# BURMUDA: A novel deep transfer learning method for single-cell RNA sequencing batch correction reveals hidden high-resolution cellular subtypes

Tongxin Wang[1], Travis S Johnson[2, 3], Wei Shao[3], Jie Zhang[4*] and Kun Huang[3, 5*]

[1]Department of Computer Science, Indiana University Bloomington, Bloomington, USA

[2]Department of Biomedical Informatics, The Ohio State University, Columbus, USA

[3]Department of Medicine, Indiana University School of Medicine, Indianapolis, USA

[4]Department of Medical and Molecular Genetics, Indiana University School of Medicine, Indianapolis, USA

[5]Regenstrief Institute, Indianapolis, USA

* Correspondence: jizhan@iu.edu, kunhuang@iu.edu

## Abstract

To fully utilize the power of single-cell RNA sequencing (scRNA-seq) technologies for generating cell lineages and identifying *bona fide* transcriptional signals, it is necessary to combine data from multiple experiments. We present *BERMUDA* (Batch-Effect ReMoval Using Deep Autoencoders), a novel transfer-learning-based method for batch-effect correction in scRNA-seq data. *BERMUDA* can be effectively applied to batches with vastly different cell population compositions, and can properly combine different batches while transferring biological information from one batch to amplify the corresponding signals in other batches. We demonstrate that *BERMUDA* outperforms multiple existing methods in both removing batch effects and separating different cell types.

**Keywords:** Single-cell, RNA-seq, Batch effect, Transfer learning, Autoencoder

## Background

Understanding cellular transcriptional dynamics at the single-cell resolution is crucial to identify disease mechanisms and understand normal cell physiology in heterogeneous tissue settings. Up until recently, the

gold standard techniques for single-cell analysis were flow cytometry [1, 2] and confocal imaging of tissue on slides [3, 4]. Though these techniques were useful, they could only focus on a small, predefined set of genetic markers [1]. In contrast, high-throughput RNA sequencing (RNA-seq) techniques could measure thousands of genetic markers, but were initially designed for tissue samples [5, 6] due to the amount of materials required for library preparation and signal detection. First proposed in 2009 [7], single-cell RNA sequencing (scRNA-seq) can measure expression profiles of thousands of genes for individual cells, which provides an unprecedented resolution of cellular heterogeneity in complex tissue mixture settings. The rapid advances in scRNA-seq techniques have produced large numbers of scRNA-seq datasets, which presents new computational challenges for evaluating experimental reproducibility and combining data from different batches and platforms [5, 8-12].

There have been many attempts to combine gene expression data from different experiments in order to achieve a more comprehensive understanding of the underlying biology, which resulted in the development of multiple software tools. The first set of tools proposed were based on linear model analysis on microarray data [13-15], which were further modified for RNA-seq data via generalized linear models [16] or negative binomial models [17]. These methods represent the foundation of batch-effect removal in scRNA-seq data. However, unlike microarray or bulk RNA-seq data, scRNA-seq data is especially prone to “drop out” events, where viable RNAs might not be amplified in library preparation [18, 19]. Moreover, due to the heterogeneity of biological samples, cell population compositions between different scRNA-seq experiments can be vastly different. As a result, scRNA-seq data usually come with large batch effects (*i.e.*, systematic variance), often greater than the signal of the underlying biological variance [20, 21].

*Seurat-CCA* (*Seurat v2*) [21] and *mnnCorrect* [22] are among the first methods proposed to combine scRNA-seq data from multiple batches properly. *Seurat v2* uses canonical correlation analysis (CCA) to project cells from different experiments to a common bias-reduced low-dimensional representation. However, this type of correction does not account for the variations in cellular heterogeneity among studies, such as different cell types or different numbers cells within each cell type. Alternatively, to account for this heterogeneity, *mnnCorrect* utilizes mutual nearest neighbors (MNN) between batches to identify matching cell types through MNN pairs [22]. By identifying the corresponding cells, a cell-specific correction can be learned for each MNN pair. As a result of performing batch-correction locally, *mnnCorrect* avoids the

assumption of similar cell population compositions between batches required by previous methods. Following *mnnCorrect*, a series of new methods have been developed to integrate scRNA-seq data from different experiments [23-27]. For example, in order to account for the cellular heterogeneity between studies, *Seurat v3* [23] utilizes MNN pairs between the reference batch and query batches to detect "anchors" in the reference batch. "Anchors" represents cells in a shared biological state across batches and are further used to guide the batch correction process through CCA. *BBKNN* [24] takes advantage of a more efficient process for creating neighborhood graphs, which are further incorporated into visualization and clustering of the cells. However, most existing batch correction methods for scRNA-seq data rely on generating neighborhood graphs based on pairwise similarities between cells, which do not fully utilize the clustering structures of different cell populations to identify the optimal batch-corrected subspace.

In this paper, by considering scRNA-seq data from different batches as different domains, we take advantage of the domain adaptation framework in deep transfer learning to properly remove batch effects by finding a low-dimensional representation of the data. Different from existing methods, the proposed method, *BERMUDA* (Batch-Effect ReMoval Using Deep Autoencoders), utilizes the similarities between cell clusters to align corresponding cell populations among different batches. We demonstrate that *BERMUDA* outperforms existing methods in combining different batches and separating different cell types in the joint dataset. By optimizing the maximum mean discrepancy (MMD) [28] between clusters across different batches, *BERMUDA* can be applied to batches with *vastly different cell population compositions* as long as there is one common cell type shared between a pair of batches. Comparing to existing methods, *BERMUDA* can also preserve biological signals that only exists in a subset of batches when removing batch effects. These improvements indeed provide a novel deep learning solution to a persisting problem in scRNA-seq data analysis, demonstrating the state-of-the-art in batch-effect correction.

## Results

### Framework of *BERMUDA*

We propose *BERMUDA*, a novel unsupervised framework to remove batch effects across different batches by training an autoencoder (Figure 1a). After preprocessing the scRNA-seq data to select highly

variable genes, we first use a graph-based clustering algorithm to detect cell clusters in each batch individually. Then, we apply a correlation-based approach to evaluate the similarity between cell clusters from different batches. Each pair of cell clusters is assigned with a similarity score, which is later used as the coefficient in the transfer loss. Next, an autoencoder is trained using the standardized and scaled transcript-per-million (TPM) value of highly variable genes to learn a low-dimensional embedding of the gene expression profiles, where the systematic biases across different batches are removed (Figure 1b). In order to successfully remove batch effects, we propose a novel approach by combining the reconstruction loss with the transfer loss when training the autoencoder. The reconstruction loss is calculated between the input and output of the autoencoder, which helps to learn a low-dimensional embedding that properly represents the original high-dimensional gene expression data. The transfer loss is applied to pairs of cell clusters in order to merge similar clusters from different batches in the low-dimensional representation. Finally, the low-dimensional code from the trained autoencoder is used for further downstream analysis.

The popularly used UMAP [29] method is chosen to visualize the cell type clustering results. Besides it, three evaluation metrics are proposed to evaluate the performance of *BERMUDA* (See the "Methods" section). $Divergence_score$ is an average of divergence of shared cell populations between pairs of batches, which indicates whether shared cell populations among different batches are mixed properly. $Entropy_score$ is an average of local entropy of distinct cell populations between pairs of batches, which can evaluate whether cell populations not shared by all the batches remain separate from other cells after batch correction. $Silhouette_score$ is calculated using cell type labels as clustering labels, which measures the quality of cell type assignment in the aligned dataset.

### ***BERMUDA* outperforms existing methods to remove batch effects on simulated data**

To assess the performance of *BERMUDA* for batch-effect removal, we first apply it on a simulated dataset with four shared virtual cell types, where the expression profiles are generated from a 2-dimensional biological subspace following the method in [22] (referred to as "2D Gaussian dataset". See the "Methods" section). In order to recover and better visualize the underlying biological subspace, we set the number of

104 neurons in the bottleneck layer of the autoencoder to two. Following the aforementioned approach, we
105 apply our method in three different scenarios:

106 - all the cells from the two batches (referred to as "*Experiment all*");
107 - holding out Type1 in *Batch1* (referred to as "*Experiment hold1*");
108 - holding out Type1 in *Batch1* and Type4 in *Batch2* at the same time (referred to as "*Experiment*
109 *hold2*").

