





Unbiased integration of single cell transcriptome replicates

Martin Loza ¹, Shunsuke Teraguchi ^{1,2,3}, Daron M. Standley ^{1,3} and Diego Diez ^{1,*}

¹Immunology Frontier Research Center, Osaka University, Suita 565-0871, Japan, ²Faculty of Data Science, Shiga University, Hikone 522-8522, Japan and ³Research Institute for Microbial Diseases, Osaka University, Suita 565-0871, Japan

Received November 10, 2021; Revised February 09, 2022; Editorial Decision February 22, 2022; Accepted February 23, 2022

ABSTRACT

Single cell transcriptomic approaches are becoming mainstream, with replicate experiments commonly performed with the same single cell technology. Methods that enable integration of these datasets by removing batch effects while preserving biological information are required for unbiased data interpretation. Here, we introduce Canek for this purpose. Canek leverages information from mutual nearest neighbor to combine local linear corrections with cell-specific non-linear corrections within a fuzzy logic framework. Using a combination of real and synthetic datasets, we show that Canek corrects batch effects while introducing the least amount of bias compared with competing methods. Canek is computationally efficient and can easily integrate thousands of single-cell transcriptomes from replicated experiments.

INTRODUCTION

Single cell sequencing technologies allow quantification of RNA expression levels within a given cell with unprecedented resolution (1). However, these approaches require integration over multiple observations to increase signal to noise. In such efforts, the true biological signal can become distorted. Even the most skilled operator using the same instrument will tend to observe systematic differences in replicates. Although such batch effects are well-known, they do not result from a single cause and thus are difficult to define or correct (2).

Many methods to integrate single cell datasets obtained from the same tissues using different technologies have been introduced (3). One of the pioneering techniques is the so-called Mutual Nearest Neighbors (MNN) correction method (4). In this method, MNN pairs are used to identify corresponding cells in two different batches. A pair-specific correction vector is then defined as the difference between the expression profiles of the cells from each MNN pair. The

correction vectors are then weighted to smooth the corrections between adjacent cells. Subsequently, other tools have been developed that use MNNs to integrate batches (5–7). One popular method, implemented in the Seurat R package, finds MNN pairs in a correlated space using canonical correlation analysis (CCA) (5). The identified pairs are used as ‘anchors’ to correct batch effects. Another interesting approach, Harmony, iteratively removes batch effects by clustering in a low dimensional space (8). LIGER applies a similar clustering approach by segmenting cells using a shared factor neighborhood graph under a low dimensional space defined with an integrative non-negative matrix factorization method (9).

In a comprehensive benchmark of 14 batch correction methods, including the ones above, the methods were tested under different scenarios to quantify the effects of data acquired by different technologies, use of dissimilar cells, data size, numbers of batches and simulated biases (3). The three top-scoring methods were Harmony, Liger and Seurat. However, the authors found that each method performed differently on each test, with no obviously superior method (3). Another benchmark done on atlas-level datasets found that the best integration approach strongly depended on the target task (10). This ambiguity makes best practices for integration of batches from replicated experiments unclear. An important question is how much bias such methods introduce and which of the methods is best suited for this task. To address these issues, we introduce Canek. Canek operates on two levels: it assumes mostly linear batch effects within a cluster of similar cells but allows non-linear corrections between different clusters of cells in a pair of datasets. This allows Canek to efficiently integrate single cell transcriptomes from replicated experiments while introducing minimal bias, thus preserving biologically relevant information.

MATERIALS AND METHODS

Canek workflow

Figure 1 shows the workflow for correcting a pair of batches. We define one of the batches as the *query batch* and the

*To whom correspondence should be addressed. Tel: +81 6 6879 4263; Email: diez@ifrec.osaka-u.ac.jp

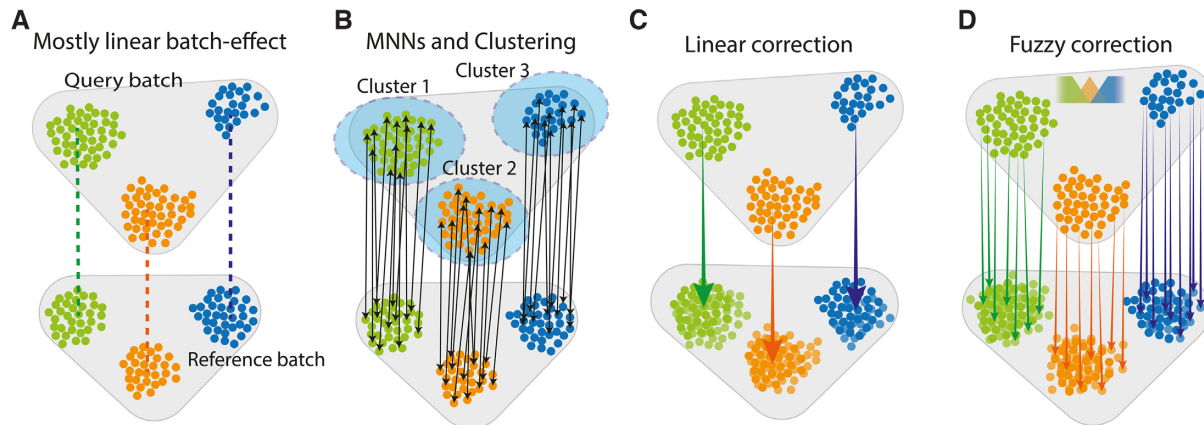


Figure 1. Overview of Canek workflow. (A) Canek starts with a reference batch and query batch, assuming a predominantly linear batch effect. (B) Cell clusters are defined on the query batch and MNN pairs (arrows) are used to define batch effect observations. (C) The MNN pairs from each cluster are used to estimate cluster specific correction vectors. These vectors can be used to correct the batch effect or, (D) a non-linear correction can be applied by calculating cell-specific correction vectors using fuzzy logic.

other one as the *reference batch*. We correct the cells from the query batch to match the cells from the reference batch. When correcting more than two batches we perform an optional hierarchical optimization of batch order (see Hierarchical integration section). The main steps of Canek's workflow are:

1. Obtaining batch effect observations using mutual nearest neighbors (MNN) pairs.
2. Clustering the query batch to define local groups of cells.
3. Calculating a batch effect correction vector for each cluster.
4. Obtaining a fuzzy correction by smoothing the transitions between the local correction vectors.

Canek expects input datasets to be log normalized. The output dataset retains the same dimensionality (number of genes) as the input batches.

Batch effect observations

The first step is to identify what we call batch effect observations. This is the gene expression differences of a set of cells from the reference and query batches that will enable us to estimate the batch effect.

To speed up computation and reduce the potentially negative impact of Euclidean distance metrics in high dimensions, we take several steps. First, we focus on the top (2000 by default) most highly variable features (HVF). Ideally, we want to choose features common to all batches, but the number of common HVF diminishes drastically as the number of datasets increases. Therefore, we used the method implemented in the Seurat package in `FindVariableFeatures`. In this method, a list of highly variable features is calculated first for each batch independently. Then, the features are ranked based on the number of times they are among the top HVF in each batch. Finally, the top features in this ranking are selected. This method will first favor features found in all batches, then those in all but one, etc. Although not perfect, we think this approach is reasonable and flexible enough for most applications. Next, we use the HVF to

calculate the first 50 principal components (PCs) (11) using the `prcomp_irlba` function from the `irlba` R package (12). This lower dimensional space is used to identify MNNs, and in the clustering and fuzzy correction steps. However, during the calculation of the correction vector step, we use the original input datasets. By focusing on the combination of features that contribute the most to the variability in the data and performing dimensionality reduction, we effectively alleviate most of the problems associated with Euclidean distances in high dimensions.

We calculate mutual nearest neighbors (MNN) pairs (4) using 50 PCs to obtain batch effect observations. We assume that at least one cell is shared between the batches to integrate. The MNN pairs are defined by the intersection of the *crossed k nearest neighbors* for each cell of two input batches. For example, for a cell c_1 from batch one, we find the k closest cells from batch two, and for cell c_2 from batch two, we find the k closest cells from batch one. If c_1 and c_2 are mutually contained on each other's nearest neighbor set, they are considered as a MNN pair. In Canek, to identify MNN pairs we first find the crossed 30 nearest neighbors of the query and reference batches using the `get.knn` function from the `FNN` R package (13). We then select those cells that fulfill the MNN criteria to form cell pairs. We treat the gene expression differences from these pairs as observations of the batch effect.

Clustering

Following Haghverdi *et al.* (2018) (4) we assume that the batch effect is almost orthogonal to the biological space, and that the variations due to the batch effect are smaller than the biological variation (see Supplementary material of (4) for a deeper discussion of these assumptions). Small variations to this orthogonality assumption can be caused by noise or by non-linearities. A common way to deal with non-linear dynamics is to linearize over bounded regions (14), to solve each of these local problems, and, if necessary, to join all the pieces back into a non-linear global solution. Following this idea, we partition the query batch into clusters, which we define as a bounded set of related cells, using

the Louvain algorithm implemented in the *igraph* R package (15). By default, clustering is done using the first 10 PCs.

