Molecules and Cells

Minireview

Integration of Single-Cell RNA-Seq Datasets: A Review of Computational Methods

Yeonjae Ryu[1,2], Geun Hee Han[1,2], Eunsoo Jung[1,2], and Daehee Hwang[1,*]

[1]School of Biological Sciences, Seoul National University, Seoul 08826, Korea, [2]These authors contributed equally to this work.
*Correspondence: daehee@snu.ac.kr
https://doi.org/10.14348/molcells.2023.0009
www.molcells.org

With the increased number of single-cell RNA sequencing (scRNA-seq) datasets in public repositories, integrative analysis of multiple scRNA-seq datasets has become commonplace. Batch effects among different datasets are inevitable because of differences in cell isolation and handling protocols, library preparation technology, and sequencing platforms. To remove these batch effects for effective integration of multiple scRNA-seq datasets, a number of methodologies have been developed based on diverse concepts and approaches. These methods have proven useful for examining whether cellular features, such as cell subpopulations and marker genes, identified from a certain dataset, are consistently present, or whether their condition-dependent variations, such as increases in cell subpopulations in particular disease-related conditions, are consistently observed in different datasets generated under similar or distinct conditions. In this review, we summarize the concepts and approaches of the integration methods and their pros and cons as has been reported in previous literature.

Keywords: batch correction, data integration, single-cell RNA-seq

INTRODUCTION

Single-cell RNA sequencing (scRNA-seq) datasets have been increasingly accumulated in public data repositories, such as the Single Cell Portal (https://singlecell.broadinstitute.org/single_cell), Gene Expression Omnibus (Barrett et al., 2013), and Human Cell Atlas (HCA) data portal (Regev et al., 2017). With an increase in the number of datasets, many efforts have been made for the integrative analysis of different scRNA-seq datasets. Multiple scRNA-seq datasets are often integrated and compared to check whether cellular features (e.g., cell subpopulations and their marker genes) identified from a certain dataset are shared or distinctive compared with those in other datasets produced under similar or different biological conditions. The integration of multiple scRNA-seq datasets has proven useful for reliably identifying shared or distinctive cellular features across datasets. Shared cellular features may not be clear when individual datasets are analyzed independently, owing to small numbers of cells or sparse expression data in each dataset and unwanted technical or biological variations within and across datasets. However, these cellular features can be corroborated by combining features from multiple datasets after correcting the unwanted variations.

In many cases, individual scRNA-seq datasets are generated from samples with distinctive characteristics (e.g., cell counts, tissue types, and conditions from which cells are isolated, such as healthy or diseased conditions) and using different experimental protocols (e.g., cell isolation and handling protocols and library preparation methods) or sequencing platforms. These differences inevitably lead to unwanted technical and biological variations across different datasets. Even within a single dataset, multiple batches with variations in sample characteristics, experimental protocols, or sequenc-

Received January 10, 2023; revised January 19, 2023; accepted January 19, 2023; published online February 24, 2023

eISSN: 0219-1032

ing platforms can exist. These unwanted variations among batches, called batch effects, within and across datasets can decrease the chances of identifying underlying cellular features by introducing inconsistent cellular expression profile structures. Therefore, batch effects arising from both systematic technical and unwanted biological variations should be corrected before integrative analysis of multiple datasets to prevent misleading conclusions. To effectively correct batch effects, several computational methods have been developed based on different concepts and approaches. In this review, we conceptually categorize existing integration methods, describe the key algorithms employed in these methods, and summarize their advantages and disadvantages as reported in literature.

RESULTS

Definition of sample batches in multiple datasets

Batch effects occur mainly due to differences in the following three factors: (1) sample characteristics (donors, tissues, species, or disease conditions), (2) experimental protocols, and (3) sequencing platforms (Fig. 1A, top). The integration of multiple datasets begins by defining batches as sets of samples that are thought to be similar in terms of these factors (Fig. 1A, boxes in different colors). The factors leading to the most significant batch effects can vary across the integrated datasets. It is common for the major factor causing the largest batch effect to be chosen subjectively, and the batches are defined based on the major factor. For example, batches were defined as sets of samples from individual donors (Reichart et al., 2022; Smillie et al., 2019; Uchimura et al., 2020; Villa et al., 2022) (Fig. 1A, Dataset 1) or as individual datasets that were generated using different protocols (Cheng et al., 2021; Morabito et al., 2021) and/or sequencing platforms (Cheng et al., 2021) (Fig. 1A, Datasets 2-3). Several studies even defined batches as individual samples (Bryois et al., 2022; Yoon et al., 2022) (Fig. 1A, Dataset *k*), assuming that different samples are subjected to distinct technical variations. Moreover, when two factors (e.g., protocol and sequencing platform) cause significant batch effects, two sets of batches can be defined based on these factors, and the batch effects in the two sets can be sequentially corrected (Cheng et al., 2021; Morabito et al., 2021). After defining the batches, multiple datasets are then merged by concatenating the expression counts of cells for the same genes in the expression count matrix for each dataset. For each cell in the merged expression count matrix, the expression counts are divided by the total expression count, multiplied by a scale factor (e.g., 10,000), and log-transformed (Fig. 1B, Normalization). For each batch, a set of highly variable genes (e.g., 2,000 genes) with large variances across the cells in the batch are then selected using various tools (e.g., 'FindVariableFeatures' function in Seurat [v3 and higher hereafter] [Stuart et al., 2019], BASiCS [Vallejos et al., 2015], Brennecke [Brennecke et al., 2013], scLVM [Buettner et al., 2015], scran [Lun et al., 2016], and scVEGs [Chen et al., 2016]). The final set of highly variable genes most frequently selected across the batches are then selected ('SelectIntegrationFeatures' function of Seurat; Fig. 1B, Selection of highly variable genes). Although data integration can be performed using only highly variable genes or all genes, the use of highly variable genes only in subsequent analyses (Fig. 1B, Batch correction, Cell clustering, and Cell annotation) has been shown to be generally more effective for identifying underlying biological differences among cell types and/or for removing unwanted variations in the data (Luecken et al., 2022).

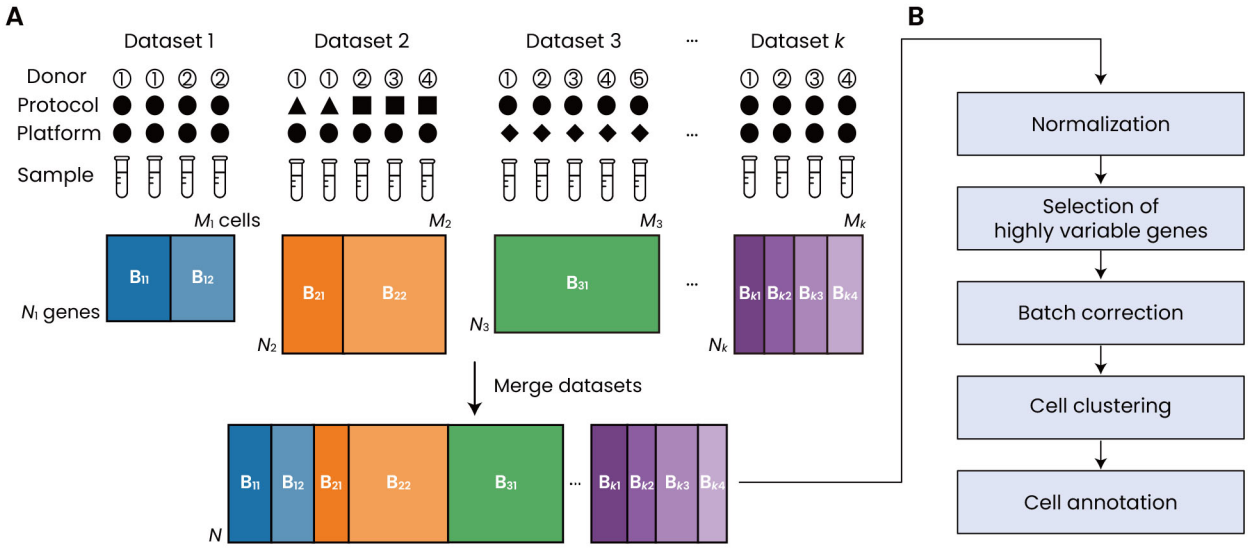

Fig. 1. Definition of batches. (A) Schematic illustration of defining batches by donors (Dataset 1), sample preparation protocols (Dataset 2), sequencing platforms (Dataset 3), and individual samples (donors; Dataset *k*). (B) Analytical flow of data integration. See text for details.

Categorization of integration methods

Batch correction is next performed for the normalized merged dataset using the selected highly variable genes (Fig. 1B, Batch correction). Batch correction methods can be conceptually classified into the following three categories depending on how they model batch effects: (1) linear decomposition methods, (2) similarity-based batch correction methods in reduced dimension space, and (3) generative models using a variational autoencoder. A similar categorization scheme for the first two categories was previously proposed by Xu et al. (2021).

Linear decomposition methods

Modeling batch effects using a generalized linear decomposition model was first introduced to remove batch effects within or across bulk RNA expression datasets when integrating multiple bulk RNA expression datasets. For example, the 'removeBatchEffect' function in limma (Ritchie et al., 2015) and ComBat (Johnson et al., 2007) have employed this linear decomposition model to delineate batch effects in bulk RNA expression datasets.

