Project Report STA 426

ScRNAseq Integration Comparison

Jennifer Probst - ETH Zürich: 16703423
Supervision: Mark Robinson, Hubert Rehrauer
Autumn Semester 2020/2021

Contents

1 Abstract

Single-cell RNA sequencing (scRNA-seq) has become an easily accessible and commercially applied tool for single-cell transcriptional profiling in study designs across a variety of biological and clinical conditions. Tasks such as cell subpopulation identification or developmental trajectory analysis in complex and heterogeneous systems require the integration of multiple datasets across studies, technologies and even donors. In this work we explored three different integrative methods for scRNA-seq data, namely Seurat v3, Conos and Harmony. These methods tackle the integration task differently, either by a gene-based, graph-based or embedding-based approach. We applied the three methods on a peripheral blood mononuclear cell dataset and integration results were investigated by identification of cell subpopulations. We visually observed that all three methods performed batch correction and identified 20 clusters with the Seurat method, 14 different clusters with Conos and 16 clusters in the case of Harmony. As more and more integration methods are developed, benchmarking studies that provide guidance in method choice will become increasingly important. Successful data integration allowing for robust comparisons of heterogeneous tissues will lay the groundwork to answer many fundamental biological questions such as cell response to perturbation, disease, and evolution.

2 Introduction

The emerging diversity of single-cell RNA sequencing (scRNA-seq) workflows [1; 2; 3] combined with technological advances in throughput and decreasing sequencing costs [4; 5; 6; 7], have made high-throughput single-cell transcriptomics an easily accessible and commercially applicable tool utilized in various study designs [8; 9]. It facilitates unbiased single-cell transcriptional profiling of huge amounts of cells in complex and heterogeneous systems in a single experiment [2]. Obtained datasets allow for full transcriptomic characterization, identification of cell types and states [10; 11], as well as spatial tissue modelling [12; 13] or analysis of developmental trajectories and fate decisions [14].

As all of this can be done across a wide range of biological and clinical conditions, a large variety of single-cell RNA-seq datasets arise; also datasets with increasing complexity. Studies often include many samples generated with complex experimental designs from heterogeneous tissues of multiple individuals across various conditions and the data might even be measured in different laboratories with varying technologies. There are various collaborative projects such as the Human Cell Atlas [15] or Accelerating Medicines Partnership [16; 17; 18] that aim to combine these RNA-seq datasets into reference atlases, e.g. for primary human tissues or mouse organs [18].

This requires the integration of multiple datasets across studies, technologies and even donors, which is not straightforward due to batch effects. Batch effects arise from technical variation in the data that inadvertently affects batches (groups of cells) [19]. This technical variation can result from multiple experimental sources such as varying data gathering and evaluation protocols, different experimental labs, distinct media, flow cells and plates used or varied sequencing depth and read length [19]. These batch effects pose challenges to the joint analysis of different scRNA-seq, because the biological variation to be studied and the technical variation are interspersed. This means for the identification of shared cell subpopulations across datasets by clustering approaches that cells will on one hand cluster by cell subpopulation, but also by dataset specific conditions, making a reliable identification impossible.

Thus, there is urgent need for the development of integrative methods that are capable of removing nested batch effects, which can be complex and non-linear, such that data can be analysed in an unbiased manner. This is currently regarded as one of the biggest challenges in scRNA-seq data analysis [20; 21].

ScRNA-seq integration methods aim to be successful and robust in diverse, scalable settings. Therefore, such algorithms have to fulfill many requirements. They must be able to align both broad and fine-grained subpopulations with potentially unique responses to confounder variables and capture shifts in subpopulation frequencies across conditions [22]. Furthermore, optimal methods should be robust to different feature scales across conditions in that multiple datasets can be normalized differently and not directed by defined cell subpopulations [22].

These points should be met while the method should scale well to large datasets considering storage complexity and runtime [23].

In the last couple of years, many integration algorithms were developed. There are currently 107 different methods for scRNA-seq data integration listed in the scRNA-tools database [24]. Each of these methods aims to combine datasets into one integrated representation to facilitate comparative downstream analysis, but they do this completely differently e.g. in an integrated graph, joint embedding, or corrected feature space. Consequently, analysts nowadays face the problem of choosing an optimal integration method for their data analysis. Multiple comparative papers aim to discuss different integration approaches and evaluate their performance by benchmarking methods on various datasets. Some of them focus on simpler, low batch complexity integration tasks [25; 26], newer papers analyse the performance of algorithms on more complex integration tasks [19], e.g. the integration of organ atlases [15].

This work aims to explore the strengths and limitations of three different data integration methods for multi single-cell RNAseq datasets and compares the integration results on a peripheral blood mononuclear cells (PBMC) dataset by cell clustering. The compared integration methods are Seurat v3 [22], Harmony [23] and Conos [27], which are all fairly new methods developed in the past two years.

3 Methods

The following subsections describe the dataset (subsection 3.1), the theory and application of the integration methods with Seurat (subsection 3.2), Conos (subsection 3.3) and Harmony (subsection 3.4).

3.1 The dataset

The dataset is an unpublished, human single-cell RNA dataset with 10x count data [28] of peripheral blood mononuclear cells (PBMCs). It consists of PBMCs from 6 patients, where two of them are 'affected' mothers (samples p2 and ko1), two of them are 'unaffected mothers' (m2, het1) and another two are 'unaffected, unrelated donors' (wt1, wt2). The whole dataset consists of 54782 cells and for each of them read counts of 18024 genes have been measured. The data is stored in an SingleCellExperiment class, which facilitates storage and retrieval of spike-in information, dimensionality reduction coordinates and size factors for each cell, along with the usual metadata for genes and libraries [29].

An important step before starting with the integration procedures is the dataset preprocessing. Quality control metrics were calculated and the data normalized. Furthermore outliers were detected by a dataset-specific outlier threshold and removed, as were genes with too low read counts. Feature selection, dimensionality reduction with UMAP [30] and PCA [31] and doublet detection were also part of the preprocessing. After these steps we were left with 46266 cells and gene counts from 10081 genes after these steps (Figure 1). Next, one could start with single cell integration to remove batch effects in this cleaned and preprocessed dataset.

```
class: SingleCellExperiment
dim: 10081 46299
metadata(2): Samples scDblFinder.stats
assays(2): counts logcounts
rownames(10081): ENSG00000237491.AL669831.5 ENSG00000230368.FAM41C ...
  ENSG00000198727.MT-CYB ENSG00000273748.AL592183.1
rowData names(3): ID Symbol Type
colnames(46299): het1.AAACCTGAGATGTTAG-1 het1.AAACCTGAGGTGGGTT-1 ...
  wt2.TTTGTCATCGCCGTGA-1 wt2.TTTGTCATCTTGACGA-1
colData names(28): Sample Barcode ... scDblFinder.originAmbiguous
  cluster
reducedDimNames(2): PCA UMAP
altExpNames(0):
```

Figure 1: SingleCellExperiment containing the whole dataset including PCA and UMAP reduction.

3.2 Integration with Seurat

3.2.1 Method description (Seurat)

Seurat is an integration approach based on common sources of variation used to properly combine multiple large scRNA-seq datasets and furthermore identify shared subpopulations across datasets. It was proposed by Butler et al. in 2018 [22] and the Seurat v3 toolkit is available at the satijalab homepage [32].

The algorithm constructs shared subspaces of two batches by identifying conserved gene-gene correlation patterns across datasets through canonical correlation analysis [19]. After that, mutual nearest neighbours, called anchor points in the algorithm, are found across the two datasets. In the next step, two datasets are integrated into a reference hyperplane via projection vectors that were inferred from the anchor points. These projection vectors can also be used if one wants to integrate new cell populations that do not have mutual neighbours. To integrate more than two datasets, anchor points have to be computed pairwise. A hierarchical clustering that is based on the distance between the datasets is formed to determine the integration order of datasets. In this order the common batch corrected gene expression or open chromatin matrix is iteratively constructed. This matrix can then be used for downstream analysis such as identification of shared cell populations across data sets, discovery of cell types or to reconstruct transitional cell states and fate decisions jointly across data sets [32].

