicating that it was able to accurately identify batch effects in the data.

As technology continues to advance and the size and complexity of datasets grow, it is becoming increasingly important to develop computational tools that can help to understand and analyze these datasets. Batch_Finder represents a significant contribution to this effort, offering a reliable and effective method for detecting and addressing batch effects in large and complex datasets. While it may not be as memory- and time-efficient as some

existing methods, we believe that Batch_Finder represents a valuable tool for researchers working with large datasets, and we hope that it will contribute to the advancement of research in the field.

6.2 Future work

While Batch_Finder has shown promising results in our evaluations, there is still room for improvement in terms of memory efficiency and runtime. We plan to address these issues in future work to make Batch_Finder more practical for use in real-world applications. Additionally, we plan to make Batch_Finder available as an R package to facilitate its use by other researchers.

In terms of the algorithm itself, we have identified a potential issue with the normalization of the euclidean distance in cases where the input data has a smaller dimension. We believe that implementing a proper normalization procedure in these cases could improve the accuracy of the algorithm. We plan to explore this issue in future work to further optimize the performance of Batch_Finder.

As mentioned before, defined test cases are acurate, but they are simplifications of the real world data while the real world data can not be classified with certainty, we need to design a test that has both high accuracy and similarity to real world data. There are some alternatives such as running a standardized clustring pipeline on the integrated data

to see if it recapitulates the known cell-types rather than the batches. We can also work on generating synthetic batch effect in real world data to test the effectiveness of Batch_Finder.

Overall, we believe that Batch_Finder has the potential to make a significant impact in the field of batch effect assessment and we look forward to exploring its capabilities further in future research.

Bibliography

[1] Brian T Wilhelm, Samuel Marguerat, Stephen Watt, Falk Schubert, Valerie Wood, Ian Goodhead, Christopher J Penkett, Jane Rogers, and Jürg Bähler. Dynamic repertoire of a eukaryotic transcriptome surveyed at single-nucleotide resolution. *Nature*, 453(7199):1239–1243, 2008.

[2] 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.

[3] Sydney Brenner, Maria Johnson, John Bridgham, George Golda, David H Lloyd, Davida Johnson, Shujun Luo, Sarah McCurdy, Michael Foy, Mark Ewan, et al. Gene expression analysis by massively parallel signature sequencing (mpss) on microbead arrays. *Nature biotechnology*, 18(6):630–634, 2000.

[4] Zhong Wang, Mark Gerstein, and Michael Snyder. Rna-seq: a revolutionary tool for

transcriptomics. *Nature reviews genetics*, 10(1):57–63, 2009.

[5] Emma Pierson and Christopher Yau. Zifa: Dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome biology*, 16(1):1–10, 2015.

[6] Serena Liu and Cole Trapnell. Single-cell transcriptome sequencing: recent advances and remaining challenges. *F1000Research*, 5, 2016.

[7] 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.

[8] Arash Dalili, Ehsan Samiei, and Mina Hoorfar. A review of sorting, separation and isolation of cells and microbeads for biomedical applications: Microfluidic approaches. *Analyst*, 144(1):87–113, 2019.

[9] Itamar Kanter and Tomer Kalisky. Single cell transcriptomics: methods and applications. *Frontiers in oncology*, 5:53, 2015.

[10] Asif Adil, Vijay Kumar, Arif Tasleem Jan, and Mohammed Asger. Single-cell transcriptomics: current methods and challenges in data acquisition and analysis. *Frontiers in Neuroscience*, 15:591122, 2021.

[11] David W McKellar, Lauren D Walter, Leo T Song, Madhav Mantri, Michael FZ Wang, Iwijn De Vlaminck, and Benjamin D Cosgrove. Large-scale integration of single-cell transcriptomic data captures transitional progenitor states in mouse skeletal muscle regeneration. *Communications biology*, 4(1):1280, 2021.

[12] Aviv Regev, Sarah A Teichmann, Eric S Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, Peter Campbell, Piero Carninci, Menna Clatworthy, et al. The human cell atlas. *elife*, 6:e27041, 2017.

[13] Po-Yuan Tung, John D Blischak, Chiaowen Joyce Hsiao, David A Knowles, Jonathan E Burnett, Jonathan K Pritchard, and Yoav Gilad. Batch effects and the effective design of single-cell gene expression studies. *Scientific reports*, 7(1):1–15, 2017.

[14] Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics*, 11(10):733–739, 2010.

[15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2022.

[16] Pierre Baldi and Soren Brunak. Bioinformatics. *The machine learning approach*, 2, 2001.

[17] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[18] 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.

[19] Georg Hager and Gerhard Wellein. *Introduction to high performance computing for scientists and engineers*. CRC Press, 2010.

[20] Ruizhi Xiang, Wencan Wang, Lei Yang, Shiyuan Wang, Chaohan Xu, and Xiaowen Chen. A comparison for dimensionality reduction methods of single-cell rna-seq data. *Frontiers in genetics*, 12:646936, 2021.

[21] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space.

The London, Edinburgh, and Dublin philosophical magazine and journal of science, 2(11):559–572, 1901.

[22] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel WH Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W Newell. Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology*, 37(1):38–44, 2019.

[23] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[24] Steven M Holland. Principal components analysis (pca). *Department of Geology, University of Georgia, Athens, GA*, pages 30602–2501, 2008.

[25] Ilya Korsunsky, Nghia Millard, Jean Fan, Kamil Slowikowski, Fan Zhang, Kevin Wei, Yuriy Baglaenko, Michael Brenner, Po-ru Loh, and Soumya Raychaudhuri. Fast, sensitive and accurate integration of single-cell data with harmony. *Nature methods*, 16(12):1289–1296, 2019.

[26] Gordon K Smyth and Terry Speed. Normalization of cdna microarray data. *Methods*, 31(4):265–273, 2003.

[27] Laleh Haghverdi, Aaron TL Lun, Michael D Morgan, and John C Marioni. Batch

effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nature biotechnology*, 36(5):421–427, 2018.

[28] A Lun. A description of the theory behind the fastmnn algorithm, 2019.

[29] A Lun. Further mnn algorithm development, 2018.

[30] Joshua Welch, Velina Kozareva, Ashley Ferreira, Charles Vanderburg, Carly Martin, and Evan Macosko. Integrative inference of brain cell similarities and differences from single-cell genomics. *BioRxiv*, page 459891, 2018.

[31] Pei-Zhuang Wang, Zeng-Liang Liu, Yong Shi, and Si-Cong Guo. Factor space, the theoretical base of data science. *Annals of data science*, 1:233–251, 2014.

[32] 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.

[33] Wolfgang Härdle, Léopold Simar, et al. *Applied multivariate statistical analysis*, volume 22007. Springer, 2007.

[34] Nikolas Barkas, Viktor Petukhov, Daria Nikolaeva, Yaroslav Lozinsky, Samuel Demharter, Konstantin Khodosevich, and Peter V Kharchenko. Joint analysis of het-

erogeneous single-cell rna-seq dataset collections. *Nature methods*, 16(8):695–698, 2019.

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

[36] Yuqing Zhang, Giovanni Parmigiani, and W Evan Johnson. Combat-seq: batch effect adjustment for rna-seq count data. *NAR genomics and bioinformatics*, 2(3):lqaa078, 2020.

[37] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[38] Maren Büttner, Zhichao Miao, F Alexander Wolf, Sarah A Teichmann, and Fabian J Theis. A test metric for assessing single-cell rna-seq batch correction. *Nature methods*, 16(1):43–49, 2019.

[39] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[40] Alina Beygelzimer, Sham M. Kakade, and John Langford. Cover trees for nearest

neighbor. *Proceedings of the 23rd international conference on Machine learning*, 2006.

[41] Karl Pearson. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.

[42] Max Reynolds. Batch effects-removing unwanted variation from your data what are batch effects and how to deal with them.

[43] Jelena Čuklina, Chloe H Lee, Evan G Williams, Tatjana Sajic, Ben C Collins, María Rodríguez Martínez, Varun S Sharma, Fabian Wendt, Sandra Goetze, Gregory R Keele, et al. Diagnostics and correction of batch effects in large-scale proteomic studies: a tutorial. *Molecular systems biology*, 17(8):e10240, 2021.

[44] Robert M Groves and Lars Lyberg. Total survey error: Past, present, and future. *Public opinion quarterly*, 74(5):849–879, 2010.

[45] Scott Menard. *Longitudinal research*, volume 76. Sage, 2002.

[46] Michael Borenstein, Larry V Hedges, Julian PT Higgins, and Hannah R Rothstein. *Introduction to meta-analysis*. John Wiley & Sons, 2021.

[47] Charlotte Soneson, Samuel Gerster, and Manuel Delorenzi. Batch effect confounding leads to strong bias in performance estimates obtained by cross-validation. *PLoS ONE*, 9(6):e100335, 2014.

[48] 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.

[49] Wilson Wen Bin Goh, Chern Han Yong, and Limsoon Wong. Are batch effects still relevant in the age of big data? *Trends in Biotechnology*, 40(9):1029–1040, 2022.

[50] 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.

[51] Luyi Tian, Xueyi Dong, Saskia Freytag, Kim-Anh Le Cao, Shian Su, Abolfazl Jalal-Abadi, Daniela Amann-Zalcenstein, Tom S Weber, Azadeh Seidi, Jafar S Jabbari, et al. scrna-seq mixology: towards better benchmarking of single cell rna-seq analysis methods. *BioRxiv*, page 433102, 2019.

[52] Jiaqi Li, Chengxuan Yu, Lifeng Ma, Jingjing Wang, and Guoji Guo. Comparison of scanpy-based algorithms to remove the batch effect from single-cell rna-seq data. *Cell Regeneration*, 9(1):1–8, 2020.

[53] Malte D Luecken, Maren Büttner, Kridsadakorn Chaichoompu, Anna Danese, Marta Interlandi, Michaela F Müller, Daniel C Strobl, Luke Zappia, Martin Dugas, Maria Colomé-Tatché, et al. Benchmarking atlas-level data integration in single-cell genomics. *Nature methods*, 19(1):41–50, 2022.

[54] 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 biology*, 21(1):1–32, 2020.

[55] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[56] Dominic Grün, Mauro J Muraro, Jean-Charles Boisset, Kay Wiebrands, Anna Lyubimova, Gitanjali Dharmadhikari, Maaike van den Born, Johan Van Es, Erik Jansen, Hans Clevers, et al. De novo prediction of stem cell identity using single-cell transcriptome data. *Cell stem cell*, 19(2):266–277, 2016.

[57] Mauro J Muraro, Gitanjali Dharmadhikari, Dominic Grün, Nathalie Groen, Tim Dielen, Erik Jansen, Leon Van Gurp, Marten A Engelse, Francoise Carlotti, Eelco Jp

De Koning, et al. A single-cell transcriptome atlas of the human pancreas. *Cell systems*, 3(4):385–394, 2016.

[58] Nathan Lawlor, Joshy George, Mohan Bolisetty, Romy Kursawe, Lili Sun, V Sivakamasundari, Ina Kycia, Paul Robson, and Michael L Stitzel. Single-cell transcriptomes identify human islet cell signatures and reveal cell-type–specific expression changes in type 2 diabetes. *Genome research*, 27(2):208–222, 2017.

[59] Åsa Segerstolpe, Athanasia Palasantza, Pernilla Eliasson, Eva-Marie Andersson, Anne-Christine Andréasson, Xiaoyan Sun, Simone Picelli, Alan Sabirsh, Maryam Clausen, Magnus K Bjursell, et al. Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes. *Cell metabolism*, 24(4):593–607, 2016.

[60] Daniela J Di Bella, Ehsan Habibi, Robert R Stickels, Gabriele Scalia, Juliana Brown, Payman Yadollahpour, Sung Min Yang, Catherine Abbate, Tommaso Biancalani, Evan Z Macosko, et al. Molecular logic of cellular diversification in the mouse cerebral cortex. *Nature*, 595(7868):554–559, 2021.

[61] Jiarui Ding, Xian Adiconis, SashaSean K Simmons, Monica S Kowalczyk, Cynthia C Hession, Nemanja D Marjanovic, Travis K Hughes, Marc H Wadsworth, Tyler Burks, Lan T Nguyen, John YH Kwon, Boaz Barak, William Ge, Amanda J Kedaigle, Shaina Carrol, Shuqiang Li, Nir Hacohen, Orit Rozenblatt-Rosen, Alex K Shalek,

Alexandra-Chloé Villani, Aviv Regev, and Joshua Z Levin. Systematic comparative analysis of single cell rna-sequencing methods. *bioRxiv*, 2019.

Appendix A

Extra Visualization of datasets

In this appendix, additional visualization techniques are presented for further analysis and investigation. These techniques provide additional insights and perspectives on the data and can help to deepen our understanding of the relationships and patterns present in the data. By examining the data using a variety of visualization methods, we can gain a more comprehensive and nuanced understanding of the data and the underlying trends and patterns. This can be especially useful when working with complex or high-dimensional data, as it allows us to identify and highlight important features and relationships that may not be immediately apparent using a single visualization method.

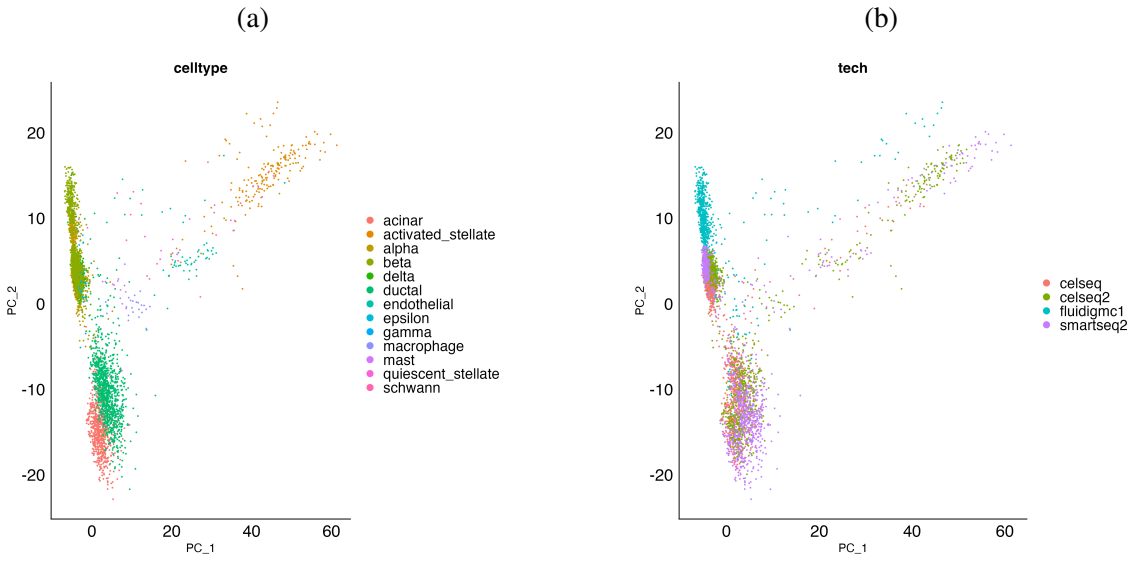


Figure A.1: PCA of the pancreas integrated data before batch removal. **a**, cell types. **b**, batches.