In the 'removeBatchEffect' function in limma, the merged data matrix (**X**) including expression levels of N genes in K batches of M samples at H conditions is represented as a linear sum of (i) the overall gene expression matrix ($\mathbf{G}_{MGE}$, $N \times M$), (ii) condition-dependent gene expression matrix represented by multiplication of the condition regression coefficient matrix ($\mathbf{R}_C$, $N \times H$) and a condition design matrix ($\mathbf{D}_C$, $H \times M$), (iii) a batch term matrix represented by multiplication of the batch regression coefficient matrix ($\mathbf{R}_B$, $N \times K$) and a batch structure matrix ($\mathbf{D}_B$, $K \times M$), and (iv) an error matrix (**E**, $N \times M$) (Fig. 2A). In the overall gene expression matrix, each column (sample) contains the same vector ($N \times 1$) including mean expression values of N genes across M samples. In the $\mathbf{D}_C$ matrix, the column ($H \times 1$) for sample m under condition h includes one in the h-th element and zeros in the other elements. Similarly, in the $\mathbf{D}_B$ matrix, the column ($K \times 1$) for sample m belonging to batch k includes one in the k-th element

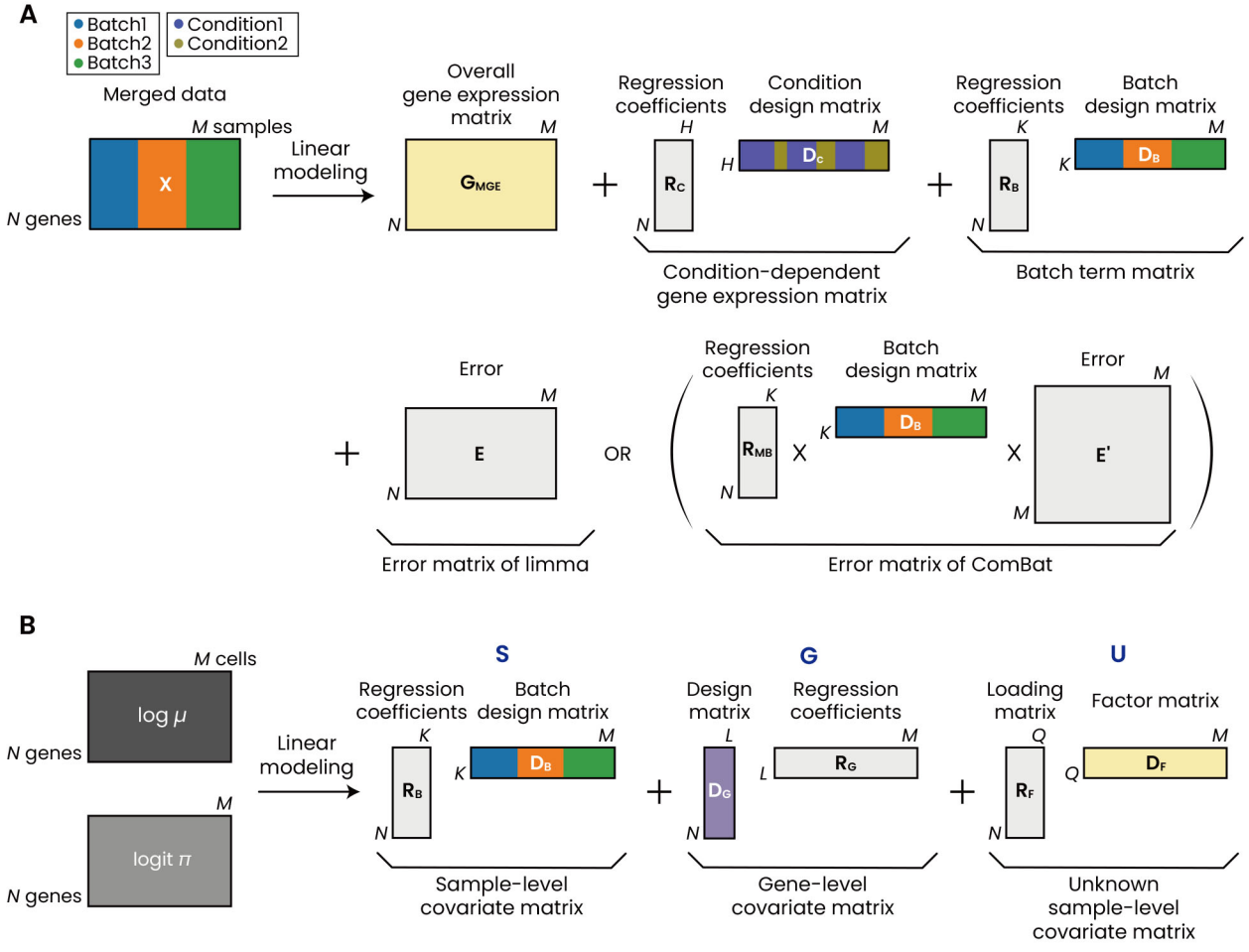

Fig. 2. Schematic view of the methods using linear decomposition models. (A) Linear decomposition scheme used in limma and ComBat. Batches and conditions for cells are indicated by colors. Matrix sizes are denoted in left bottom (number of rows) and right top (number of columns) corners: N genes, M cells, H conditions, and K batches. The error matrix used in ComBat is depicted in parentheses. (B) Decomposition scheme used in ZINB-WaVE involving L gene-level covariates and Q unknown sample-level covariates.

and zeros in the other elements. The condition ($\mathbf{R}_C$) and batch regression coefficients ($\mathbf{R}_B$) are estimated by minimizing the sum of the squared errors in $\mathbf{E}$. The effects of the individual batches are linearly combined using regression coefficients (Fig. 2A, Batch term). Finally, the batch corrected matrix is generated by subtracting $\mathbf{R}_B\mathbf{D}_B$ from $\mathbf{X}$.

In addition to the above batch effects, called additive batch effects, ComBat estimates multiplicative batch effects. To this end, $\mathbf{E}$ is modified into multiplication of a regression coefficient matrix ($\mathbf{R}_{MB}$, $N \times K$), a batch structure matrix ($\mathbf{D}_B$, $K \times M$) and an error matrix ($\mathbf{E}'$, $M \times M$) (Fig. 2A, Error matrix of ComBat). Unlike the linear regression in limma, the regression coefficients in this modified linear decomposition model are estimated using an empirical Bayesian method. With the multiplicative batch effect terms, batch correction was performed by $[\mathbf{X} - (\mathbf{G}_{MGE} + \mathbf{R}_C\mathbf{D}_C + \mathbf{R}_B\mathbf{D}_B)]/\mathbf{R}_{MB}\mathbf{D}_B + (\mathbf{G}_{MGE} + \mathbf{R}_C\mathbf{D}_C)$.

These approaches have been also used to remove batch effects when integrating multiple scRNA-seq datasets in early studies (Giustacchini et al., 2017; Kadoki et al., 2017; Smillie et al., 2019; Young et al., 2018). However, another linear decomposition model ZINB-WaVE (Risso et al., 2018) with an alternative model structure was developed specifically for integrating scRNA-seq datasets that have unique features such as zero inflation (dropouts), overdispersion, and different count distributions from bulk RNA-seq datasets. In this method, the expression count in the merged data matrix ($\mathbf{X}$, $N \times M$) is defined as a random variable (y) following a zero-inflated negative binomial (ZINB) distribution: $\pi\ \delta(y) + (1 - \pi)\ f_{NB}(y; \mu, \theta)$ where π is the probability of dropout (π), and δ and f_{NB} are Dirac delta function and negative binomial probability mass function with the mean μ and an inverse dispersion parameter θ, respectively (Greene, 1994). The model decomposes the log μ matrix ($N \times M$) into sample- ($\mathbf{S}$) and gene-level covariate matrices ($\mathbf{G}$) and unknown sample-level covariate matrix ($\mathbf{U}$) (Fig. 2B). Although $\mathbf{S}$ is defined for K batches, as in the aforementioned $\mathbf{R}_B\mathbf{D}_B$ (Fig. 2B, Sample-level covariate), $\mathbf{G}$ is represented by multiplication of the design matrix ($\mathbf{D}_G$, $N \times L$) and gene regression coefficients ($\mathbf{R}_G$, $L \times M$). $\mathbf{G}$ was introduced to capture variations in L gene sets, each of which had different GC content and gene length distributions, possibly causing differences in the counts and quality of reads (Fig. 2B, Gene-level covariate). $\mathbf{U}$ was represented by multiplying the loading matrix ($\mathbf{R}_F$, $N \times Q$) and the factor matrix ($\mathbf{D}_F$, $Q \times M$) to capture the systematic cellular gene expression profiles in a low dimension of Q latent factors (Fig. 2B, Factor matrix). Notably, U may also include the effects of unknown factor-driven batches that may be missed when defined in $\mathbf{S}$. The logit π (i.e., $\ln[\pi/(1 - \pi)]$) matrix is also decomposed into $\mathbf{S}_\pi$, $\mathbf{G}_\pi$, and $\mathbf{U}_\pi$ (Fig. 2B). However, the same factor matrix ($\mathbf{D}_F$) is shared for both $\mathbf{U}$ and $\mathbf{U}_\pi$. The regression coefficients in $\mathbf{R}_B$, $\mathbf{R}_G$, and $\mathbf{R}_F$ for the log μ and logit π matrices are then collectively estimated using the maximum likelihood method. $\mathbf{D}_F$ was finally used as a cellular gene expression profile summarized in the Q-dimensional space during subsequent analyses (e.g., clustering and visualization).