The following paragraphs aim to describe the individual steps in more detail; mathematical derivations can be found in the Methods section of the developers' paper [22].

For the Seurat integration workflow a list of at least two scRNA-seq datasets is needed. First, these datasets are preprocessed by the same gene expression value normalization and natural-log transformed. As in this method only highly variable genes are used for the alignment procedure, genes being highly variable in at least two datasets were selected for the multi-dataset alignment.

The second step tries to identify shared gene correlation structures conserved across datasets. This is done by canonical correlation analysis (CCA), first in a pairwise manner. CCA aims to find linear combinations of features across two datasets that are maximally correlated and to construct shared subspaces of two batches. This is done only on the subset of shared genes of both datasets, exhibiting high single-cell variation in at least one data set. For cases with more cells than genes, a variation of CCA, namely diagonal CCA, is employed, which assumes diagonal covariance matrices for each dataset. Canonical correlation vectors can then be found by singular value decomposition.

This two-set, canonical correlation can be extended to multi-set analysis (Multi-CCA) for the

integration of multiple data sets. Here we try to identify projection vectors that maximize the correlation between all datasets [36]. An iterative algorithm, with convergence threshold and a maximal number of iterations is used to solve for canonical correlation vectors. In this manner a set of canonical basis vectors exploiting high correlated variation is found, which embed all cells in a combined low-dimensional space.

As an optional third step, individual cells can be flagged for further analysis. In particular, there are cases, where CCA does not pick up certain cell populations as it only focusses on sources of variance common to both datasets. If one cell population is rare and non-overlapping between datasets, it might blend in with an abundant cluster and will not be described by the shared CCA structure. Also, abundant populations might not be present in one dataset and could therefore throw off the whole integration.

The vectors obtained from the CCA are by definition correlated between datasets, but not necessarily aligned. Therefore, in the next step of the workflow, CCA basis vectors must be aligned across datasets to consequently define a single, integrated low-dimensional space. This is done by finding mutual nearest neighbors across two datasets that in the algorithm are called anchor points and integrating these two datasets into a reference hyperplane using projection vectors that were inferred from these anchor points. Briefly, metagenes are obtained from the basis vectors and a mapping between these metagenes is determined using dynamic time warping. Consequently, vectors get locally stretched or compressed during alignment, what corrects changes in population density. A single, low-dimensional and aligned space is constructed by application of the procedure to each basis vector pair.

To integrate more than two datasets, a pairwise computation of anchor points is required. The datasets then get integrated based on the order of a hierarchical clustering which depends on the distance between datasets. We end up with a common batch corrected gene expression or open chromatin matrix that is defined by the aligned canonical basis vectors.

In a final step the aligned canonical basis vectors can be utilized for downstream analysis. The Seurat developers found that the results of the downstream analysis were quite robust to the exact choice of number of basis vectors used; as a default 20 are utilized [22]. An example of integrated downstream analysis is the identification of discrete subpopulations through clustering. The Seurat toolkit includes its own functions for cluster analysis. The clustering is graph-based and done in two steps by first constructing a KNN graph and then using modularity optimization techniques such as the Louvain algorithm (default) or SLM, to iteratively form grouped cell clusters while optimizing the standard modularity function. Furthermore, developmental processes can be reconstructed by trajectory building or changes in population density and gene expression can be explored.

3.2.2 Method application (Seurat)

Dataset preprocessing and integration with Seurat (version 3.1.1) was run in R according to the developers' integration tutorial [33] for dataset processing and integration. Furthermore, a clustering tutorial of PBMC cells [34] was followed for cell clustering. Code for the whole analysis can be found on the project website [35].

First, the data was read in and a Seurat object was created. Seurat integrates the individual datasets in a pairwise order determined by a hierarchical clustering using anchor points. For this to work, the Seurat object had to be split into the different original datasets as well as the data normalized and scaled. For each dataset the 2000 most variable features (genes) were identified and pairwise integration anchors were calculated. We did not use the optional step (previously referred to as step two in the workflow) to identify rare populations. Instead, the list of individual datasets was directly integrated into a reference hyperplane via projection vectors inferred by the before determined anchors using the IntegrateData function, where a default of 1:30 dims was used.

The returned batch corrected gene expression matrix could then be used for downstream analysis. The success of the integration was visually explored in an Uniform Manifold Approximation and Projection (UMAP) plot [30]. Furthermore, we used the in the Seurat package provided functions FindNeighbors and FindClusters to detect PBMC cell types in a clustering analysis. This was done on the original uncorrected matrix and on the integrated matrix. For the clustering, a resolution parameter of 0.8 was chosen and the clustering was visualized in a two-dimensional space using the UMAP algorithm.

3.3 Integration with Conos

3.3.1 Method description (Conos)

Conos (Clustering of network of samples) is a graph-based integration approach for multiple large single-cell RNA-seq datasets, proposed by Barkas et al. in Aug 2019 [27]. More information about the package can be found at the authors' github [37]. The approach makes use of various plausible inter- and intra-sample mappings to build a global graph capturing likely relationships between all measured cells. The resulting corrected neighbourhood graph can then be used to produce a clustered grouping to identify recurrent cell clusters of similar cell subpopulations in a way robust to inter-sample variation.

Conos processing can be divided into a three-step process. The first step consists of preprocessing and normalization, while in the second phase an initial mapping between cells of different datasets is determined (weighted inter-sample cell-to-cell links). This is done by pairwise projection of samples onto a common space and analysing the cell-cell similarity after

dimensionality reduction. In a final step a joint graph is constructed by including inter-batch connections reduced by a mutual nearest neighbour approach and lower-weight intra-sample edges from within-sample k-nearest neighbors to integrate all cells into the graph [38].

In the following, the individual steps will be described in more detail; explained with some more mathematical notation in Barkas et al. [27].

For the filtering and normalization step, common single-cell data processing packages like Seurat or pagoda2 are used in the Conos workflow. Thereby, cell filtering, library size normalization and identification of overdispersed genes is applied to each individual dataset. Additionally, pagoda2 supports variance normalization.

The second step is in simplified terms a pairwise dataset alignment, where pairwise initial mappings between cells of individual datasets are determined. Therefore, pairs of cell samples get projected onto a common space which only includes overdispersed genes, common to both cells. Furthermore, dimensionality reduction through common principle component analysis (CPCA) or joint non-negative matrix factorization (JNMF) is applied on these sets of common, overdispersed genes, resulting in one reduced projection matrix for each cell in a pairwise comparison. The cell-cell similarity is defined as the maximal Pearson linear correlation of row-wise comparisons of the cell projection matrices. These similarities are then used as weights of the connections across datasets, in other words the weight of the edge between the two compared cells in the graph.
It is worth noting that the inter-sample edge values differ depending on the choice of rotation space and neighbour mapping strategy and might be error-prone due to the alignment method of sample pairs.

The third step reduces the total number of connections in the graph by a (mutual) nearest neighbour approach; their edge-weight being the in step two computed cell-cell similarity in the case of inter-sample connections and the by a constant downweighed cell-cell similarity in the case of intra-sample connections. This reduces contributions of intra-sample edges, as cells from the same community (same cell type) will have much higher similarities.

Integration of multiple datasets with Conos results in a single joint nearest neighbour graph, where each cell is represented by a node and edges connecting combinations of inter- and intra-samples. The obtained graph can then be used for downstream analysis as in our case the detection of subpopulations across different samples by clustering. The clustering follows the idea that cells from the same subpopulation will likely map to each other in the many pairwise comparisons and form cliques (graph communities). These can be detected by commonly used community detection methods as the proposed walktrap.communities function from the igraph package or the Leiden community detection method. The use of other community detection methods could result in a more sensitive cluster or, if wished,

hierarchical subpopulation view. The link between hierarchical clustering of cells and graph communities is also discussed in the paper of Barkas et al. [27], where they state that cutting the tree hierarchy at varying levels is equal to the difference between separating or joining subpopulations.