110 *Experiment hold2* represents the most difficult case in these three experiments due to only two cell types
111 shared by both batches. We evaluate the results by inspecting the 2-dimensional visualizations (Figure 2a,
112 Additional file 1: Figure S1). Ideally, the visualizations generated after proper batch correction should
113 contain four separate cell clusters, each representing a cell type, and the clusters for shared cell types
114 should contain a homogenous mixture of cells from both batches. In *Experiment hold2*, *BERMUDA* properly
115 removes batch effects (Figure 2a1-2) while both *mnnCorrect* and *BBKNN* generate false similarities
116 between cell types that do not exist in the original data (Figure 2a4-5). Specifically, in *Experiment hold2*,
117 clusters corresponding to Type1 and Type4 are closely connected in the *mnnCorrect* results (Figure 2a4),
118 and clusters corresponding to Type1, Type3, and Type4 are not as well separated in the *BBKNN* results
119 (Figure 2a5). Moreover, only *BERMUDA* produces proper batch correction consistently across three
120 different experiments (Figure 2a, Additional file 1: Figure S1). For example, *BBKNN* incorrectly separates
121 Type1 from the same batch into two distinct clusters in *Experiment hold1* (Additional file 1: Figure S1b5).
122 This suggests that although *mnnCorrect* and *BBKNN* can deal with differences in cell population
123 composition among different batches, their performance is less optimal compared with BURMUDA when
124 such differences are large.

125 In order to compare with the popularly used scRNA-seq data analysis tool *Seurat v2* [21] and the
126 recently-proposed *Seurat v3* [23], where the workflows are designed for scRNA-seq count data, we then
127 evaluate *BERMUDA* using simulated single-cell RNA sequence counts generated by Splatter [30] (referred
128 to as "Splatter dataset"). Similarly, we also conduct multiple experiments using different cell populations,
129 with *Experiment hold2* representing the most difficult case:

130 - all the cells from two batches (referred to as "*Experiment all*");
131 - holding out Type1 in *Batch1* (referred to as "*Experiment hold1*");

132 • holding out Type1 in *Batch1* and Type4 in *Batch2* (referred to as "*Experiment hold2*").

133 When only two cell types are shared between two batches in *Experiment 2*, only *BERMUDA* can properly
134 remove batch effects and align corresponding cell types (Figure 2b1-2), while other methods improperly
135 merge Type1 with Type4 in the UMAP visualizations (Figure 2b4-7). Moreover, *BERMUDA* is the only
136 method that can consistently remove batch effects in all three cases. When all the four cell types are shared
137 in both batches, we observe that all the methods properly merge cells of the same type (Additional file 1:
138 Figure S2a). However, *BBKNN* and *Seurat v2* produce much lower $silhouette_score$ values due to the
139 inflated variance within cell clusters comparing with the original data (Additional file 1: Figure S2c). Also,
140 when the difference in cell type distributions is introduced in *Experiment hold1*, we observe that *BBKNN*
141 and *Seurat v2* can no longer mix the same cell type from different batches properly (Additional file 1: Figure
142 S2b4-5). The results of *Seurat v2* are anticipated since it is designed to align different batches globally
143 without considering the population differences among batches. The inconsistent results of *BBKNN* under
144 different cell type distributions indicate that the neighborhood graphs generated by *BBKNN* may not always
145 be reliable since the method mainly focused on computational efficiency.

146 The observations through UMAP visualizations are further confirmed by evaluating the results through
147 the proposed $divergence_score$ , $entropy_score$ , and $silhouette_score$ (Figure 2c). The axis of
148 $silhouette_score$ is reserved so that points closer to the bottom-left corner in all the plots indicate more
149 desirable results. We observe that *BERMUDA* consistently yield the best performance in all three cases
150 when evaluating using the proposed metrics (Figure 2c, Additional file 1: Figure S2c-d).

151 By evaluating *BERMUDA* on two different simulated datasets, we discover that it achieves better
152 performance than existing methods in removing batch effects, especially when the difference of cell
153 population compositions among batches is large.

154 **BERMUDA outperforms existing methods to remove batch effects on human pancreas data**

155 To further evaluate *BERMUDA* on real biological data, we apply it on publicly available human pancreas
156 datasets that were generated with different scRNA-seq protocols. Muraro et al. [31] used CEL-Seq2, a
157 multiplexed linear amplification RNA sequencing technique (referred to as "*Muraro batch*"), while Baron et
158 al. [32] used a droplet RNA-seq technology (referred to as "*Baron batch*"). Besides applying our method on

159 the entire dataset (referred to as "*Experiment all*", Figure 3a-b, Additional file 1: Figure S4a), we also hold
160 out alpha and beta cells from *Baron batch* (referred to as "*Experiment hold*", Additional file 1: Figure S4b-
161 d) to simulate the situation where vast cell type distribution difference exists across different batches in real
162 scRNA-seq data.

163 *BERMUDA* achieves competitive results in both cases compared with *mnnCorrect*, *BBKNN*, *Seurat v2*,
164 and *Seurat v3*. Specifically, the performance of *Seurat v2* deteriorates when alpha and beta cells are
165 removed from the *Baron batch* (Figure 3b, Additional file 1: Figure S4d) because it utilizes CCA not capable
166 to address the case that only a subset of cell types is shared among batches. *BBKNN* can generally
167 separate different cell types into different clusters, but fails to mix cells from different batches within clusters
168 properly (Figure 3a5, Additional file 1: Figure S4a5), producing a high $divergence_score$. *Seurat v3* and
169 *mnnCorrect* are designed to cope with the situation where cell type distributions across datasets are
170 different. However, *Seurat v3* cannot produce consistent results with clear separation between cell types
171 when the cell population composition varies within the same set of data. For example, when holding out
172 alpha and beta cells from *Baron batch*, *Seurat v3* produces a UMAP visualization that represents alpha,
173 beta, gamma and delta cells as tightly connected clusters, which is different from both the original data and
174 the *Seurat v3* result in *Experiment all* (Figure 3a7, Additional file 1: Figure S4b7). The *mnnCorrect* method
175 produces relatively good results across both cases (Figure 3a4, Additional file 1: Figure S4b4). However,
176 *BERMUDA* still outperforms all these existing methods when evaluating using the three proposed metrics
177 jointly (Figure 3b, Additional file 1: Figure S4d).

178 ***BERMUDA* preserves batch-specific biological signals in human pancreas data**

179 In this paper, we use "batch-specific" biological signals to refer to *bona fide* biological signals that are
180 not shared by all the batches. A homogeneous mixture of the same cell type among different batches is
181 one of the indications of proper batch corrections. However, an overly batch-corrected homogeneous
182 mixture could result in the loss of the subtle batch-specific biological signals, and defeat the purpose of
183 carrying out scRNA-seq. One of the advantages of *BERMUDA* is the flexibility of adjusting between a more
184 homogeneous mixture of cell types while still retaining the subtle structures within cell types by decreasing
185 and increasing the value of $S_{thr}$. Sometimes, the latter could be more desirable since it could relate to

186 potential batch-specific cell subtypes in the original data. We demonstrate experimentally that choosing
187 $S_{thr}$ between 0.85 and 0.90 can consistently produce proper batch correction in different datasets through
188 assessing the performance of *BERMUDA* using a wide range of $S_{thr}$ values (Additional file 1: Figure S8-
189 S9). For example, in *Experiment all* on the pancreas dataset, *BERMUDA* produces competitive results in
190 both $S_{thr}$ settings (Figure 3b). However, the difference is that, when $S_{thr} = 0.85$, each cell type is separated
191 as a single cluster with a homogenous mixture of cells from both batches (Figure 3a1), while when $S_{thr} =$
192 $0.90$, we observe two closely connected cell clusters in both alpha and beta cells within the *Baron batch*
193 (Figure 3a2), which is consistent with the original uncorrected data (Figure 3a3). We also notice that when
194 $S_{thr} = 0.90$, only one of two connected clusters in alpha or beta cells in *Baron batch* merges with the
195 corresponding cells in the *Muraro batch*, while the other mostly remains unmixed (Figure 3a2, Additional
196 file 1: Figure S4a2). This observation is further confirmed by the *mnnCorrect* and *Seurat v3* results, although
197 with a subtler distinction of clusters (Figure 3a4, 3a7). The consistent observation indicates that the distinct
198 cell clusters might relate to certain alpha or beta cell subtypes that only appear in the *Baron batch*, which
199 can be visualized through choosing a higher $S_{thr}$ value of 0.90.

200 To further identify the potential cell subtypes, we perform differential gene expression analysis within
201 each individual cell type of alpha or beta cells on two pairs of cell populations. For alpha cells, we perform
202 differential gene expression analysis on the ones from the mixed cluster in *Muraro batch* versus the ones
203 from the unmixed cluster in *Baron batch*, as well as on the ones from the mixed cluster versus the ones
204 from the unmixed cluster within *Baron batch*. For beta cells, we perform the same two sets of differential
205 gene expression analysis between the respective mixed cluster and the unmixed cluster. We use the
206 "FindMarkers" function in *Seurat v2* [21] to identify genes that have significantly different expression
207 patterns in both pairs of cell populations (Additional file 1: Table S1-S2), which results in genes related to
208 important pancreatic functions. For example, *ARX* and *MAFB* (Table S1, Figure 3c1, 3c4) are significantly
209 under-expressed in the unmixed alpha cells compared with the mixed alpha cells. *PDX1*, *MAFA*, and *MAFB*
210 are significantly under-expressed in the unmixed beta cells compared with the mixed beta cells (Table S2,
211 Figure 3c2-4). Previous studies [33, 34] have shown that *ARX* plays a key role in the differentiation of
212 pancreatic islet cells. Moreover, the decrease of *MAFA*, *MAFB*, and *PDX1* expression levels have been
213 found in human Type 2 diabetes islet alpha and beta cells, which can be associated to islet cell dysfunction

214 [35]. The identification of above transcriptomic signatures suggests that the refined cell clusters that only
215 exist in *Baron batch* may contain cell subpopulations with altered pancreatic functions which may warrant
216 a further biological investigation. As mentioned above, such cell subpopulations have been also identified
217 in *Seurat v3* and *mnnCorrect*.