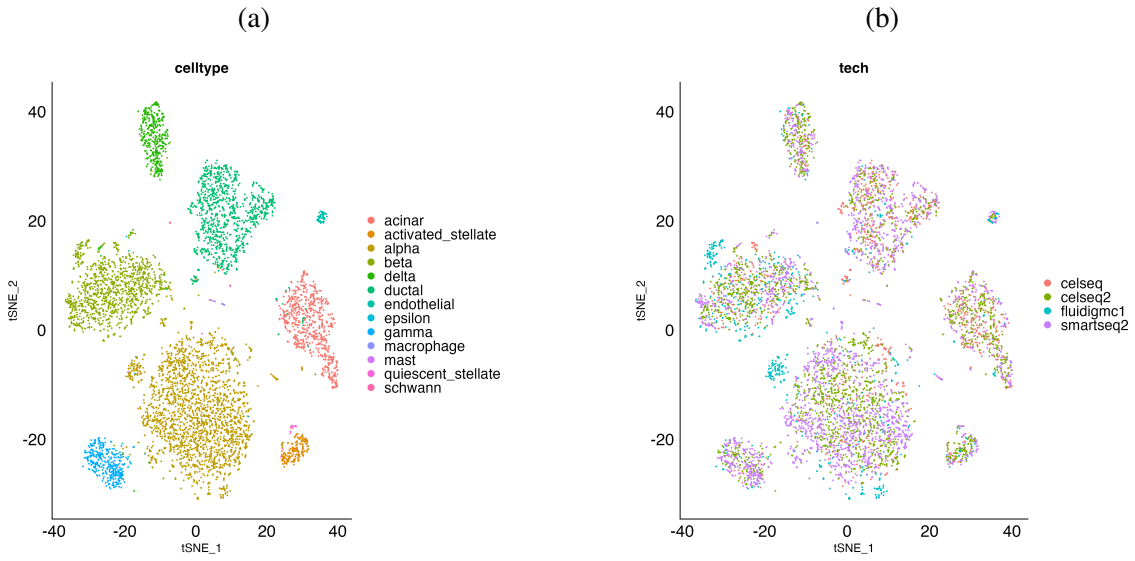


Figure A.2: t-SNE of the pancreas integrated data after harmony method. **a**, cell types. **b**, batches.

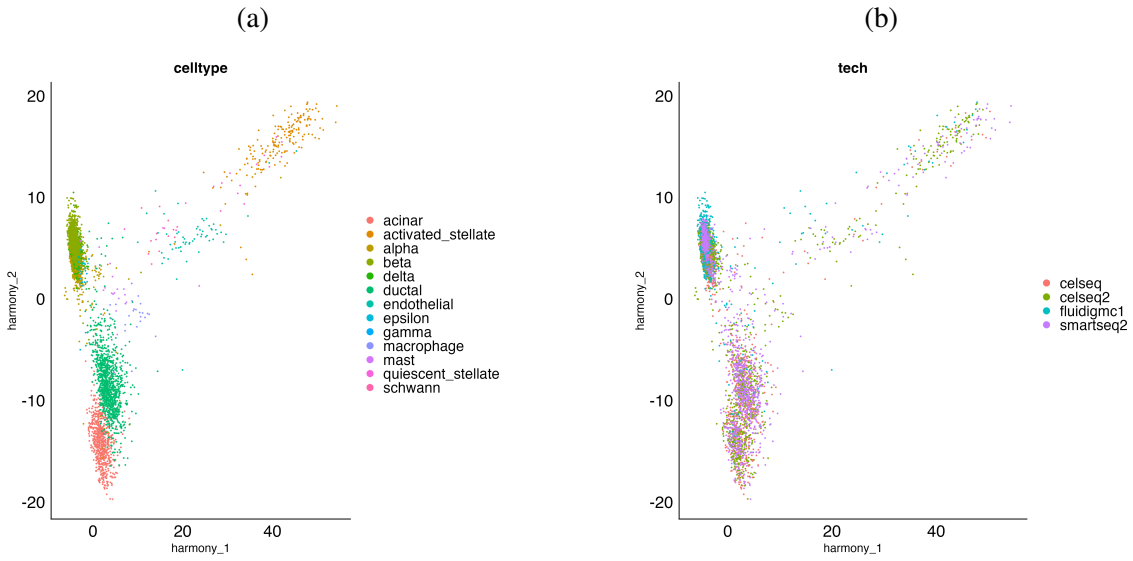


Figure A.3: PCA of the pancreas integrated data after harmony method. **a**, cell types. **b**, batches.

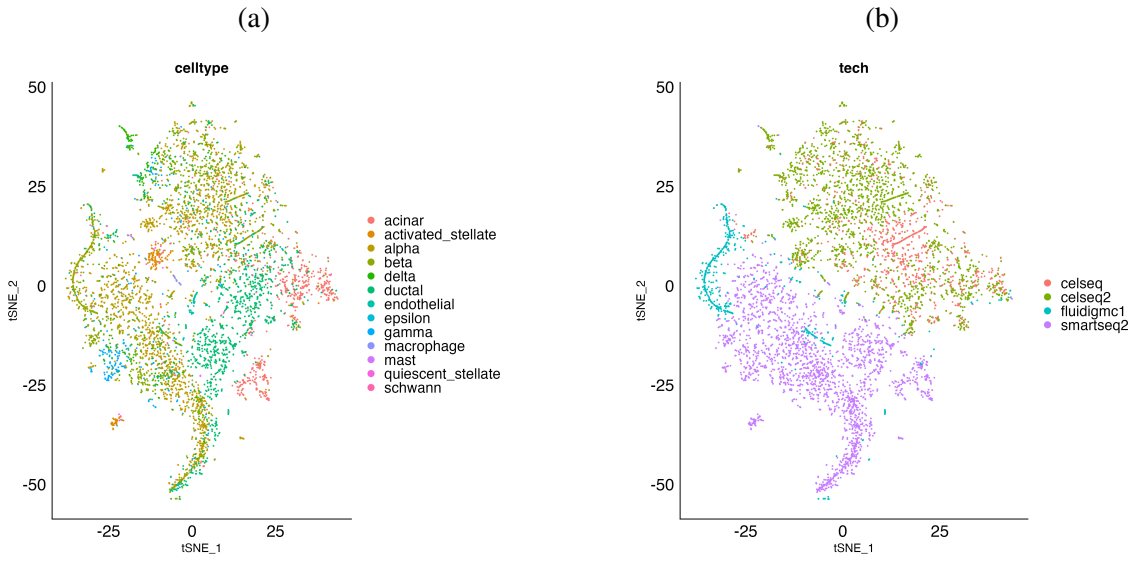


Figure A.4: t-SNE of the pancreas integrated data after limma method. **a**, cell types. **b**, batches.

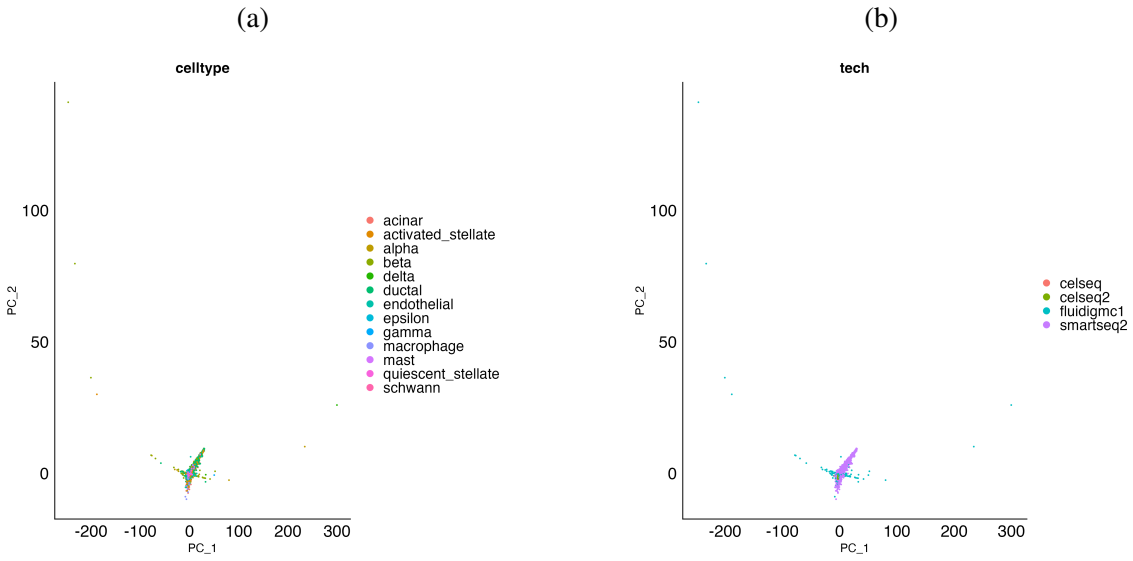


Figure A.5: PCA of the pancreas integrated data after limma method. **a**, cell types. **b**, batches.

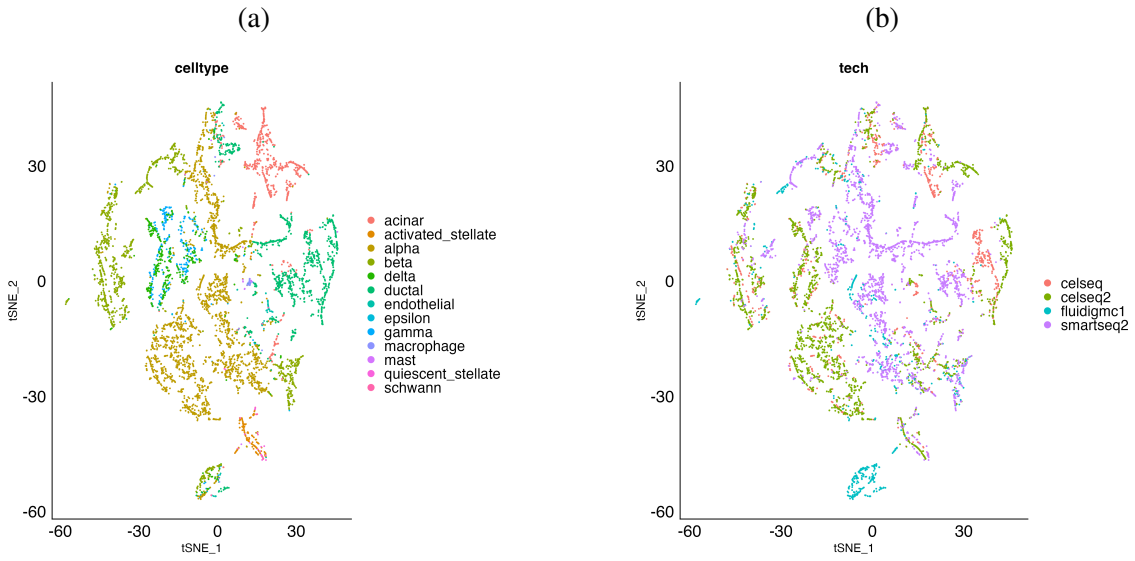


Figure A.6: t-SNE of the pancreas integrated data after liger method. **a**, cell types. **b**, batches.

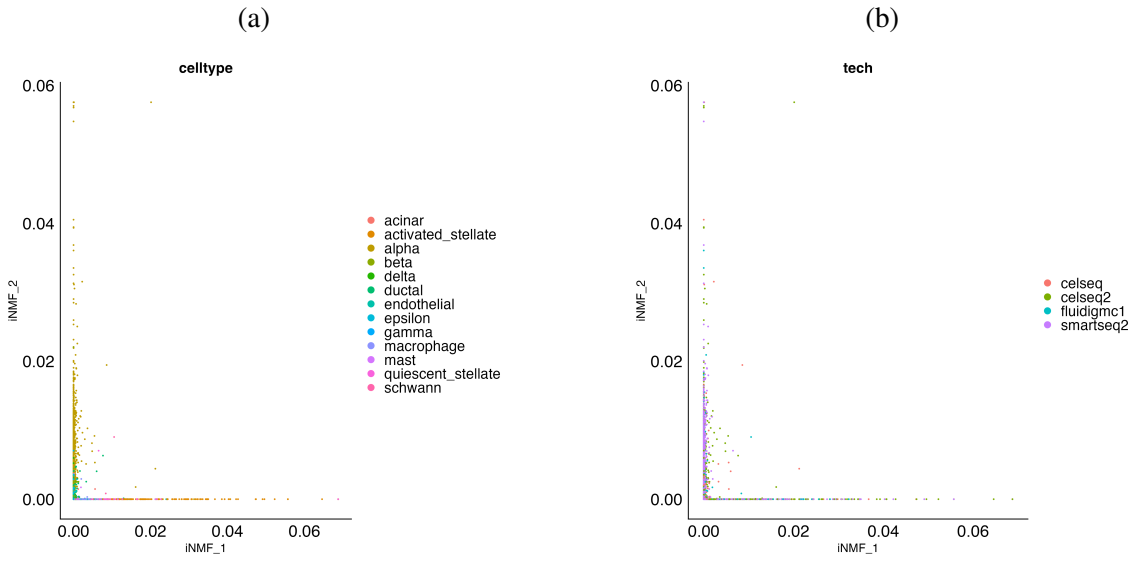


Figure A.7: PCA of the pancreas integrated data after liger method. **a**, cell types. **b**, batches.