3.3.2 Method application (Conos)

The Conos (version 1.3.1) integration was adapted to this vignette [38] by the developers of Conos. Dataset preprocessing within the Conos workflow is supported with Seurat or Pagoda2. We chose preprocessing with Seurat to simplify result comparison to the other methods and followed the vignette in [39]. Code for the whole analysis can be found on the project website [35].

The workflow started by reading in the data and setting up a Seurat object. Similar to the Seurat workflow, the combined SingleCellExperiment had to be split into a list of the six original datasets. These were each normalized, variable features were found, data was scaled and PCs were computed following the Seurat workflow. In addition, a T-Distributed Stochastic Neighbor Embedding (t-SNE) [40], used by some of the standard Conos functions, was calculated.

As a next step, a Conos object was set up and was combined to one single joint nearest neighbour graph in the Conos integration step (buildGraph function). Integration was done in the PCs space with 30 components and a default neighbourhood size of 15. This integrated graph could then be used for downstream analysis. We utilized the Leiden community detection to detect PBMC cell types in a clustering analysis with a resolution parameter of 0.8. The clustering was visualized in a two-dimensional space using the largeVis algorithm [41], which is default for Conos. Furthermore, cells were plotted in the original t-SNE embedding, split by dataset and cell labels obtained from the joint clustering were added. Thereof, the individual cell cluster compositions of the six samples could be visualized in a barplot. Conos would offer many more visualization options to explore such as visualizing gene expression paths or hierarchical community structure. For more information visit following vignette [38].

3.4 Integration with Harmony

3.4.1 Method description (Harmony)

The Harmony algorithm tries to project all cells of multiple large single-cell RNA-seq datasets into a reduced, joint embedding, where cells should cluster only by cell type and not by dataset-specific conditions. The algorithm was proposed by Korsunsky et al. [23] in Nov 2019 and is available as an R package on github [42] with standalone functions that can be integrated into Seurat [43] pipeline analyses.

Concerning runtime and storage complexity, the developers of Harmony found that their algorithm scales well to large datasets from 30'000 to 500'000 cells (runtime: 4min on 30k up to 68min on 500k cells, maximum memory usage: 0.9 GB on 30k up to 7.2 GB on 500k cells) [23]. They therefore conclude that Harmony is capable of analysing large datasets of up to 10^6 cells on a personal computer. As mentioned before, Harmony attempts to embed cells into a joint space with reduced dimensionality.

In short, the algorithm initializes all cells in a low dimensional, linear PCA (principal component) space [19]. Two main concepts are then used to produce an integrated embedding over all datasets. In a first step, maximum diversity clustering, a soft k-means clustering with a diversity maximizing regularization term, is utilized. This method tries to assign cells into multi-dataset clusters, to push clusters with same cells apart, meanwhile maximizing diversity of datasets within each cluster by the penalty term. The second concept is a mixture model based linear batch correction. During this process cell-specific linear correction functions are calculated for each cell from global cluster centroids, as well as dataset-specific centroids for each cluster and weighted by the cell's soft cluster assignments made in the previous step and applied to get adjusted coordinates for each cell. The steps of maximum diversity clustering and mixture model based linear batch correction are repeated until convergence, in other words till the assignment of cells to clusters does not change anymore. Thereby should the dependence between dataset and cluster assignment decrease in each iteration. When the algorithm has converged, Harmony outputs a corrected embedding, which can then be used for downstream analysis such as cluster visualization or trajectory analysis.

The following section describes the iterated Harmony algorithm of maximum diversity clustering and mixture model based linear batch correction in more detail. More detailing mathematical derivations and formulas are to be found in the Methods section of the original paper by Korsunsky et al. [23].

Algorithm 1 Harmony
  function HARMONIZE (Z, ϕ)
    $\hat{Z} \leftarrow Z$
    repeat
      $R \leftarrow$ CLUSTER ($\hat{Z}$, ϕ)
      $\hat{Z} \leftarrow$ CORRECT (Z, R, ϕ)
    until convergence
    return $\hat{Z}$

Figure 2: Pseudocode of the Harmony algorithm

Figure 2 shows the Harmony algorithm in pseudocode. The function is initialized with a PCA embedding of all cells (Z) normalized by library size and their batch assignments (ϕ). The

output of the algorithm is the batch corrected, integrated embedding ($\hat{Z}$). The algorithm itself iterates between the two steps 'Maximum diversity clustering' (CLUSTER(($\hat{Z}, \phi$))) and 'batch effect correction' by a linear mixture model (CORRECT(($\hat{Z}, R, \phi$)) until convergence. It is worth noticing that the clustering step uses the batch corrected embedding $\hat{Z}$ from the last iteration to compute a soft assignment of cells to clusters, encoded in the matrix R. The batch effect correction step however is restricted to a linear model of the original embedding by only using the original Z and the updated soft clusters as input.

The algorithm step CLUSTER($\hat{Z}, \phi$) is a newly developed soft k-means clustering with a diversity maximizing regularization term. The clustering algorithm aims to minimize an objective function composed by a normal soft k-means clustering objective function (obtained following [44]) plus a diversity maximizing regularization term. Multiple batch penalty terms are allowed to account for multiple technical or biological factors and they are simply additive in the objective function.
This clustering approach thereby allows each cell to be assigned to multiple clusters because of the soft clustering and favors clusters with cells from multiple datasets as the penalty term ensures maximal diversity of datasets within each cluster. The objective function is optimized by an expectation-maximization framework that iterates between cluster assignment and centroid estimation. It returns a soft cluster assignment matrix, that consists of probability distributions for each cell belonging to the different clusters, where each probability distribution is potentially unique as cells can be assigned to multiple, different clusters.

The algorithm step (CORRECT(($\hat{Z}, R, \phi$)) aims to correct batch effects from the batch-diverse clusters defined in the CLUSTER step by a linear mixture model. This is addressed by a variation of the original mixture of experts model by Jordan and Jacobs [45]. For a given clustering, Harmony calculates a global centroid for each cluster, as well as dataset-specific centroids for each cluster, from which cluster specific correction factors can be computed. This is done by defining a Gaussian probability distribution for response variables (input embedding of cells) conditioned on cluster. These distributions can be solved for the parameters of the linear model named W_k in the original paper for each cluster independently. The batch specific correction factors for the i-th cell can be obtained from the i-th column in W_k: a batch-dependent intercept stored in $W_k[1, i]$ and its batch dependent terms represented by the remaining rows $W_k[2 : B, i]$. Since each cell is probabilistically assigned to multiple clusters, its cell-specific linear correction factor is a weighted average of the cluster specific correction factors and therefore unique. Each cell embedding is then corrected by these factors.

The output of the CORRECT step is a matrix with an adjusted embedding for each cell. Once convergence is reached, the algorithm outputs its most recently batch corrected and integrated embedding, which can then be used for downstream analysis.

3.4.2 Method application (Harmony)

The Harmony (version 1.0) integration was done within the Seurat workflow in R and followed steps from a vignette published by the developers of the algorithm [43]. R code for the whole analysis can be found on the project website [35].

The workflow started by reading in the data and setting up a Seurat object, identically to the Seurat and Conos workflow. In contrast, the integration is then done on one single Seurat object combining all datasets that have to be integrated. Preprocessing such as normalizing, scaling and finding variable features, as well as PCA are then run on the Seurat object with Seurat functions.

Subsequently, to run the Harmony integration, one simply has to pass the Seurat object into the RunHarmony function and specify the variable to integrate out, in our case the samples (in the code referred to as 'Sample'), and the number of max.interations that should be run. The returned embedded alignments can then be used for downstream analysis. We explored a clustering analysis to find PBMC cell types. Clustering was done with the Seurat functions FindNeighbors and FindClusters, where the resolution parameter 0.8 was chosen. The clustering was visualized in a two dimensional space using the UMAP algorithm [30].