218 By applying our method on human pancreas datasets, we demonstrate that *BERMUDA* can effectively
219 remove batch effects under different cell population compositions across batches. *BERMUDA* outperforms
220 existing methods in combining corresponding cell types, preserving subtle cell clusters that are not shared
221 by all the batches, and properly separating different cell types. We also show that *BERMUDA* can preserve
222 and even amplify biological meaningful structures within cell types when integrating different batches.

223 ***BERMUDA* properly maps human pan T cells to PBMCs**

224 In order to show that *BERMUDA* can transfer information between batches with more complicated cell
225 populations and reveal biological signals that might remain hidden when analyzing each batch individually,
226 we also apply *BERMUDA* to two large scRNA-seq datasets of peripheral blood mononuclear cells (PBMCs)
227 generated from the 10X Genomics Chromium platform (*i.e.*, "*PBMC batch*" and "*pan T cell batch*"). *PBMC*
228 *batch* and *pan T cell batch* are acquired from different healthy donors. We expect that cell types in the *pan*
229 *T cell batch* should be a subset of those in the *PBMC batch*, while the *PBMC batch* could contain other cell
230 types such as B cells and monocytes. From the UMAP visualization of the uncorrected data (Additional file
231 1: Figure S5c), we speculate that the *pan T cell batch* should roughly correspond to the largest cell cluster
232 in the *PBMC batch*, while the other two smaller clusters should represent non-T cells in the *PBMC batch*.
233 As shown in Figure 4b, *BERMUDA* successfully retains the two PBMC-specific clusters, while mapping the
234 T cell population properly.

235 Since the obtained data do not include pre-defined cell type labels, we evaluate the results by inspecting
236 the expression patterns of marker genes using the UMAP visualizations. We identify seven major cell types
237 in our results through corresponding marker genes: CD4 T cells (Figure 4a1, 4a3), CD8 T cells (Figure 4a3),
238 B cells (Figure 4a2), CD1c+ dendritic cells (Figure 4a6), CD14+ monocytes (Figure 4a4), CD16+ monocytes
239 (Figure 4a5), and NK/NKT cells (Figure 4a7-8). Cell types that only exist in the *PBMC batch* (B cells, CD14+
240 monocytes, CD16+ monocytes, CD1c+ dendritic cells) retain a similar structure with the uncorrected original

241 data (Additional file 1: Figure S5a-c), which further demonstrate that our method can properly combine
242 different batches while preserving biological information only exists in certain batches. Surprisingly and
243 interestingly, the UMAP visualization of *pan T cell batch* alone does not show any clear clustering structures.
244 However, different cell types within *pan T cell batch* are further distinguished as different clusters by
245 combining with PBMC cells. This indicates that *BERMUDA* is capable of effectively transferring the
246 biological information from one batch to another, and extracting biological insights from batches that might
247 not be evident by studying each batch individually.

248 Contrarily, *Seurat v2* and *Seurat v3* fail to align the overlapping cell types from two batches properly
249 (Additional file 1: Figure S5f-g). *BBKNN* overly corrects the batch effects and lost the structure within the T
250 cell cluster and the monocyte cluster observed in the original uncorrected data (Additional file 1: Figure
251 S5e). *mnnCorrect* aligns the shared cell types between two batches properly, while producing a more
252 heterogeneous structure within the T cell cluster when comparing with the uncorrected data visually
253 (Additional file 1: Figure S5g).

254 ***BERMUDA* can be generalized to combine multiple batches**

255 Since *BERMUDA* is based on applying MMD loss to similar clusters from different batches for batch-
256 effect removal, it can be easily generalized to deal with multiple batches simultaneously. Here, we
257 demonstrate such capability by including another human pancreas dataset generated by Segerstolpe et al.
258 [36] (referred to as "*Segerstolpe batch*"). The *Segerstolpe batch* was generated using completely different
259 sequencing platform SMART-seq2 technology and contains 2,061 cells of interest.

260 Similarly, two experiments are performed on the human pancreas dataset: one with all the cells in the
261 dataset (referred to as "*Experiment all*", Additional file 1: Figure S6) and one with alpha and beta cells
262 removed from the *Baron batch* and the *Segerstolpe batch* (referred to as "*Experiment hold*", Additional file
263 1: Figure S7). *BERMUDA* consistently produces competitive results in both experiments (Additional file 1:
264 Figure S6c, S7c). Moreover, the aforementioned potential alpha and beta cell subpopulations in the *Baron*
265 *batch* are also consistently observed when setting $S_{thr} = 0.90$. This demonstrates that our method can
266 naturally handle multiple batches simultaneously and generate consistent and robust results when number
267 of batches increases.

## Discussion

With the rapid advances of single cell sequencing technologies and accumulation large scRNA-seq data, it is important to combine data from different samples or studies to fully harness the power of scRNA-seq techniques, infer cell lineages, and identify *bona fide* transcriptional signals from vast amount of cellular signals. Proper removal of batch effects between different scRNA-seq experiments has become an important and urgent issue. However, batch effects may arise from multiple sources such as different protocols, platforms, or lab-specific artifacts. Besides various sources of batch effects, due to the heterogenous nature of biological samples, cell type compositions across different batches could also be substantially different, which increases the difficulty to distinguish *bona fide* biological signals from systematic biases and to correctly combine data from multiple batches.

In this study, we propose *BERMUDA*, a novel batch correction method based on the deep transfer learning framework. We consider scRNA-seq data from different batches as different domains and utilize domain adaptation approaches to project the original data to a lower-dimensional space without batch effects. We demonstrate that *BERMUDA* can properly combine data from different batches in both simulated data and real datasets as long as there is one common cell type shared a pair of batches. We show that *BERMUDA* outperforms existing methods (*mnnCorrect*, *BBKNN*, *Seurat v2*, and *Seurat v3*) in three aspects: homogeneously mixing the same cell type from different batches, maintaining the purity of batch-specific cell clusters, and separating different cell types in the combined dataset. By training an autoencoder with MMD loss on pairs of similar clusters, *BERMUDA* can effectively remove batch effects without considering their specific sources. Because of this, it is capable of removing batch effects resulted from various sources, such as simulated batch effects, batch effects from the same platform in the PMBC dataset, and batch effects generated by different platforms in the human pancreas dataset. Moreover, by using a deep neural network framework and training through mini-batch gradient descent, *BERMUDA* is scalable to large number of cells, which fits to the rapid growth of the scale of scRNA-seq experiments.

Different from some existing methods that do not account for the differences in cell type distributions across batches, *BERMUDA* only requires each batch shares at least one common cell type with another batch. Since our method performs batch correction based on similarity between cell clusters, it can naturally

295 accommodate not only the variation of cell population compositions across batches, but also variation of
296 batch effects across different cell populations. Rather than considering a global batch effect, *BERMUDA*
297 removes batch effects locally through combining similar clusters, while maintaining the global structure of
298 the data by reconstructing the original data through autoencoders. By applying our method to analyze
299 multiple batches jointly, knowledge from one batch can be transferred to another to shed light on more
300 biological insights, which might be hard to discover by examining each batch individually. For example,
301 when directly visualizing the *pan T cell batch*, it did not show obvious clustering structures (Figure 4c).
302 However, by combining it with PBMCs, pan T cells are properly mapped to the corresponding cell clusters
303 in the *PBMC batch*, providing a higher resolution of cell subtypes within the *pan T cell batch* (Figure 4b).
304 Usually, it is a trade-off between removing batch effects and retaining experimental specific biological
305 signals, and some existing methods (such as *BBKNN* and *Seurat v2*) may lose subtle biological signals
306 while merging multiple batches for a more homogenously mixture of cells. *BERMUDA* can preserve and
307 amplify the sensitivity to biological differences in the original data while properly remove batch effects.
308 Specifically, the trade-off between a more homogeneous mixture within cell types and preserving more
309 batch-specific biological signals could be adjusted by changing the $S_{thr}$ value, where a higher $S_{thr}$ value
310 help to retain more biological signals that only exist in a subset of batches. We demonstrate experimentally
311 that *BERMUDA* is robust and can outperform existing methods in a wide range of $S_{thr}$ values, and choosing
312 $S_{thr}$ between 0.85 and 0.90 can consistently yield a good balance between proper batch correction and
313 preserving batch-specific biological signals (Additional file 1: Figure S8, S9).
314 *BERMUDA* also has limitations. Since the design of *BERMUDA* is based on scRNA-seq data with
315 distinct cell populations, it may suffer in cases of data with continuous structures, such as data generated
316 by experiments investigating cell differentiation. Although such data could still fit into our framework by
317 adjusting the resolution in the graph-based clustering algorithm and the trade-off between reconstruction
318 loss and transfer loss to align clusters at a more granular level, improvements will be made to *BERMUDA*
319 in the future to handle this kind of data more effectively. Another limitation is that, while our neural-network-
320 based framework for batch correction is designed to handle large datasets, the use of k-nearest neighbor
321 in the clustering algorithm (integrated in *Seurat v2*) before aligning cells may not scale well to extremely
322 large datasets [24]. However, novel clustering algorithms could be applied in the future to speed up the

323 clustering process. For example, Ren et al. [37] proposed a novel framework for clustering large scRNA-
324 seq data, which reduced the computational complexity to O(n) while maintaining a high clustering accuracy.