Similarity-based batch correction methods in reduced dimension space

The above methods assume that there are no cell-level covariates and that all cells vary equivalently between different samples (or batches) in response to the same sources of variation. However, owing to distinct sensitivity in the response, cells can vary differentially between samples and thus between batches. For example, the three groups of cells shown in Fig. 3A exhibited different distribution changes between batches 1 and 2 in terms of their mean and standard deviation when visualized in a uniform manifold approximation and projection (UMAP) space. To address this cell-level covariate issue in batch correction, a number of methods have been developed, including canonical correlation analysis (CCA) (Butler et al., 2018), mutual nearest neighbor (MNN) (Haghverdi et al., 2018), fastMNN (Haghverdi et al., 2018), 'IntegrateData' function in Seurat (Stuart et al., 2019), Scanorama (Hie et al., 2019), BBKNN (Polański et al., 2020), Conos (Barkas et al., 2019), Harmony (Korsunsky et al., 2019), DESC (Li et al., 2020), LIGER (Welch et al., 2019), scMerge (Lin et al., 2019), and SAUCIE (Amodio et al., 2019). These methods start with the projection of cells in the merged dataset ($\mathbf{X}$) onto a reduced space defined by several dimension reduction methods, such as principal component analysis (PCA), CCA, non-negative matrix factorization (NMF), and an autoencoder (e.g., a 2-dimentional space defined by two latent variables [LV1-2] in Fig. 3B). They then identify similar cells sharing expression profiles, which can be identified as pairs of cells between batches at the individual cell level (Fig. 3B, connected cells between batches 1 and 2 in a 2-dimentional LV space) or as cells from different batches in the same cluster at the cluster level. Batch effects are then corrected such that similar cells followed a common distribution in the reduced space (see batch-corrected cluster-level similar cells in Fig. 3C). Each step is described in detail below.

Dimension reduction

scRNA-seq data are vulnerable to technical and biological noise owing to high dropout rates and low expression counts, leading to reduced power to decipher the underlying intrinsic biological differences between cells. Dimension reduction has been commonly employed to focus on the intrinsic information in the data and remove non-systematic noise during subsequent analyses of scRNA-seq data, such as searching for similar cells between batches, cell clustering and visualization (Bzdok et al., 2018). Furthermore, dimension reduction makes computation more convenient and efficient (Argelaguet et al., 2021). Hence, most methods use dimension reduction strategies for subsequent analyses.

Linear dimension reduction approaches have been the most frequently employed, including PCA/singular value decomposition (SVD) (Haghverdi et al., 2018), CCA (Butler et al., 2018), and NMF (Welch et al., 2019). PCA defines LVs, called principal components (PCs), that are orthogonal to each other to capture the largest variances in the data (Fig. 3D, PC1 and PC2). The number of PCs is determined such that the projection of the scRNA profiles ($\mathbf{X}$) of individual cells onto the PCs can sufficiently capture the variation (covariance) in the data by minimizing the reconstruction error $\mathbf{E} = \mathbf{X} - \mathbf{P}\mathbf{T}^T$ where $\mathbf{P}$ ($N \times F$) and $\mathbf{T}$ ($M \times F$) for N genes and M cells are the loading (F PCs) and score matrices (projections onto F PCs), respectively. While PCA defines the PCs using the

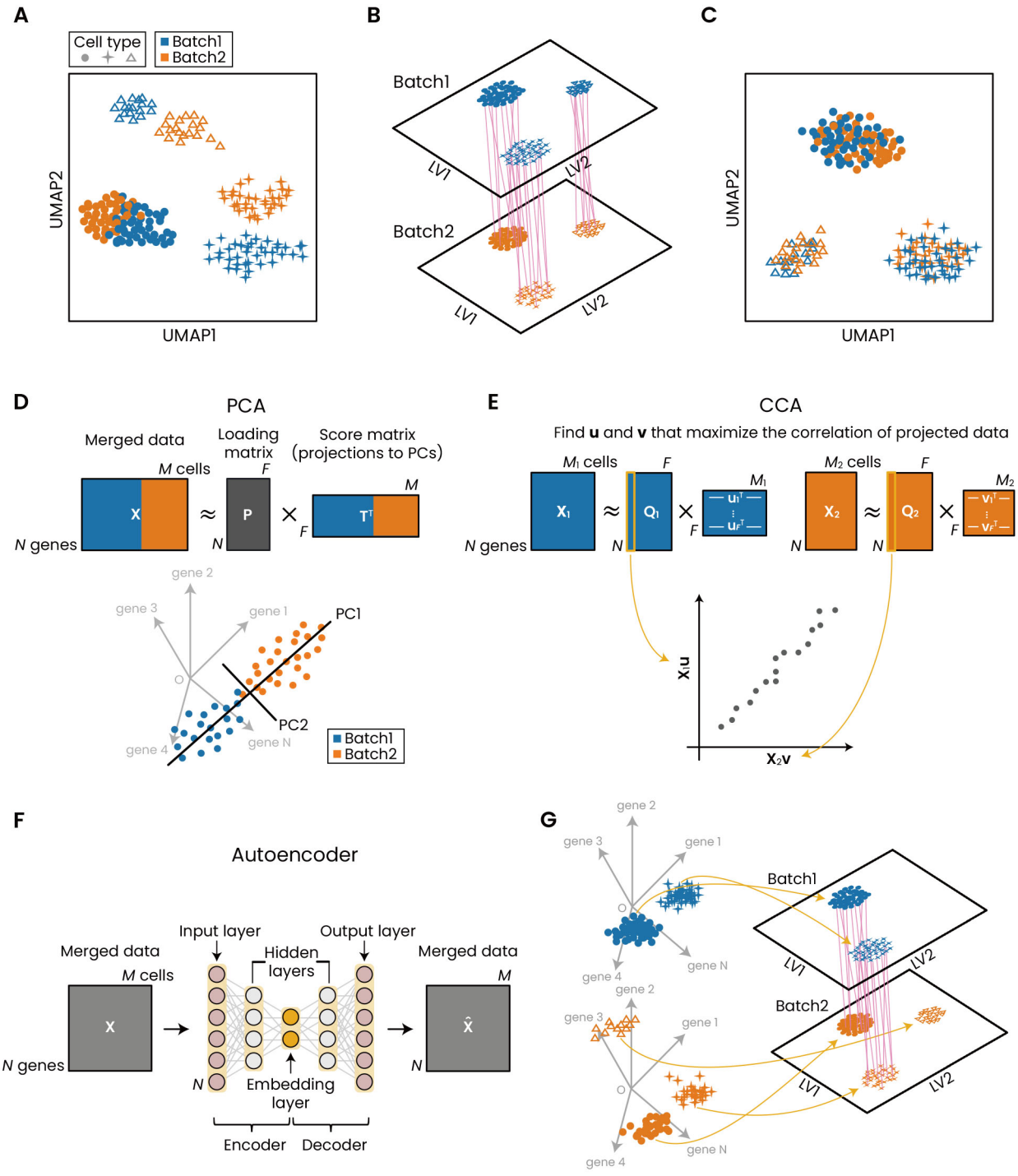

Fig. 3. Dimension reduction methods. (A) Cell-level covariates. Three cell types (clusters) show differential variations between batches 1 and 2. (B) Anchored cell pairs between batches 1 and 2 on two-dimensional LV space. (C) Distributions of cells after batch correction on the UMAP. (D and E) Schematic illustration of PCA (D) and CCA (E). PC1 and PC2 are defined to capture the largest and 2nd largest variance in the distribution of cells while **u** and **v** are defined to maximize the correlation between projections of $\mathbf{X}_1$ (batch 1) and $\mathbf{X}_2$ (batch 2) onto **u** and **v**. Decomposition schemes of $\mathbf{X}_1$ and $\mathbf{X}_2$ are also shown. (F) Architecture of the autoencoder that takes **X** as an input and tries to reconstruct **X** itself. During this reconstruction, the essential features of **X** are extracted in the nodes of the embedding layer. UMAP, uniform manifold approximation and projection; PCA, principal component analysis; CCA, canonical correlation analysis; PC, principal component.

merged dataset, CCA defines the first LVs for each batch (**u** for batch 1 data matrix $\mathbf{X}_1$ and **v** for batch 2 data matrix $\mathbf{X}_2$) such that the correlation between the projections of $\mathbf{X}_1$ onto **u** and $\mathbf{X}_2$ onto **v** is maximized (Fig. 3E). The remaining LVs are similarly determined to maximize the remaining correlation between $\mathbf{X}_1$ and $\mathbf{X}_2$. By default, fastMNN, BBKNN, Conos, and Harmony use PCA for dimension reduction while 'FindIntegrationAnchors' function in Seurat uses CCA. Because both PCA and CCA capture information to maximize the variance in the data and the correlation between the batches, respectively, nonsystematic noises with small variance and correlation are disregarded in principle when the data are projected onto LVs, thereby enabling us to focus on the underlying intrinsic biological signals.

NMF defines K nonnegative factors (**W**, $N \times K$) and sample projections (**H**, $K \times M$) for a non-negative merged data matrix (**X**, $N \times M$) such that they minimize the reconstruction error $\mathbf{E} = \mathbf{X} - \mathbf{WH}$. For non-negative factorization, **X** can be unfolded to $[\mathbf{X}_{pos}\ |\mathbf{X}_{neg}|]$ such that $\mathbf{X}_{pos}$ and $|\mathbf{X}_{neg}|$ include positive and absolute negative elements in **X**, as previously described (Kim et al., 2011). Among the NMF variants, integrative NMF (iNMF) (Yang and Michailidis, 2016) and UINMF (Kriebel and Welch, 2022) have been used for dimensionality reduction in scRNA-seq datasets. For the data matrices ($\mathbf{X}_i$) for L batches, iNMF has an additional term to reflect batch-specific factor loadings ($\mathbf{V}_i$), and these **W**, $\mathbf{V}_i$, and $\mathbf{H}_i$ are determined to minimize the reconstruction errors for L batches: $\Sigma\|\mathbf{E}_i\|^2 = \Sigma\|\mathbf{X}_i - (\mathbf{W} + \mathbf{V}_i)\mathbf{H}_i\|^2$. The data integration tool LIGER uses iNMF. Its variant UINMF has another additional factor loadings ($\mathbf{U}_i$) in the error $\Sigma\|\mathbf{X}_i - ([\mathbf{W};\ \mathbf{0}] + [\mathbf{V}_i;\ \mathbf{U}_i])\mathbf{H}_i\|^2$ to estimate $\mathbf{U}_i$ for the genes uniquely detected in I batches by minimizing the reconstruction error. Compared with PCA and CCA, which focus on shared signals between batches, iNMF and UINMF can effectively handle batch-specific sources of variation by identifying both batch-specific ($\mathbf{V}_i$ or $[\mathbf{V}_i;\ \mathbf{U}_i]$) and shared factors (**W**).