4 Results

As described in the Methods part, three different integration methods were used to align multiple single-cell RNA datasets and subsequently identify cell population clusters. The results of the integration approaches are discussed in the following subsections in detail. The full analysis code in R (version 4.0.2) and higher quality figures are available on the project's webpage [35] and github [46].

4.1 Integration with Seurat

After setting up a Seurat object and splitting it into a list of the original six samples, we identified the 2000 most variable features for each sample dataset. The most variable features of the first sample (het1) can be seen in the Appendix in Figure 19.

In the last iteration of the FindIntegrationAnchors function 17359 anchors were found and 7623 anchors were retained and used for integration.

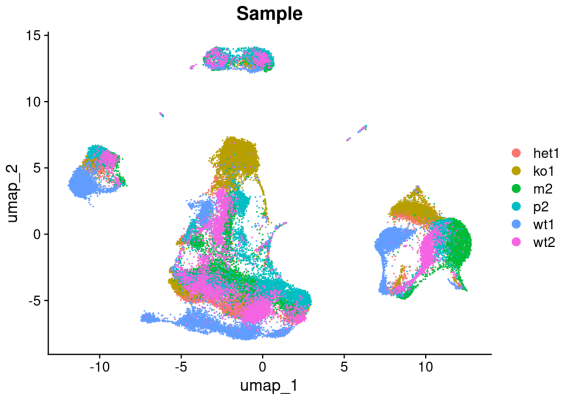

Figure 3: Plot of all cells in the initial, uncorrected UMAP embedding. Colors represent the samples.

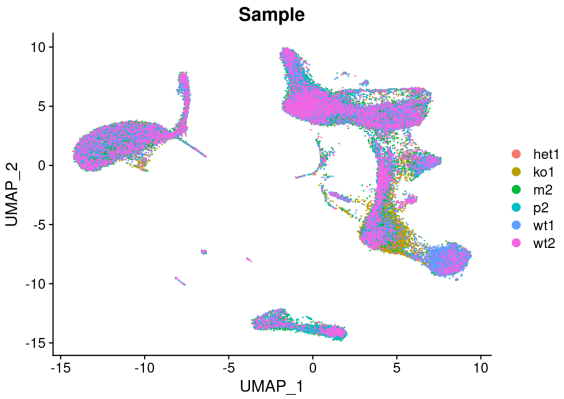

Figure 4: Plot of all cells in the Seurat integrated UMAP embedding. Colors represent the samples.

To see how well our integration worked, UMAP plots of all cells in the initial embedding (Figure 3) and in the Seurat integrated embedding (Figure 4) were visually inspected. One can observe that cells are not well integrated in the original embedding and cluster by sample. In the Seurat integrated UMAP plots the batch effect has been removed and individual samples seem well integrated.

We aim to identify different clusters from the integrated dataset in the UMAP embedding. Comparing the identified clusters from the uncorrected embedding to the ones found in the Seurat aligned space, we find 20 clusters identified in both the Seurat corrected embedding (Figure 6), as well as the original embedding (Figure 5).

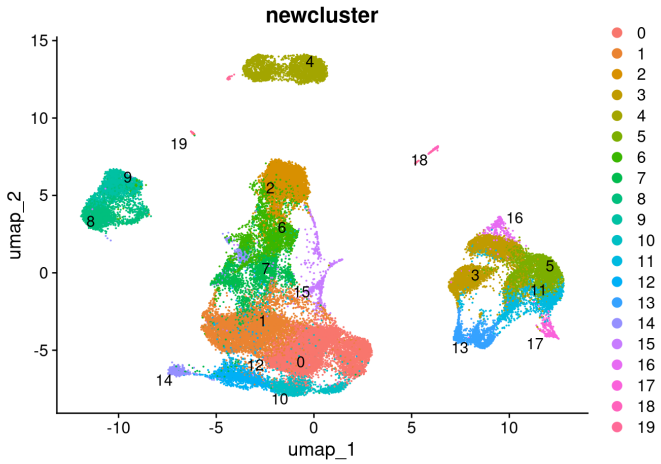

Figure 5: Clusters identified from the uncorrected, initial UMAP embedding. Colors represent the 20 recognized clusters.

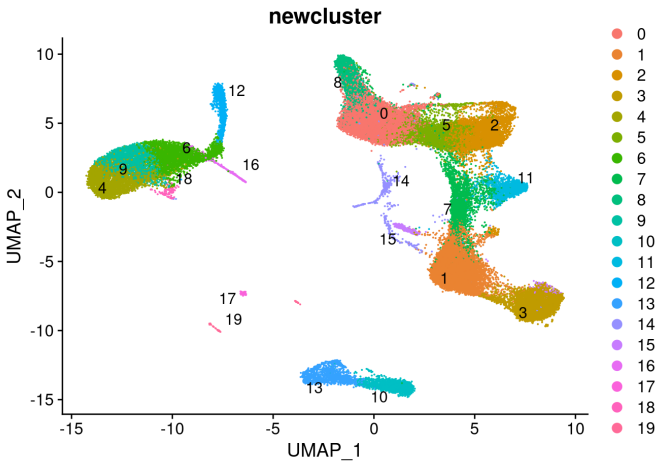

Figure 6: Clusters identified from the by Seurat integrated UMAP embedding. Colors represent the 20 recognized clusters.

4.2 Integration with Conos

According to the Conos integration workflow with Seurat preproccessing [39], a Seurat object was set up, preprocessed and used to construct a Conos object. During the Conos integration step a single joint nearest neighbour graph in a PCA space with 30 components was built. The estimated amount of variance explained by successive PCs is visualized in Figure 20 in the Appendix. We observe that the first three PCs already explain a large fraction of the variance.

To find out how well the integration worked, the integrated graph is plotted in PCA space (Figure 7). We observe that batch effects have been removed and the graph captures the population structure irrespective of the origin of each cell. This integrated graph could then be used for cluster identification. The exact number of clusters recognized varies slightly in parallelized computational settings due to path dependency of random number subsequences used in sub-processes (even when fixing initial random number seeds). Therefore we find between 13 and 15 clusters with the Leiden community detection method (Figure 8).

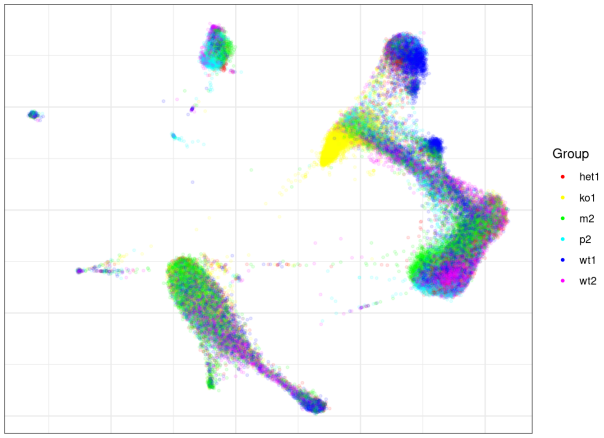

Figure 7: Plot of all cells in the Conos-integrated PC embedding visualized with LargeVis. Different colors represent the six samples.

Furthermore, cells were plotted in the original t-SNE embedding split by dataset and cell labels obtained from the joint clustering were added (Figure 9). We observe that different amounts of the individual cell types are present in the samples. For further analysis, compositions of individual clusters were visualized by dataset (Figure 10). Many interesting things can be noticed in this plot. The number of cells belonging to one cluster has a big range from many 1000 cells in cluster 1 to less than ten cells in cluster 14. In almost every cluster cells from all samples were observed, except for cluster 14 which only appeared in samples ko1 and wt1 and cluster 12 which was not present in het1. It is notable that most of the cells in ko1 belong to cluster 2. Furthermore, there is no obvious trend, judged by eye, that wildtype sample cells seemed to be differently distributed than the remaining samples het1, m2 and p2.