## 325 Conclusions

326 Removing batch effects has become essential to the analysis of data from multiple scRNA-seq
327 experiments and multiple technical platforms. Here, we introduce *BERMUDA*, a novel batch-correction
328 method for scRNA-seq data based on deep transfer learning. We use an autoencoder to learn a low-
329 dimensional representation of the original gene expression profiles while removing the batch effects locally
330 by incorporating MMD loss on similar cell clusters. *BERMUDA* provides several improvements compared
331 to existing methods. Firstly, by introducing three different metrics to evaluate the batch correction
332 performance, we demonstrate that *BERMUDA* outperforms existing methods in merging the same cell types,
333 preserving cell types that are not shared by all the batches, and separating different cell types. Secondly,
334 *BERMUDA* can properly remove batch effects even when the cell population compositions across different
335 batches are vastly different. Thirdly, *BERMUDA* can preserve batch-specific biological signals and discover
336 new information that might be hard to extract by analyzing each batch individually. Finally, *BERMUDA* can
337 be easily generalized to handle multiple batches and can scale to large datasets.

## 338 Methods

### 339 Framework of BERMUDA

340 *BERMUDA* is a novel unsupervised framework for batch correction across different batches (Figure
341 1a). The workflow of *BERMUDA* includes the following five steps: 1. Preprocessing of scRNA-seq data. 2.
342 Clustering of cells in each batch individually. 3. Identifying similar cell clusters across different batches. 4.
343 Removing batch effects by training an autoencoder (Figure 1b). 5. Utilizing the batch-corrected codes for
344 further downstream analysis. We introduce each step in detail in the following sections.

### 345 Preprocessing

Gene expression level from each cell is first quantified using TPM. Seurat [21] (v2.3.4) is then used to identify highly variable genes for further analysis. We normalize the gene expression level of variable genes as

$$G_{ij} = \log_2(\mathrm{TPM}_{ij} + 1),$$

where $\mathrm{TPM}_{ij}$ is the TPM of gene $i$ in cell $j$. We further use z-score to standardize the expression level of each batch as

$$G'_{ij} = \frac{G_{ij} - \bar{G}_i}{\sigma_i},$$

where $G'_{ij}$ is the standardized expression level of gene $i$ in cell $j$. $\bar{G}_i$ is the mean expression level for gene $i$ and $\sigma_i$ is the standard deviation of expression level for gene $i$. Finally, we linearly scale the expression level of each gene to $[0, 1]$.

**Clustering of cells and identifying similar cell clusters**

We first identify cell clusters from each batch individually following the pipeline in *Seurat v2* [21]. Seurat implements a clustering algorithm based on optimizing the modularity function on a k-nearest neighbor graph. We then take advantage of MetaNeighbor [9] to determine the similarity between clusters from different batches based on Spearman correlation. For $n$ batches each contains $c_i$ clusters, $i = 1, 2, \dots, n$, MetaNeighbor produces a similarity score for each pair of cell clusters by calculating the mean of area under the receiver operator characteristic curve (AUROC) in cross-validation. We denote $M_{i_1,j_1,i_2,j_2}$ as the similarity score between cluster $j_1$ in batch $i_1$ and cluster $j_2$ in batch $i_2$. Since we are only interested in similar clusters across different batches, we set the similarity score between clusters within the same batch to 0. For each cluster, we only consider the most similar cluster in each of the other batches, such that

$$M_{i_a,j_a,i_b,j_b} = \begin{cases} M_{i_a,j_a,i_b,j_b}, & \text{if } M_{i_a,j_a,i_b,j_b} = \max\{M_{i_a,j_a,i_b,j}, j = 1, 2, \dots, c_{i_b}\} \\ 0, & \text{otherwise} \end{cases}.$$

To account for the cases where one cluster in a batch could corresponds to multiple clusters in another batch and make the similarity scores between two clusters symmetrical, we further modify $M_{i_a,j_a,i_b j_b}$ as

$$M_{i_a,j_a,i_b,j_b} = \max\{M_{i_a,j_a,i_b,j_b}, M_{i_b,j_b,i_a,j_a}\}, i_a, i_b = 1,2, \dots, n, j_a = 1,2, \dots, c_{i_a}, j_b = 1,2, \dots, c_{i_b}.$$

Finally, we binarize $M_{i_a,j_a,i_b,j_b}$ with a threshold value $S_{thr}$, where

$$M_{i_a,j_a,i_b,j_b} = 1 \text{ if } M_{i_a,j_a,i_b,j_b} \geq S_{thr}, i_a, i_b = 1,2,\dots,n, j_a = 1,2,\dots,c_{i_a}, j_b = 1,2,\dots,c_{i_b}.$$

$S_{thr}$ is chosen empirically and we observe that *BERMUDA* achieves robust and competitive results across different datasets when $S_{thr}$ is between 0.85 and 0.90 (Additional file 1: Figure S8, S9).

**Batch correction using an autoencoder**

We use an autoencoder to project the original uncorrected gene expression profiles to a low-dimensional space to remove the experimental artifacts across different batches (Figure 1b). An autoencoder can be represented as a function $\mathbf{x}' = f(\mathbf{x}) = f_{decoder}(f_{encoder}(\mathbf{x}))$ , where $f$ is trying to reconstruct the input gene expression profile $\mathbf{x}$ through the neural network. To avoid trivial solutions, autoencoders usually have a bottleneck layer that learns a low-dimensional embedding of the input data called code, *e.g.*, $\mathbf{z} = f_{encoder}(\mathbf{x})$. Here, we use an autoencoder with three hidden layers, and the default number of neurons in each hidden layer are 200, 20, 200. For the synthetic data generated from bivariate Gaussian distributions, we set the neurons in each hidden layer as 20, 2, 20 to reconstruct the 2-dimensional biological plane.

Training of the autoencoder is performed with mini-batches $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_\mathrm{B}\}$, where each mini-batch contained 50 cells from each cluster ($\mathrm{B} = 50\sum_{i=1}^{n} c_i$). The loss function for training the autoencoder consists of two parts. The first part is a reconstruction loss between the output layer and the input layer defined by mean squared error

$$L_1(\mathbf{X}, \mathbf{X}') = \sum_{i=1}^{B} \|\mathbf{x}_i - \mathbf{x}_i'\|_2^2,$$

where $\mathbf{x}_i$ and $\mathbf{x}_i'$ are input and reconstructed expression profile of the $i$-th cell in a mini-batch.

The second part of the loss function is a novel MMD based loss [28] that estimates the differences in distributions between similar cell clusters in different batches. MMD is a non-parametric distance estimate between distributions based on the reproducing kernel Hilbert space (RKHS). Considering the case where only a subset of the cell population is shared among batches, instead of applying MMD loss on batches entirely, we only consider the loss between pairs of similar cell clusters among different batches. So, the MMD-based loss can be defined as

$$L_2(\mathbf{X}, \mathbf{X}') = \sum_{i_a,i_b,j_a,j_b} M_{i_a,j_a,i_b,j_b} \cdot MMD(\mathbf{Z}_{i_a,j_a}, \mathbf{Z}_{i_b,j_b}),$$

397 where $\mathbf{Z}_{i,j}$ is the code of the input $\mathbf{X}_{i,j}$, and $\mathbf{X}_{i,j}$ is the expression profiles of cells from cluster $j$ of dataset $i$
398 in the mini-batch $\mathbf{X}$. $MMD(\cdot)$ equals to zero when the underlying distributions of the observed samples are
399 the same. By minimizing the MMD loss between the distributions of similar clusters, the autoencoder can
400 be trained remove batch effects in the bottleneck layer. In summary, the total loss function on a mini-batch
401 can be written as

402 $$L(\mathbf{X}, \mathbf{X}') = L_1(\mathbf{X}, \mathbf{X}') + \lambda L_2(\mathbf{X}, \mathbf{X}'),$$

403 where $\lambda$ is a regularization parameter. We gradually increase the value of $\lambda$ during the training process,
404 which can help the autoencoder to first find a proper low-dimensional representation of the original gene
405 expression data, then focus on aligning the distributions of similar clusters in the low-dimensional space.