Neural-network-based dimensionality reduction methods have been employed to effectively handle nonlinear correlations among variables in datasets. Among these methods, the autoencoder (Amodio et al., 2019) is one of the most frequently used methods. For the merged data matrix (**X**), the autoencoder is designed to reconstruct **X** itself and thus has (1) input and output layers comprising N nodes for N genes, (2) an embedding layer in the center, and (3) hidden layers with a symmetric structure between the input and output layers (Fig. 3F). In the embedding layer, the number of nodes (i.e., nonlinear LVs) is smaller than N nodes in the input layer, thereby enabling dimension reduction. The embedded values in this layer represent nonlinear projection of scRNA profiles onto the reduced dimension defined by the nonlinear LVs, also called 'nonlinear PCA' in that the autoencoder defines the nonlinear LVs to minimize the reconstruction error. DESC and SAUCIE use autoencoders for dimension reduction.

Projection values in the reduced dimension from the above linear or nonlinear methods are used to identify similar cells at the individual or cluster level (Fig. 3G), followed by batch correction for similar cells to follow a common distribution in the reduced space. However, SAUCIE does not identify similar cells, but directly performs batch correction to minimize discrepancies between batches (Amodio et al., 2019). To this end, SAUCIE randomly sets one batch as the reference batch, and corrects the mean and standard deviation of the embedded values for each non-reference batch with those of the reference batch while minimizing the reconstruction error. The weights in the autoencoder are thus determined to balance the reconstruction error and batch correction.

Identification of similar cells between batches

Several methods have been developed to identify similar cells among different batches, which can be categorized into two groups according to whether similar cells between batches are identified as cell pairs at the individual cell level (Fig. 4A, left) or as sets of cells in the same cell cluster (Fig. 4A, right).

Cell-level similarity search: The methods to identify similar cells at the individual cell level mostly employ nearest neighbor-based methods, including MNN, 'FindIntegrationAnchors' function in Seurat, and the algorithms in Scanorama, BBKNN, and Conos. Suppose that there are two batches (batches 1 and 2) to be aligned. The MNN computes the cosine distance (i.e., Euclidean distance after normalizing the gene expression profile vector of each cell to have the unit length) for a pair of cells i (Fig. 4B, dark blue dot) in batch 1 and cell j (Fig. 4B, red dot) in batch 2, and then finds k MNNs of cell i in batch 2 (Fig. 4B, orange dots within dark blue dotted circle) and cell j in batch 1 (Fig. 4B, blue dots within red dotted circle). When the MNNs include cells i and j, this pair is anchored between batches 1 and 2 (Fig. 4B, pink line). While MNN performs the search in the original dimension, fastMNN and 'FindIntegrationAnchors' function in Seurat do it on the reduced PCA and CCA spaces, respectively. MNN then selects a reference batch (e.g., batch 1 with the largest cell count) and computes expression differences between the reference and query (e.g., batch 2) batches for all anchored cell pairs (e.g., $N \times 1$ $\mathbf{d}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ for anchored pairs of cell i in batch 1 and cell j in batch 2). For cell I in batch 2, the gene expression vector ($N \times 1$ $\mathbf{x}_I$) is corrected by subtracting a batch vector ($N \times 1$ $\mathbf{u}_I$), which is a weighted sum of the differences ($\mathbf{d}_{ij}$) for all anchored cell pairs between batches 1 and 2 (Fig. 4B, right). Gaussian kernel weights for all anchored cells in batch 2 from cell I are used such that the anchored cells closer to cell I have higher weights and are collectively used to ensure robustness in the batch correction (Fig. 4B, Batch vector). The same procedure can be performed in a reduced space using a PC vector ($F \times 1$ $\mathbf{x}_I$ for F PCs in fastMNN) instead of $\mathbf{x}_I$. Scanorama uses the same MNN strategy while 'FindIntegrationAnchors' uses a reliable set of anchored cell pairs and a different weighting scheme (Stuart et al., 2019).

BBKNN and Conos explicitly perform no batch correction, but provide weighted graphs. BBKNN connects cell i in batch 1 to its k nearest neighbors within batch 1 and in batch 2 as well and then determines the weight for each connected pair such that a smaller distance between the connected cells (e.g., cell i and a neighbor) in the reduced space has a higher weight (Polański et al., 2020). Conos connects cell i in batch 1 to its k nearest neighbors within batch 1 and the anchored pair (cell j in Fig. 4B) in batch 2 identified by the above MNN strategy, and then determines the weight as Pearson's correlation of projection values onto PCs between two connected cells, followed by multiplication of 0.1 to the weights of

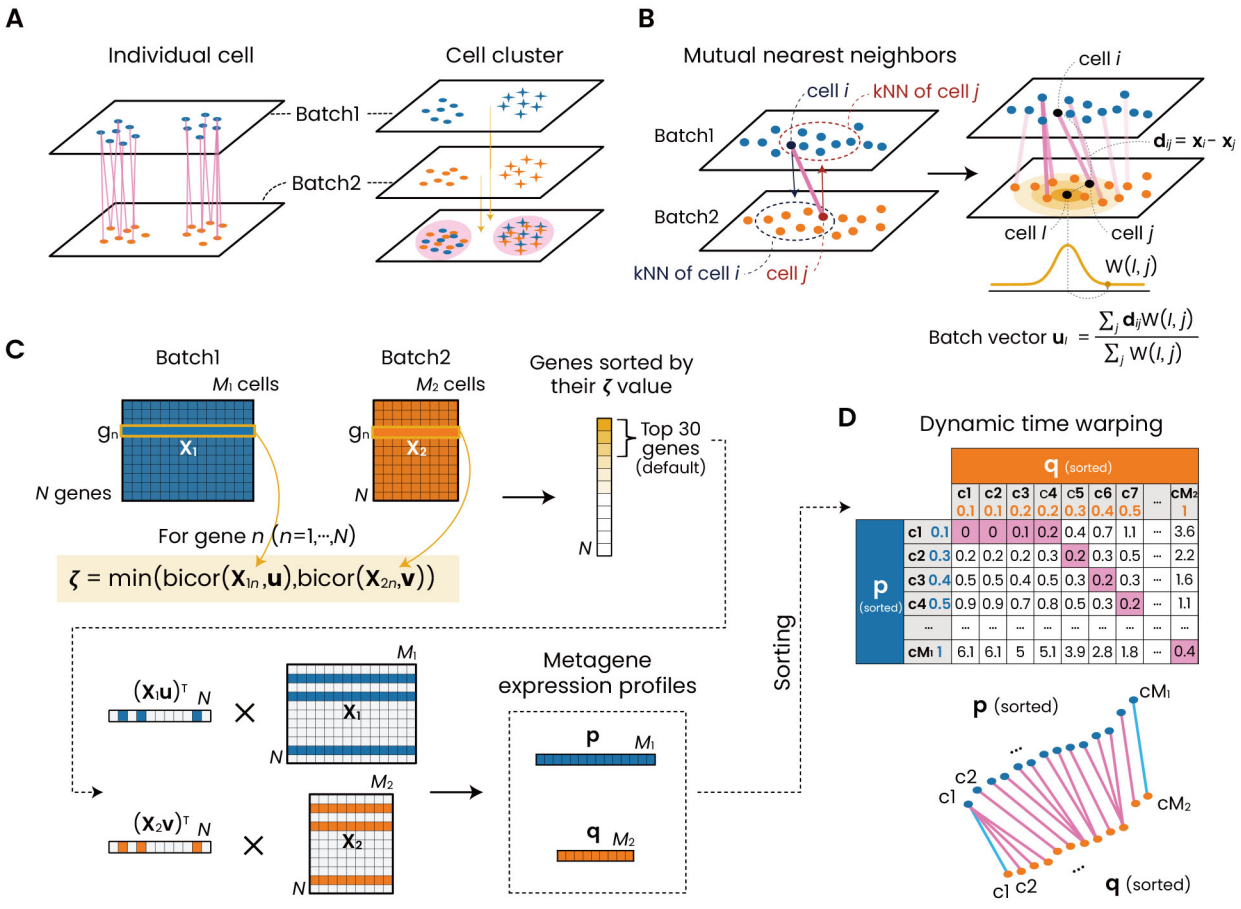

Fig. 4. Cell-level similarity search. (A) Similar cell pairs identified by cell-level similarity search (left) and similar clusters identified by clustering (right). (B) Schematic illustration of MNN strategy for identifying anchored cell pairs (left) and batch correction strategy (right). (C and D) Dynamic time warping involving selection of metagenes (C, top), determination of metagene expression profiles (C, bottom), generation of cumulative distance matrix (D, top), and dynamic time warping strategy (D, bottom). See text for details (B-D).

k nearest neighbors within batch 1 to reduce the effect of the within batch neighbors (Barkas et al., 2019). A weighted cell graph from BBKNN or Conos is then used as an input for downstream analyses such as clustering (e.g., Louvain clustering; Blondel et al., 2008) and pseudotime inference.