Correction vector

Following our *local orthogonal batch effect* assumption, for each cluster we state the relation:

$$g_{Q_k}^i = g_{R_k}^i + g_{BE_k}^i + \epsilon$$

where g^i , $i = 1, \dots, n$, is the log-normalized gene expression level of the n genes from the input batches. The batch effect g_{BE_k} is represented as an additive value in the query batch g_{Q_k} in terms of the same gene in the reference batch g_{R_k} , $k = 1, \dots, p$, being p the number of MNN pairs from the membership under analysis. Finally, ϵ represents a normally distributed random error term with mean zero and standard deviation σ , which we assume to be independent of g^i on each cluster. Thus, using

$$g_{Q_k} - g_{R_k} = g_{BE_k} + \epsilon$$

on each gene i , the term $g_{BE_k} + \epsilon$ would be normally distributed with mean $\mu = g_{BE}$ and standard deviation σ . Accordingly, a good estimation of the batch effect would be the mean of the gene expression subtraction between

MNN cells pairs (e.g. $\hat{g}_{BE} = \frac{1}{n} \sum_{k=1}^p (g_{Q_k} - g_{R_k})$). But there

is a complication with this approach, since *erroneous pairs* between cells from distinct but related cell types could be formed, resulting in the incorrect integration of dissimilar subpopulations (5). To tackle this problem, reasoning that abnormal pairs would appear as outliers to the normal distribution of $g_{BE_k} + \epsilon$, we estimate a correction vector

$$CV = - \begin{bmatrix} \hat{g}_{BE_k}^1 = \text{Med}(g_{Q_k}^1 - g_{R_k}^1) \\ \vdots \\ \hat{g}_{BE_k}^n = \text{Med}(g_{Q_k}^n - g_{R_k}^n) \end{bmatrix}$$

where the function *Med* represents the statistical median, which is less affected by outliers than the mean. Canek uses this approach by default to reduce the impact from outliers, but it is possible to perform an optional filtering step (with extra computational cost) based on the interquartile range to detect MNN outliers (see *Filtering* section).

Fuzzy correction

From the steps described above, each cell from the same cluster will be assigned the same correction vector. We use fuzzy logic to smoothly join the cluster-specific corrections into a cell-specific one, where each cell has a unique correction vector (see Figure 1). Within this fuzzy logic framework, the clusters previously identified will be considered as memberships.

Using the PCs of the query batch, we create a minimum spanning tree (MST) over the memberships' center points (*MCs*) using the *mst* function from the R package *igraph* (15) (Supplementary Figure S1a, b). For each edge of the MST, we construct a pair of membership functions (*MFs*). These *MFs* are used to calculate a fuzzy score for the cells

(Supplementary Figure S1c, d). For example, let us consider an edge that joins the centers of memberships number 1 (MC_1) and 2 (MC_2). For each cell j that belongs to memberships 1 or 2, we define the vector V_j as a vector for cell j from MC_1 in the PCs embeddings. Similarly, let V_{MC_2} be the vector corresponding to MC_2 . Then, we obtain the scalar projection p_j for each cell j onto the line connecting MC_1 and MC_2 as:

$$p_j = V_j \cdot \frac{V_{MC_2}}{\|V_{MC_2}\|}$$

where the operator \cdot denotes the dot product, and $\|V_{MC_2}\|$ is the length of V_{MC_2} . We then construct the *MFs* (i.e. MF_1 and MF_2) as

$$MF_1(j) = 1 - \frac{p_j - p_{min}}{p_{max} - p_{min}}$$

$$MF_2(j) = \frac{p_j - p_{min}}{p_{max} - p_{min}}$$

Here, p_{max} and p_{min} are the maximum and the minimum of the scalar projections of the cells in the memberships ($p_{max} = \max_j p_j$ and $p_{min} = \min_j p_j$). In this way, the membership function MF_1 (MF_2) takes the maximum value 1 (the minimum value 0) for p_{min} and the minimum value 0 (the maximum value 1) for p_{max} , respectively, and linearly interpolates for the other values of the projections. (Supplementary Figure S2).

We calculate cell specific correction vectors CV_j by using the Takagi-Sugeno approach (16) to combine the membership's correction vector $CV^{(l)}$ (see *Correction Vector* section) with the membership functions:

$$CV_j = \frac{\sum_l MF_l(j) CV^{(l)}}{\sum_l MF_l(j)}$$

Each cell from the query batch gets a score between 0 and 1 for each of the memberships, which can be interpreted as a probability of belonging to that membership. We note that when a cell belongs to a membership connected to more than one membership, the cell gets as many scores as the number of other connected memberships. In this case, we use the mean of these scores as the cell's membership score, while the scores from the other memberships remain unchanged.

Even though the fuzzy scheme is applied in a low dimensional representation, the final output is in the original dimensionality of the input datasets. We recommend using Canek with the fuzzy step, but to skip it users can set the boolean parameter *fuzzy* to *FALSE*. In this case the final integration will be done using a membership-specific correction instead of a cell-specific one.

Hierarchical integration

The choice of reference batch may have a strong effect on the integration. To address this potential problem, we carry out hierarchical integration when there are more than two input batches. The purpose is to determine the optimal order of batches. First, we define the reference as the batch

with the largest number of cells. Then, the query batch is chosen to be the batch sharing the highest number of MNN pairs with the reference, an indication of higher similarity between batches. To this end, we obtain the first three PCs using the `prcomp_irlba` function in the `irlba` R package (12), find the MNN pairs and select the query batch as the one with the highest number of pairs. Once the reference and the selected query batch are integrated, we define the integrated batch as the new reference, and again select the query batch following the same procedure as before. We continue this process until all the input batches are integrated. The hierarchical integration is optional and can be deactivated by setting the boolean parameter `hierarchical` to `FALSE`. In this case, the order of integration follows the order in the input list.

Filtering

We assume that erroneous MNN pairs would appear as outliers from the normal distribution of $g_{BE_k} + \epsilon = g_{Q_k}^n - g_{R_k}^n$ (see Correction Vector section). We use the median function to reduce the impact of these outliers on the correction vector estimation. In addition, the user can select an extra filtering step based on the interquartile range:

$$IQR = Q_{75} - Q_{25}$$

where Q_{75} and Q_{25} are the 75th and 25th percentiles of the distribution of the p MNN pairs' Euclidean distance $d(k)$, $k = 1, \dots, p$. Therefore, we will select and filter any outlier MNN pairs as:

$$MNN_k \text{ IS outlier IF } (d(MNN_k) < (Q_{25} - 1.5IQR) \mid d(MNN_k) > (Q_{75} + 1.5IQR)).$$

Analysis details

Data pre-processing. We performed the same data pre-processing for all the analyses. We implemented the 'Standard Workflow' from the Seurat R package (17), which involves:

- Normalization: using the function `NormalizeData`. Gene expression levels are divided by the total number of transcripts and multiplied by 10,000. The results are then log normalized.
- Identification of high variable features: using the function `FindVariableFeatures`. Genes that show high variations among cells are selected using the `vst` method.

Batch-correction algorithms. Table 1 lists the batch correction methods used.

To objectively compare the batch effect correction methods, we used the same pre-processed data and the same variable genes on each method. We obtained the variable genes from Seurat's integration using the `VariableFeatures(assay = 'integrated')` function from the Seurat R package (17) and used the genes to reduce the pre-processed uncorrected datasets. This is equivalent to running `FindVariableFeatures` in each batch independently and then obtaining (with `Seurat::SelectIntegrationFeatures`) the top features, as indicated in the section 'Batch effect observations' (above). We implemented the integration methods as follows:

Seurat. We used the `FindIntegrationAnchors` and `IntegrateData` functions with default parameters from the Seurat R package (17).

Canek. We used the `RunCanek` function from the Canek R package with default parameters.

MNN. We used the `mnnCorrect(cos.norm.out = FALSE)` function with default parameters from the `batchelor` Bioconductor package (4).

Scanorama. We used the `scanorama.correct(return.dense=TRUE)` with default parameters from the `scanorama` Python library (7).

ComBat. We used the `ComBat` function with default parameters from the `sva` R package (18).

Harmony. We used the `RunHarmony` function with default parameters from the `harmony` R package (8).

Liger. We used the `RunOptimizeALS(k = 20, lambda = 5)` and the `RunQuantileNorm` functions with default parameters from the `SeuratWrappers` R package (9).

ComBat-seq. We used the `ComBat_seq` function with default parameters from the `sva` R package (19).

scMerge. We used the `scMerge` function with default parameters from the `scMerge` R package (20). For the parameter `kmeansK`, we used the number of cell types when known, otherwise the number of clusters.

After correcting batch effects, we scaled each of the integrated and uncorrected datasets using the `ScaleData` function from the Seurat R package, except for Harmony and Liger integrations, as their output is already a low dimensional space.