406 **Datasets used for performance evaluation**

407 For the consistency of the presentation, in this paper, we refer a set of datasets investigating similar
408 biological problems as a dataset and each individual dataset within this set of datasets as a batch. We
409 apply *BERMUDA* on simulated datasets, human pancreas cell datasets, and PBMC datasets to assess its
410 performance (Table 1). We use two methods to generate simulated datasets for evaluating the performance
411 of *BERMUDA*. For the 2D Gaussian dataset, we follow [22] to generate highly synthetic data with batch
412 effects. We simulate two batches of four different cell types according to different bivariate normal
413 distributions in a 2-dimensional biological subspace. The cell population composition of each batch is
414 generated randomly. Then we randomly project the data to a 100-dimensional space to simulate the high-
415 dimensional gene expression data. Gene-specific batch effects are generated by adding Gaussian noise
416 to the high-dimensional data. For the Splatter dataset, we use the Splatter [30] package to simulate RNA
417 sequence counts of two batches with four different cell populations. Splatter can directly simulate multiple
418 batches following similar cell type compositions at the same time. We set the cell population composition
419 to be 0.4, 0.3, 0.2, and 0.1 among the four simulated cell types.

420 To evaluate whether *BERMUDA* can remove batch effects in real scRNA-seq data and extract
421 meaningful biological insights, we also apply it to datasets of human pancreas cells and PBMCs. The
422 pancreas dataset is obtained from Gene Expression Omnibus (GSE85241 for *Muraro batch* [31],
423 GSE84133 for *Baron batch* [32]) and The European Bioinformatics Institute (E-MTAB-5061 for *Segerstolpe*

*batch* [36]). The PBMC dataset is obtained from 10X Genomics support datasets. To effectively compare the difference between the cases where all the cell types or only a subset of those are shared among batches, we only retain the shared cell types in the pancreas dataset. The details of the datasets are shown in 错误!未找到引用源。.

**Performance Evaluation**

To evaluate the performance of *BERMUDA* when different subsets of the cell population are shared among batches, we perform different experiments on a dataset by holding out certain cell types in certain batches. We propose three different metrics to evaluate three different aspects of the results.

Firstly, we use a k-nearest-neighbor based divergence estimation method [38] to evaluate the quality of merging the shared cell population among batches. For $N$ scRNA-seq batches with gene expression profiles $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N$ and their corresponding batch-corrected low-dimensional embeddings $\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_N$, we define

$$divergence_score = \frac{1}{n_{div}} \sum_{i \neq j, s(\mathbf{Z}_i, \mathbf{Z}_j) \neq \emptyset, s(\mathbf{Z}_j, \mathbf{Z}_i) \neq \emptyset} D\left(s(\mathbf{Z}_i, \mathbf{Z}_j), s(\mathbf{Z}_j, \mathbf{Z}_i)\right),$$

where $s(\mathbf{Z}_i, \mathbf{Z}_j)$ is the cell population in $\mathbf{Z}_i$ that is shared by $\mathbf{Z}_j$, $D(\mathbf{Z}_i, \mathbf{Z}_j)$ is the divergence estimation of the two distributions given samples $\mathbf{Z}_i$ and $\mathbf{Z}_j$, and $n_{div} = |\{(i,j) | \, i \neq j, s(\mathbf{Z}_i, \mathbf{Z}_j) \neq \emptyset, s(\mathbf{Z}_j, \mathbf{Z}_i) \neq \emptyset\}|$ is the number of pairs of batches with shared cell population. Since proper removal of batch effects should produce results where the distributions of shared cell populations among different batches are similar, a smaller $divergence_score$ is preferred, indicating that the shared cell population between different datasets are homogeneously mixed.

Secondly, we use entropy to evaluate whether a cell population that only exists in a certain batch remains distinct from other populations after correction. We define

$$entropy_score = \frac{1}{n_{ent}} \sum_{i \neq j, d(\mathbf{Z}_i, \mathbf{Z}_j) \neq \emptyset} \sum_{\mathbf{k} \in d(\mathbf{Z}_i, \mathbf{Z}_j)} E(\mathbf{k}),$$

where $d(\mathbf{Z}_i, \mathbf{Z}_j)$ is the cell population in $\mathbf{Z}_i$ that is not shared by $\mathbf{Z}_j$, and $n_{ent} = |\{(i,j) | \, i \neq j, d(\mathbf{Z}_i, \mathbf{Z}_j) \neq \emptyset\}|$ is the number of pairs of batches where there exists distinct cell population in $\mathbf{Z}_i$ from $\mathbf{Z}_j$. $E(\mathbf{k})$ is the estimation of entropy locally around cell $\mathbf{k}$ defined as

$$E(\mathbf{k}) = \sum_{i=1}^{N} p_i \log(p_i),$$

where $p_i$ is the proportion of cells from batch $i$ among the 100-nearest neighbors of cell $\mathbf{k}$. When batch effects are removed properly, a cell type that only exists in a batch should not be mixed with cells from other batches. So, a smaller $entropy_score$ is desired, suggesting that biological signals only contained in a subset of batches are properly preserved during correction. Note that when all the batches share the same cell types, we do *not* calculate $entropy_score$ during evaluation since there is no batch-specific cell population.

The above two scores are based on divergence and entropy estimations on pairs of batches and then averaged to acquire a summary of the batch-correction performance among multiple batches. When the dimensionality of the embedding is high, in order to get robust estimations of divergence and entropy, $divergence_score$ and $entropy_score$ are calculated based on the 2-dimensional UMAP [29] embeddings of the data. UMAP is a general dimensionality reduction algorithm that can achieve competitive results compared to t-SNE [39], while preserves more global structures of the data.

Thirdly, since $divergence_score$ and $entropy_score$ are both proposed to evaluate mixture of cells among batches, we also need a metric to evaluate the separation of different cell types after batch effects being removed. Here, we use the silhouette coefficient with clusters defined by cell types. For a cell $\mathbf{k}$, let $a(\mathbf{k})$ be the average distance between $\mathbf{k}$ and all the other cells within the same cluster and $b(\mathbf{k})$ be the smallest average distance between $\mathbf{k}$ and all the cells in any other cluster, we define the silhouette coefficient of cell $\mathbf{k}$ as

$$S(\mathbf{k}) = \frac{b(\mathbf{k}) - a(\mathbf{k})}{\max\{a(\mathbf{k}), b(\mathbf{k})\}}.$$

The average silhouette coefficient of all the cells in different batches is calculated after batch-effect removal, such that

$$silhouette_score = \frac{1}{\sum_{i=1}^{N} |\{\mathbf{k} | \mathbf{k} \in \mathbf{Z}_i\}|} \sum_{i=1}^{N} \sum_{\mathbf{k} \in \mathbf{Z}_i} S(\mathbf{k}).$$

A larger $silhouette_score$ indicates that the cell type assignment in the aligned dataset is more appropriate, where a cell is close to cells of the same type and distant from cells of different types. $S(\mathbf{k})$ is calculated using Euclidean distance on the 2-dimensional UMAP embeddings of the results.

**Performance comparison with popular batch correction methods**

476 We compare *BERMUDA* with several existing state-of-the-art batch correction methods for scRNA-seq
477 data, including *mnnCorrect* [22], *BBKNN* [24], *Seurat v2* (v2.3.4) [21], and *Seurat v3* (v3.0.0) [23]. *BBKNN*
478 and *mnnCorrect* are applied to log-transformed TPM data of variable genes. *Seurat v2* and *Seurat v3* are
479 applied on the datasets following the recommended workflow [21, 23]. Due to the restriction of the Seurat
480 workflow, we do not apply *Seurat v2* and *Seurat v3* on the Gaussian simulated gene expression data. To
481 demonstrate the necessity of batch correction methods for scRNA-seq data, we also compare *BERMUDA*
482 with batch correction methods for microarray and bulk RNA-seq data, such as *combat* [15] and *limma* [16]
483 (Additional file 1: Figure S3).

## 484 List of Abbreviations

485 **AUROC**: area under the receiver operator characteristic curve
486 **CCA**: canonical correlation analysis
487 **MMD**: maximum mean discrepancy
488 **MNN**: mutual nearest neighbors
489 **PBMC**: peripheral blood mononuclear cell
490 **RKHS**: reproducing kernel Hilbert space
491 **scRNA-seq**: single-cell RNA sequencing
492 **TMP**: transcript-per-million

## 493 Declarations

### 494 Ethics approval and consent to participate

495 Not applicable.

### 496 Consent for publication

497 Not applicable.

### 498 Availability of data and materials

499 The implementation of BERMUDA can be downloaded from Github

500 (https://github.com/txWang/BERMUDA).