'AlignSubspace' function in Seurat v2 employed alternatively dynamic time warping for cell-cell similarity search. It first selects the batch with the largest number of cells as the reference batch (e.g., batch 1). For a given query batch (e.g., batch 2), it applies CCA to identify the first LVs for batches 1 (**u**) and 2 (**v**) as illustrated in Fig. 3E; selects top 30 genes with the highest contribution [i.e., highest min(bicor($\mathbf{X}_{1n}$,**u**), bicor($\mathbf{X}_{2n}$,**v**)) for gene *n*] to **u** and **v** (Fig. 4C, top); and then computes the metagene expression profiles for M_1 cells (**p**) in batch 1 and M_2 cells (**q**) in batch 2 by multiplying the data matrix and the loading values [$\mathbf{p} = (\mathbf{X}_1\mathbf{u})^T\mathbf{X}_1$ and $\mathbf{q} = (\mathbf{X}_2\mathbf{v})^T\mathbf{X}_2$] of the 30 genes on the 1st CCA space (Fig. 4C, bottom). After sorting **p** and **q**, a cell-cell distance matrix ($M_1 \times M_2$) is generated between M_1 and M_2 cells (Fig. 4D, top). For the dynamic time warping of cells in batch 2 to those in batch 1, the pairs of cells with the first and last **p** and **q** values are first linked (Fig. 4D, light blue lines). A warping path indicates a set of connections between the remaining cells in batches 1 and 2 based on the sorted orders. Among all possible warping paths, the path that minimizes the cumulative sum of the distances of the linked cells (e.g., a path comprising the linked cells highlighted by magenta boxes in Fig. 4D) is selected (see 'dtw' package [Giorgino, 2009] in R for details). Finally, **v** is modified to have the same value as **u** for the linked cells for batch correction. This procedure is repeated for each pair of LVs (e.g., 2nd **u** and **v**) in *F*-dimensional CCA space.

Cluster-level similarity search: Unlike the above methods, cells with similar RNA expression profiles in the same cluster can be considered a set of similar cells, and batch effects can be corrected at the cluster level, assuming that cells in the same cluster are subjected to similar variations. This concept has been employed in Harmony, DESC, LIGER, and scMerge. Harmony first applies PCA to the merged data matrix (**X**) and performs a modified soft *k*-means clustering (k = min(100, M/30), where M is the total cell count in **X**) using normalized

projections onto PCs with the unit length. The centroid vector ($\mathbf{c}_k$) for cluster k is first determined by regular hard k-means clustering (Fig. 5A, left). Given the clustering results, the probability (R_{ki}) of cell i to belong to cluster k ($\Sigma_k R_{ki} = 1$) and $\mathbf{c}_k$ are iteratively determined to minimize the objective function (Fig. 5A, middle): $\Sigma_{i,k}[R_{ki}\|\mathbf{z}_i-\mathbf{c}_k\|^2 + \lambda_1 R_{ki}\log(R_{ki}) + \lambda_2 R_{ki}\log(O_{ki}/E_{ki})\phi_i]$ where $\mathbf{z}_i$ is the PC projection vector for cell i (Fig. 5A, maximum diversity clustering), and ϕ_i is a batch index vector where cell i belonging to batch j includes one in the j-th element and zeros in the other elements. The second entropy term $[R_{ki}\log(R_{ki})]$ is added for soft clustering, which probabilistically assigns cell i to k clusters in a parsimonious manner. The third term is added to minimize the Kullback Leibler (KL) divergence between the observed (O_{ki}) and expected (E_{ki}) distributions of cell counts from batches over clusters, thereby maximizing the variance of cell counts from batches in each cluster (i.e., maximum diversity). Assuming independence between the cluster assignment and batch, each cluster is expected to include cells from different batches with uniform probability. In each iteration, projection of cell i ($\mathbf{z}_i$) is corrected by subtracting $\Sigma_k R_{ki} a_k$ where a_k is a portion of the projection that can be explained by batch-dependent R_{ki} in cluster k, called linear mixture model correction (Fig. 5A, middle). These steps of R_{ki} estimation and $\mathbf{z}_i$ update are repeated until $\mathbf{z}_i$ converges (Fig. 5A, right).

DESC also performs cluster-level batch correction; however, its algorithm is different from that of Harmony. The stacked autoencoder, a variation of the autoencoder with pre-training and model fine tuning steps, is first applied to the merged data matrix ($\mathbf{X}$) to obtain nonlinear LV projections ($\mathbf{z}_i$ for cell i; Fig. 5B, Encoder and Latent space). A graph was then constructed based on the similarity between cell pairs using their nonlinear LV projections, and Louvain clustering (Blondel et al., 2008) is performed to determine the number (k) of clusters and initialize the cluster centroids ($\mathbf{c}_k$ for cluster k; Fig. 5B, Louvain clustering). Using $\mathbf{z}_i$ and $\mathbf{c}_k$, DESC estimates q_{ki} as $(1 + \|\mathbf{z}_i - \mathbf{c}_k\|^2/\alpha)^{-1}/\Sigma_k(1 + \|\mathbf{z}_i - \mathbf{c}_k\|^2/\alpha)^{-1}$ where α is the degree of freedom of the Student's t distribution ($\alpha = 1$ by default). Of note, q_{ki} becomes high when cell i is close to the centroid of cluster k. Using $\mathbf{z}_i$ and $\mathbf{c}_k$, DESC computes KL divergence as $\Sigma_i\Sigma_k[p_{ki}\log(p_{ki}/q_{ki})]$ where $p_{ki} = [q_{ki}^2/\Sigma_i q_{ki}]/[\Sigma_k(q_{ki}^2/\Sigma_i q_{ki})]$, called an auxiliary distribution, which represents the probability that cell i belongs to cluster k, similar to R_{ki} in Harmony. The weights in the encoder, as well as $\mathbf{z}_i$ and $\mathbf{c}_k$, are updated to minimize the KL divergence, which improves cluster purity (Fig. 5B, Model fine tuning). This iteration continues until the KL divergence converges, and the final updated $\mathbf{z}_i$ is considered a batch-corrected projection.

LIGER also performs cluster-level batch correction. For clustering of M cells in the merged dataset ($\mathbf{X}$), it assigns cell i to an iNMF factor (cluster) with the maximum factor loading for the cell (Fig. 5C, node boundary colors). To handle the uncertainty in the maximum factor loadings, LIGER identifies k nearest neighbors of cell i (Fig. 5C, cells within the red dotted circle) and then computes a factor neighborhood vector ($1 \times K$) for cell i in which the k-th element represents the count of the k nearest neighbors belonging to cluster k (Fig. 5C, Factor neighborhood vector). A shared factor neighborhood graph is then built based on the Manhattan distance between pairs of cells (Fig. 5C, SFN graph construction), which is subjected to Louvain clustering. In Fig. 5C, after Louvain clustering, cells i and j, originally assigned to factor 2, were clustered together with the cells assigned to factor 1, due to their factor neighborhood vectors similar to those of the cells assigned to factor 1. When there are L batches, for each factor, the loadings ($M \times 1$) of the cells assigned to cluster k are split into L sets of the loadings for cells, according to their batch information. The L sets of the loadings are then subjected to quantile normalization (Bolstad et al., 2003) to match the distribution of factor loadings in each batch with that in the reference batch (Fig. 5C, right). The corrected factor loadings (low dimensional representations) are used for the subsequent analyses.

Finally, scMerge performs first k-means clustering for cells in each batch and then identifies the pairs of anchored clusters between batches based on mutual nearest clusters (MNCs; Fig. 5D, Identification of MNC). A cluster graph having edges as the anchored cluster pairs is built, and subgraphs are then identified using 'igraph' package (Csardi and Nepusz, 2006) in R (e.g., Sub1-3 in Fig. 5D, Identification of subgraphs). For every cluster in a subgraph, the core cells are defined as half of the cells with the smallest Euclidean distances to the cluster centroid, and the set of all core cells are then defined as a pseudoreplicate for the subgraph (Fig. 5D, Identification of pseudoreplicates). Variations in the expression profiles of the core cells within the same pseudoreplicate are considered unwanted variations (e.g., batch effects). To sort these variations in the merged data ($\mathbf{X}$), scMerge defines a replicate matrix ($\mathbf{B}$) with pseudoreplicate and non-core cell columns (e.g., three pseudoreplicate columns for Sub1-3 and the remaining non-core cell columns in Fig. 5D, Replicate matrix). One is then added to $\mathbf{B}(i,l)$ for core cell i in pseudoreplicate l while one is to $\mathbf{B}(j,m)$ for non-core cell j (e.g., $m = 3 + j$ in Fig. 5D, Replicate matrix). Multiplication of the residual operator $[\mathbf{R} = \mathbf{I} - \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T]$ to the auto-scaled $\mathbf{X}_s$ over genes sorts out the unwanted variations from $\mathbf{X}_s$, and $\mathbf{RX}_s$ is then decomposed into $\mathbf{RW}\boldsymbol{\alpha} + \mathbf{E}$ (Molania et al., 2019). The regression coefficients ($\boldsymbol{\alpha}$) are first estimated to minimize $\mathbf{E}$ by SVD using all the genes, and $\mathbf{W}$ is re-estimated only using stably expressed genes (SEGs) showing minimal changes across cells by $\mathbf{X}_{SEG}\boldsymbol{\alpha}_{SEG}^T(\boldsymbol{\alpha}_{SEG}^T\boldsymbol{\alpha}_{SEG})^{-1}$ where $\mathbf{X}_{SEG}$ and $\boldsymbol{\alpha}_{SEG}$ include only the rows of $\mathbf{X}_s$ and the coefficients (loadings) for SEGs, respectively. Finally, batch effects are corrected by $\mathbf{X}_s - \mathbf{W}\boldsymbol{\alpha}$, which corrects unwanted variations explained by the correlations ($\boldsymbol{\alpha}_{SEG}$) of SEGs.

In practice, after batch correction using the aforementioned methods, the outputs, such as batch corrected merged data matrix ($\mathbf{X}$ from linear decomposition methods) or projections ($\mathbf{z}$) on the reduced space (e.g., Harmony and DESC), or updated factor loadings (e.g., LIGER) from these methods are subjected to another clustering (e.g., Louvain clustering using the kNN graph in Seurat), and identification of differentially expressed genes (e.g., 'FindMarkers' function in Seurat) and annotation of cell types (e.g., SingleR; Aran et al., 2019) are then performed for the resulting clusters.