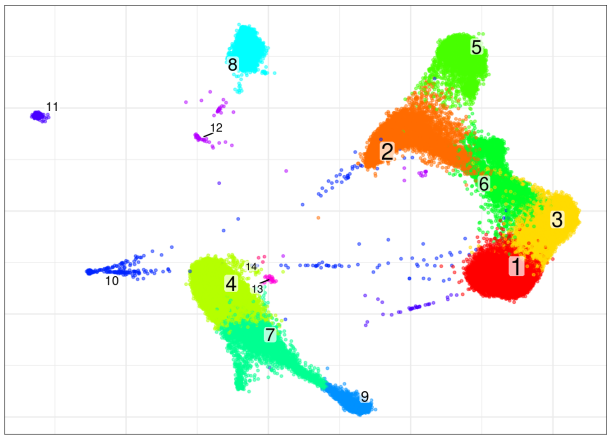

Figure 8: Clusters identified from the Conos integrated PC embedding. The colors represent the 14 recognized cell clusters.

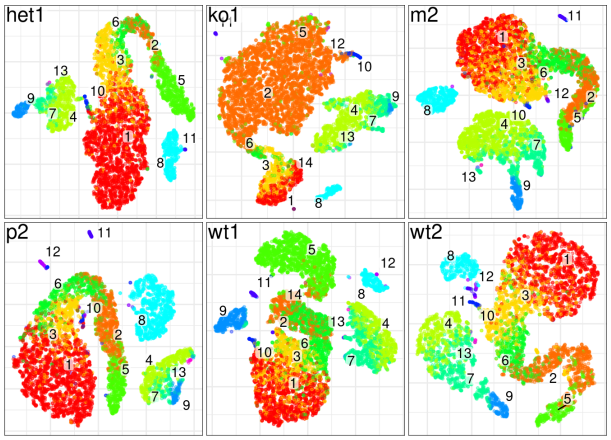

Figure 9: Plot of cells in the original t-SNE embedding divided by sample. Individual cell labels obtained from the joint clustering are added by color coding.

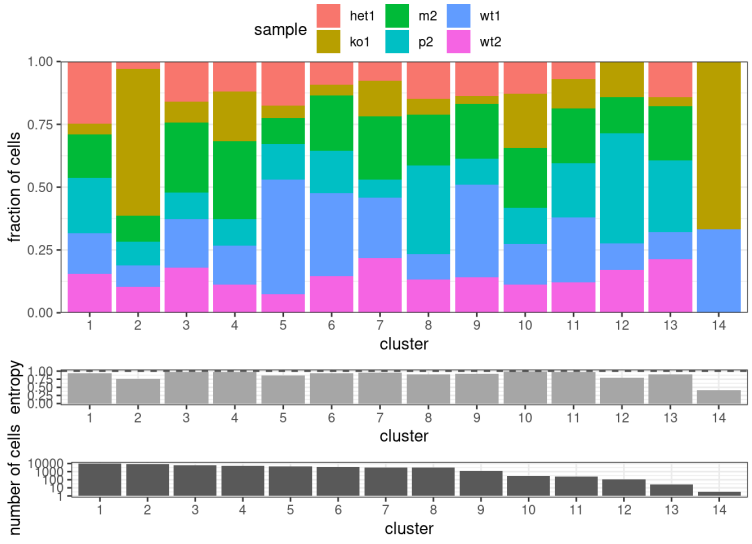

Figure 10: Composition of the individual clusters identified by the Harmony method.

4.3 Integration with Harmony

After preprocessing of the Seurat object, we first had a look at the dataset and its uncorrected PCs. In the Dimension plot (Figure 11) we can see all cells colored by sample scattered by the first two uncorrected PCs. The cells don't seem to be well integrated and cluster by sample. This is even more the case for Figure 12 where we observe that embedding values for the first PC vary across samples.

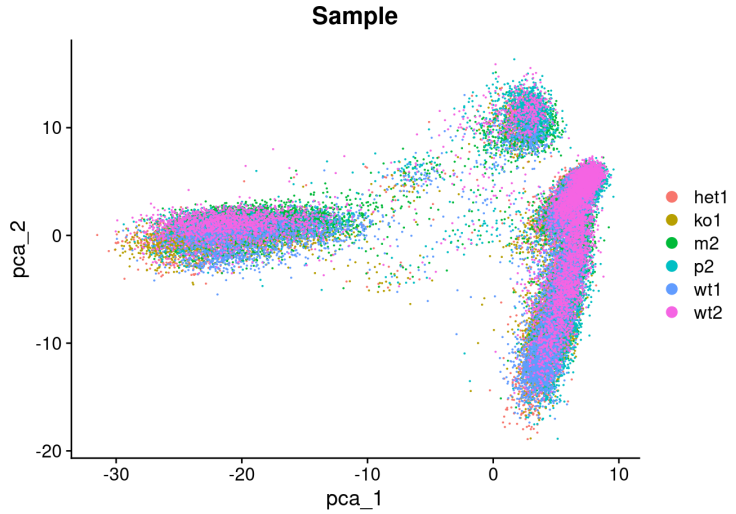

Figure 11: Cells represented with uncorrected PCs. The six different samples are represented by colors.

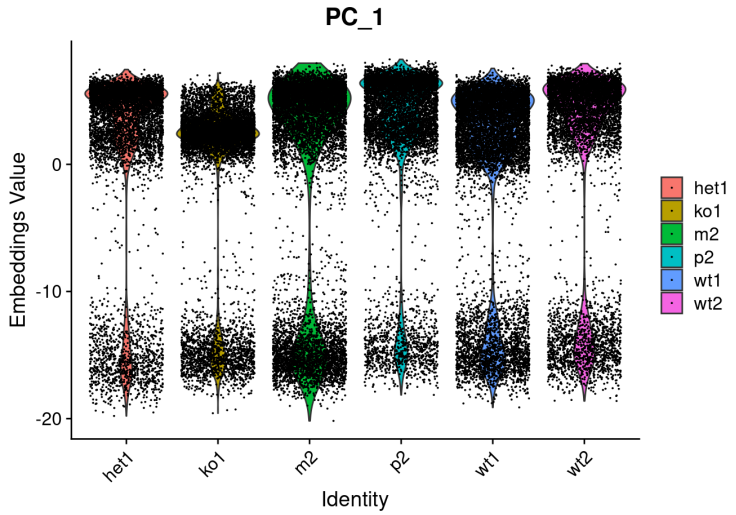

Figure 12: Embedding values of the first uncorrected PC of each cell split by sample.

Running the integration with Harmony, we observe that the algorithm converges after 14 cycles (Figure 21 in the Appendix).

In Figure 13, which shows the cells represented in the integrated PCs, cells seem to be better integrated in the first two reduced harmony dimensions compared to the original unprocessed PCs. This is conformed in the violin plot in Figure 14 as the embedding values are much more similarly distributed.

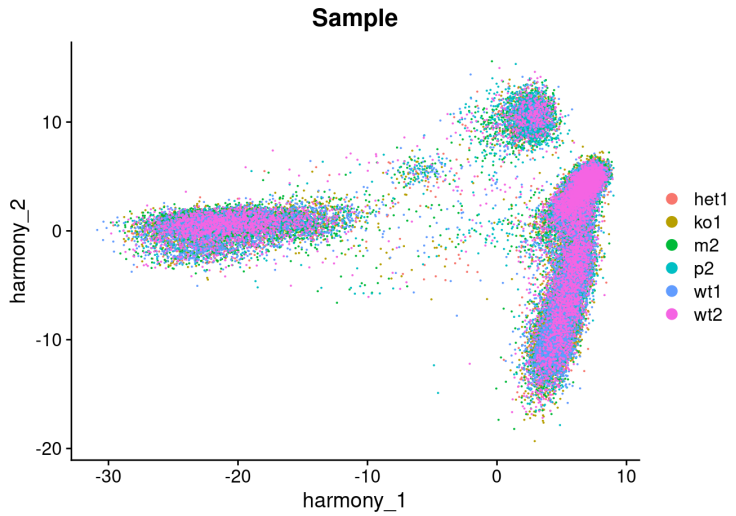

Figure 13: Datasets represented with integrated PCs, corrected by the Harmony algorithm. Different colors represent the six samples.