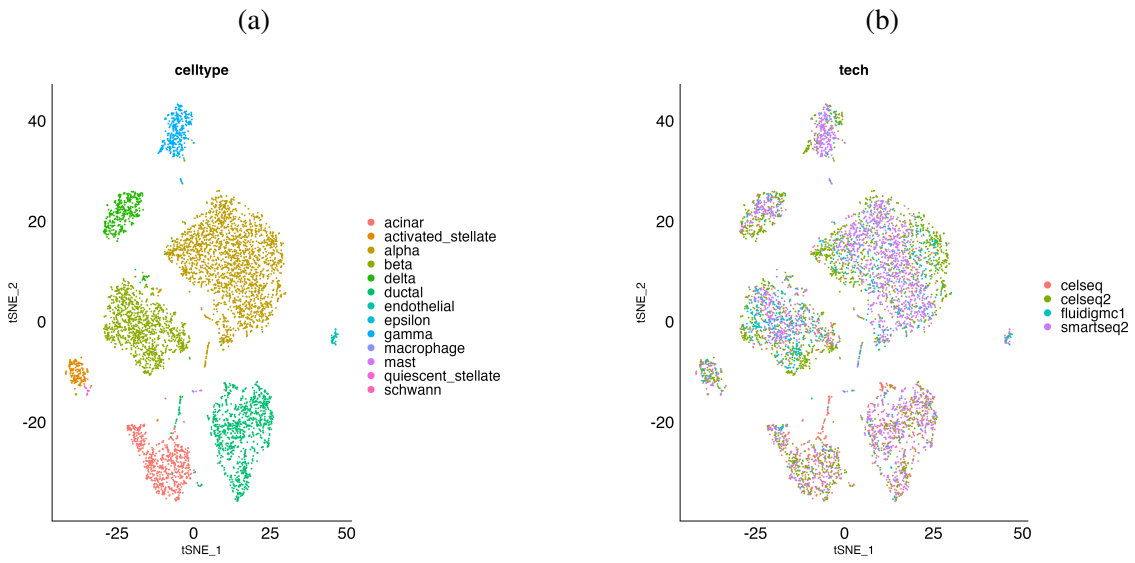


Figure A.8: t-SNE of the pancreas integrated data after CCA method. **a**, cell types. **b**, batches.

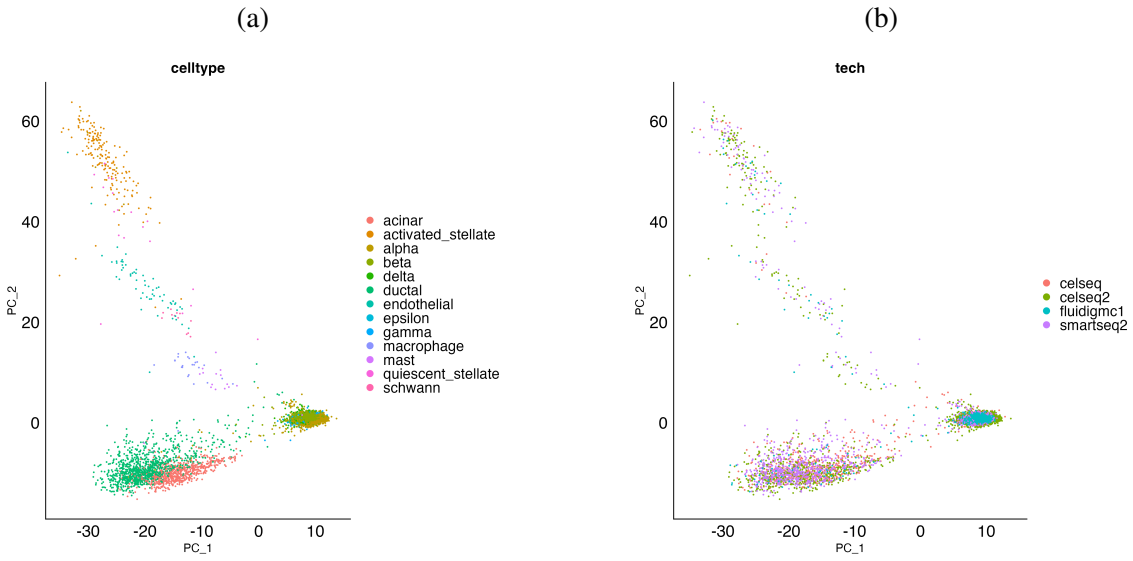


Figure A.9: PCA of the pancreas integrated data after CCA method. **a**, cell types. **b**, batches.

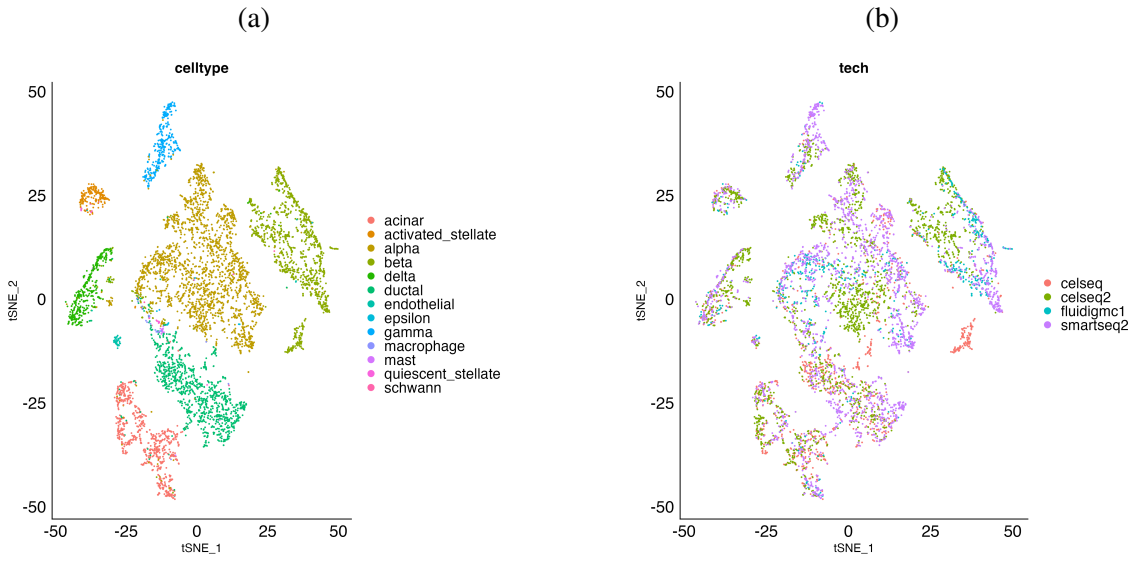


Figure A.10: t-SNE of the pancreas integrated data after fastMNN method. **a**, cell types. **b**, batches.

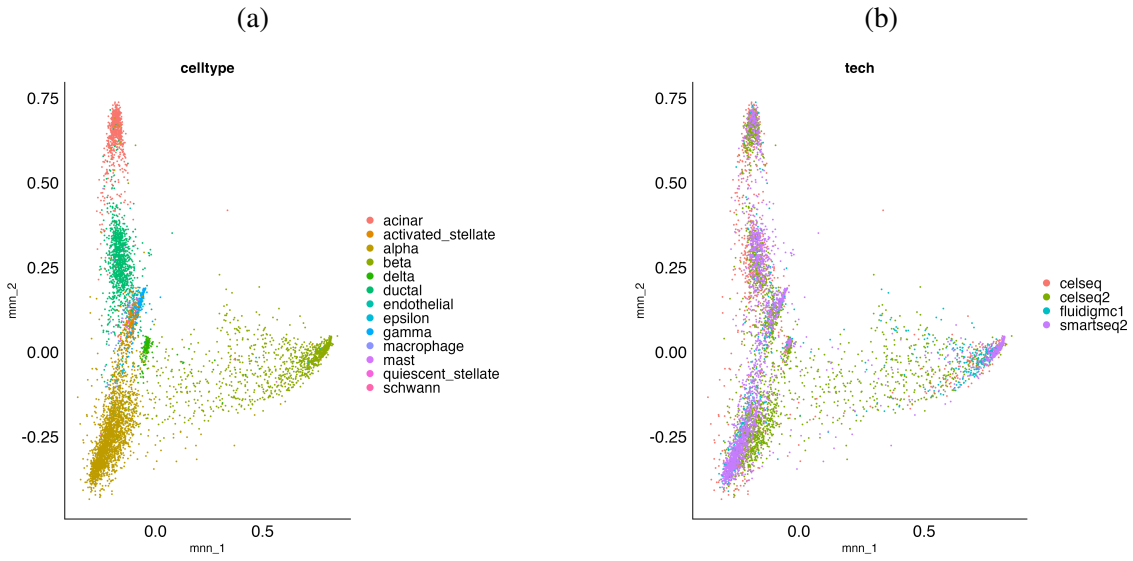


Figure A.11: PCA of the pancreas integrated data after fastMNN method. **a**, cell types. **b**, batches.

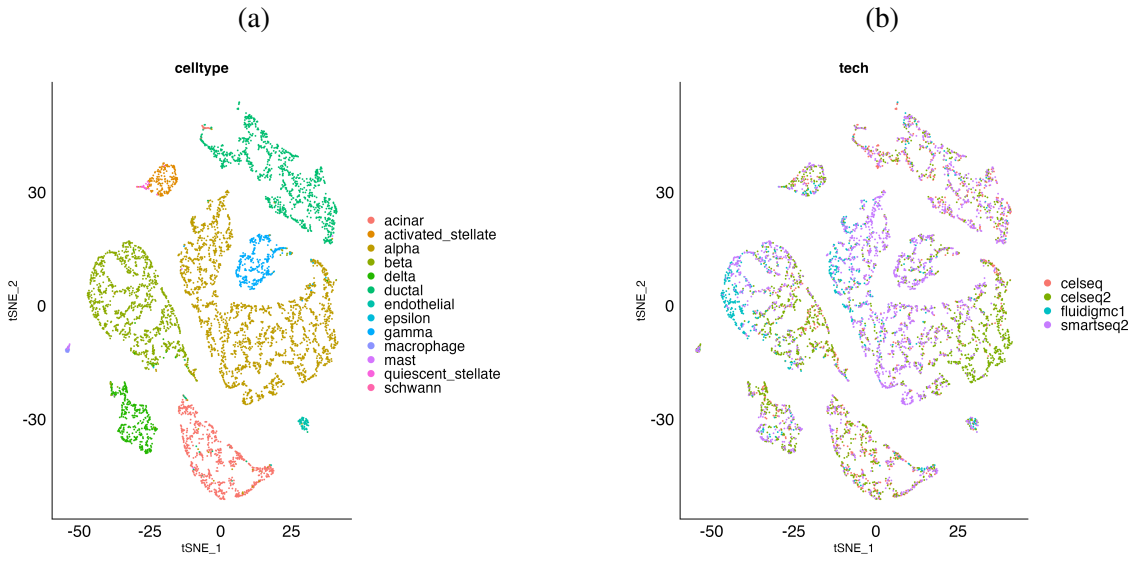


Figure A.12: t-SNE of the pancreas integrated data after conos method. **a**, cell types. **b**, batches.

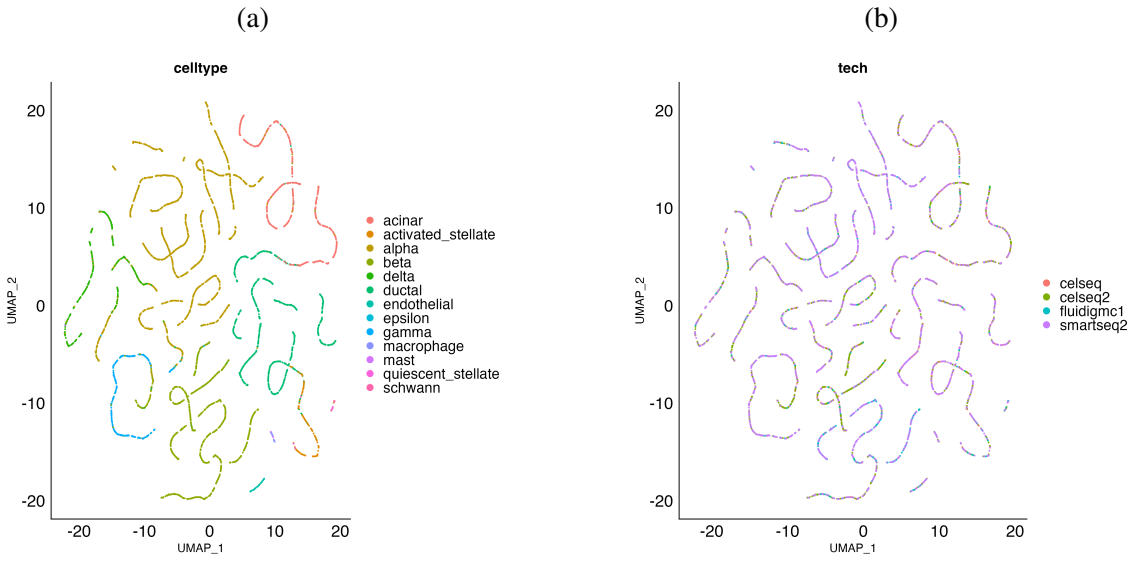


Figure A.13: UMAP of the pancreas integrated data after conos method. **a**, cell types. **b**, batches.

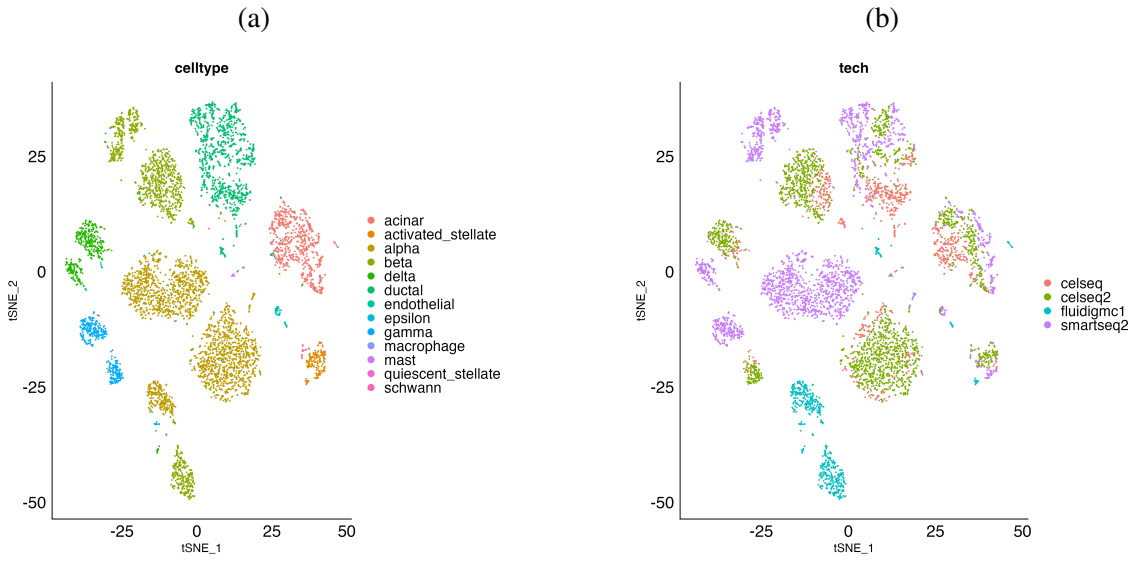


Figure A.14: t-SNE of the pancreas integrated data after combat method. **a**, cell types. **b**, batches.