Dimensionality reduction. We obtained the principal components (11) from the corrected and uncorrected datasets using the `RunPCA` function from the Seurat R package. We used the first 10 PCs as the standard in all the tests. In the case of the Uniform Manifold Approximation and Projection (UMAP) representation (21), we applied the `RunUMAP` from the Seurat R package to the selected PCs.

Simulated data. We used the `splatSimulate` function from the `splatter` Bioconductor package (22) to simulate three batches with batch effect. `Splatter` allows us to simulate cell types whether as groups or paths. Because we wanted to assess cell type preservation on a mixed population scenario with clearly defined groups along with a differentiation process, we simulated paths and groups separately and merged them. Then, we manually removed cells such that the batches shared only one cell type. The final cell type composition is:

After removing the cell types, the number of cells per batch is: 1671 cells for Batch 1, 975 cells for Batch 2 and 964 cells for Batch 3, all of them with the same 2000 genes.

To obtain the gold standard without batch effects, we used `splatSimulate(batch.rmEffect = TRUE)` and removed the same cells as the simulations with batch effects.

Running time benchmark. We used the `splatSimulate` function from the `Splatter` Bioconductor package (22) to simulate two datasets with a 2,000 genes and a varying number of cells in the range of 10k to 100k. Each dataset contained three cell types with appearance probabilities of 0.3, 0.3 and

Table 1. Batch effect correction methods used.

Method	Batch effect corrected output	Package version
Seurat	Normalized gene expression matrix	Seurat version 3.2.2 (5)
MNN	Normalized gene expression matrix	Bioconductor's batchelor version 1.6.2 (4)
Scanorama	Normalized gene expression matrix	Scanorama version 1.6 (7)
ComBat	Normalized gene expression matrix	sva version 3.38.0 (18)
Harmony	Normalized feature reduction vectors	Harmony version 1.0 (8)
Liger	Normalized feature reduction vectors	Liger version 0.5.0 (9)
ComBat-seq	Normalized gene expression matrix	sva version 3.38.0 (19)
scMerge	Normalized gene expression matrix	scMerge version 1.6.0 (20)
Canek	Normalized gene expression matrix	Canek version 0.1.7

0.4 respectively. We applied each of the correction methods five times on each of the simulated datasets and recorded the time. We used the *geom_smooth* function from the *ggplot2* R package (23) to plot the time trend lines.

Public datasets. Table 3 lists the public datasets we used.

Jurkat/t293 data analysis. We used the following publicly available datasets:

- 293T cells. <https://www.10xgenomics.com/resources/datasets/293-t-cells-1-standard-1-1-0>
- Jurkat cells. <https://www.10xgenomics.com/resources/datasets/jurkat-cells-1-standard-1-1-0>
- 50:50 Jurkat:293T cell mixture. <https://www.10xgenomics.com/resources/datasets/50-percent-50-percent-jurkat-293-t-cell-mixture-1-standard-1-1-0>

In the 50:50 Jurkat:293T cell dataset, we used the *kmeans* function from the *stats* R package (31) with $k = 2$, checked the expression of the XIST and CD3D3 genes and assigned the appropriate cell type labels.

Spleen data analysis (pseudo-batch).

- We use the publicly available dataset from Tabula Muris (25) <https://ndownloader.figshare.com/files/13090478>.

Pancreatic data analysis.

- We obtained the five public datasets (26–28,32,33) from the SeuratData R package(5) and used the provided cell type labels.

PBMC unstimulated and IFN- β -stimulated data analysis.

- We obtained the public datasets (29) from the SeuratData R package (5) and used the provided cell type labels.
- We identified clusters using the Seurat functions FindNeighbors with 10 PCA dimensions and FindClusters with the Leiden algorithm. Differentially expressed genes were identified for the two clusters of CD14+ monocytes using FindMarkers.

Human lung dataset. Single cell RNA-seq data and associated metadata from human lung was downloaded from GSE136831, loaded into R and converted into a Seurat object (5). This object was converted into a list of objects with the SplitObject function using Library_Identity as batch variable. The standard processing workflow was used

to normalize and find highly variable features in each of the libraries (see Data preprocessing above). This data was passed to the function RunCanek with hierarchical integration set to FALSE. The integrated dataset was scaled using ScaleData and Principal Component Analysis dimensions were calculated with RunPCA. The top 25 principal components were used to calculate UMAP using RunUMAP, and to create the Shared Nearest Neighbors graph with FindNeighbors. Clustering was done using FindClusters with the Louvain algorithm and a resolution of 1. Cells annotated as multiplets in the original publication were removed.

Metrics. We evaluated the results from the batch-correction methods by scoring the mixing between batches with the k-nearest-neighbor batch effect test (kBET) (34), and the cell purity preservation with the average Silhouette width (Silhouette) (35). We used the kBET and batch_sil functions from the kBET R package (34).

The kBET metric provides a rejection rate within 0 and 1 after testing batch mixing at the local level. The kBET's score could be affected by the choice in the number of k-nearest neighbors (kNN). To objectively assess the different integration methods, following the idea of Tran et al. (3), we obtained the mean cell number of the datasets and performed the scoring by fixing the kNN size as the 5%, 15% and 30% of this mean. To ease the interpretation of this metric, we calculated an 'acceptance rate', defined in the kBET publication as 1 minus the rejection rate (34).

We used the Silhouette coefficient to assess cell purity after integration (35). This metric analyzes the separation among cells from the same cluster as compared with cells from other clusters. Let $a(i)$ be the average Euclidean distance of cell i to all other cells from the same cluster as i , then the Silhouette width $s(i)$ is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

where $b(i)$ is calculated as

$$b(i) = \min_C d(i, C)$$

being $d(i, C)$ the average distance of cell i to all the other cells assigned to different clusters C . We used the cell type labels provided for each dataset as inputs to the Silhouette coefficient, except on the pseudo-batch experiment, where we obtained the cluster labels using the FindNeighbors and FindClusters(resolution = 0.5) functions from the Seurat R package.

For both kBET and Silhouette metrics we used the ‘harmony’ and ‘iNMF’ embeddings from Harmony and Liger corrections respectively. For the rest of the methods and the Uncorrected data we used the first 10 PCs.

A high kBET acceptance rate represents a better mix of the cells from both batches. A high Silhouette score represents a better separation between clusters. However, a high kBET or Silhouette scores do not necessarily indicate a better batch correction result. This is because the nature of the dataset influence how much mixing is necessary to remove batch effect. For example, a method that integrates all cells into a single cluster disregarding cell types would yield a much higher kBET score than a method that mixes batches while preserving known cell types. For this reason, we consider that kBET and Silhouette scores should be used as objective metrics only when there is an objective truth to compare with (see pseudobatch and simulation experiments results). In the other cases, these metrics can be used as quantitative metrics to support PCA or UMAP representations.

RESULTS

Overview of Canek

Canek corrects multiple batches by integrating pairs of batches sequentially. The dataset pairs that are input to Canek are denoted reference batch and query batch (Figure 1A). Then the integrated dataset becomes the reference to integrate the following batch. *Batch effect observations are defined* using *mutual nearest neighbors* (MNN) (4) and groups of similar cells are identified from the *query batch using clustering* (Figure 1B). Canek estimates a *correction vector* for each cluster using the median gene expression differences between cells in each cluster of the query batch and the corresponding cells in the reference as identified by MNN (arrows on Figure 1c). The correction vector can thus be used to remove the batch effect from each cluster in the query batch. In this linear correction, the same correction is applied to all the cells in the cluster (Figure 1C). Subsequently, Canek performs a *non-linear correction* by calculating a cell-specific transformation using fuzzy logic. This is done by defining a minimum spanning tree among clusters and then smoothing the transitions between the correction vectors (Figure 1D). Using a combination of real and simulated data, we show that, Canek exhibits unbiased corrections of single cell transcriptome data.

Canek successfully corrects batch effects in a Jurkat/293T mixture dataset

An example where batch effects are clearly visible is the mixture of cells used to demonstrate the 10× chromium sequencing technology (24). The dataset consists of three batches: one containing only 293T HEK (Human Embryonic Kidney) cells, another containing only Jurkat cells (immortalized human T lymphocytes), and a third comprised of a 50:50 mixture of 293T and Jurkat cells (24). Principal component analysis (PCA) of the Uncorrected dataset is shown in Figure 2A. Looking at cell-specific markers we can see there is a cluster of cells expressing XIST (293T cells) and two clusters of cells expressing CD3D (Jurkat cells). While the cluster of 293T cells shows mixing of cells from

both batches, the two clusters of Jurkat cells show batch specific distributions, suggesting an unknown systematic bias. We used different integration methods and assessed their ability to correct the systematic differences in the Jurkat cell data without introducing additional bias. To this end, we applied batch correction using Canek and 8 state-of-the-art methods: Combat, ComBat-seq, Harmony, Liger, MNN, Scanorama, scMerge and Seurat (4,5,7–9,18–20). Both Canek and MNN corrected the batch effect and enabled the identification of the expected cell population clusters (Figure 2b,d). However, other methods, including Combat and Seurat, resulted in incorrect mixing of cell populations (Figure 2C, E). The results for all methods are shown in Supplementary Figure S3 for PCA, and Supplementary Figure S4 for UMAP plots.