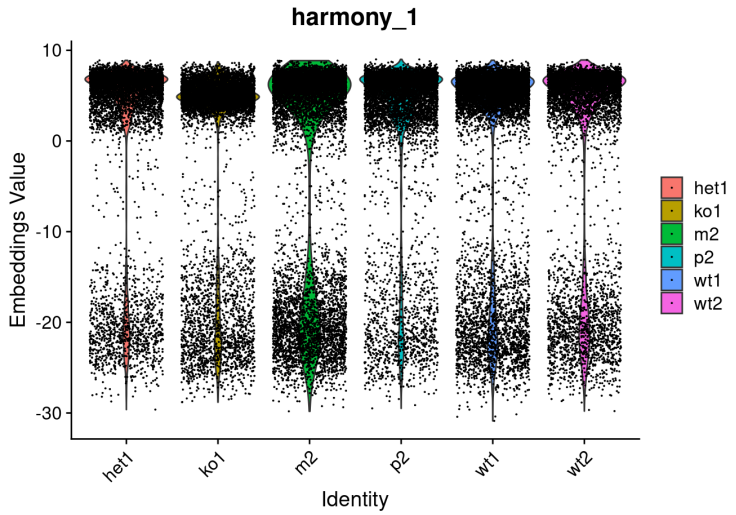

Figure 14: Embedding values of the first Harmony-corrected PC of each cell split by sample.

Comparing the UMAP plots of cells in the initial embedding (Figure 15) and the integrated embedding (Figure 16), we find cells clustering by sample in the original embedding and can clearly see that batch effects have been removed in the Harmony integrated UMAP plots.

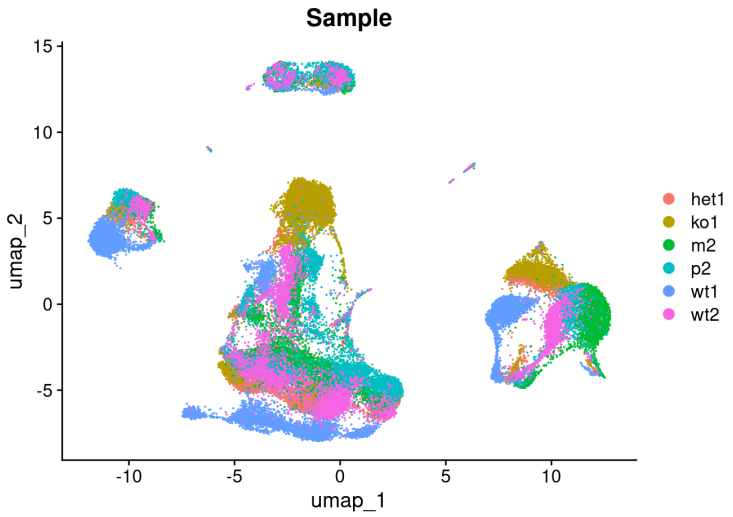

Figure 15: Plot of the initial, uncorrected UMAP embedding. Colors represent the six different samples.

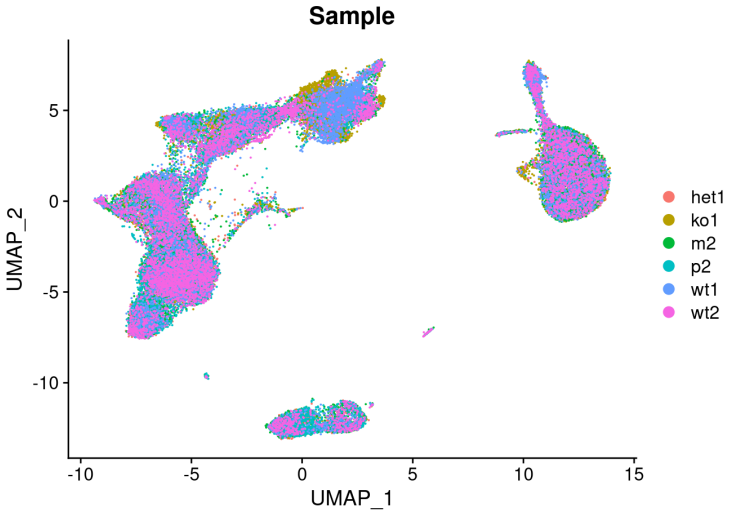

Figure 16: Plot of the Harmony-corrected UMAP embedding. Colors represent the six different samples.

Looking at the identified clusters in the uncorrected embedding (Figure 17), we see that 20 clusters have been identified. The cluster identification for the Harmony integrated embedding results in 16 clusters (Figure 18).

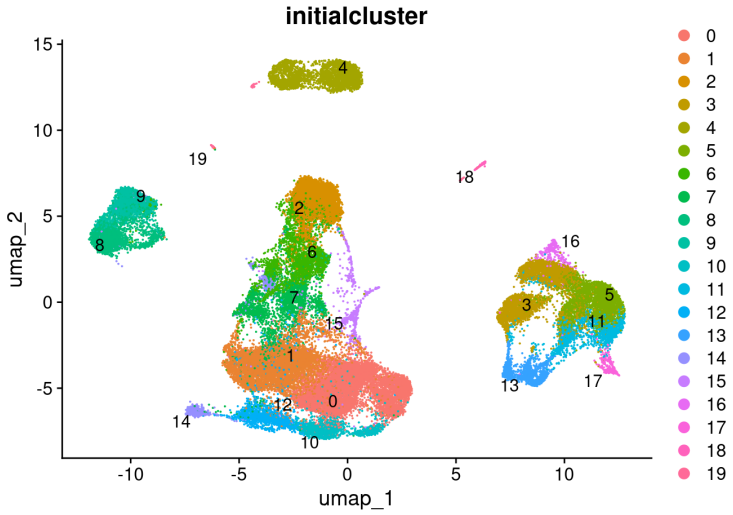

Figure 17: Clusters identified from the uncorrected, initial UMAP embedding. The colors represent the 20 recognized clusters.

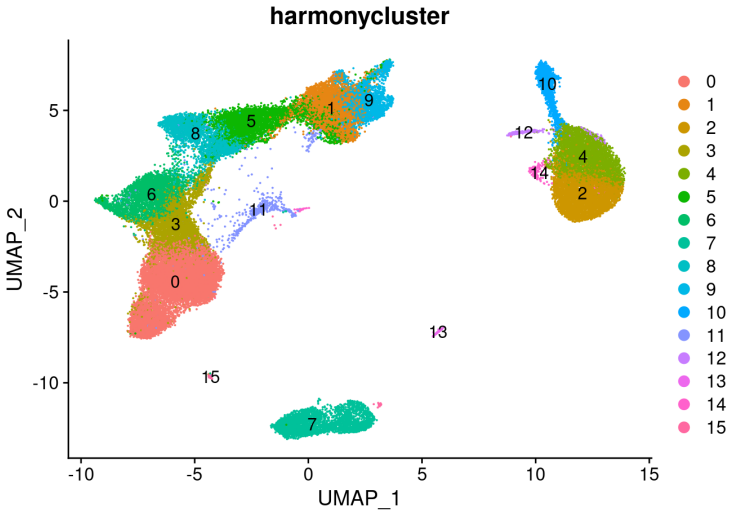

Figure 18: Clusters identified from the Harmony integrated UMAP embedding. The colors represent the 16 recognized clusters.

5 Discussion

Integrating single-cell RNA-seq data to combine data sets produced across multiple technologies and laboratories is an integral step in the general framework for case-control studies [22]. These integrated datasets can then be robustly utilized to study the effects of manipulation and genetic variation in heterogeneous cell populations.

In this project we have introduced three different single-cell RNAseq integration methods, namely Seurat, Conos and Harmony, and applied them on a PBMC dataset. To make the results of the individual methods comparable, the preprocessing for all methods was done using the same Seurat functions and identical dimensionality reduction methods such as PCA and UMAP. Whenever possible, identical parameters were chosen in all methods, e.g. the resolution parameter for cluster identification.