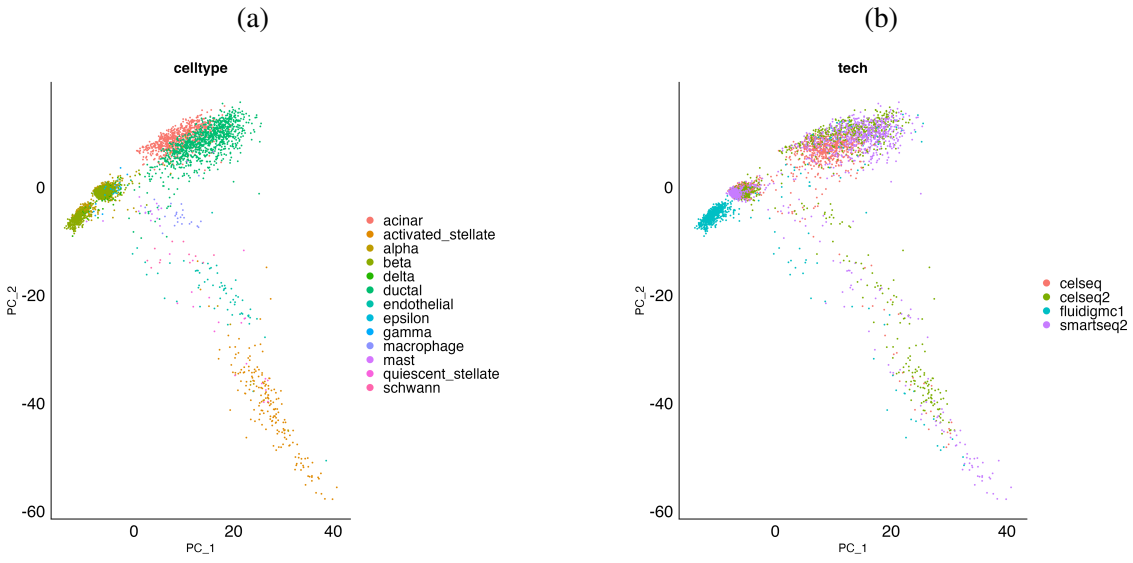


Figure A.15: PCA of the pancreas integrated data after combat method. **a**, cell types. **b**, batches.

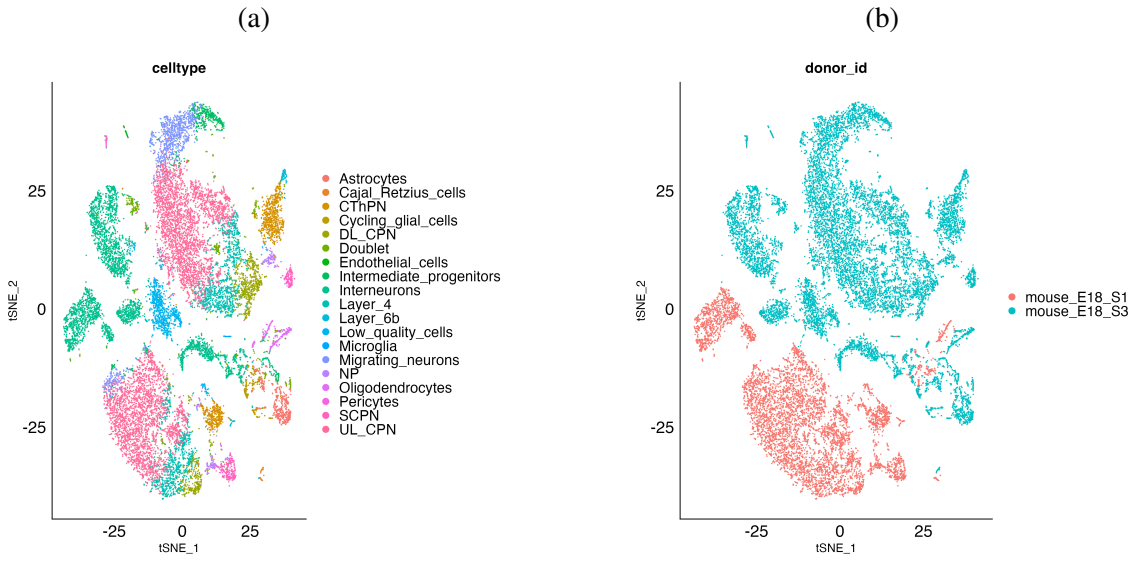


Figure A.16: t-SNE of the brain integrated data before batch removal. **a**, cell types. **b**, batches.

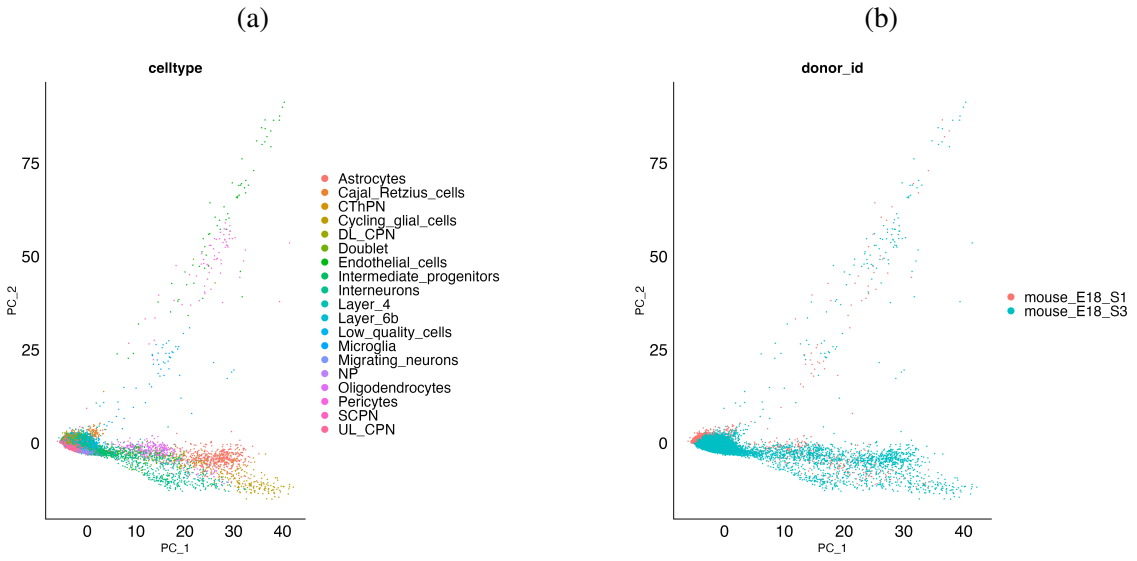


Figure A.17: PCA of the brain integrated data before batch removal. **a**, cell types. **b**, batches.

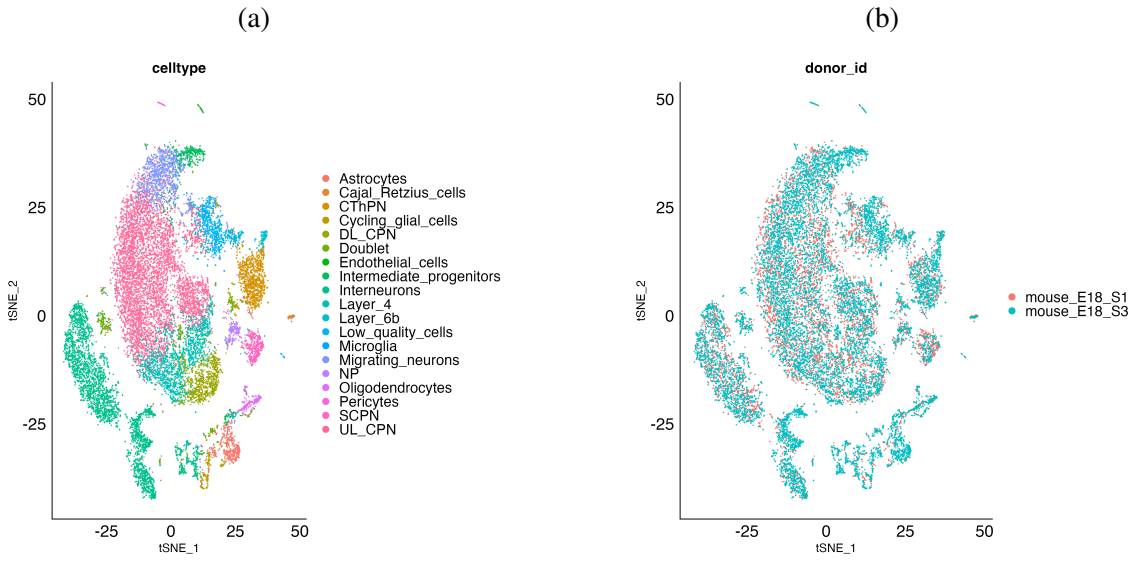


Figure A.18: t-SNE of the brain integrated data after harmony method. **a**, cell types. **b**, batches.

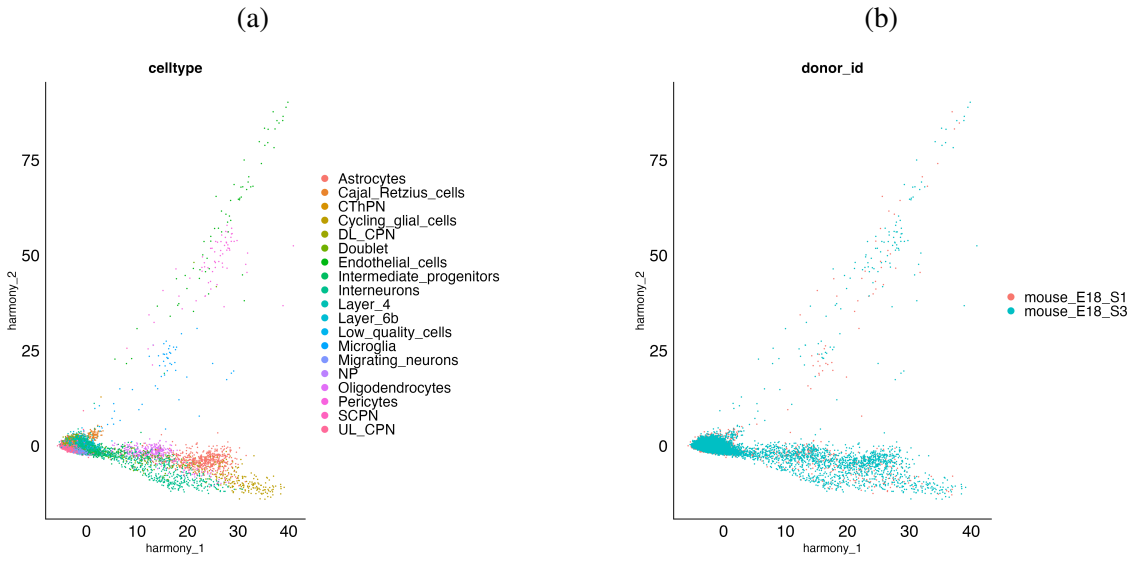


Figure A.19: PCA of the brain integrated data after harmony method. **a**, cell types. **b**, batches.

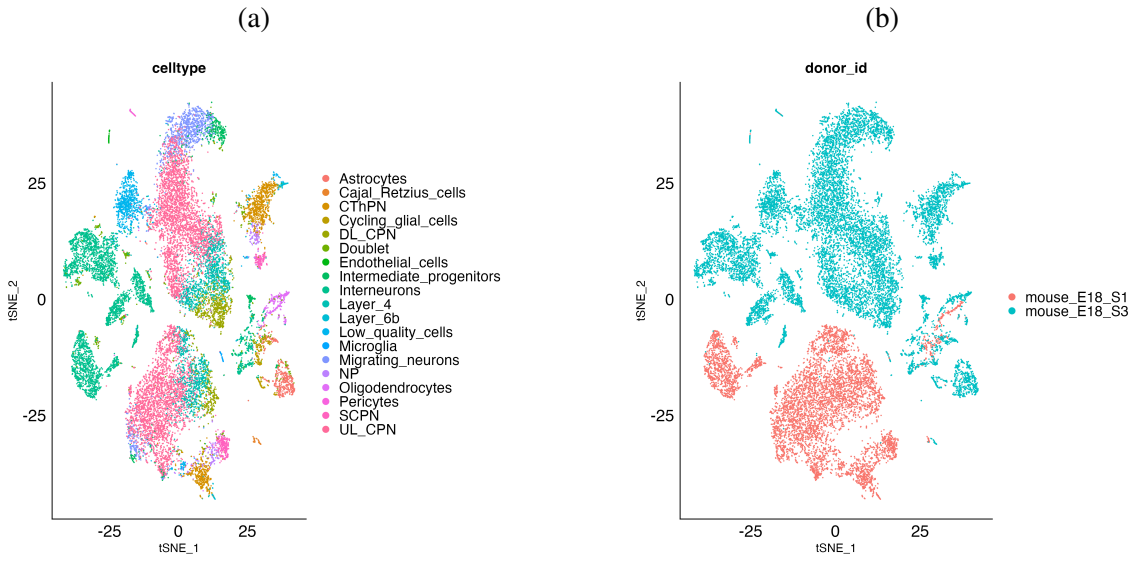


Figure A.20: t-SNE of the brain integrated data after limma method. **a**, cell types. **b**, batches.

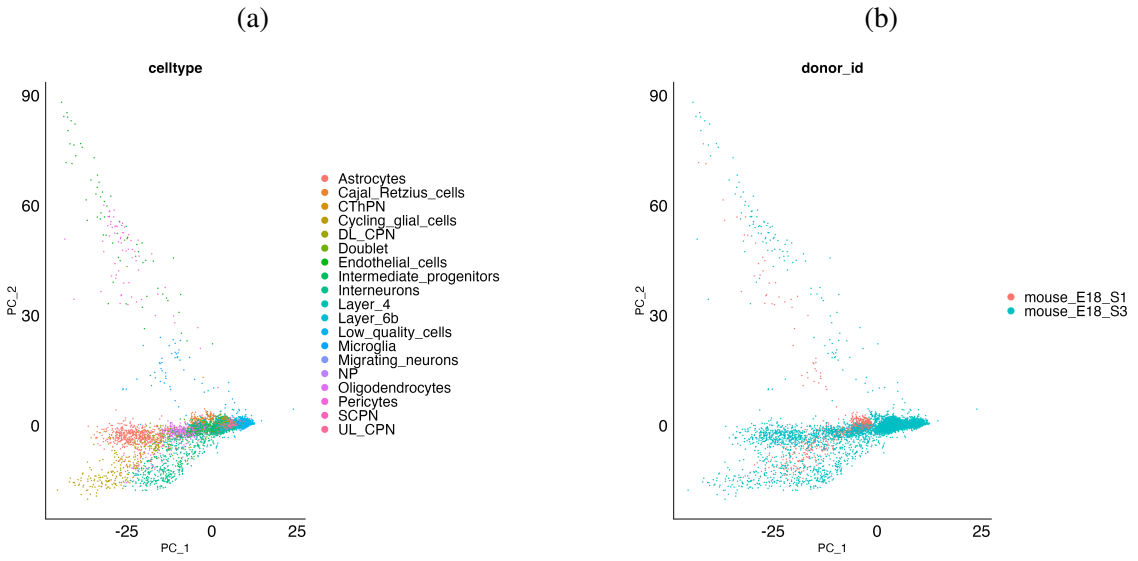


Figure A.21: PCA of the brain integrated data after limma method. **a**, cell types. **b**, batches.