Next, we computed kBET and Silhouette scores for the Uncorrected and corrected datasets (Figure 2F). We chose the kBET metric to estimate the mixing of batches after correction, while the Silhouette metric enabled us to assess the preservation of the original cell clusters. As described in the Materials and Methods section in more detail, the metrics in this case cannot be used to assess the performance of the integration. However, they can be useful to identify how much mixing or change in cell populations are induced by the different methods. Furthermore, in this dataset, given its simplicity, we know how a proper integration should look, making the metrics plots more useful. Most methods show similar values of kBET, indicating similar levels of mixing. Methods that successfully integrated the batches while preserving cell populations had higher values in the Silhouette score. Canek, Harmony, MNN, and Scanorama all have similar values, and lead to visually successful integrations as seen in the PCA and UMAP plots. Combat, ComBat-seq and Seurat resulted in good integration but different levels of success in preserving the cell populations. Liger showed very different behavior (high kBET and low Silhouette), suggesting excessive mixing while not preserving cell populations. This agrees with the PCA/UMAP plots in Supplementary Figures S3 and S4. This is a good example where a larger kBET score does not necessarily corresponds with better batch correction result. These results show that Canek was able to successfully identify and correct the local batch effect while preserving biologically meaningful cell type differences.

Evaluation of correction bias

As shown above, batch correction methods can introduce biases in the data that disturb the biological information or alter the structure of cell populations (10,36). As single-cell genomics technologies become mainstream, more laboratories will perform experiments under different conditions with biological replicates obtained using a common technology. In this scenario, integration of datasets with minimal impact on cell phenotype is essential.

We define batch correction bias as undesired correction that may alter the original biological signal. To quantify how much bias correction methods introduce, we use a pseudo-batch approach (shown schematically in Figure 3A). Starting from a single dataset we identified clusters to define cell populations. Then we generated two

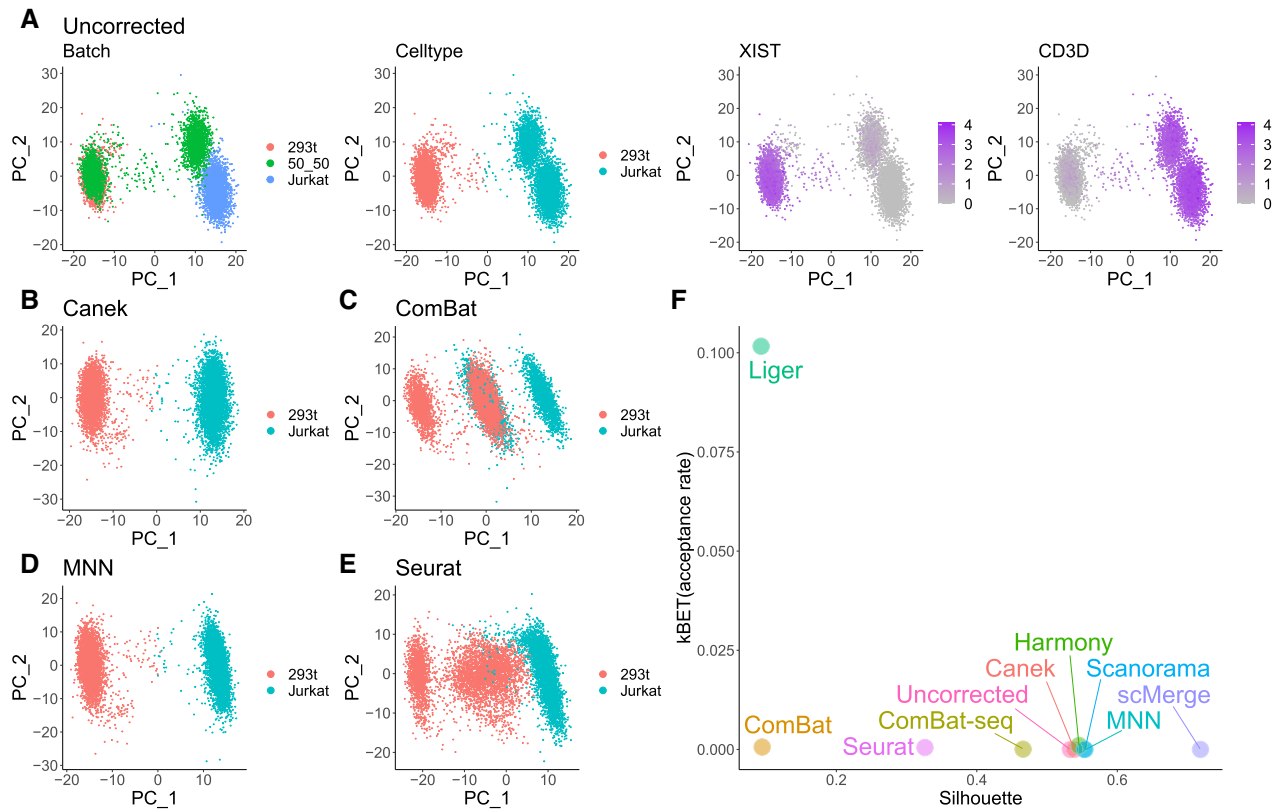


Figure 2. Batch effect correction methods may incorrectly mix dissimilar cell types. Batch effect correction of three batches, two containing pure Jurkat and HEK293T cells, and one with a 50:50 mix of Jurkat and HEK293T cells. (A) Jurkat and HEK293T cells are characterized by the expression of CD3D and XIST genes respectively. Before correction Jurkat cells grouped by batch. (B–E) show the results of batch effect correction using Canek, Combat, MNN and Seurat. Canek and MNN correctly integrated the Jurkat cells. Combat and Seurat incorrectly mixed Jurkat and HEK293T cells. (F) Scatterplot with kBET and Silhouette metrics calculated for the Uncorrected dataset and after correction with Canek and 8 other methods. Higher kBET (acceptance rate) and Silhouette scores mean better mixing of batches and a better delineation of cell types respectively.

pseudo-batches by sampling cells without replacement. Each pseudo-batch preserves the information about the original cell populations (clusters). Because no modifications to the original expression values were introduced during the sampling process, the batch effect between the two pseudo-batches is effectively zero. We assume that batch correction methods should not correct in this scenario since no batch effect exists, and identification of clusters from the integrated batches should preserve the clusters obtained from the original (uncorrected) dataset.

We applied this strategy to the droplet spleen dataset from Tabula Muris (25). In Figure 3B, the first UMAP plot shows the original dataset with cell clusters indicated with colors. The next two UMAP plots show the cells from the two pseudo-batches obtained after sampling. We applied batch correction to these two pseudo-batches. To quantify the introduced bias, we computed kBET and Silhouette scores for the Uncorrected and corrected datasets. Since there was no batch effect, the scores for the Uncorrected dataset corresponded to the optimal values.

Figure 3C shows that Canek integration resulted in even distribution of the cells from both batches and cluster distribution that resembled the original dataset. Figure 3D shows that MNN failed to completely integrate this dataset, with uneven distribution of cells from the pseudo-batches.

In Supplementary Figure S5 we show UMAP plots with results for each of the evaluated methods. Although in some cases it was trivial to identify differences with the Uncorrected dataset due to obvious changes in cell distributions, it was not always easy to evaluate the relative performance. To do so, we calculated kBET and Silhouette scores and compared them with those obtained from the Uncorrected dataset, which represented the optimal values. Figure 3E shows the scores for kBET and Silhouette metrics obtained from this experiment. In this plot, the dashed lines indicate the scores for the Uncorrected dataset, with the crossing point representing the optimal value. Canek, Combat, ComBat-seq and Harmony resulted in scores very close to the optimal value. To estimate the variability of the results due to pseudo-batch sampling, we repeated this experiment 10 times. Supplementary Figure S6 shows that Canek obtained scores closest to the values of the Uncorrected dataset, demonstrating that it introduced the least amount of bias when no batch effect was present.