501 **Competing interests**

502 Not applicable.

503 **Funding**

504 This work was partially supported by Indiana University School of Medicine (IUSM) start-up fund, the

505 National Cancer Institute Informatics Technology for Cancer Research (NCI ITCR) U01 [CA188547], and

506 Indiana University Precision Health Initiative.

507 **Authors' contributions**

508 Conceived and designed the experiments: TW, TSJ, WS. Performed the experiments and analyzed the

509 data: TW, TSJ. Developed the structure and arguments for the paper: TW, KH. Wrote the manuscript:

510 TW, TSJ, JZ. Edited the manuscript: WS, JZ, KH. All the authors reviewed and approved of the final

511 manuscript.

512 **Acknowledgements**

513 Not applicable.

## 514 Additional files

515 Additional file 1: Supplementary figures and tables.

## 516 References

517 1. Saeys Y, Van Gassen S, Lambrecht BN: **Computational flow cytometry: helping to make sense of**
518 **high-dimensional immunology data.** *Nat Rev Immunol* 2016, **16:**449.
519 2. Wang D, Bodovitz S: **Single cell analysis: the new frontier in 'omics'.** *Trends Biotechnol* 2010,
520 **28:**281-290.

521 3. Huang WE, Stoecker K, Griffiths R, Newbold L, Daims H, Whiteley AS, Wagner M: **Raman-FISH:**
522 **combining stable-isotope Raman spectroscopy and fluorescence in situ hybridization for the**
523 **single cell analysis of identity and function.** *Environ Microbiol* 2007, **9:**1878-1889.
524 4. Puppels GJ, de Mul FF, Otto C, Greve J, Robert-Nicoud M, Arndt-Jovin DJ, Jovin TM: **Studying**
525 **single living cells and chromosomes by confocal Raman microspectroscopy.** *Nature* 1990,
526 **347:**301-303.
527 5. Consortium SM-I: **A comprehensive assessment of RNA-seq accuracy, reproducibility and**
528 **information content by the Sequencing Quality Control Consortium.** *Nat Biotechnol* 2014,
529 **32:**903-914.
530 6. Marioni JC, Mason CE, Mane SM, Stephens M, Gilad Y: **RNA-seq: an assessment of technical**
531 **reproducibility and comparison with gene expression arrays.** *Genome Res* 2008, **18:**1509-1517.
532 7. Tang F, Barbacioru C, Wang Y, Nordman E, Lee C, Xu N, Wang X, Bodeau J, Tuch BB, Siddiqui A, et
533 al: **mRNA-Seq whole-transcriptome analysis of a single cell.** *Nat Methods* 2009, **6:**377-382.
534 8. Consortium M, Shi L, Reid LH, Jones WD, Shippy R, Warrington JA, Baker SC, Collins PJ, de
535 Longueville F, Kawasaki ES, et al: **The MicroArray Quality Control (MAQC) project shows inter-**
536 **and intraplatform reproducibility of gene expression measurements.** *Nat Biotechnol* 2006,
537 **24:**1151-1161.
538 9. Crow M, Paul A, Ballouz S, Huang ZJ, Gillis J: **Characterizing the replicability of cell types defined**
539 **by single cell RNA-sequencing data using MetaNeighbor.** *Nat Commun* 2018, **9:**884.
540 10. Draghici S, Khatri P, Eklund AC, Szallasi Z: **Reliability and reproducibility issues in DNA**
541 **microarray measurements.** *Trends Genet* 2006, **22:**101-109.
542 11. Labaj PP, Kreil DP: **Sensitivity, specificity, and reproducibility of RNA-Seq differential**
543 **expression calls.** *Biol Direct* 2016, **11:**66.
544 12. Shi L, Campbell G, Jones WD, Campagne F, Wen Z, Walker SJ, Su Z, Chu TM, Goodsaid FM,
545 Pusztai L, et al: **The MicroArray Quality Control (MAQC)-II study of common practices for the**
546 **development and validation of microarray-based predictive models.** *Nat Biotechnol* 2010,
547 **28:**827-838.
548 13. Chen C, Grennan K, Badner J, Zhang D, Gershon E, Jin L, Liu C: **Removing batch effects in**
549 **analysis of expression microarray data: an evaluation of six batch adjustment methods.** *PLoS*
550 *One* 2011, **6:**e17238.
551 14. Diboun I, Wernisch L, Orengo CA, Koltzenburg M: **Microarray analysis after RNA amplification**
552 **can detect pronounced differences in gene expression using limma.** *BMC Genomics* 2006,
553 **7:**252.
554 15. Johnson WE, Li C, Rabinovic A: **Adjusting batch effects in microarray expression data using**
555 **empirical Bayes methods.** *Biostatistics* 2007, **8:**118-127.
556 16. Ritchie ME, Phipson B, Wu D, Hu Y, Law CW, Shi W, Smyth GK: **limma powers differential**
557 **expression analyses for RNA-sequencing and microarray studies.** *Nucleic Acids Res* 2015,
558 **43:**e47.
559 17. Leek JT: **svaseq: removing batch effects and other unwanted noise from sequencing data.**
560 *Nucleic Acids Res* 2014, **42:**e161.
561 18. Li WV, Li JJ: **An accurate and robust imputation method scImpute for single-cell RNA-seq data.**
562 *Nat Commun* 2018, **9:**997.
563 19. Lin P, Troup M, Ho JW: **CIDR: Ultrafast and accurate clustering through imputation for single-**
564 **cell RNA-seq data.** *Genome Biol* 2017, **18:**59.
565 20. Johnson T, Abrams Z, Zhang Y, Huang K: **Mapping Neuronal Cell Types Using Integrative Multi-**
566 **Species Modeling of Human and Mouse Single Cell Rna Sequencing.** *Pac Symp Biocomput* 2016,
567 **22:**599-610.

21. Butler A, Hoffman P, Smibert P, Papalexi E, Satija R: **Integrating single-cell transcriptomic data across different conditions, technologies, and species.** *Nat Biotechnol* 2018, **36:**411-420.
22. Haghverdi L, Lun ATL, Morgan MD, Marioni JC: **Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors.** *Nat Biotechnol* 2018, **36:**421-427.
23. Stuart T, Butler A, Hoffman P, Hafemeister C, Papalexi E, Mauck WM, Stoeckius M, Smibert P, Satija R: **Comprehensive integration of single cell data.** *bioRxiv* 2018**:**460147.
24. Park J-E, Polanski K, Meyer K, Teichmann SA: **Fast Batch Alignment of Single Cell Transcriptomes Unifies Multiple Mouse Cell Atlases into an Integrated Landscape.** *bioRxiv* 2018**:**397042.
25. Barkas N, Petukhov V, Nikolaeva D, Lozinsky Y, Demharter S, Khodosevich K, Kharchenko PV: **Wiring together large single-cell RNA-seq sample collections.** *bioRxiv* 2018**:**460246.
26. Korsunsky I, Fan J, Slowikowski K, Zhang F, Wei K, Baglaenko Y, Brenner M, Loh P-R, Raychaudhuri S: **Fast, sensitive, and accurate integration of single cell data with Harmony.** *bioRxiv* 2018**:**461954.
27. Welch J, Kozareva V, Ferreira A, Vanderburg C, Martin C, Macosko E: **Integrative inference of brain cell similarities and differences from single-cell genomics.** *bioRxiv* 2018**:**459891.
28. Gretton A, Sejdinovic D, Strathmann H, Balakrishnan S, Pontil M, Fukumizu K, Sriperumbudur BK: **Optimal kernel choice for large-scale two-sample tests.** In *Adv Neural Inf Process Syst*. 2012: 1205-1213.
29. McInnes L, Healy J, Melville J: **Umap: Uniform manifold approximation and projection for dimension reduction.** *arXiv* 2018**:**03426.
30. Zappia L, Phipson B, Oshlack A: **Splatter: simulation of single-cell RNA sequencing data.** *Genome Biol* 2017, **18:**174.
31. Muraro MJ, Dharmadhikari G, Grun D, Groen N, Dielen T, Jansen E, van Gurp L, Engelse MA, Carlotti F, de Koning EJ, van Oudenaarden A: **A Single-Cell Transcriptome Atlas of the Human Pancreas.** *Cell Syst* 2016, **3:**385-394 e383.
32. Baron M, Veres A, Wolock SL, Faust AL, Gaujoux R, Vetere A, Ryu JH, Wagner BK, Shen-Orr SS, Klein AM, et al: **A Single-Cell Transcriptomic Map of the Human and Mouse Pancreas Reveals Inter- and Intra-cell Population Structure.** *Cell Syst* 2016, **3:**346-360 e344.
33. Collombat P, Mansouri A, Hecksher-Sorensen J, Serup P, Krull J, Gradwohl G, Gruss P: **Opposing actions of Arx and Pax4 in endocrine pancreas development.** *Genes Dev* 2003, **17:**2591-2603.
34. Gage BK, Asadi A, Baker RK, Webber TD, Wang R, Itoh M, Hayashi M, Miyata R, Akashi T, Kieffer TJ: **The Role of ARX in Human Pancreatic Endocrine Specification.** *PLoS One* 2015, **10:**e0144100.
35. Guo S, Dai C, Guo M, Taylor B, Harmon JS, Sander M, Robertson RP, Powers AC, Stein R: **Inactivation of specific beta cell transcription factors in type 2 diabetes.** *J Clin Invest* 2013, **123:**3305-3316.
36. Segerstolpe A, Palasantza A, Eliasson P, Andersson EM, Andreasson AC, Sun X, Picelli S, Sabirsh A, Clausen M, Bjursell MK, et al: **Single-Cell Transcriptome Profiling of Human Pancreatic Islets in Health and Type 2 Diabetes.** *Cell Metab* 2016, **24:**593-607.
37. Ren X, Zheng L, Zhang Z: **SSCC: a computational framework for rapid and accurate clustering of large-scale single cell RNA-seq data.** *bioRxiv* 2018**:**344242.
38. Wang Q, Kulkarni SR, Verdu S: **Divergence Estimation for Multidimensional Densities Via k-Nearest-Neighbor Distances.** *IEEE Trans Inf Theory* 2009, **55:**2392-2405.
39. Maaten Lvd, Hinton G: **Visualizing data using t-SNE.** *J Mach Learn Res* 2008, **9:**2579-2605.