(a) (b)

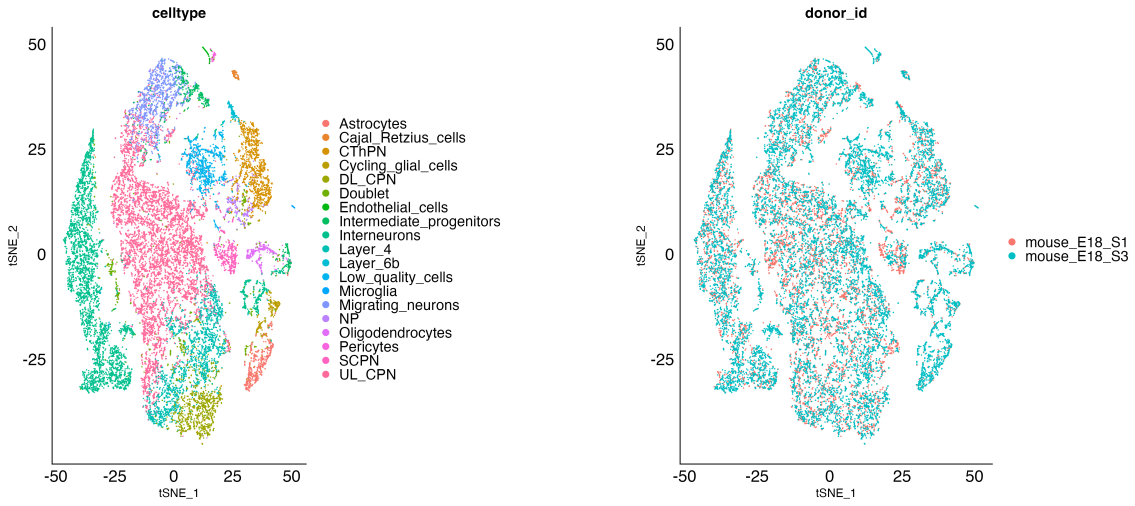


Figure A.22: t-SNE of the brain integrated data after liger method. **a**, cell types. **b**, batches.

(a) (b)

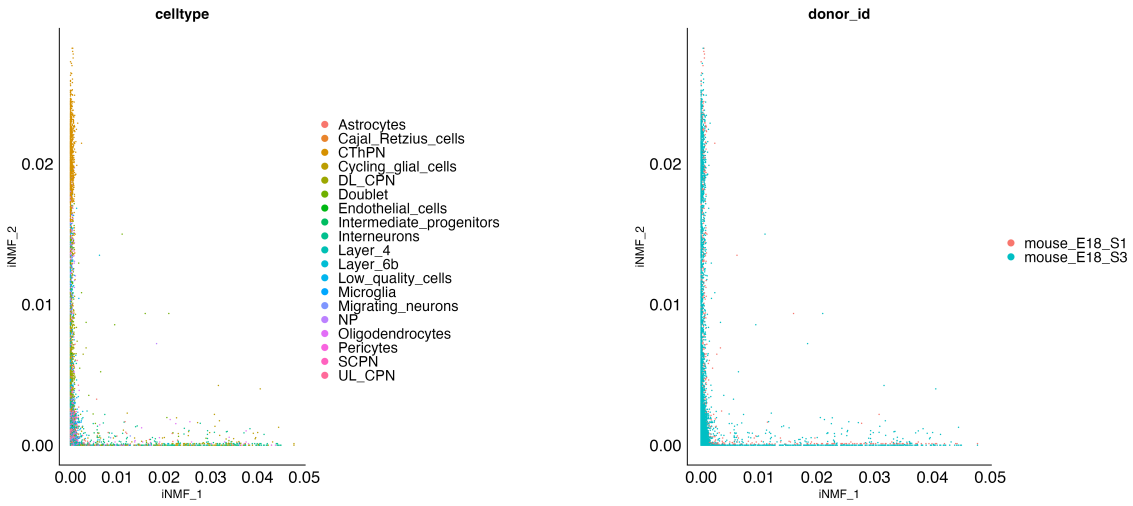


Figure A.23: PCA of the brain integrated data after liger method. **a**, cell types. **b**, batches.

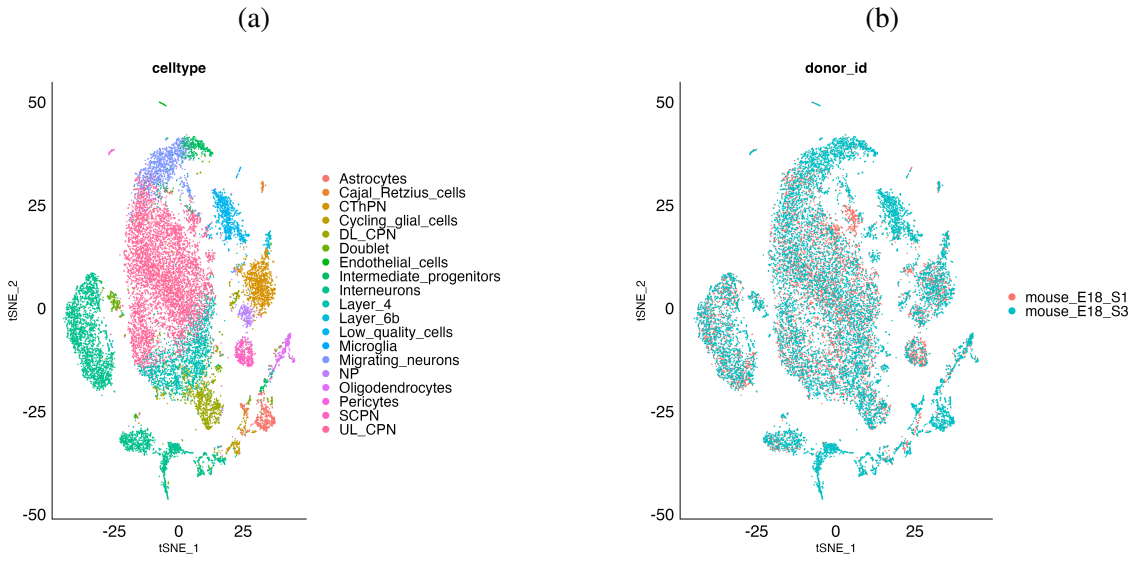


Figure A.24: t-SNE of the brain integrated data after CCA method. **a**, cell types. **b**, batches.

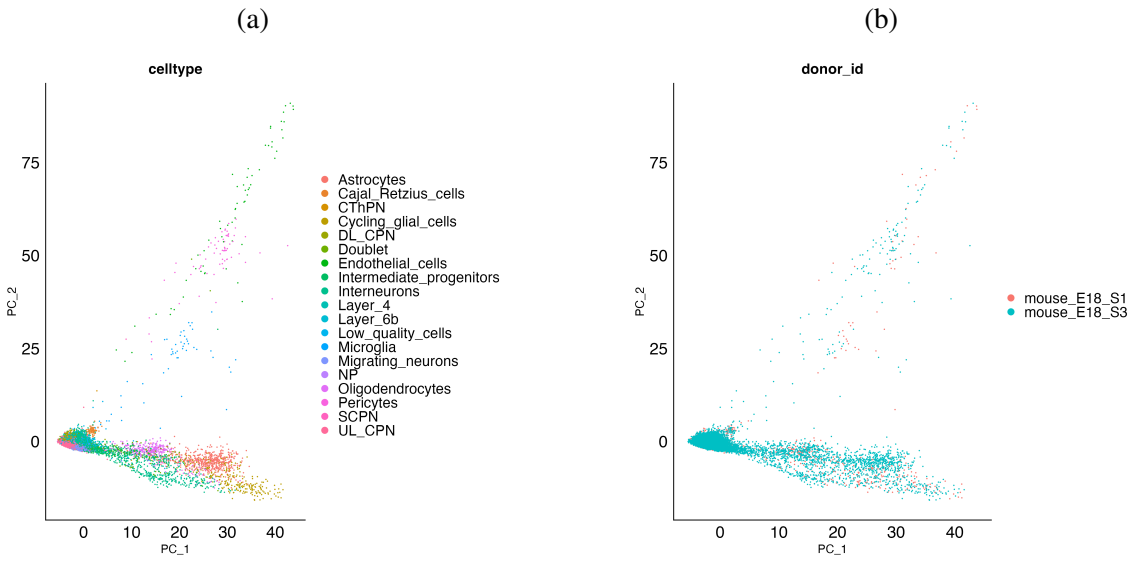


Figure A.25: PCA of the brain integrated data after CCA method. **a**, cell types. **b**, batches.

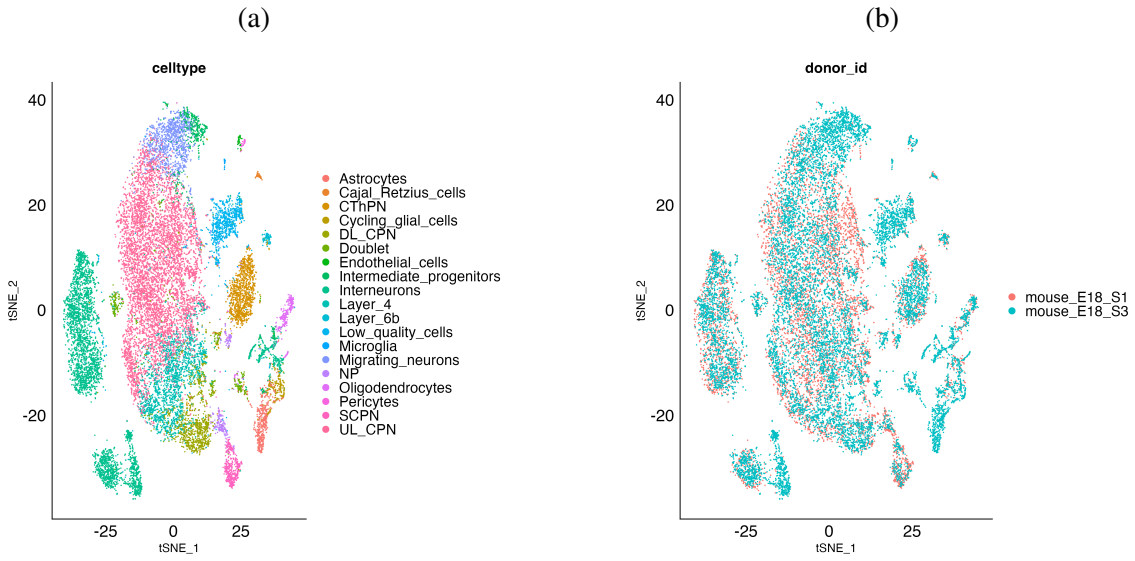


Figure A.26: t-SNE of the brain integrated data after fastMNN method. **a**, cell types. **b**, batches.

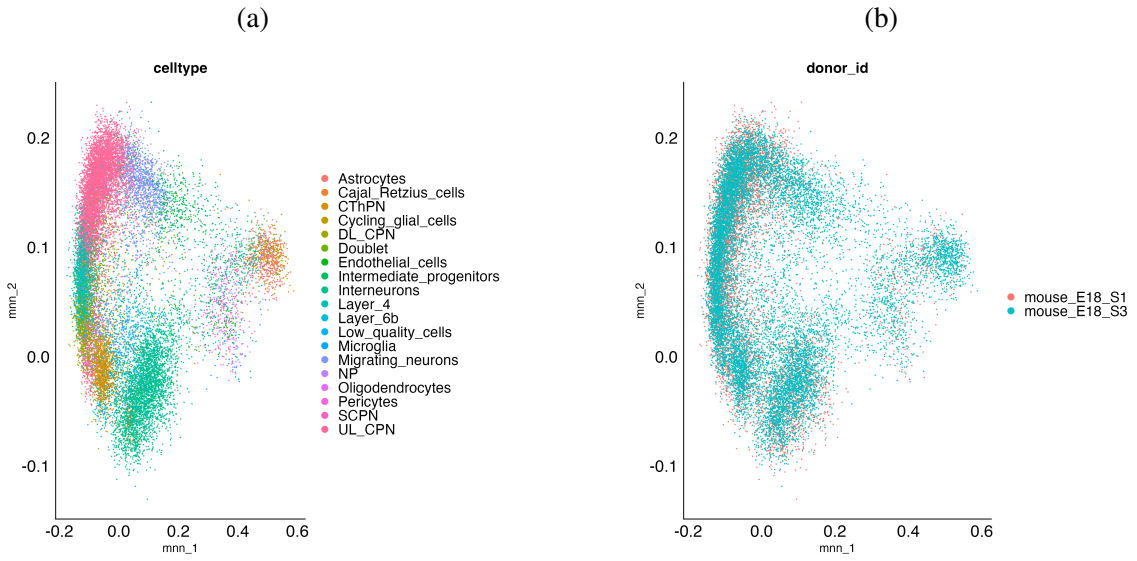


Figure A.27: PCA of the brain integrated data after fastMNN method. **a**, cell types. **b**, batches.