Evaluation of integration in simulated data

To estimate the ability to correct batch effects when the effect is known exactly, we compared Canek with other methods using simulated data. We simulated three batches with

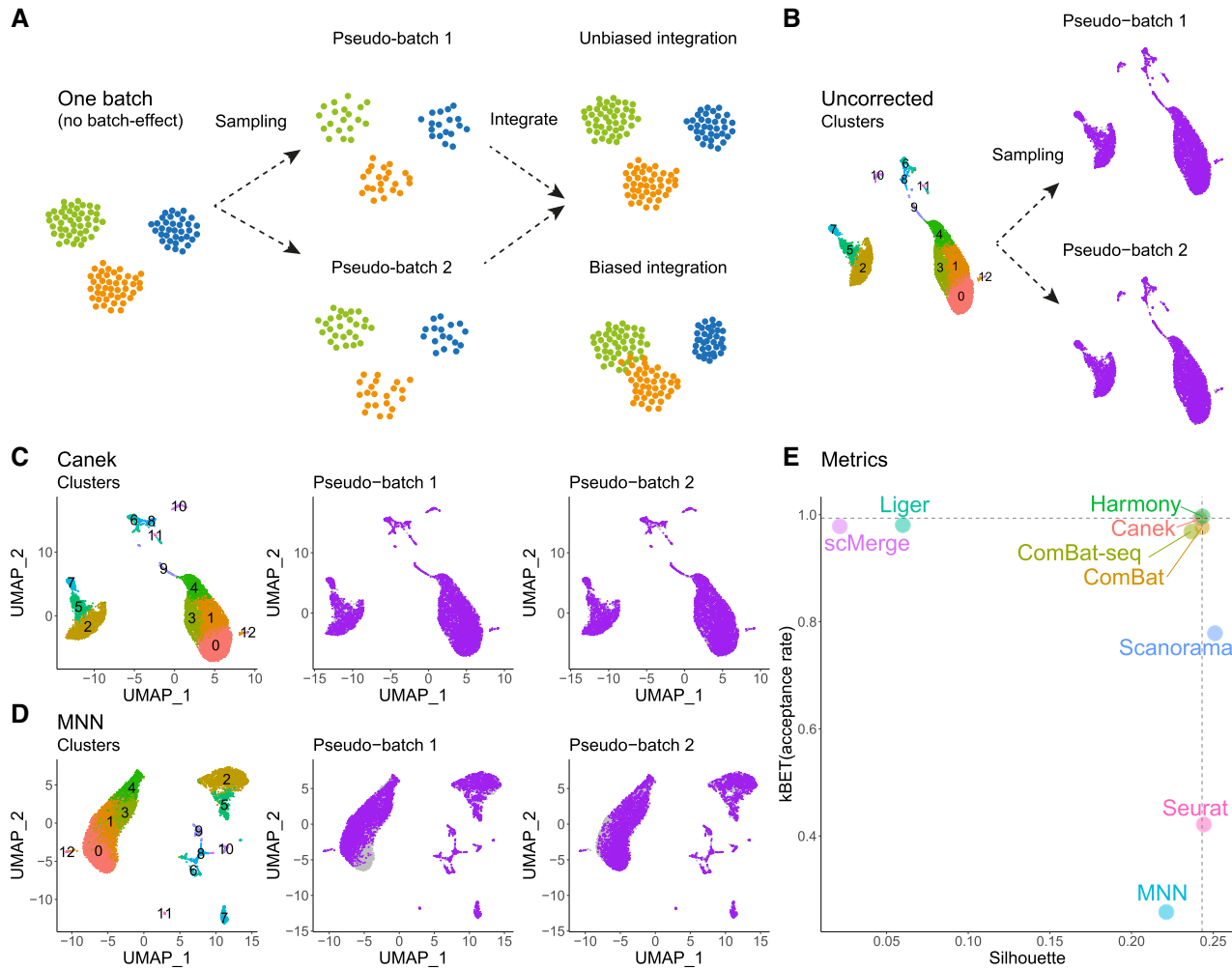


Figure 3. Correction methods may introduce biases. (A) Strategy for pseudo-batch generation: Starting from a single dataset with identified cell populations (clusters), we sample without replacement to generate two pseudo-batches. Then, we integrate these pseudo-batches and check whether the integration introduced biases comparing the result with the original dataset. (B) Pseudo-batch generation using the spleen dataset from Tabula Muris. The cells from each batch are colored purple while all other cells are in grey. (C) Canek integrated the pseudo-batches without introducing biases to the cell distribution. (D) MNN integration led to uneven distribution of cells from the pseudo-batches in the UMAP plot with e.g. some cells in batch 1 not covering all areas, causing the grey cells in batch 2 to be seen. (E) Using kBET and Silhouette metrics, the mixing among batches and the cluster preservation were evaluated. The optimal scores from the Uncorrected data are shown as dashed lines, while the scores from the correction methods are indicated as colored points. Unbiased methods are those whose metrics are closest to the intersection of the gray lines.

Table 2. Cell type distribution on simulated data

Method	Path cells		Group cells		
	Cell-1	Cell-2	Cell-3	Cell-4	Cell-5
Batch-1	✓	✓	✓	-	-
Batch-2	-	✓	-	✓	-
Batch-3	✓	-	-	-	✓

shared cell types using the splatter package (22) from which we can obtain an integrated dataset to use as a gold standard (GS). Batch 1 was composed of two shared and one unique cell type, whereas batches 2 and 3 had one shared and one unique cell type (see Table 2 for complete description). Supplementary Figure S7 shows UMAP plots with results for each of the evaluated methods. Figure 4A and

Table 3. Public datasets used

Dataset	Number of cells	Technology	Reference
Jurkat cells	3258	10x	(24)
HEK293T cells	2885		
Jurkat:HEK293T 50:50 mixture	3388		
Mouse spleen	1697	SMART-seq2	(25)
	9552	10x	
Human pancreas	8569	inDrop	(26)
	2285	CEL-seq	(27)
	1004	CEL-seq2	
	638	Fluidigm C1	
	2394	Smart-seq2	(28)
Human PBMCs (Interferon beta)	13 999	10x	(29)
Human lung	312 928	10x	(30)

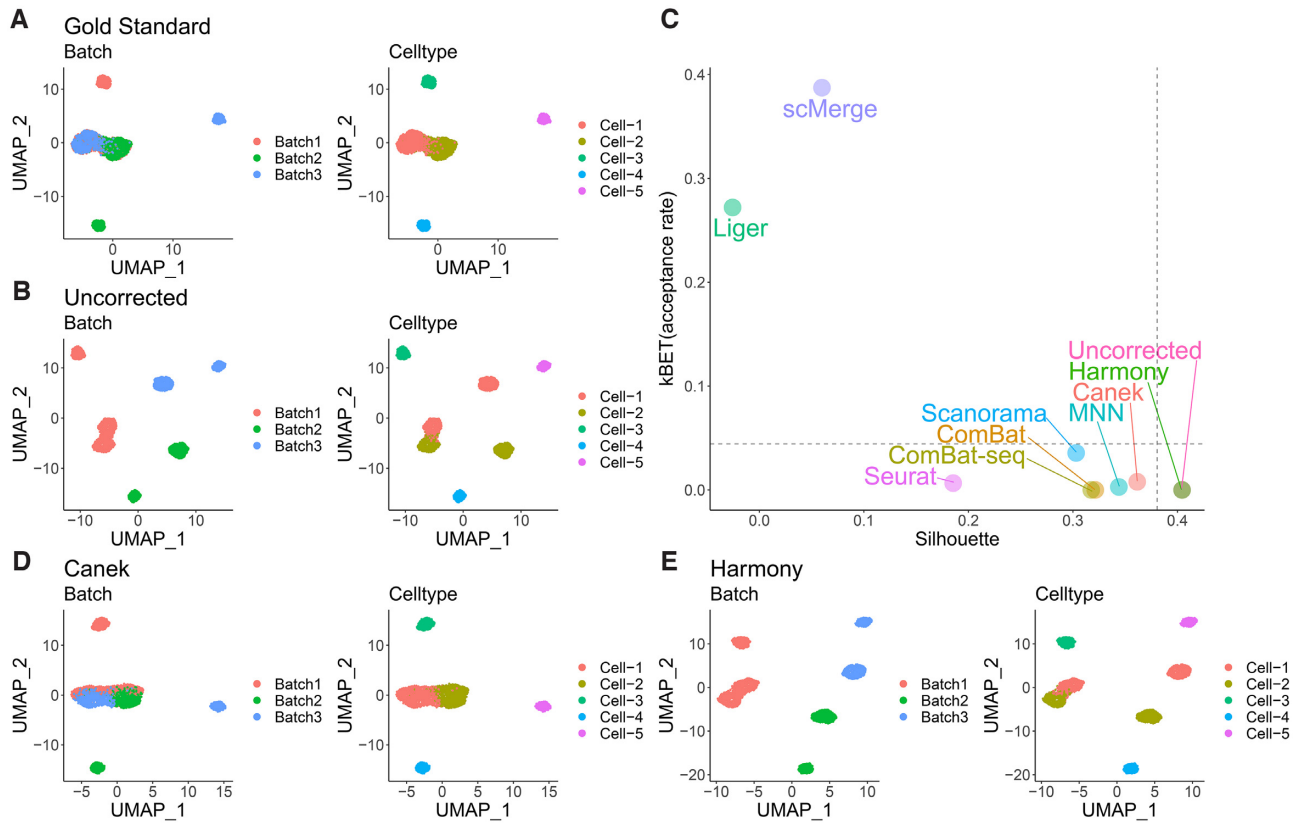


Figure 4. Batch effect correction on simulated data with a known gold standard. Three batches were simulated to test the integration methods in a scenario with a known gold standard. (A) The gold standard shows the UMAP plot for the batches without batch effect. Two cell types (Cell-1 and Cell-2) are shared among different batches. (B) The Uncorrected dataset shows batch-specific differences in cells of the same type. (C) kBET and Silhouette metrics for Uncorrected, Gold Standard (dashed lines) and the nine evaluated methods. Canek shows scores closest to the Gold Standard. (D) UMAP plot shows that Canek correctly integrated the shared cell types while maintaining the identity of non-shared ones. (E) Harmony correction failed to integrate cells from the same type.