Generative models with variational autoencoder

Linear decomposition and similarity-based methods using linear dimension reduction cannot effectively capture the

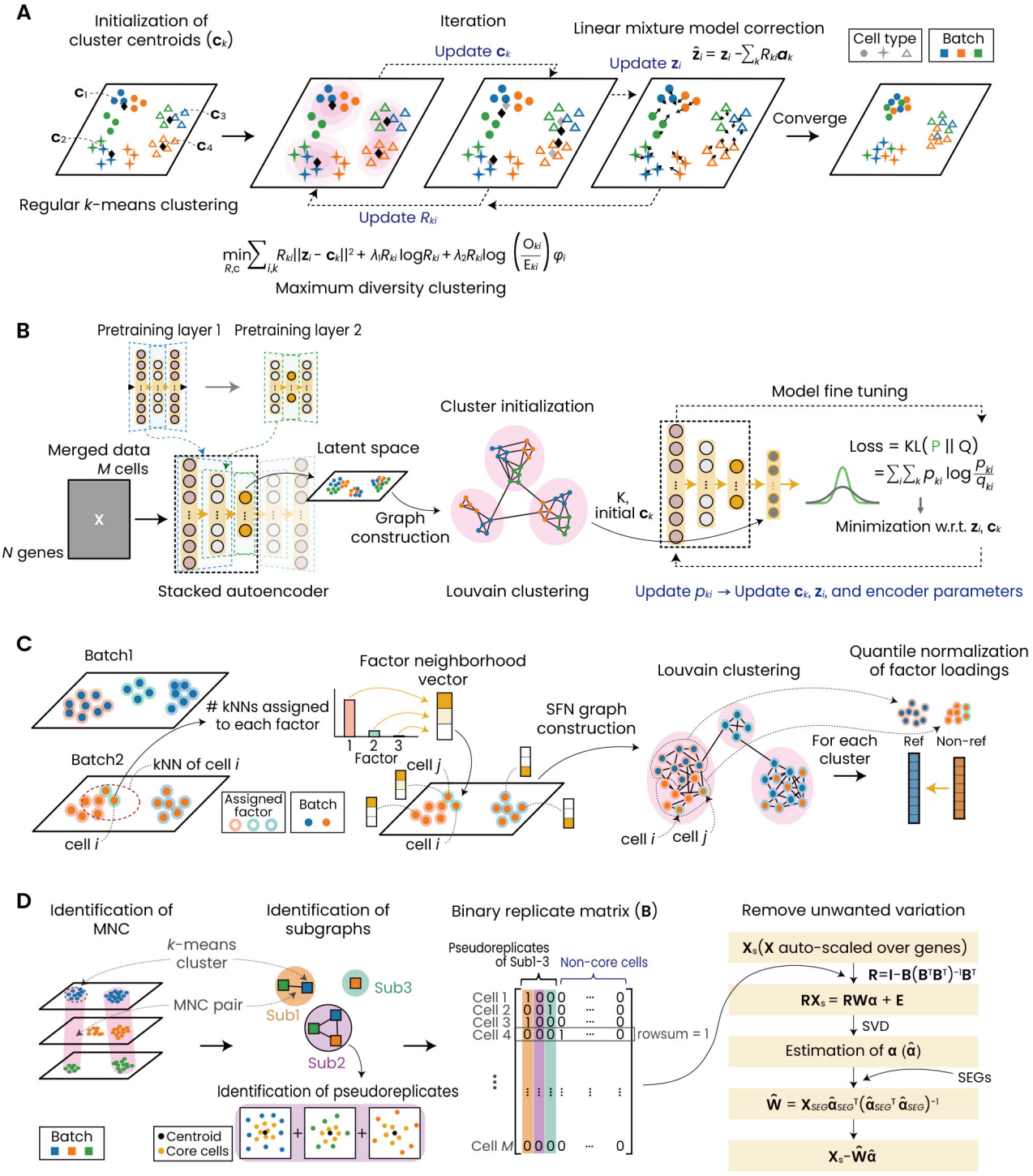

Fig. 5. Cluster-level similarity search. Schematic illustration of the analytical steps in Harmony (A), DESC (B), LIGER (C), and scMerge (D). See text for details.

nonlinear characteristics of batch effects and systematic biological signals. To address these issues, several methods using a generative model with variational autoencoder (e.g., scVI [Lopez et al., 2018], scGen [Lotfollahi et al., 2019], and trVAE [Lotfollahi et al., 2020]) have been developed.

Similarity-based methods often suffer from heavy computational loads during similarity searches and batch corrections, when hundreds of thousands of cells are integrated. scVI (Lo-

pez et al., 2018) was developed to effectively model the nonlinear characteristics of data and resolve the computational load issue. The scVI assumes that the expression count (x_m) for each gene in cell m follows the ZINB distribution $p(x_m|z_m, s_m, l_m)$ where z_m is the nonlinear LVs, s_m is the batch information for cell m; and l_m is a cell-specific size factor that accounts for variations during library construction and sequencing, which is not explicitly considered in the aforementioned methods. To estimate the ZINB probability, scVI employs a variational autoencoder model composed of 'variational posterior' and 'generative model' parts (Fig. 6A): the first part performs the inference of a variational posterior distribution $q(z_m, l_m|x_m, s_m)$ for two unknown variables z_m and l_m given x_m and s_m using neural networks (NNs; Fig. 6A, NN1-4), and the second part estimates $p(x_m|z_m, s_m, l_m)$ using other NNs (Fig. 6A, NN5-6). NN1-4 are trained to estimate the parameters (mean and standard deviation) of the Gaussian distributions that z_m and l_m are assumed to follow based on variational inference. After z_m and l_m (mean values) are sampled from their estimated distributions, z_m, considered as batch corrected projections, are used to train NN5-6 to estimate the expected dropout (π) and frequency in the NB distribution, respectively. These expected values are finally used together with the sampled l_m to estimate the expected expression counts (x_m) that follow $p(x_m|z_m, s_m, l_m)$. The weights of the NNs are updated such that the sampled z_m and l_m explain the observed x_m given s_m based on the ZINB distribution. The sampled z_m can be used for clustering analysis, and the expected counts can be used to identify differentially expressed genes for cell clusters. scANVI (Xu et al., 2021), an extension of scVI, uses additional cell-type information such that the distribution of cells in the latent space (z_m) reflects the cell types, thereby enabling batch corrections with the cell-type information considered.

Similar to scANVI, scGen (Lotfollahi et al., 2019) uses the cell-type information obtained from the cell-type annotation of the clusters after cell clustering (e.g., three cell types in Fig. 6B, left). A variational autoencoder is used to estimate the distribution parameters (means and standard deviations) of the nonlinear LVs (z_m) based on variational inference (Fig. 6B, middle top). The sampled z_m are used to determine per-

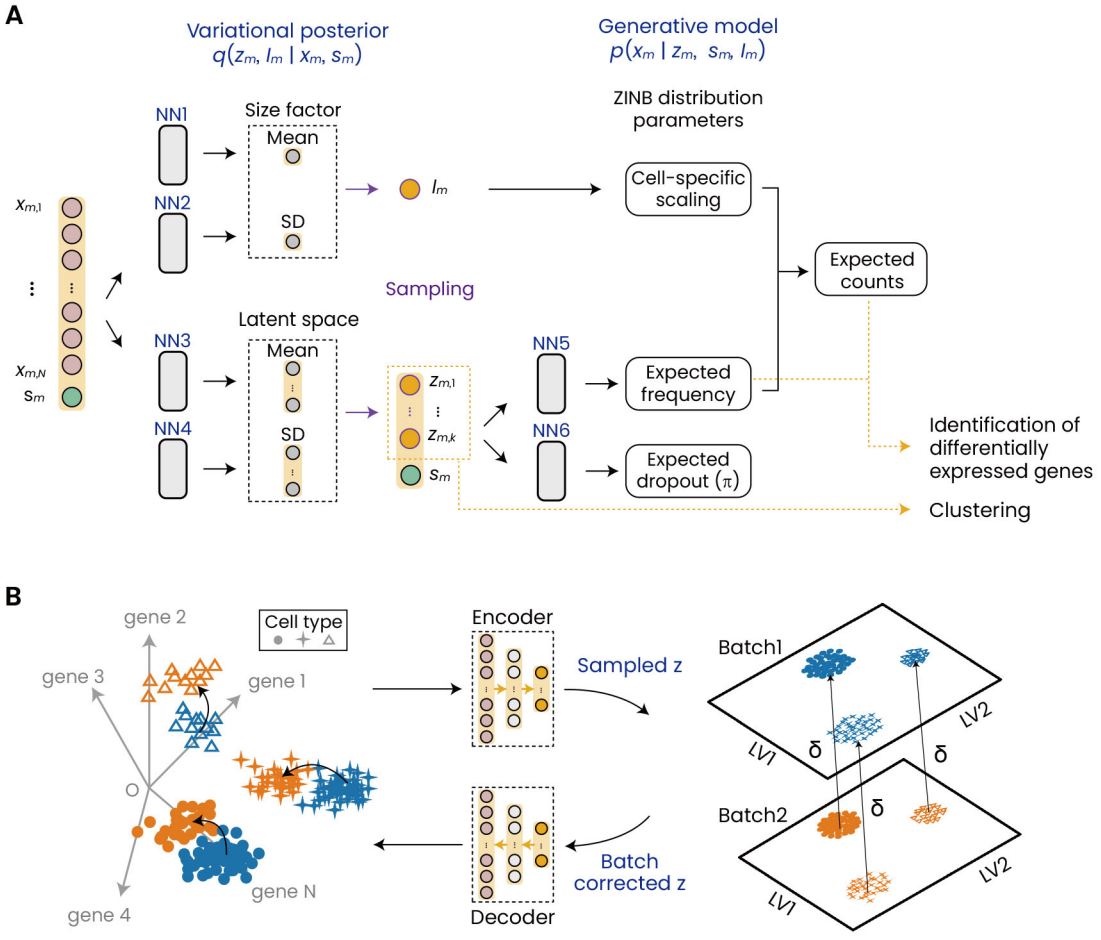

Fig. 6. Generative models with variational autoencoder. (A) Architecture of scVI and schematic illustration of analytical steps in scVI. The outputs from NN5-6 are used to estimate the ZINB distribution $p(x_m|z_m,s_m,l_m)$. (B). Schematic illustration of analytical steps in scGen. See text for details (A and B).

turbation parameters ($K \times 1$ δ) for K LVs as the difference between the mean vectors of z_m for cells in batches (e.g., δ on 2-dimensional LV space between batches 1-2 in Fig. 6B, right). Batch effects are then corrected by applying δ to z_m for cells in batch 2, and the batch corrected z_m are then rescaled back to produce the batch corrected x_m using the decoder NN (Fig. 6B, right). Finally, trVAE (Lotfollahi et al., 2020) employs a variational encoder similar to that in scGen to estimate the distribution parameters of z_m; however, the input layer has additional L nodes that take the batch information for L batches. Optionally, nonlinear LVs can first be decoded in two sequential stages: from the LVs to an intermediate y_m (g_1 decoder to handle the discrepancy between batches) and then from y_m to x_m (g_2 decoder).