## Figures

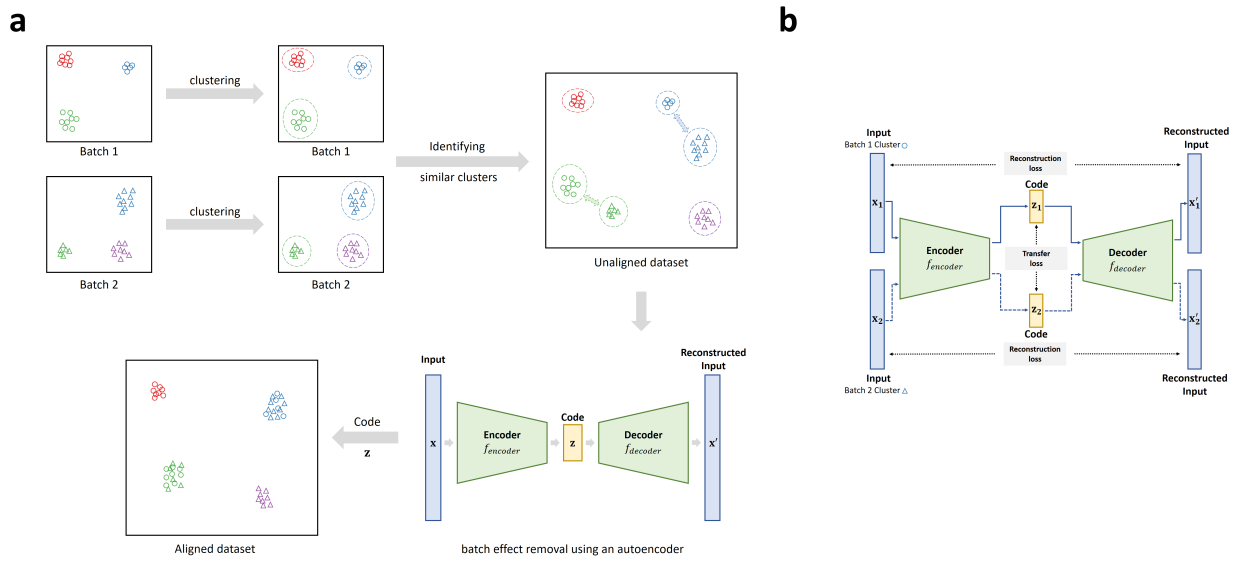


**Figure 1. Overview of BERMUDA for removing batch effects in scRNA-seq data.** a. The workflow of BERMUDA. Circles and triangles represent cells from *Batch 1* and *Batch 2*, respectively. Different colors represent different cell types. A graph-based clustering algorithm is first applied on each batch individually to detect cell clusters. Then, MetaNeighbor, a method based on Spearman correlation, is used to identify similar clusters between batches. We subsequently train an autoencoder to perform batch-correction on the code of the autoencoder. The code of the autoencoder is a low-dimensional representation of the original data without batch effects and can be used for further analysis. b. Training an autoencoder to remove batch effects. The blue solid lines represent training with the cells in *Batch 1* and the blue dash lines represents training with cells in *Batch 2*. The black dash lines represent the calculation of losses. The loss function we optimized contains two components: the reconstruction loss between the input and the output of the autoencoder, and the MMD-based transfer loss between the codes of similar clusters.

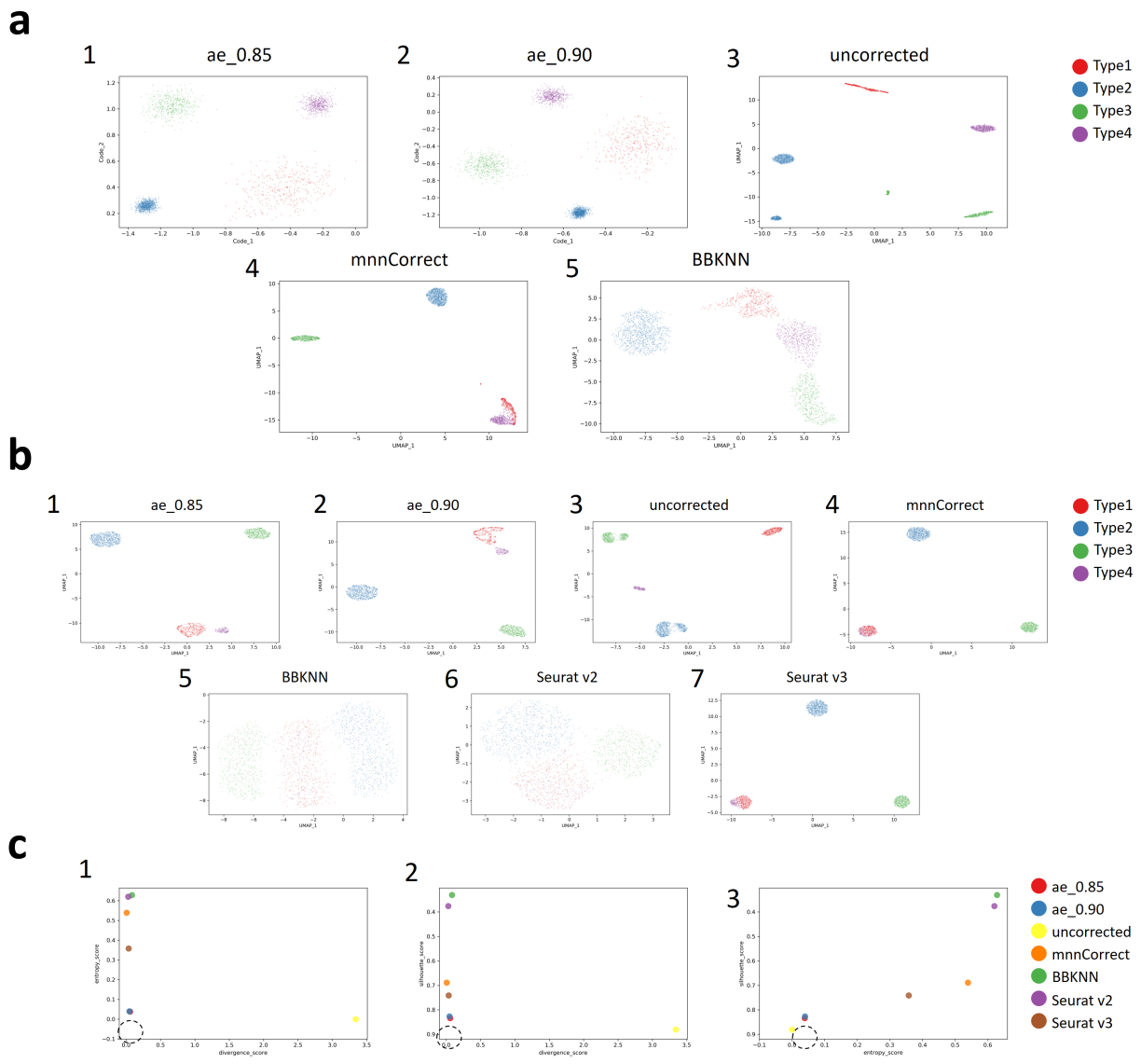


**Figure 2. Removing batch effects in simulated scRNA-seq data.** a. UMAP visualizations of results for 2D Gaussian dataset *Experiment hold2*, where Type1 from *Batch1* and Type4 from *Batch2* are held out. ae_0.85 and ae_0.90 represents results of BERMUDA with $S_{thr} = 0.85$ and $0.90$, respectively. b. UMAP visualizations of results for Splatter dataset *Experiment hold2*, where Type1 from *Batch1* and Type4 from *Batch2* are held out. c. Evaluation of batch-correction performance on Splatter dataset *Experiment hold2* using the proposed metrics. c1. $entropy_score$ *vs.* $divergence_score$. c2. $silhouette_score$ *vs.* $divergence_score$. c3. $silhouette_score$ *vs.* $entropy_score$. The $silhouette_score$ axis is reversed so that points close to the bottom-left corner indicate better results. The results of BERMUDA are highlighted by dashed circles. Our method outperforms existing methods in both parameter settings.