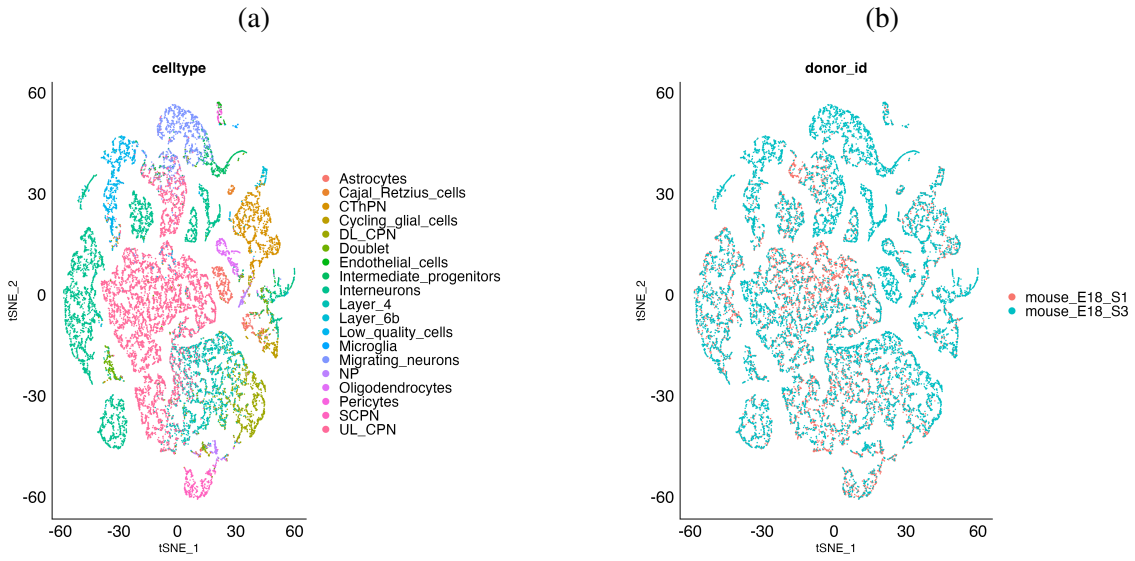


Figure A.28: t-SNE of the brain integrated data after conos method. **a**, cell types. **b**, batches.

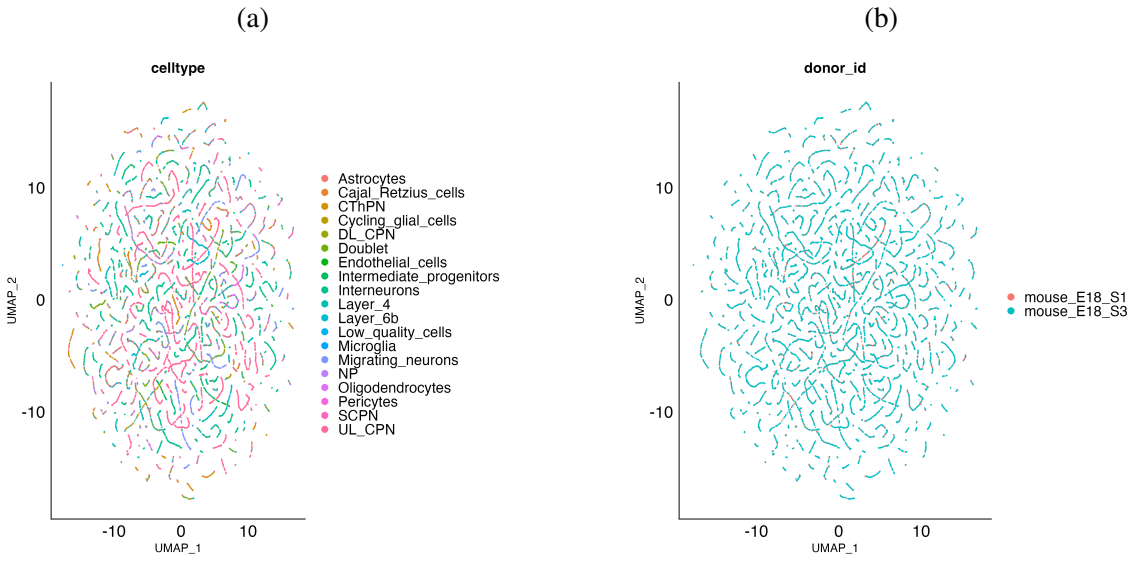


Figure A.29: UMAP of the brain integrated data after conos method. **a**, cell types. **b**, batches.

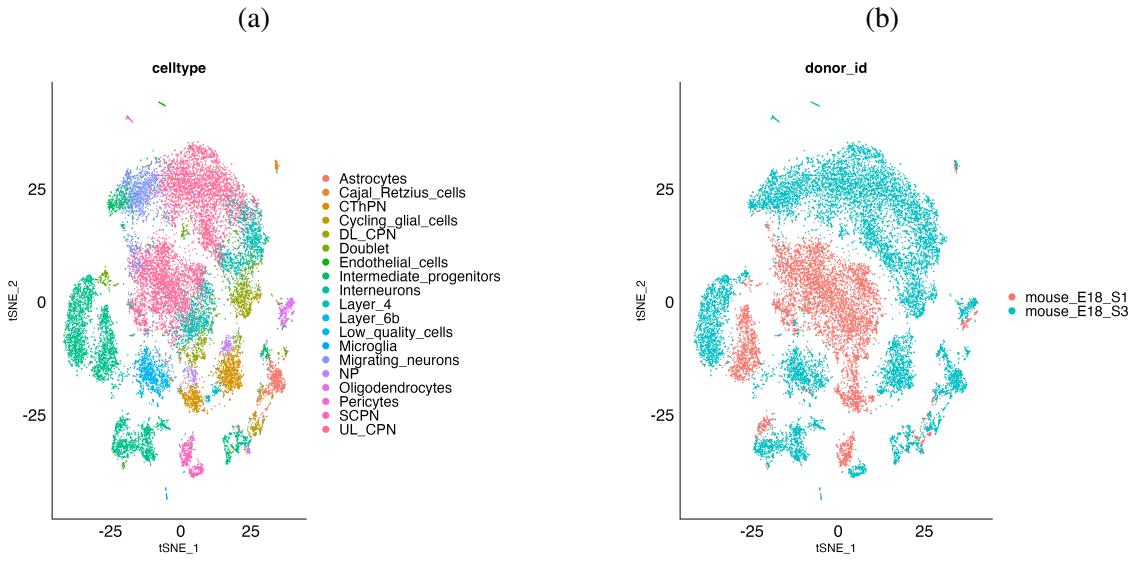


Figure A.30: t-SNE of the brain integrated data after combat method. **a**, cell types. **b**, batches.

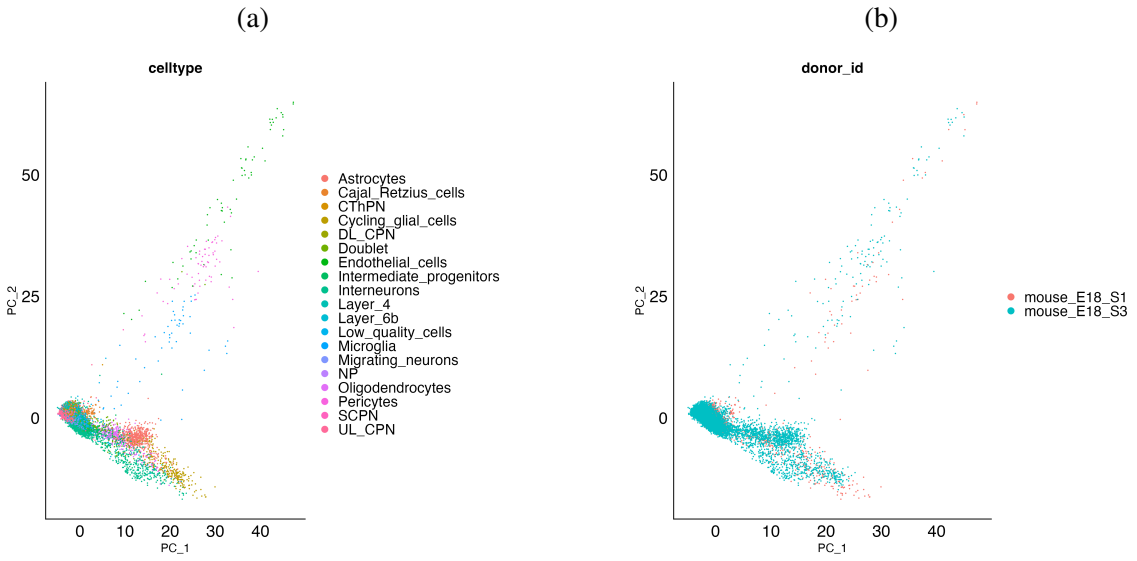


Figure A.31: PCA of the brain integrated data after combat method. **a**, cell types. **b**, batches.

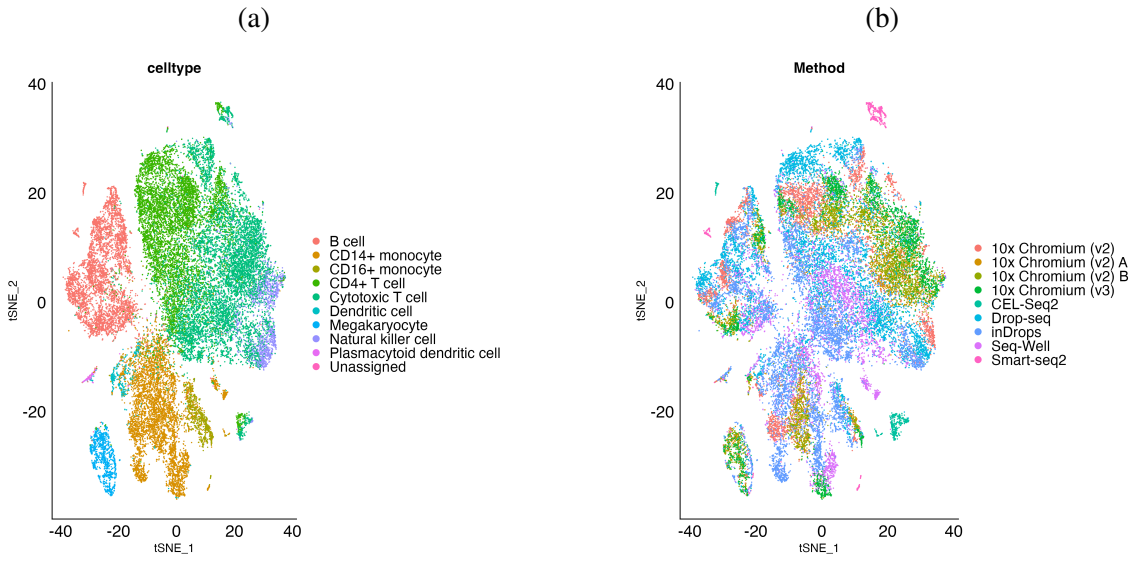


Figure A.32: t-SNE of the PBMC integrated data before batch removal. **a**, cell types. **b**, batches.

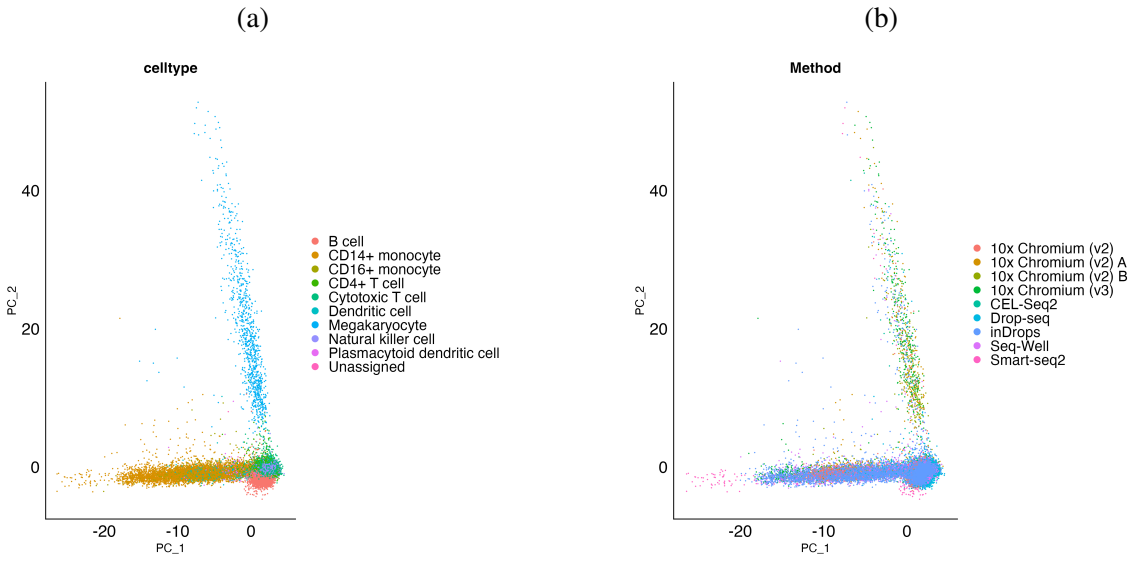


Figure A.33: PCA of the PBMC integrated data before batch removal. **a**, cell types. **b**, batches.

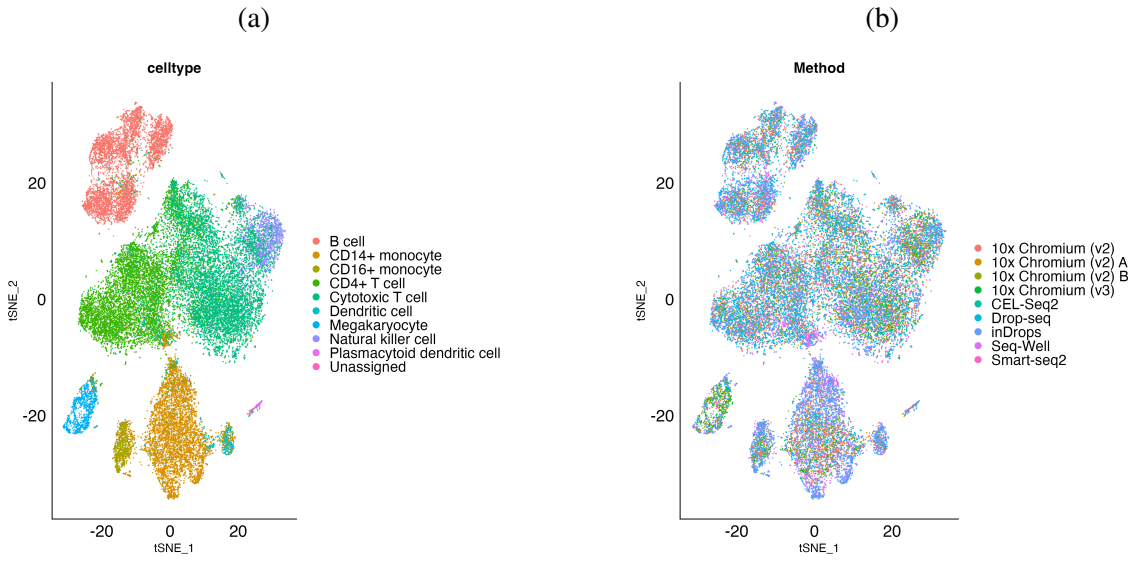


Figure A.34: t-SNE of the PBMC integrated data after harmony method. **a**, cell types. **b**, batches.