Pros and cons

All the aforementioned methods have differences in model structures and parameters, dimension reduction, similarity search, and batch correction. These aspects of the individual methods are summarized in Supplementary Table S1. The unique characteristics of these methods provide advantages and disadvantages in terms of their performance (output type, speed, peak memory use, GPU support, etc.), which are summarized in Supplementary Table S1. 'High' performance in the table indicates that the corresponding methods were found to show good performance in terms of speed, memory, and batch correction performance evaluated based on diversity of batches in each cluster (i.e., cell type).

The characteristics of the individual methods are associated with the algorithms or approaches employed. Linear decomposition methods are the simplest, thereby providing an advantage in terms of analysis speed (Tran et al., 2020) (Supplementary Table S1, Speed). ComBat could model diverse batch effects using both additive and multiplicative batch terms compared to other linear decomposition methods, and also incorporated the empirical Bayes shrinkage of parameters to pool the information across genes, which provides robustness in correction of batches with small sample sizes. ZINB-WaVE includes unique gene-level covariates in the model. However, how its addition to the model improves performance has not been systematically tested, and the benefit of the gene-level covariate term is unclear, considering that the discrepancy among gene-level covariates may be handled by normalization strategies in other methods. Linear decomposition methods assume that the cell types between batches are similar (Haghverdi et al., 2018; Lin et al., 2019) and cannot effectively handle heterogeneity among cell-level covariates across samples or batches.

Although similarity-based batch correction methods effectively handle cell-level covariate issues, they commonly involve additional steps to search for similar cells and correct batch effects to match the distributions of similar cells. Because of these additional steps, some of them (e.g., MNN-based methods) often suffer from heavy computational loads (Supplementary Table S1, Speed and Peak memory use), which render limited applicability or scalability for scRNA-seq datasets, including hundreds of thousands of cells or more (Li et al., 2020). Both linear decomposition and similarity-based methods effectively handle unwanted non-systematic variations. For systematic unwanted variations, however, MNN-based methods can handle more effectively the systematic unwanted variations than the linear decomposition methods, as long as they identify pairs of similar cells with reasonable accuracy.

Similarity-based methods perform similarity searches and batch corrections mostly on reduced dimensions. Among the dimension reduction tools, CCA captures the shared sources of variations between batches and thus performs well when cell populations are largely shared between batches, whereas it is less likely to capture variations for cell subpopulations uniquely present in small numbers of batches. By contrast, PCA captures distinctive sources of variation with a sufficient number of LVs, which may make similarity search among these distinctive cell subpopulations possible. iNMF specifically models cells that are uniquely present in a small number of batches to effectively address this issue. Moreover, batch corrections based on batch vectors require large amounts of memory (Hie et al., 2019), thereby causing memory issues, particularly for datasets that include a large number of cells. Scanorama endeavored to resolve this memory issue by reducing peak memory usage, thereby improving the scalability of analyses with limited computing resources (Li et al., 2020; Luecken et al., 2022; Tran et al., 2020) (Supplementary Table S1, Speed, Peak memory use, and Performance). BBKNN and Conos provide no batch-corrected **X** or **z**, but several downstream analyses, such as functional gene program identification (Kotliar et al., 2019) or trajectory inference (Trapnell et al., 2014), require corrected **X** or **z**, thereby limiting the applicability of BBKNN and Conos (Luecken et al., 2022).

Among the cluster-level similarity search methods, Harmony shows high scalability in terms of both runtime and memory usage (Korsunsky et al., 2019; McKellar et al., 2021; Tran et al., 2020) (Supplementary Table S1, Speed). However, the KL divergence term in its objective function can make Harmony biased toward major cell types (clusters), including large numbers of cells. There may be a chance for Harmony to overcorrect batch effects for small cell types when they are integrated with major cell types. DESC does not require batch information, but corrects batch effects and simultaneously performs soft clustering. The exact number of batch-effect sources (or unwanted variations) is typically unknown. Although DESC may effectively handle these unknown sources of batch effects, systematic analyses are needed to understand the benefits from no use of batch information. LIGER was shown to have a tendency of focusing on batch effect correction rather than biological conservation, favoring its application to integration of cross-species datasets (Luecken et al., 2022) (Supplementary Table S1, Bio-conservation vs Batch correction).

scVI and scANVI are 'all-inclusive' tools that perform a range of analyses including normalization, dimension reduction, batch effect correction, imputation, clustering, and differential expression. They can effectively capture the nonlinear characteristics in data based on probabilistic models that statistically bound variations in random variables (x_m or z_m) using neural networks. Probabilistic models enable the propagation of the statistically bounded x_m or z_m to clustering, differential analysis, or cell-type annotation, providing effective handling

of uncertainties in downstream analyses. These methods have been shown to scale up to very large datasets (Stuart et al., 2019). scVI and scANVI use the ZINB distribution for probabilistic modeling by default, but it is possible to use the NB distribution instead of the ZINB distribution. It is debatable which distribution better fits the read counts derived from either the droplet-based method or the full-length plate-based method. However, in the benchmarking analysis using both distributions, similar results in four datasets tested were shown (Xu et al., 2021).

As no single method performs well in all integration settings, several representative methods in the aforementioned categories (e.g., ComBat, Seurat's anchoring method, harmony, and scVI) should be compared to select an appropriate method for data integration. Clusters of cells identified by each method on UMAP can be compared across the methods. Among the clusters, several are consistently identified with similar shapes and memberships in all methods. Focusing on these clusters provides the most reliable conclusions. However, these methods often produce inconsistent clusters, particularly for small clusters. For example, a small cluster can be identified as a distinct cluster in the ComBat and anchoring methods but merged into another large cluster in Harmony and scVI. Moreover, the relative position of the small cluster with respect to the other clusters on the UMAP cells can differ across methods. Furthermore, cells within a cluster can be evenly distributed or exhibit a skewed distribution. The patterns of these cluster characteristics across the methods can suggest whether batch effects are corrected too weakly, too much, or appropriately for particular clusters of interest, from linear and similarity-based (cell- or cluster-level) methods to variational autoencoders. Nevertheless, whether a small cluster is a biologically meaningful cell subpopulation or an artifact from cell isolation, and batch correction should be determined by detailed functional experiments for small clusters.

CONCLUSION

Data integration methods provide unique opportunities to systematically compare and federate cell types present in multiple sets of samples under similar or distinct disease conditions, thereby enabling a more comprehensive interpretation of the functional roles of different cell types under diverse disease conditions. Moreover, the integrative analysis of multiple datasets generated from diverse disease conditions can provide insights into the interplay of particular cell-type pairs by providing shared count variations of the cell-type pairs under disease conditions. Although data integration methods have improved from linear decomposition to nonlinear probabilistic methods, many problems remain to be resolved. A definition of batches is required for most tools. Although batches are defined subjectively based on the major sources of unwanted variation, there may still be unrecognized sources that cause batch effects. Moreover, most methods assume that batch effects are smaller than biological differences. However, there could be systematic unwanted variations that are similar in magnitude to biological differences. These systematic unwanted variations cannot be effectively distinguished from biological differences using the current methods. In addition, there is a significant need for tools that can effectively evaluate how robust or stable correction data integration methods can achieve in the presence of diverse types of non-systematic and systematic noises. Furthermore, data integration tends to be biased toward the major cell types that are commonly abundant across the integrated datasets, thereby not providing stable integration for the small cell types present only in particular small sets of datasets. Finally, owing to recent technical advances, the number of detected cells has substantially increased. Thus, the scalability of these methods needs to be improved to effectively handle large numbers of cells. Therefore, there are still plenty of room for improvements. Nonetheless, data integration methods have been applied to answer diverse single-cell-level biological and medical questions. Along with the improvement of these methods, their continuous application will shed new insights into cellular players and their interactions underlying disease pathogenesis, and provide new cellular targets for the treatment of various diseases.

Note: Supplementary information is available on the Molecules and Cells website (www.molcells.org).

ACKNOWLEDGMENTS

This study was supported by the Bio & Medical Technology Development Program of the National Research Foundation (NRF), funded by the Korean government (MSIT) (No. 2019M3A9B6066967).

AUTHOR CONTRIBUTIONS

Y.R., G.H.H., and E.J. wrote the original draft. Y.R., G.H.H., E.J., and D.H. reviewed and revised the manuscript.

CONFLICT OF INTEREST

The authors have no potential conflicts of interest to disclose.