B shows UMAP plots from the GS and the Uncorrected dataset, respectively. Figure 4C shows kBET and Silhouette scores from the GS (cross of dashed lines), Uncorrected data, and integrated datasets. We expected the best correction methods to be close to the metrics from GS. These results show that Canek scores were closest to those of the GS. This is consistent with the UMAP plot shown in Figure 4D, where Canek corrected the differences among shared cell types. Other methods including MNN, Scanorama, Combat and Combat-seq returned similar results to Canek. Interestingly, Harmony returned scores very close to the Uncorrected data, suggesting that it performed almost no correction, consistent with the UMAP plot in Figure 4E.

Application to real datasets

Next, we compared Canek with other methods on three real datasets: Tabula Muris spleen, human pancreatic islets and interferon beta stimulation (25–28,32,33).

First, we tested a scenario in which the same sample was used with two different technologies simultaneously. For this, we integrated the droplet and FACS batches from the Tabula Muris spleen datasets (25). Supplementary Figure S8 shows UMAP plots for the Uncorrected data, and the correction done by Canek and the other 8 correction meth-

ods. Except for scMerge, which merged some cell populations, all the methods successfully integrated the datasets, with cells from the same type found in the same clusters. This demonstrates that Canek can integrate datasets even from different technologies.

Next, we tested the scenario in which similar tissues were used with different technologies. For this we integrated eight human pancreatic islet datasets from five different technologies. Figure 5A shows the Uncorrected data, where the batch effect caused the cells to cluster by batch. The results for all methods are shown in Supplementary Figure S9. Figure 5 highlights the results from Canek and Seurat. Seurat (Figure 5C) and MNN (Supplementary Figure S9g) mixed the batches almost perfectly. Canek (Figure 5B) was able to integrate the batches, but some differences remained. This dataset consists of samples from different technologies with varied numbers of cells sampled, so it possible that this falls outside Canek’s assumptions of small non-linear batch effects. Interestingly, the differences remaining in Canek integration are correlated with disease state (Supplementary Figure S10), with some of the samples containing type 2 diabetes whereas other containing only healthy individuals. Thus, it is also possible to speculate that some of the observed differences represent true biological differences.

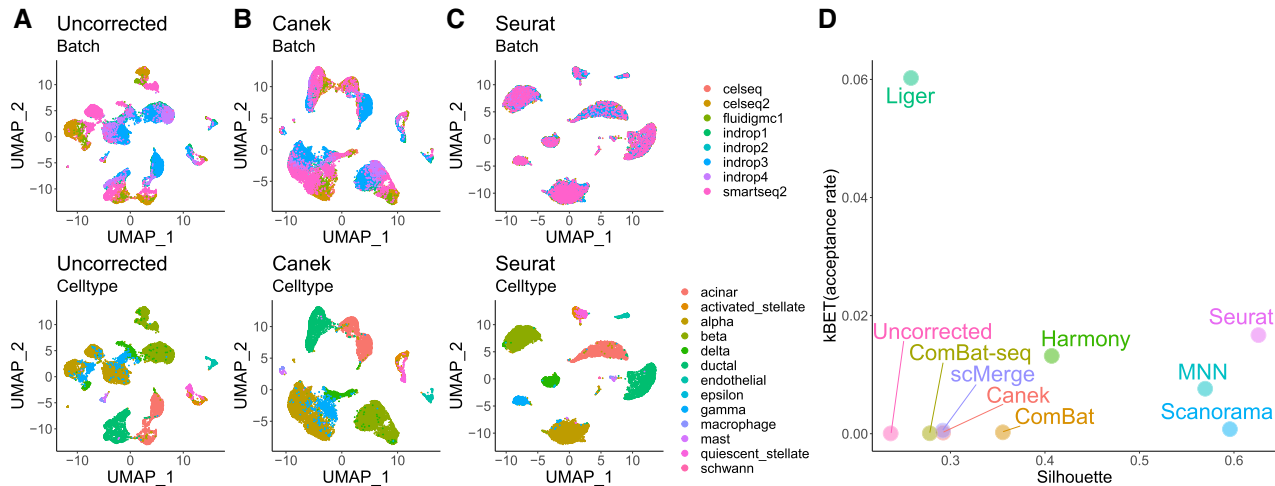


Figure 5. Integration datasets from different technologies. Eight pancreatic datasets obtained using different technologies were corrected. (A) UMAP of Uncorrected dataset. Batch effects caused the cells to cluster by batch instead of by cell type. (B, C) UMAP of dataset after Canek and Seurat integrations. (D) kBET and Silhouette metrics for Uncorrected, and the nine methods compared.

Finally, we evaluated the scenario wherein two samples from different conditions were assayed using the same technology. For this we integrated a dataset obtained from PBMCs with and without interferon beta stimulation (29). In this scenario, differences between the same cell types due to the stimulation were expected. Supplementary Figure S11a shows that in the Uncorrected data, cells separate by batch. Supplementary Figures S11b-j shows the correction done by Canek and 8 other methods. After integrating with Canek, B cells and T cells were almost completely integrated but some differences remained. Differences in monocytes and dendritic cells in the stimulated vs. non-stimulated cells were more prominent. This is in agreement with experiments showing that interferon beta induces stronger changes in gene expression in monocytes compared to T cells (37). To further validate this observation, we identified clusters in the integrated Canek dataset and identified two clusters of CD14+ monocytes, cluster 1 coming primarily from control and cluster 3 made primarily from interferon beta stimulated cells (Supplementary Figure S12). We calculated differentially expressed genes between these two clusters and found many differentially expressed genes, as shown in the volcano plot in Supplementary Figure S12D. Among the genes, the expression of EDN1 was up-regulated in the cluster of IFNB stimulated monocytes, whereas IL1B and RXRA were down-regulated (Supplementary Figure S12E, F). This is in agreement with the results reported in the original publication where these genes were experimentally validated (29).

Integration of a human lung dataset

To evaluate how Canek performs on the task of integrating samples from replicated experiments, we used a human lung single cell dataset with 78 samples including IPF ($n = 32$; idiopathic pulmonary fibrosis), COPD ($n = 18$; chronic obstructive pulmonary disease), and control donors ($n = 28$) (30). This dataset consisted of 312 928 cells distributed over 107 sequencing libraries that we treated as different batches.

Figure 6A and B shows that Canek integration resulted in good mixing among libraries while preserving the cell populations identified in the original publication. These cell types closely correlated with cell clusters based on Canek integration (Figure 6C). Most cells were distributed evenly among all cell types and disease conditions (Figure 6D). Enrichment and depletion in cell populations associated with disease were preserved (Supplementary Figure S13). A group of macrophages enriched in IPF in clusters 12 (interstitial macrophages expressing matrix metalloproteinase 9; MMP9, Figure 6E–G) and 16 (alveolar macrophages) showed cells in a transitional state almost exclusively from IPF donors (30). This demonstrates that Canek can integrate a high number of replicated datasets while preserving biologically meaningful information.

Computational performance

To compare the computational performance and scalability of Canek and eight other batch correction methods, we simulated datasets using splatter and recorded the integration time. We fixed the number of genes to 2000 and varied the number of cells from 10k to 100k. Figure 7 shows run times as a function of the number of cells. The fastest method was ComBat, followed by Scanorama, Harmony and Canek, all of which showed near linear run time dependence and ability to integrate 100k cells in under 20 min. On the other side of the spectrum MNN, Seurat, and ComBat-seq showed a strong dependence of run time on data size. These results demonstrated that Canek is a scalable method that can integrate hundreds of thousands of single-cell transcriptomes efficiently.