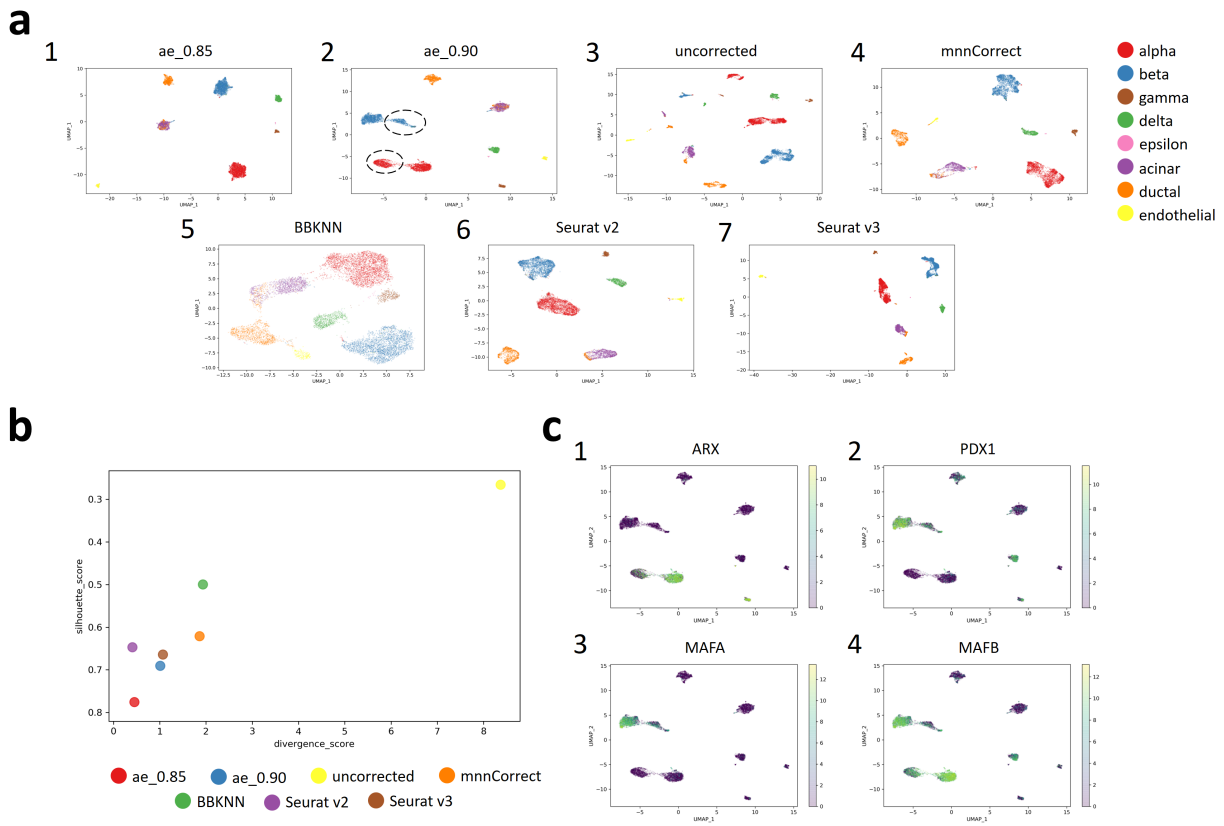


**Figure 3. Removing batch effects in scRNA-seq data of pancreas cells.** a. UMAP visualizations of batch-effect removal results for pancreas dataset *Experiment all*. Identified alpha and beta cell subpopulations in the *Baron batch* are highlighted with dashed circles. b. Evaluation of batch-correction performance on pancreas dataset *Experiment all* using the proposed metrics ($silhouette_score$ *vs.* $divergence_score$). *BERMUDA* outperforms existing methods in both parameter settings. c. Expression patterns of differently expressed genes within alpha and beta cells colored by log-transformed TPM values. Statistical significance of differential expression analysis is listed in Table S1-S2.

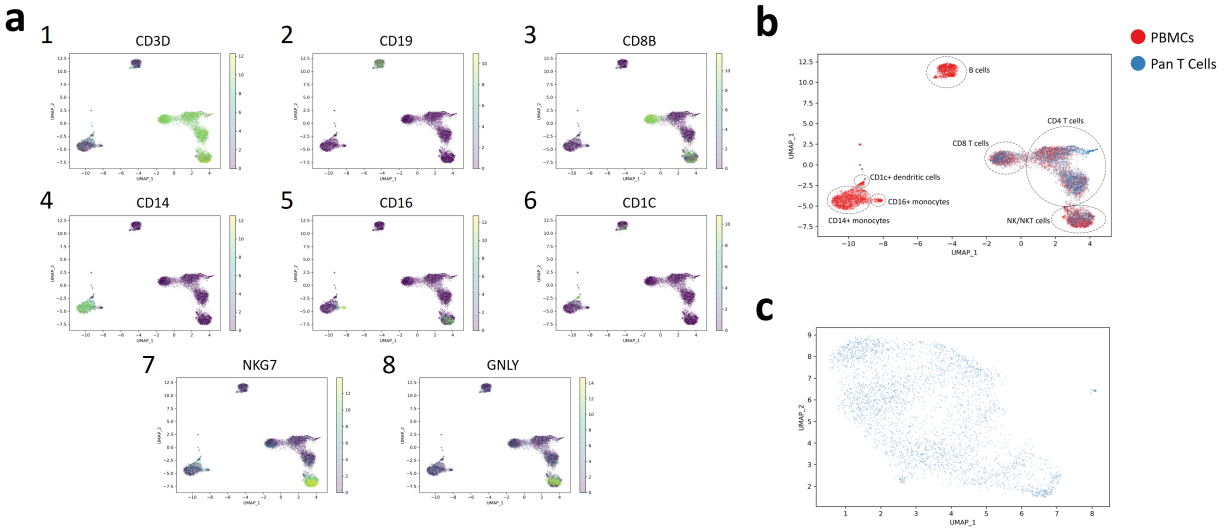


**Figure 4. Removing batch effects in scRNA-seq data of PBMCs.** a. Expression patterns of marker genes of immune cells colored by log-transformed TPM values. b. The UMAP visualization of results of *BERMUDA* colored by batches. Different cell types are identified by analyzing expression patterns of eigengenes and are highlighted by dashed circles. *BERMUDA* correctly combines the same cell types between different batches while preserves cell types not shared by both batches as separate clusters. c. The UMAP visualization of *pan T cell batch*. No obvious clustering structure is shown when visualizing *pan T cell batch* individually.

## Tables

**Table 1.** Datasets used for evaluation of *BERMUDA*.

| Dataset | Batch | Protocol | Number of cells | Cell Types (Number of cells) |
|---|---|---|---|---|
| 2D Gaussian | Batch1 | Simulation | 2000 | Type1 (855), Type2 (237), Type3 (373), Type4 (535) |
| | Batch2 | Simulation | 2000 | Type1 (379), Type2 (656), Type3 (102), Type4 (863) |
| Splatter | Batch1 | Splatter | 2000 | Type1 (810), Type2 (616), Type3 (388), Type4 (186) |

| | | | | |
|---|---|---|---|---|
| | Batch2 | Splatter | 1000 | Type1 (439), Type2 (262), Type3 (201), Type4 (98) |
| Pancreas | Muraro | CEL-Seq2 | 2042 | alpha (812), beta (448), gamma (101), delta (193), epsilon (3), acinar (219), ductal (245), endocrine (21) |
| | Baron | inDrop | 8012 | alpha (2326), beta (2525), gamma (255), delta (601), epsilon (18), acinar (958), ductal (1077), endocrine (252) |
| | Segerstolpe | SMART-seq2 | 2061 | alpha (886), beta (270), gamma (197), delta (114), epsilon (7), acinar (185), ductal (386), endocrine (16) |
| PBMC | PBMC | 10X Chromium | 8381 | |
| | Pan T Cell | 10X Chromium | 3555 | |

For the consistency in the paper, we refer to each pancreas and PBMC dataset, *e.g.*, the Murano dataset, as a batch, and refer to the set of datasets investigating similar biological systems, *e.g.*, pancreas, as a dataset. Type1 to Type4 in the 2D Gaussian and Splatter dataset refer to the virtual cell types that are generated using simulation techniques.