ORCID

Yeonjae Ryu	https://orcid.org/0000-0003-3608-0885
Geun Hee Han	https://orcid.org/0000-0003-4723-3410
Eunsoo Jung	https://orcid.org/0000-0003-2192-5603
Daehee Hwang	https://orcid.org/0000-0002-7553-0044

REFERENCES

Amodio, M., van Dijk, D., Srinivasan, K., Chen, W.S., Mohsen, H., Moon, K.R., Campbell, A., Zhao, Y., Wang, X., Venkataswamy, M., et al. (2019). Exploring single-cell data with deep multitasking neural networks. Nat. Methods *16*, 1139-1145.

Aran, D., Looney, A.P., Liu, L., Wu, E., Fong, V., Hsu, A., Chak, S., Naikawadi, R.P., Wolters, P.J., Abate, A.R., et al. (2019). Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. Nat. Immunol. *20*, 163-172.

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.

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

Barrett, T., Wilhite, S.E., Ledoux, P., Evangelista, C., Kim, I.F., Tomashevsky, M., Marshall, K.A., Phillippy, K.H., Sherman, P.M., Holko, M., et al. (2013). NCBI GEO: archive for functional genomics data sets--update. Nucleic Acids Res. *41*(Database issue), D991-D995.

Blondel, V.D., Guillaume, J.L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. J. Stat. Mech. *2008*, P10008.

Bolstad, B.M., Irizarry, R.A., Åstrand, M., and Speed, T.P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. Bioinformatics *19*, 185-193.

Brennecke, P., Anders, S., Kim, J.K., Kołodziejczyk, A.A., Zhang, X., Proserpio, V., Baying, B., Benes, V., Teichmann, S.A., Marioni, J.C., et al. (2013). Accounting for technical noise in single-cell RNA-seq experiments. Nat. Methods *10*, 1093-1095.

Bryois, J., Calini, D., Macnair, W., Foo, L., Urich, E., Ortmann, W., Iglesias, V.A., Selvaraj, S., Nutma, E., Marzin, M., et al. (2022). Cell-type-specific cis-eQTLs in eight human brain cell types identify novel risk genes for psychiatric and neurological disorders. Nat. Neurosci. *25*, 1104-1112.

Buettner, F., Natarajan, K.N., Casale, F.P., Proserpio, V., Scialdone, A., Theis, F.J., Teichmann, S.A., Marioni, J.C., and Stegle, O. (2015). Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. Nat. Biotechnol. *33*, 155-160.

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.

Bzdok, D., Altman, N., and Krzywinski, M. (2018). Statistics versus machine learning. Nat. Methods *15*, 233-234.

Chen, H.I., Jin, Y., Huang, Y., and Chen, Y. (2016). Detection of high variability in gene expression from single-cell RNA-seq profiling. BMC Genomics *17* Suppl *7*, 508.

Cheng, S., Li, Z., Gao, R., Xing, B., Gao, Y., Yang, Y., Qin, S., Zhang, L., Ouyang, H., Du, P., et al. (2021). A pan-cancer single-cell transcriptional atlas of tumor infiltrating myeloid cells. Cell *184*, 792-809.e23.

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. InterJournal, Complex Systems *1695*, 1-9.

Giorgino, T. (2009). Computing and visualizing dynamic time warping alignments in R: the dtw Package. J. Stat. Softw. *31*, 1-24.

Giustacchini, A., Thongjuea, S., Barkas, N., Woll, P.S., Povinelli, B.J., Booth, C.A.G., Sopp, P., Norfo, R., Rodriguez-Meira, A., Ashley, N., et al. (2017). Single-cell transcriptomics uncovers distinct molecular signatures of stem cells in chronic myeloid leukemia. Nat. Med. *23*, 692-702.

Greene, W.H. (1994). Accounting for Excess Zeros and Sample Selection in Poisson and Negative Binomial Regression Models (New York: New York University).

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.

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

Johnson, W.E., Li, C., and Rabinovic, A. (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics *8*, 118-127.

Kadoki, M., Patil, A., Thaiss, C.C., Brooks, D.J., Pandey, S., Deep, D., Alvarez, D., von Andrian, U.H., Wagers, A.J., Nakai, K., et al. (2017). Organism-level analysis of vaccination reveals networks of protection across tissues. Cell *171*, 398-413.e21.

Kim, Y., Kim, T.K., Kim, Y., Yoo, J., You, S., Lee, I., Carlson, G., Hood, L., Choi, S., and Hwang, D. (2011). Principal network analysis: identification of subnetworks representing major dynamics using gene expression data. Bioinformatics *27*, 391-398.

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.

Kotliar, D., Veres, A., Nagy, M.A., Tabrizi, S., Hodis, E., Melton, D.A., and Sabeti, P.C. (2019). Identifying gene expression programs of cell-type identity and cellular activity with single-cell RNA-Seq. Elife *8*, e43803.

Kriebel, A.R. and Welch, J.D. (2022). UINMF performs mosaic integration of single-cell multi-omic datasets using nonnegative matrix factorization. Nat. Commun. *13*, 780.

Li, X., Wang, K., Lyu, Y., Pan, H., Zhang, J., Stambolian, D., Susztak, K., Reilly, M.P., Hu, G., and Li, M. (2020). Deep learning enables accurate clustering with batch effect removal in single-cell RNA-seq analysis. Nat. Commun. *11*, 2338.

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.

Lopez, R., Regier, J., Cole, M.B., Jordan, M.I., and Yosef, N. (2018). Deep generative modeling for single-cell transcriptomics. Nat. Methods *15*, 1053-1058.

Lotfollahi, M., Naghipourfar, M., Theis, F.J., and Wolf, F.A. (2020). Conditional out-of-distribution generation for unpaired data using transfer VAE. Bioinformatics *36*(Suppl_2), i610-i617.

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

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.

Lun, A.T., McCarthy, D.J., and Marioni, J.C. (2016). A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. F1000Res. *5*, 2122.

McKellar, D.W., Walter, L.D., Song, L.T., Mantri, M., Wang, M.F.Z., De Vlaminck, I., and Cosgrove, B.D. (2021). Large-scale integration of single-cell transcriptomic data captures transitional progenitor states in mouse skeletal muscle regeneration. Commun. Biol. *4*, 1280.

Molania, R., Gagnon-Bartsch, J.A., Dobrovic, A., and Speed, T.P. (2019). A new normalization for Nanostring nCounter gene expression data. Nucleic Acids Res. *47*, 6073-6083.

Morabito, S., Miyoshi, E., Michael, N., Shahin, S., Martini, A.C., Head, E., Silva, J., Leavy, K., Perez-Rosendahl, M., and Swarup, V. (2021). Single-nucleus chromatin accessibility and transcriptomic characterization of Alzheimer's disease. Nat. Genet. *53*, 1143-1155.

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.

Regev, A., Teichmann, S.A., Lander, E.S., Amit, I., Benoist, C., Birney, E., Bodenmiller, B., Campbell, P., Carninci, P., Clatworthy, M., et al. (2017). The human cell atlas. Elife *6*, e27041.

Reichart, D., Lindberg, E.L., Maatz, H., Miranda, A.M.A., Viveiros, A., Shvetsov, N., Gartner, A., Nadelmann, E.R., Lee, M., Kanemaru, K., et al. (2022). Pathogenic variants damage cell composition and single cell transcription in cardiomyopathies. Science *377*, eabo1984.

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

Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., and Smyth, G.K. (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. *43*, e47.

Smillie, C.S., Biton, M., Ordovas-Montanes, J., Sullivan, K.M., Burgin, G.,

Graham, D.B., Herbst, R.H., Rogel, N., Slyper, M., Waldman, J., et al. (2019). Intra- and inter-cellular rewiring of the human colon during ulcerative colitis. Cell *178*, 714-730.e22.

Stuart, T., Butler, A., Hoffman, P., Hafemeister, C., Papalexi, E., Mauck, W.M., 3rd, Hao, Y., Stoeckius, M., Smibert, P., and Satija, R. (2019). Comprehensive integration of single-cell data. Cell *177*, 1888-1902.e21.

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.

Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., Lennon, N.J., Livak, K.J., Mikkelsen, T.S., and Rinn, J.L. (2014). The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nat. Biotechnol. *32*, 381-386.

Uchimura, K., Wu, H., Yoshimura, Y., and Humphreys, B.D. (2020). Human pluripotent stem cell-derived kidney organoids with improved collecting duct maturation and injury modeling. Cell Rep. *33*, 108514.

Vallejos, C.A., Marioni, J.C., and Richardson, S. (2015). BASiCS: Bayesian analysis of single-cell sequencing data. PLoS Comput. Biol. *11*, e1004333.

Villa, C.E., Cheroni, C., Dotter, C.P., Lopez-Tobon, A., Oliveira, B., Sacco, R., Yahya, A.C., Morandell, J., Gabriele, M., Tavakoli, M.R., et al. (2022). CHD8 haploinsufficiency links autism to transient alterations in excitatory and inhibitory trajectories. Cell Rep. *39*, 110615.

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.e17.

Xu, C., Lopez, R., Mehlman, E., Regier, J., Jordan, M.I., and Yosef, N. (2021). Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. Mol. Syst. Biol. *17*, e9620.

Yang, Z. and Michailidis, G. (2016). A non-negative matrix factorization method for detecting modules in heterogeneous omics multi-modal data. Bioinformatics *32*, 1-8.

Yoon, B.K., Oh, T.G., Bu, S., Seo, K.J., Kwon, S.H., Lee, J.Y., Kim, Y., Kim, J.W., Ahn, H.S., and Fang, S. (2022). The peripheral immune landscape in a patient with myocarditis after the administration of BNT162b2 mRNA vaccine. Mol. Cells *45*, 738-748.

Young, A.L., Marinescu, R.V., Oxtoby, N.P., Bocchetta, M., Yong, K., Firth, N.C., Cash, D.M., Thomas, D.L., Dick, K.M., Cardoso, J., et al. (2018). Uncovering the heterogeneity and temporal complexity of neurodegenerative diseases with Subtype and Stage Inference. Nat. Commun. *9*, 4273.