DISCUSSION

Existing batch effect correction methods focus on integrating single-cell transcriptomics datasets obtained from different technologies and/or species, minimizing the differences among batches to obtain correlated cell types. While

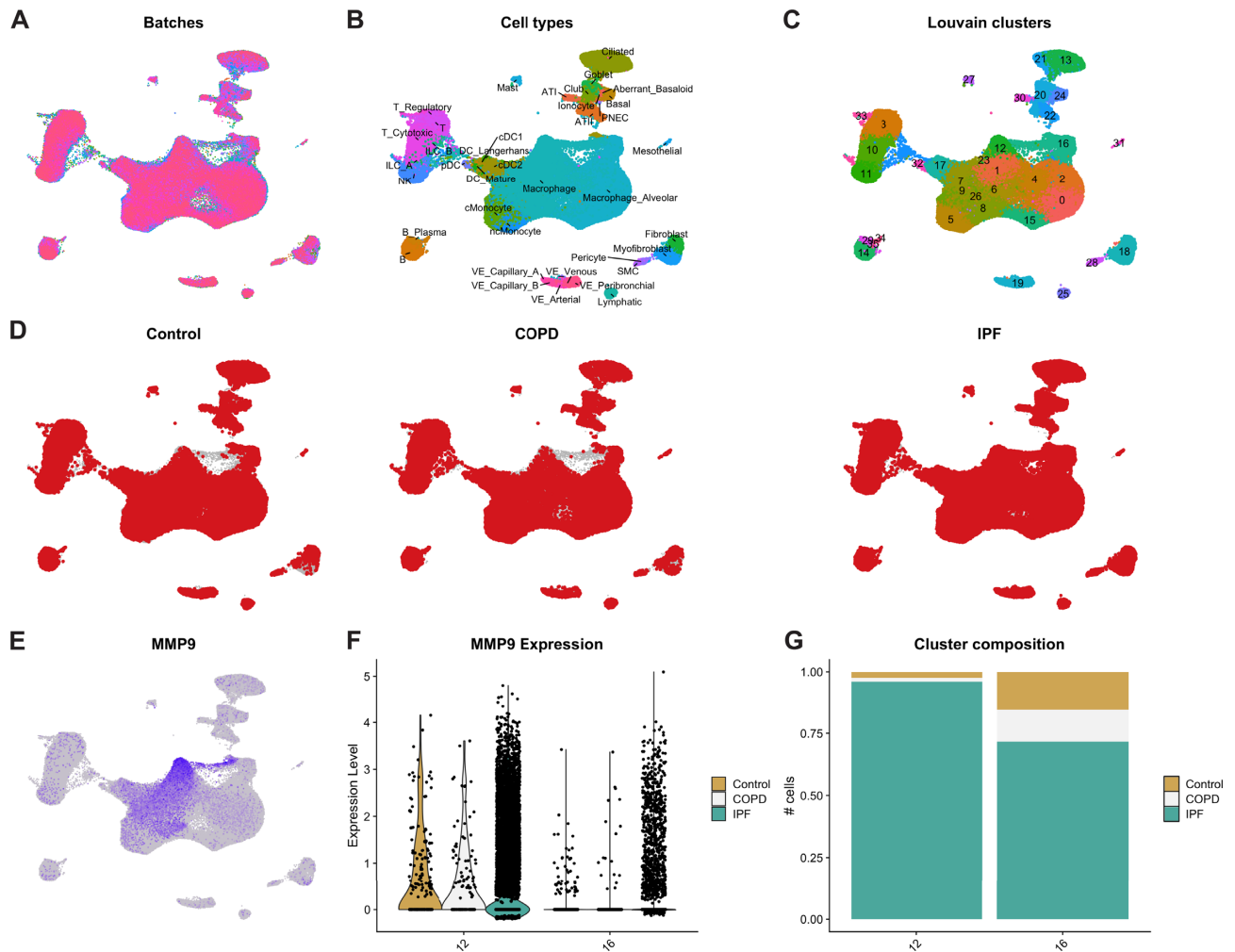


Figure 6. Integration of the lung dataset from Adams (2020). (A) UMAP plot showing the mixing between batches. (B) Cell populations described in the original publication are preserved. (C) Clustering based on Canek integration matches with cell populations. (D) Distribution of cells by disease condition shows even distribution except for a group of IPF specific cells in clusters 12 and 16. (E) MMP9 is highly expressed in cluster 12 (interstitial macrophages), (F) in IPF donors. (G) Cluster 12 and 16 are enriched in cells from IPF.

these frameworks offer a powerful solution to integrate datasets with strong differences between batches, they may also introduce significant biases due to over-correction. This represents a potential problem when these methods are applied to datasets where we wish to preserve biological differences (e.g. replicated experiments obtained with the same technology). Over-correction could negatively affect downstream tasks such as clustering or differential gene expression analysis. Canek provides an unbiased batch effect correction method for single-cell transcriptomics data that is suited for such integration of experimental replicates. We focused on preserving the inherent biological structure while being flexible enough to deal with small non-linear differences that might appear on heterogeneous datasets. We applied Canek to simulated and real datasets and showed its ability to correct batch effects without masking real biological signals. We also tested Canek on a pseudo-batch scenario with no batch effect and observed that it preserved the biological structure and introduced the least undesirable bias among tested methods.

We further showed that Canek successfully integrated datasets from different technologies (e.g. the Tabula Muris spleen dataset). Depending on the nature of the dataset, Canek did not necessarily lead to the best batch mixing (e.g. in the human pancreatic islet integration). However, latent variables other than batch effects (i.e. disease condition) may influence the integration of these datasets. It is an open question how to integrate complex datasets with co-founding variables.

The main goal of Canek is to enable efficient and unbiased integration of replicated experiments. Thus, we applied Canek to a large dataset from human lung disease. We identified enrichment of cell populations reported in the original publication, including cells in an apparently transitional state between interstitial to alveolar macrophages. This showed that Canek was able to integrate large numbers of replicated experiments while preserving biological information.

Single cell genomics datasets performed on other modalities like protein (CITE-seq) and chromatin accessibil-

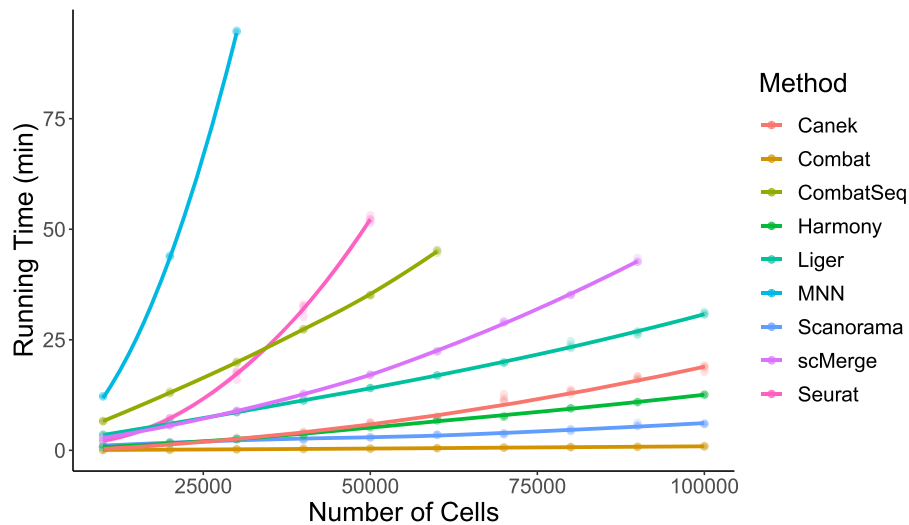


Figure 7. Runtime benchmark of Canek and other eight batch correction methods. Each method was run 5 times on different datasets with the number of genes fixed to 2k and the number of cells varying in a range of 5k to 100k. The color code differentiates each of the methods, the dots represent the runtime, and the lines represent the time increasing trends. Canek displayed a linear time increase over these conditions.

ity (scATAC-seq) are becoming more popular. Furthermore, integration of datasets with multi-modal (e.g. RNA + ATAC) measurements are beginning to appear. We are working to extend Canek's approach to the integration of multi-modal datasets.

As single-cell RNA-seq from replicated experiments using the same technology become more common, batch effect correction methods that conserve local differences will become more important. Canek provides a solution to this problem with an unbiased and computationally efficient batch effect correction.

DATA AVAILABILITY

The datasets used for the simulation and pseudo-batch tests are available from figshare: <https://figshare.com/projects/Canek/122638>.

CODE AVAILABILITY

Canek is implemented as an R package and is available from GitHub: <https://github.com/MartinLoza/Canek>. Code used to replicate the analyses presented in this paper is available from: <https://github.com/MartinLoza/WorkflowsCanek>.

SUPPLEMENTARY DATA

Supplementary Data are available at NARGAB Online.

ACKNOWLEDGEMENTS

We thank Dr Alexis Vandenbon for valuable comments on our manuscript.

Author contributions: M.L. conceived this work with contributions from S.T. and D.D. D.M.S. provided guidance for benchmarking and performance evaluation. M.L. and

D.D. implemented the methods. M.L. performed the analyses with contributions from D.D. M.L. and D.D. wrote the manuscript with contributions from S.T. and D.M.S. All authors edited and approved the submitted manuscript.