All integration methods appeared to work in that batch effects in PCA or UMAP plots were visually removed. In other words, all three integration methods succeeded in representing cells such that they no longer clustered by sample. Furthermore, we could observe in the case of the Harmony corrected embedding that the embedding values appeared more similarly distributed than the values of the initial, uncorrected PCs.

From the initial, uncorrected dataset, 20 clusters could be identified, compared to 20 clusters found in the Seurat integrated UMAP embedding, 14 different clusters found with Conos in PCA space and 16 clusters in the case of Harmony integration. There are various reasons how such big differences in cluster detection numbers could arise. First, the three methods apply different integration approaches using varied data structures and resulting in completely different outputs. Seurat removes batch effects by computing a common, batch corrected gene expression matrix (gene-based method), Conos calculates a corrected neighbourhood graph (graph-based method) and Harmony attempts to embed cells in a joint space with reduced dimensionality (embedding-based method). Moreover, each integration approach utilizes a different community detection method, which can differ in sensitivity, even when the same resolution parameter was used. Another point to notice is that all three methods make use of a k-nearest neighbour (k-NN) approach to cluster cells. In contrast to Conos and Seurat which use a discrete k-NN clustering, Harmony employs a soft clustering approach for probabilistic cluster assignments. The developers of the Harmony algorithm argue that discrete clustering can over-discretize the space and therefore smooth transitions between populations can be lost [23]. Preprocessing could also have an impact on the results. Within the Harmony workflow, normalization, finding variable features and scaling was done on the complete dataset, while Seurat and Conos preprocessing was applied to each sample individually before pooling cells into a single matrix. This strategy might facilitate integration of datasets in which all cell populations appear in all samples, but may cause problems for datasets with overlapping but not identical populations [23].

We analysed the composition of clusters detected in the Conos integration approach. The number of cells belonging to one cluster has a big range from many 1000 cells in cluster 1 to less than ten cells in cluster 14. In almost every cluster cells from all samples were observed, except for cluster 14 which only appeared in samples ko1 and wt1 and cluster 12 which was not present in het1. As the rare cluster 14 was observed both in an affected individual as well as in an unaffected control, we cannot say anything about its connection to the disease state. More interestingly, the diseased individual ko1 has a rather different distribution of cell clusters. All other samples have many cells belonging to cluster 1, but most cells of sample ko1 belong to cluster 2. Other than that, there is no obvious visual trend, that wildtype sample cells seemed to be differently distributed than the remaining samples het1, m2 and p2.

Other consideration, when comparing these three integration methods are runtime and memory storage. Harmony beats Seurat in both categories: at 125'000 cells, Harmony runs 30 to 200 times faster and requires 30-50 times less memory [23]. The effect gets even more evident for larger datasets and Seurat fails to integrate datasets $> 100'000$ cells [19]. Conos had the highest memory requirements in a comparative study [19], but is still a fast approach, particularly when a simple PCA space is used.

It was also difficult to choose resolution parameters for the clustering methods. An Analysis with a high resolution parameter will result in more fine-grained clusters while reducing cluster breadth and obtaining more specific clusters which don't appear in all samples. We chose a value of 0.8 as many PBMC integration tutorials chose values of 0.5 to 1 [43; 38; 34], but another choice would definitely alter the findings.

This project only focussed on the application of three integration methods. In future work it would also be interesting to have a closer look at the cell composition of the identified clusters, to examine shifts in cell population sizes and types. This can be done by pre-established sets of cell type and cell subtype markers as for example CD14+ or CD16+ for monocytes or CD20+ for B cells and CD3+ for T cells [22].

Moreover, it would be desirable to use metrics to assess integration performance of the individual methods. This has been done in previous comparative studies that discussed different integration approaches and evaluated their performance by benchmarking methods on various datasets. Some papers focussed on simpler, low batch complexity integration tasks [25; 26; 47]. These studies found Harmony and Seurat 3 to perform generally well, besides Seurat being prone to overcorrect batch effects in samples with huge amounts of biologically distinct cell types [47]. Harmony was recommended as a first approach due to its significantly shorter runtime. Conos being quite a new method, was only included in a recent paper from 2020 that benchmarked 10 popular data integration tools on nine more complex integration tasks [19]. To assess the performance of the integration methods, 14 different metrics were used to determine batch effect removal versus conservation of biological variation. An inter-

esting finding was that Seurat v3 tended to focus on removal of batch variation, while Conos prioritized bio-conservation. An impact of task complexity on algorithm performance was also observed. Seurat v3 and Harmony perform particularly well in simulations and simple integration tasks with clear batch and biological structure, which is in line with previously mentioned benchmark studies. Ranked best overall for RNA-seq integration were the methods BBKNN [48], Scanorama [49] and scVI [50], which worked particularly well on more complex, real data integration tasks. Conos performed well overall, but did not stand out in one specific integration task. The authors considered all ten methods as usable, though recommending though Harmony, BBKNN and Seurat v3 for new users due to excellent documentation.

We learn from these benchmarking studies that there is no single superior method for all kinds of scRNA-seq integration tasks. Each method has distinct strengths and weaknesses, such that benchmarking studies will become even more important as new methods arise. In addition, these papers can be references for method developers to directly assess performance of new integration approaches on benchmarking datasets. An interesting development in the future might be deep learning approaches as scVI, which could have great potential with increases of data availability and accessibility of GPU hardware [19].

Functioning data integration approaches allowing for robust comparisons of heterogeneous tissues will lay the groundwork to answer many fundamental biological questions such as cell response to perturbation, disease or evolution.

6 Appendix

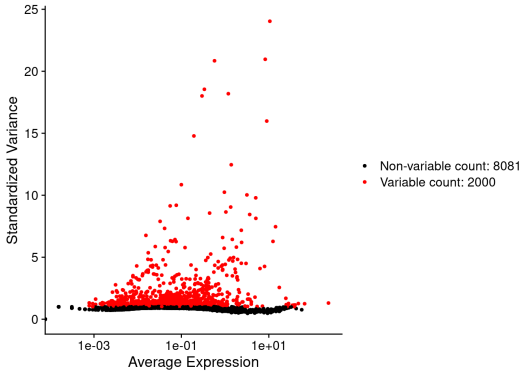

Figure 19: Standardized variance of all features (genes) of the first sample (het1) detected by the Seurat FindVariableFeatures function. In red visible are the 2000 most variable features, black shows the remaining features of the first dataset.

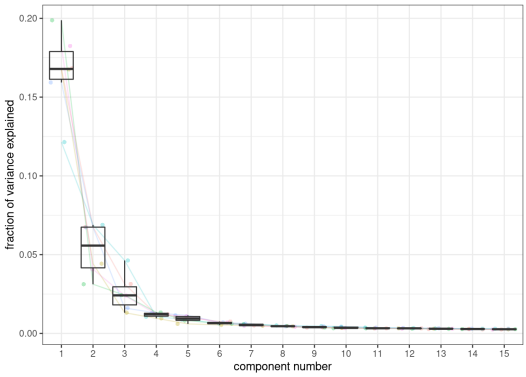

Figure 20: Estimated amount of variance explained by successive PCs identified by the Conos integration function (buildGraph()).

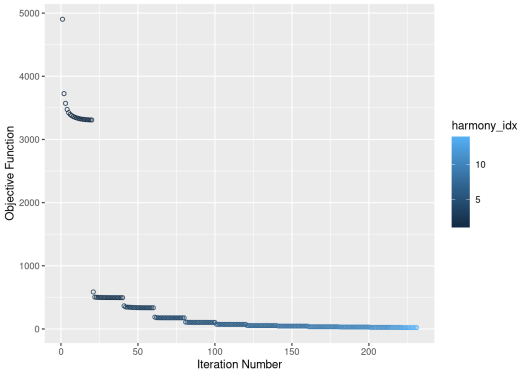

Figure 21: Value of the objective function per iteration of the Harmony function. The harmony-idx shows the cycles run by the algorithm with a colour gradient.