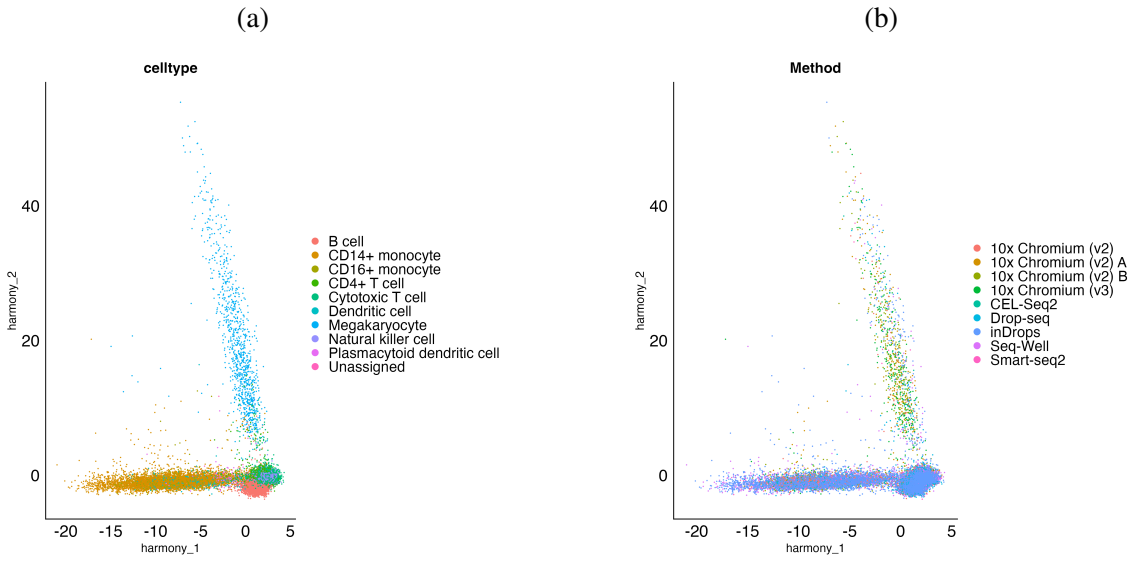


Figure A.35: PCA of the PBMC integrated data after harmony method. **a**, cell types. **b**, batches.

(a) (b)

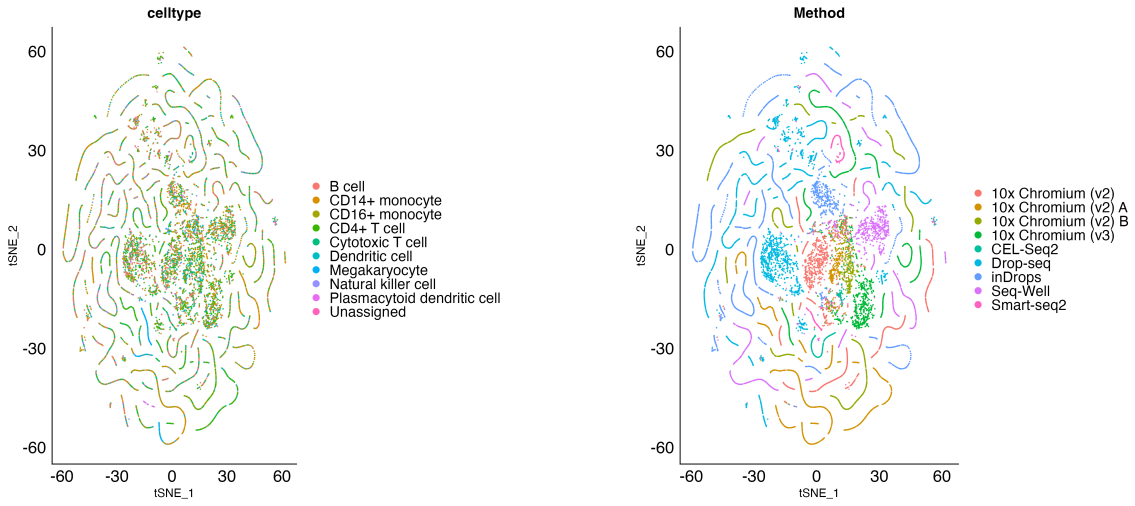


Figure A.36: t-SNE of the PBMC integrated data after limma method. **a**, cell types. **b**, batches.

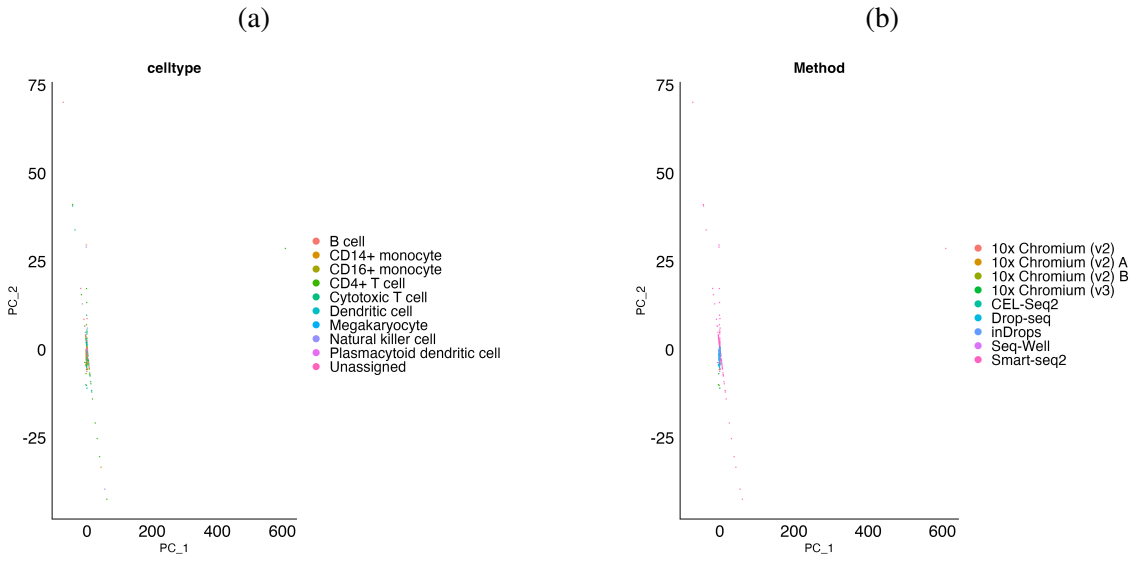


Figure A.37: PCA of the PBMC integrated data after limma method. **a**, cell types. **b**, batches.

(a) (b)

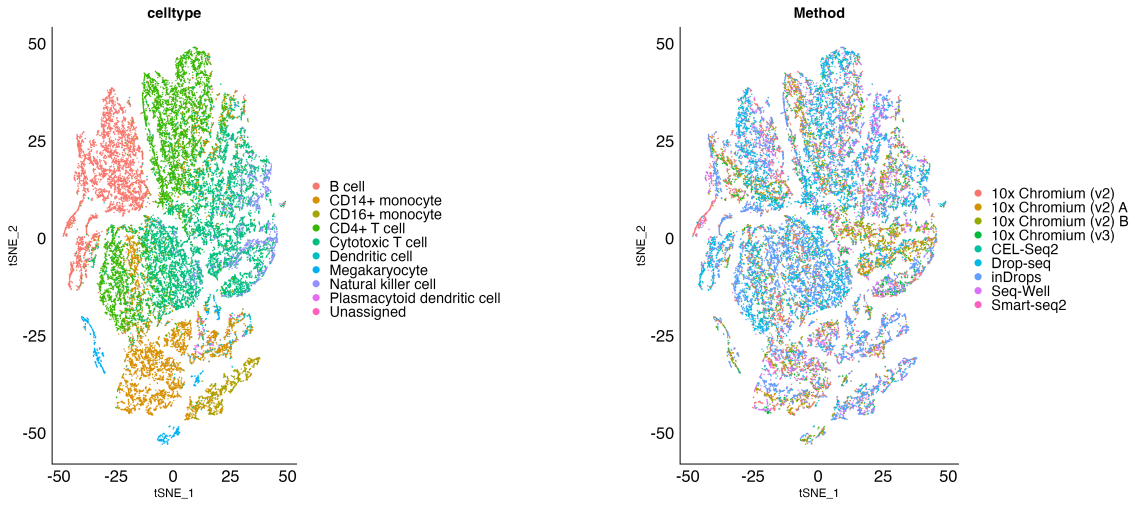


Figure A.38: t-SNE of the PBMC integrated data after liger method. **a**, cell types. **b**, batches.

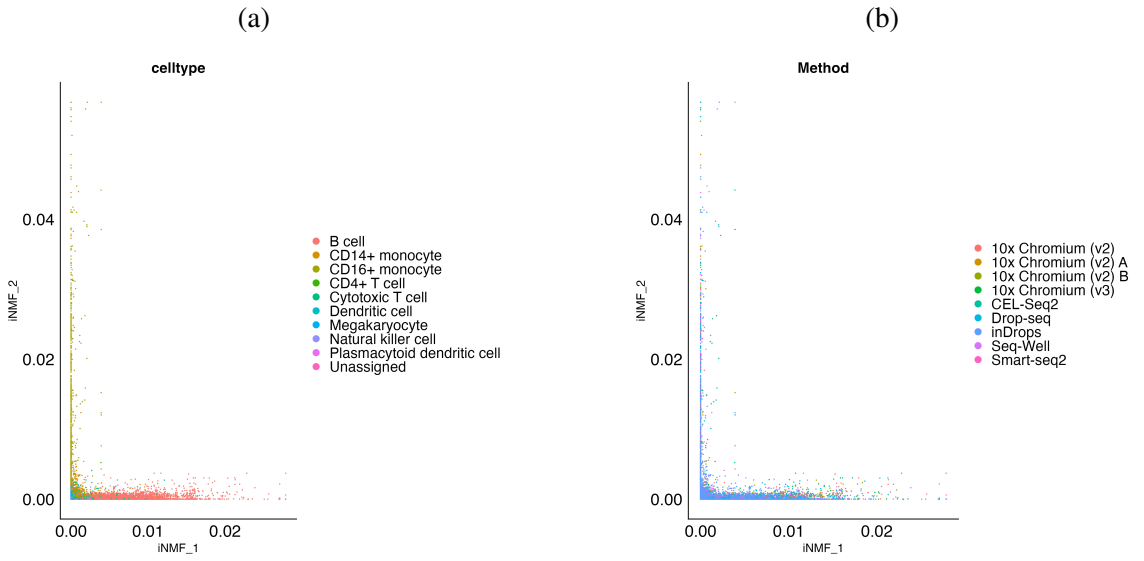


Figure A.39: PCA of the PBMC integrated data after liger method. **a**, cell types. **b**, batches.

(a) (b)

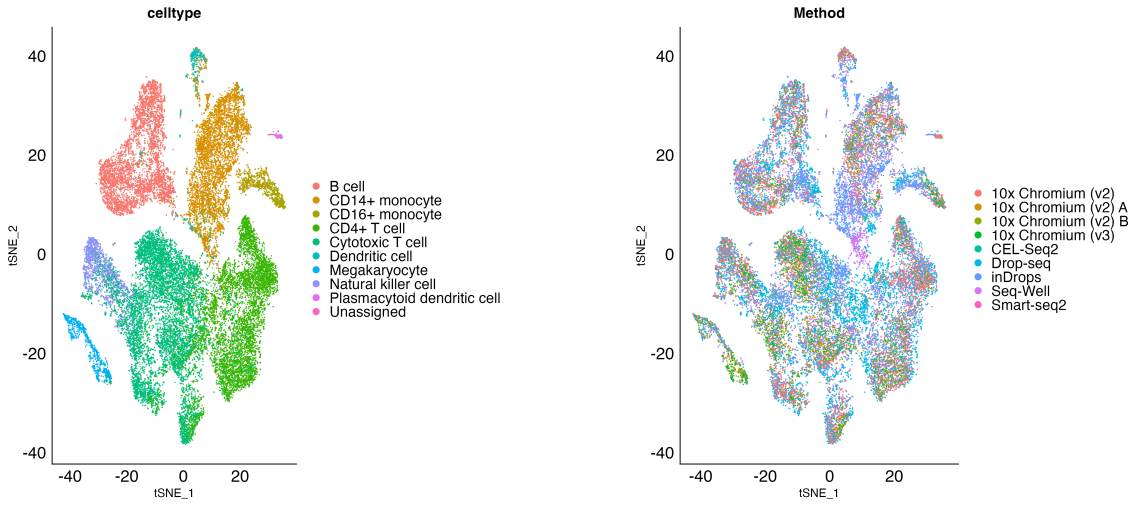


Figure A.40: t-SNE of the PBMC integrated data after CCA method. **a**, cell types. **b**, batches.

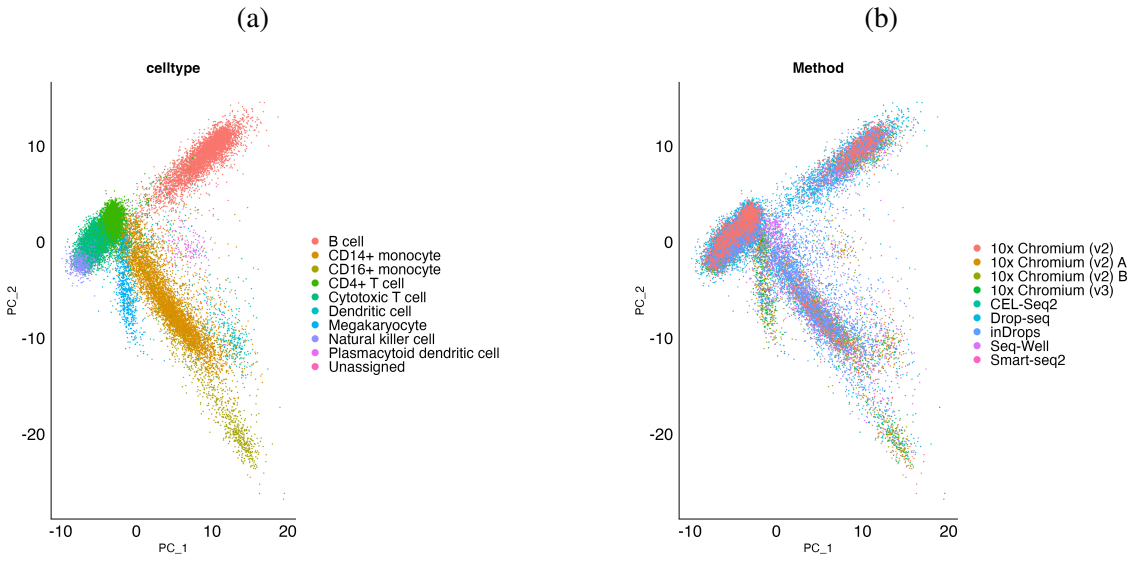


Figure A.41: PCA of the PBMC integrated data after CCA method. **a**, cell types. **b**, batches.