FUNDING

This work was supported by JSPS KAKENHI Grant Numbers JP20K06610 and JP20K07538, and by the Platform Project for Supporting Drug Discovery and Life Science Research (Basis for Supporting Innovative Drug Discovery and Life Science Research (BINDS)) from AMED under Grant Number 21am0101108j0005. M.L. was supported by a scholarship from The Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan. *Conflict of interest statement.* None declared.

REFERENCES

- Svensson,V., Vento-Tormo,R. and Teichmann,S.A. (2018) Exponential scaling of single-cell RNA-seq in the past decade. *Nat. Protoc.*, **13**, 599–604.
- Leek,J.T., Scharpf,R.B., Bravo,H.C., Simcha,D., Langmead,B., Johnson,W.E., Geman,D., Baggerly,K. and Irizarry,R.A. (2010) Tackling the widespread and critical impact of batch effects in high-throughput data. *Nat. Rev. Genet.*, **11**, 733–739.
- Tran,H.T.N., Ang,K.S., Chevrier,M., Zhang,X., Lee,N.Y.S., Goh,M. and Chen,J. (2020) A benchmark of batch-effect correction methods for single-cell RNA sequencing data. *Genome Biol.*, **21**, 12.
- Haghverdi,L., Lun,A.T.L., Morgan,M.D. and Marioni,J.C. (2018) Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.*, **36**, 421–427.
- Stuart,T., Butler,A., Hoffman,P., Hafemeister,C., Papalexi,E., Mauck,W.M., Hao,Y., Stoeckius,M., Smibert,P. and Satija,R. (2019) Comprehensive integration of single-cell data. *Cell*, **177**, 1888–1902.
- Polański,K., Young,M.D., Miao,Z., Meyer,K.B., Teichmann,S.A. and Park,J.E. (2020) BBKNN: fast batch alignment of single cell transcriptomes. *Bioinformatics*, **36**, 964–965.
- Hie,B., Bryson,B. and Berger,B. (2019) Efficient integration of heterogeneous single-cell transcriptomes using scanorama. *Nat. Biotechnol.*, **37**, 685–691.

8. Korsunsky, I., Millard, N., Fan, J., Slowikowski, K., Zhang, F., Wei, K., Baglaenko, Y., Brenner, M., Loh, P.R. and Raychaudhuri, S. (2019) Fast, sensitive and accurate integration of single-cell data with harmony. *Nat. Methods*, **16**, 1289–1296.
9. Welch, J.D., Kozareva, V., Ferreira, A., Vanderburg, C., Martin, C. and Macosko, E.Z. (2019) Single-cell multi-omic integration compares and contrasts features of brain cell identity. *Cell*, **177**, 1873–1887.
10. Luecken, M.D., Büttner, M., Chaichoompu, K., Danese, A., Interlandi, M., Mueller, M.F., Strobl, D.C., Zappia, L., Dugas, M., Colomé-Tatché, M. *et al.* (2022) Benchmarking atlas-level data integration in single-cell genomics. *Nat. Methods*, **19**, 41–50.
11. Pearson, K. (1901) LIII. On lines and planes of closest fit to systems of points in space. *London Edinburgh Dublin Philos. Mag. J. Sci.*, **2**, 559–572.
12. Baglama, J., Reichel, L. and Lewis, B.W. (2021) irlba: Fast Truncated Singular Value Decomposition and Principal Components Analysis for Large Dense and Sparse Matrices. R package version 2.3.5.
13. Beygelzimer, A., Kakadet, S., Langford, J., Arya, S., Mount, D. and Li, S. (2019) FNN: Fast Nearest Neighbor Search Algorithms and Applications. R package version 1.1.3.
14. Strogatz, S. (2001) *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering (studies in nonlinearity)*.
15. Csardi, G. and Nepusz, T. (2006) The igraph software package for complex network research. *InterJ., Complex Syst.*, **1695**, 1–9.
16. Takagi, T. and Sugeno, M. (1985) Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst. Man Cybern.*, 116–132.
17. Butler, A., Hoffman, P., Smibert, P., Papalexi, E. and Satija, R. (2018) Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.*, **36**, 411–420.
18. Johnson, W.E., Li, C. and Rabinovic, A. (2007) Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, **8**, 118–127.
19. Zhang, Y., Parmigiani, G. and Johnson, W.E. (2020) ComBat-seq: batch effect adjustment for RNA-seq count data. *NAR Genom Bioinform*, **2**, lqaa078.
20. Lin, Y., Ghazanfar, S., Wang, K.Y.X., Gagnon-Bartsch, J.A., Lo, K.K., Su, X., Han, Z.G., Ormerod, J.T., Speed, T.P., Yang, P. *et al.* (2019) scMerge leverages factor analysis, stable expression, and pseudoreplication to merge multiple single-cell RNA-seq datasets. *Proc. Natl. Acad. Sci. U.S.A.*, **116**, 9775–9784.
21. McInnes, L., Healy, J., Saul, N. and Großberger, L. (2018) UMAP: Uniform Manifold Approximation and Projection. *J. Open Source Softw.*, **3**, 861.
22. Zappia, L., Phipson, B. and Oshlack, A. (2017) Splatter: simulation of single-cell RNA sequencing data. *Genome Biol.*, **18**, 174.
23. Wickham, H. (2016) *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag, NY.
24. Zheng, G.X., Terry, J.M., Belgrader, P., Ryvkin, P., Bent, Z.W., Wilson, R., Ziraldo, S.B., Wheeler, T.D., McDermott, G.P., Zhu, J. *et al.* (2017) Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.*, **8**, 14049.
25. Tabula Muris, C. and Overall, C., Logistical, C., Organ, c., processing, Library, p., sequencing, Computational data, a., Cell type, a., Writing, g. *et al.* (2018) Single-cell transcriptomics of 20 mouse organs creates a tabula muris. *Nature*, **562**, 367–372.
26. Baron, M., Veres, A., Wolock, S.L., Faust, A.L., Gaujoux, R., Vetere, A., Ryu, J.H., Wagner, B.K., Shen-Orr, S.S., Klein, A.M. *et al.* (2016) A single-cell transcriptomic map of the human and mouse pancreas reveals Inter- and Intra-cell population structure. *Cell Syst.*, **3**, 346–360.
27. Muraro, M.J., Dharmadhikari, G., Grun, D., Groen, N., Dielen, T., Jansen, E., van Gurp, L., Engelse, M.A., Carlotti, F., de Koning, E.J. *et al.* (2016) A single-cell transcriptome atlas of the human pancreas. *Cell Syst.*, **3**, 385–394.
28. Segerstolpe, A., Palasantza, A., Eliasson, P., Andersson, E.M., Andreasson, A.C., Sun, X., Picelli, S., Sabirsh, A., Clausen, M., Bjursell, M.K. *et al.* (2016) Single-Cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes. *Cell Metab.*, **24**, 593–607.
29. Kang, H.M., Subramaniam, M., Targ, S., Nguyen, M., Maliskova, L., McCarthy, E., Wan, E., Wong, S., Byrnes, L. and Lanata, C.M. (2018) Multiplexed droplet single-cell RNA-sequencing using natural genetic variation. *Nat. Biotechnol.*, **36**, 89.
30. Adams, T.S., Schupp, J.C., Poli, S., Ayaub, E.A., Neumark, N., Ahangari, F., Chu, S.G., Raby, B.A., DeLullis, G., Januszyn, M. *et al.* (2020) Single-cell RNA-seq reveals ectopic and aberrant lung-resident cell populations in idiopathic pulmonary fibrosis. *Sci. Adv.*, **6**, eaba1983.
31. R. C. Team (2021) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.
32. Grün, D., Muraro, M.J., Boisset, J.C., Wiebrands, K., Lyubimova, A., Dharmadhikari, G., van den Born, M., van Es, J., Jansen, E., Clevers, H. *et al.* (2016) De novo prediction of stem cell identity using single-cell transcriptome data. *Cell Stem Cell*, **19**, 266–277.
33. Lawlor, N., George, J., Bolisetty, M., Kursawe, R., Sun, L., Sivakamasundari, V., Kycia, I., Robson, P. and Stitzel, M.L. (2017) Single-cell transcriptomes identify human islet cell signatures and reveal cell-type-specific expression changes in type 2 diabetes. *Genome Res.*, **27**, 208–222.
34. Büttner, M., Miao, Z., Wolf, F.A., Teichmann, S.A. and Theis, F.J. (2019) A test metric for assessing single-cell RNA-seq batch correction. *Nat. Methods*, **16**, 43–49.
35. Rousseeuw, P.J. (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, **20**, 53–65.
36. Argelaguet, R., Cuomo, A.S.E., Stegle, O. and Marioni, J.C. (2021) Computational principles and challenges in single-cell data integration. *Nat. Biotechnol.*, **39**, 1202–1215.
37. Henig, N., Avidan, N., Mandel, I., Staun-Ram, E., Ginzburg, E., Paperna, T., Pinter, R.Y. and Miller, A. (2013) Interferon-beta induces distinct gene expression response patterns in human monocytes versus t cells. *PLoS One*, **8**, e62366.