References

[1] Svensson, Valentine, et al. "Power analysis of single-cell RNA-sequencing experiments." Nature methods 14.4 (2017): 381-387.

[2] Kolodziejczyk, Aleksandra A., et al. "The technology and biology of single-cell RNA sequencing." Molecular cell 58.4 (2015): 610-620.

[3] Ziegenhain, Christoph, et al. "Comparative analysis of single-cell RNA sequencing methods." Molecular cell 65.4 (2017): 631-643.

[4] Klein, Allon M., et al. "Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells." Cell 161.5 (2015): 1187-1201.

[5] Zheng, Grace XY, et al. "Massively parallel digital transcriptional profiling of single cells." Nature communications 8.1 (2017): 1-12.

[6] Svensson, Valentine, Roser Vento-Tormo, and Sarah A. Teichmann. "Exponential scaling of single-cell RNA-seq in the past decade." Nature protocols 13.4 (2018): 599-604.

[7] Macosko, Evan Z., et al. "Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets." Cell 161.5 (2015): 1202-1214.

[8] Haque, Ashraful, et al. "A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications." Genome medicine 9.1 (2017): 1-12.

[9] Bacher, Rhonda, and Christina Kendziorski. "Design and computational analysis of single-cell RNA-sequencing experiments." Genome biology 17.1 (2016): 63.

[10] Villani, Alexandra-Chloé, et al. "Single-cell RNA-seq reveals new types of human blood dendritic cells, monocytes, and progenitors." Science 356.6335 (2017).

[11] Shekhar, Karthik, et al. "Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics." Cell 166.5 (2016): 1308-1323.

[12] Tirosh, Itay, et al. "Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq." Science 352.6282 (2016): 189-196.

[13] Satija, Rahul, et al. "Spatial reconstruction of single-cell gene expression data." Nature biotechnology 33.5 (2015): 495-502.

[14] Welch, Joshua D., Alexander J. Hartemink, and Jan F. Prins. "SLICER: inferring branched, nonlinear cellular trajectories from single cell RNA-seq data." Genome biology 17.1 (2016): 1-15.

[15] "Human Cell Atlas", `https://www.humancellatlas.org/` (accessed: 09/01/21)

[16] "Accelerating Medicines Partnership (AMP), https://www.nih.gov/research-training/accelerating-medicines-partnership-amp (accessed: 09/01/21)

[17] Zhang, Fan, et al. "Defining inflammatory cell states in rheumatoid arthritis joint synovial tissues by integrating single-cell transcriptomics and mass cytometry." Nature immunology 20.7 (2019): 928-942.

[18] Der, Evan, et al. "Tubular cell and keratinocyte single-cell transcriptomics applied to lupus nephritis reveal type I IFN and fibrosis relevant pathways." Nature immunology 20.7 (2019): 915-927.

[19] Luecken, Malte D., et al. "Benchmarking atlas-level data integration in single-cell genomics." BioRxiv (2021).

[20] Lähnemann, David, et al. "Eleven grand challenges in single-cell data science." Genome biology 21.1 (2020): 1-35.

[21] Eisenstein, Michael. "Single-cell RNA-seq analysis software providers scramble to offer solutions." Nature Biotechnology 38.3 (2020): 254-257.

[22] Butler, Andrew, et al. "Integrating single-cell transcriptomic data across different conditions, technologies, and species." Nature biotechnology 36.5 (2018): 411-420.

[23] Korsunsky, Ilya, et al. "Fast, sensitive and accurate integration of single-cell data with Harmony." Nature methods (2019): 1-8.

[24] "scRNA-tools", https://www.scrna-tools.org/tools?sort=name&cats=Integration, (accessed: 09/01/21)

[25] Büttner, Maren, et al. "A test metric for assessing single-cell RNA-seq batch correction." Nature methods 16.1 (2019): 43-49.

[26] Tran, Hoa Thi Nhu, et al. "A benchmark of batch-effect correction methods for single-cell RNA sequencing data." Genome biology 21.1 (2020): 1-32.

[27] Barkas, Nikolas, et al. "Joint analysis of heterogeneous single-cell RNA-seq dataset collections." Nature methods 16.8 (2019): 695-698.

[28] "10xGenomics: Chromium Single Cell Gene Expression", https://www.10xgenomics.com/products/single-cell-gene-expression, (accessed 03/01/21)

[29] "Bioconductor: SingleCellExperiment", http://www.bioconductor.org/packages/release/bioc/html/SingleCellExperiment.html, (accessed: 03/01/21)

[30] "Uniform Manifold Approximation and Projection in R", (https://cran.r-project.org/web/packages/umap/vignettes/umap.html) (accessed: 01/01/2021)

[31] "Principal Component Analysis (PCA)", https://www.rdocumentation.org/packages/FactoMineR/versions/2.4/topics/PCA (accessed: 09/01/21)

[32] "Seurat: R toolkit for Single Cell Genomics", https://satijalab.org/seurat/ (accessed: 02/01/21)

[33] "Integration and Label Transfer", (2019), https://satijalab.org/seurat/v3.0/integration.html (accessed 02/01/2021)

[34] "Seurat - Guided Clustering Tutorial", (2020), https://satijalab.org/seurat/v3.2/pbmc3k_tutorial.html, (accessed 02/01/2021)

[35] "Comparison of Single Cell RNA sequencing integration approaches" (2021), https://jeprob.github.io/project-scrnaseq-integration-comparison/index.html (accessed: 08/01/21)

[36] Witten, Daniela M., Robert Tibshirani, and Trevor Hastie. "A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis." Biostatistics 10.3 (2009): 515-534.

[37] Kharchenkolab, "Conos", Github repository, https://github.com/kharchenkolab/conos (accessed: 04/01/21)

[38] Kharchenkolab, "Conos Walkthrough", Github repository, https://github.com/kharchenkolab/conos/blob/master/vignettes/walkthrough.md (accessed: 08/01/21)

[39] "Integration of datasets using Conos", (2019), http://htmlpreview.github.io/?https://github.com/satijalab/seurat-wrappers/blob/master/docs/conos.html (accessed: 08/01/21)

[40] "Package 'tsne'", (2016) (https://cran.r-project.org/web/packages/tsne/tsne.pdf) (accessed: 08/01/21)

[41] Lferry007, "LargeVis", Github repository, (https://github.com/lferry007/LargeVis) (accessed: 08/01/21)

[42] Immunogenetics, "Harmony", (2020), GitHub repository, https://github.com/immunogenomics/harmony

[43] "Seurat V3 Interface", http://htmlpreview.github.io/?https://github.com/immunogenomics/harmony/blob/master/docs/SeuratV3.html (accessed: 01/01/2020)

[44] Mao, Qi, et al. "Dimensionality reduction via graph structure learning." Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2015.

[45] Jordan, Michael I., and Robert A. Jacobs. "Hierarchical mixtures of experts and the EM algorithm." Neural computation 6.2 (1994): 181-214.

[46] Jeprob, "Comparison of Single Cell RNA sequencing integration approaches", Github repository, `https://github.com/jeprob/project-scrnaseq-integration-comparison` (accessed: 11/01/21)

[47] Chen, Wanqiu, et al. "A multicenter study benchmarking single-cell RNA sequencing technologies using reference samples." Nature Biotechnology (2020): 1-12.

[48] Polański, Krzysztof, et al. "BBKNN: fast batch alignment of single cell transcriptomes." Bioinformatics 36.3 (2020): 964-965.

[49] Hie, Brian, Bryan Bryson, and Bonnie Berger. "Efficient integration of heterogeneous single-cell transcriptomes using Scanorama." Nature biotechnology 37.6 (2019): 685-691.

[50] Lopez, Romain, et al. "Deep generative modeling for single-cell transcriptomics." Nature methods 15.12 (2018): 1053-1058.