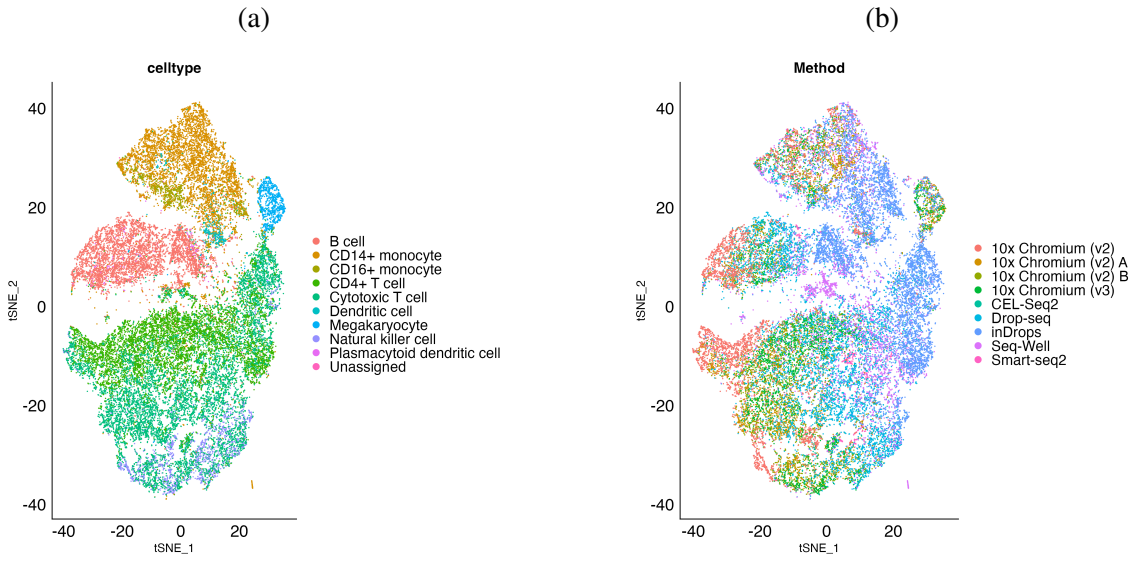


Figure A.42: t-SNE of the PBMC integrated data after fastMNN method. **a**, cell types. **b**, batches.

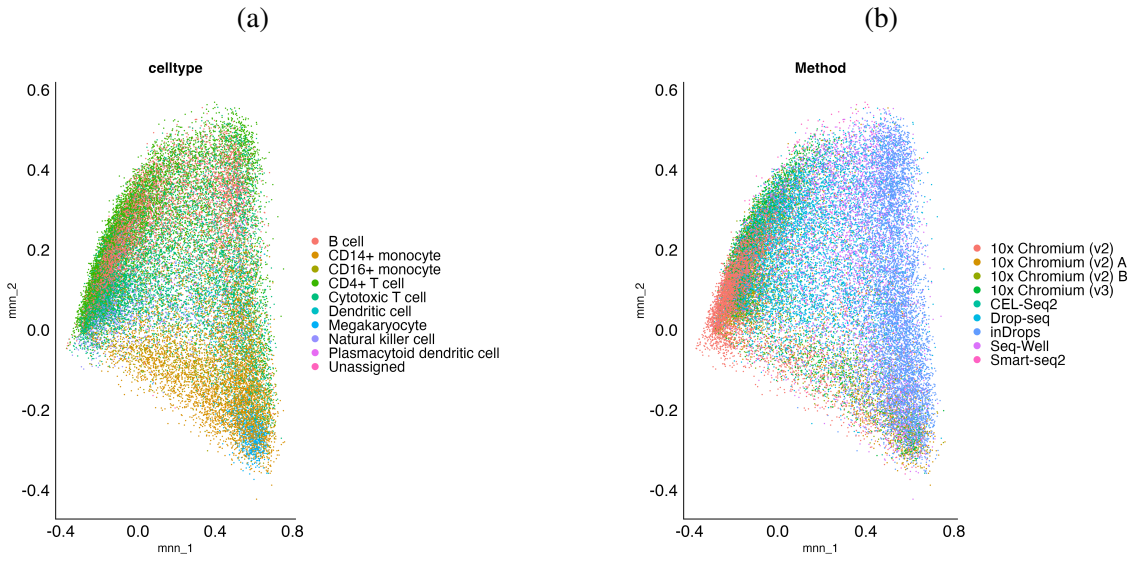


Figure A.43: PCA of the PBMC integrated data after fastMNN method. **a**, cell types. **b**, batches.

(a) (b)

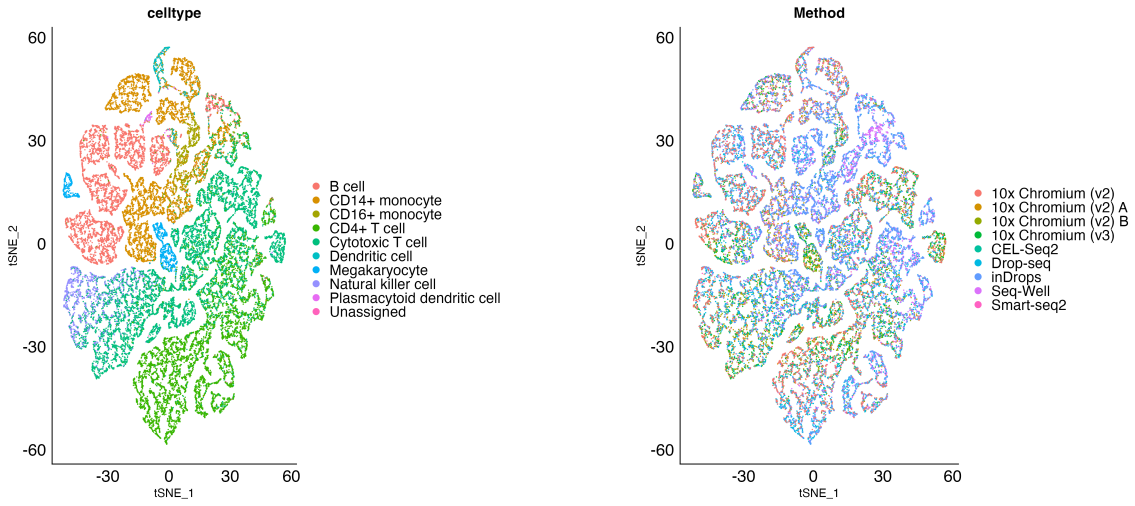


Figure A.44: t-SNE of the PBMC integrated data after conos method. **a**, cell types. **b**, batches.

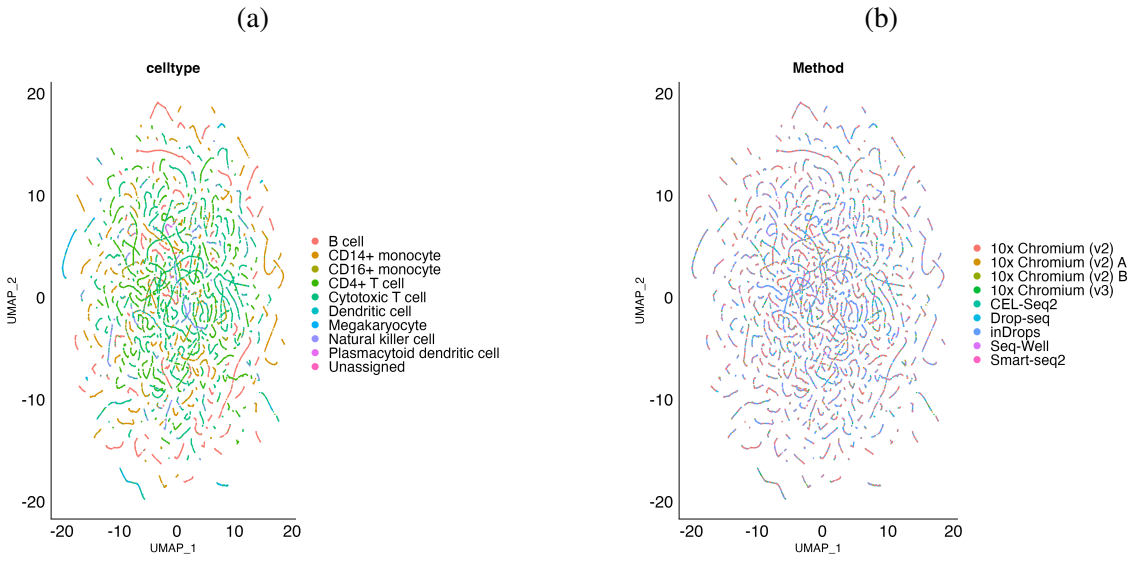


Figure A.45: UMAP of the PBMC integrated data after conos method. **a**, cell types. **b**, batches.

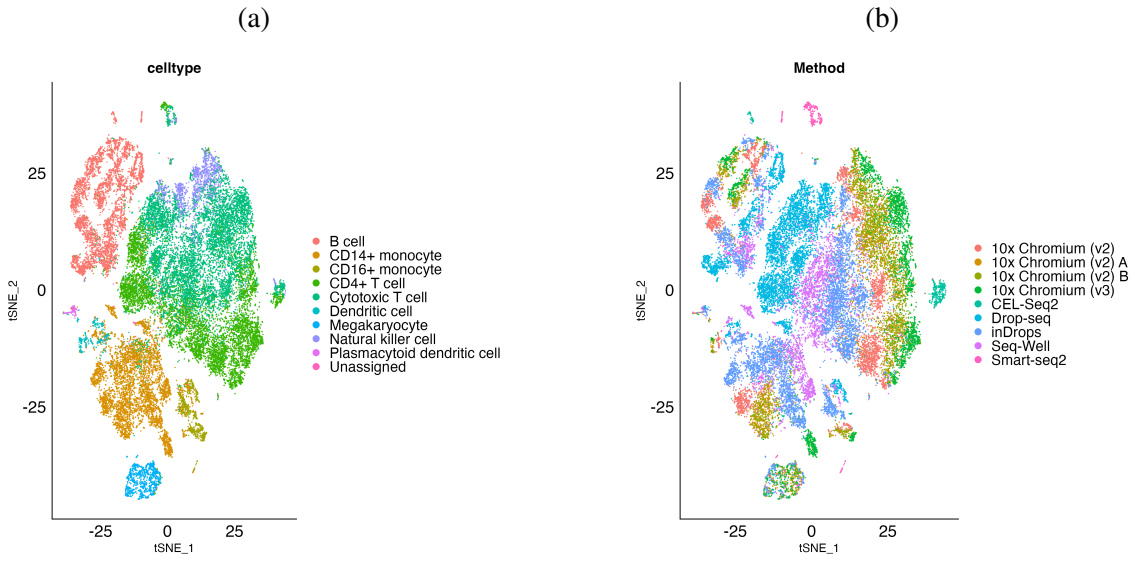


Figure A.46: t-SNE of the PBMC integrated data after combat method. **a**, cell types. **b**, batches.

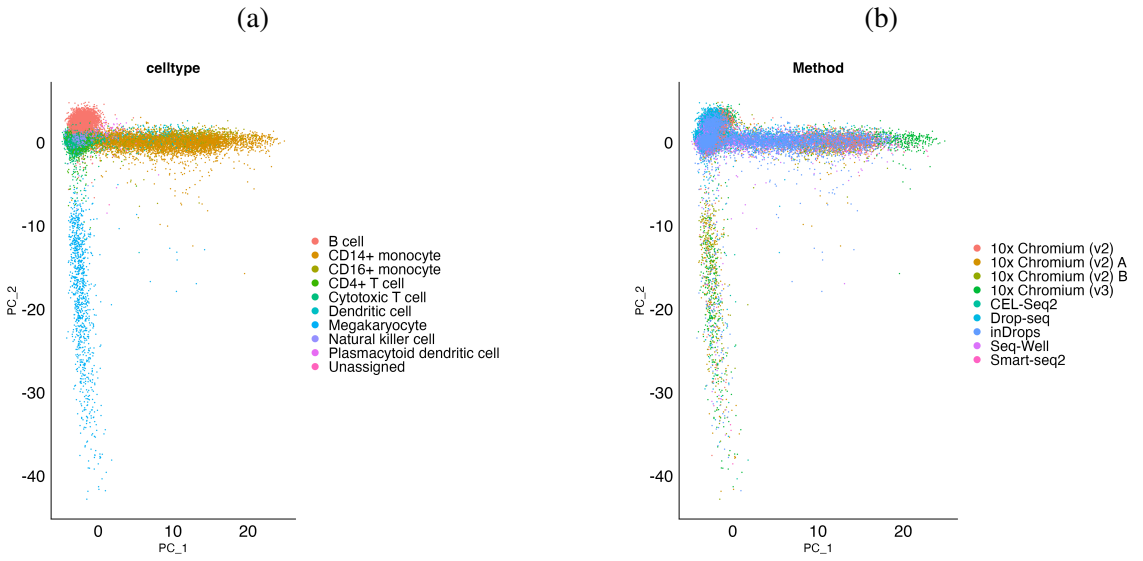


Figure A.47: PCA of the PBMC integrated data after combat method. **a**, cell types. **b**, batches.

Curriculum Vitae

Publications:

Marufkhani, Hanie, Behnam Jabbari Zadeh, SeyedReza Hashemirad, and Iman Sharifi. ”Sliding Mode Controller via Extended Kalman Filters For Mobile Robot.” In 2019 7th International Conference on Robotics and Mechatronics (ICRoM), pp. 167-174. IEEE, 2019.

Name: Behnam JabbariZadeh

Post-Secondary Education and Degrees:

Amirkabir University of Technology (Tehran Polytechnic)

Tehran, Iran

B.Sc. Electrical Engineering 2015 - 2020

University of Western Ontario

London, ON

Master's Studies in Computer Science, 2021-2022

Honours and Awards:

WGRS

2021-2022

Identified as an Exceptional Talent during undergraduate studies

2015 - 2020

Related Work Experience:

Teaching Assistant

The University of Western Ontario

2021 - 2022

Teaching Assistant

Amirkabir University of Technology

2018 - 2020