Performance Assessment and Selection of Normalization Procedures for Single-Cell RNA-Seq

Michael B. Cole*[1], Davide Risso*[2], Allon Wagner[3,4], David DeTomaso[4], John Ngai[5], Elizabeth Purdom[4,6], Sandrine Dudoit†[4,6,7], and Nir Yosef†[3,4]

[1]Department of Physics, University of California, Berkeley
[2]Division of Biostatistics and Epidemiology, Department of Healthcare Policy and Research, Weill Cornell Medicine, New York, NY
[3]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley
[4]Center for Computational Biology, University of California, Berkeley
[5]Department of Molecular and Cell Biology, University of California, Berkeley
[6]Department of Statistics, University of California, Berkeley
[7]Division of Biostatistics, School of Public Health, University of California, Berkeley

December 13, 2017

Abstract

Due to the presence of systematic measurement biases, data normalization is an essential preprocessing step in the analysis of single-cell RNA sequencing (scRNA-seq) data. While a variety of normalization procedures are available for bulk RNA-seq, their suitability with respect to single-cell data is still largely unexplored. Furthermore, there may be multiple, competing considerations behind the assessment of normalization performance, some of them study-specific. The choice of normalization method can have a large impact on the results of downstream analyses (e.g., clustering, inference of cell lineages, differential expression analysis), and thus it is critically important to assess the performance of competing methods in order to select a suitable procedure for the study at hand.

We have developed *scone* – a framework that implements a wide range of normalization procedures in the context of scRNA-seq, and enables the assessment of their performance based on a comprehensive set of data-driven performance metrics. The accompanying open-source Bioconductor R software package scone (available at `https://bioconductor.org/packages/scone`) also provides numerical and graphical summaries of expression measures, data quality assessment, and data-adaptive gene and sample filtering criteria. We demonstrate the effectiveness of *scone* on a selection of scRNA-seq datasets across a variety of protocols, ranging from plate- to droplet-based methods. We show that *scone* is able to correctly rank normalization methods according to their performance in a given dataset and that selecting the best performing normalization leads to higher agreement with independent validation data than lowly-ranked methods.

Introduction

Normalization is a common preprocessing step in the analysis of -omic data, such as high-throughput transcriptome microarray and sequencing (RNA-seq) data. The goal of normalization is to account for observed differences in measurements between samples and/or features (e.g., genes) resulting from technical artifacts or unwanted biological effects (e.g., batch effects) rather than biological effects of interest. In order to derive gene expression measures from single-cell RNA sequencing (scRNA-seq) data and compare these measures between cells, one first needs to normalize read counts (or other expression unit) to adjust for obvious differences in sequencing depths as well as more complex unwanted technical artifacts linked to sample and library

*These authors contributed equally.
†These authors contributed equally.

preparation. This manuscript focuses on normalization as part of a distinct preprocessing step, although methods have been proposed for combining normalization with downstream analyses such as dimensionality reduction and the identification of differentially expressed genes [1, 2].

As previously discussed [3, 4], normalization of scRNA-seq data is often performed using methods developed for bulk RNA-seq or even microarray data. These methods tend to neglect prominent features of scRNA-seq data such as: *zero inflation*, i.e., an excessive proportion of zero read counts [1, 5]; large nuisance technical effects (e.g., Fluidigm C1 run), comparable in magnitude to the biological effects of interest [6]; uneven sample quality, e.g., in terms of alignment rates and nucleotide composition of the reads [7]. In particular, widely-used global-scaling procedures, such as reads per million (RPM) [8], trimmed mean of M values (TMM) [9], and DESeq [10], are not well suited to handle large batch effects and may be biased by zero inflation [4]. Other more flexible methods, such as remove unwanted variation (RUV) [11, 12] and surrogate variable analysis (SVA) [13, 14], depend on tuning parameters (e.g., the number of unknown factors of unwanted variation).

A handful of normalization methods specifically designed for scRNA-seq data have recently been proposed. These include scaling methods [15, 16], regression-based methods for known nuisance factors [17, 18], and methods that rely on spike-in sequences from the External RNA Controls Consortium (ERCC) [19, 20]. While these methods alleviate some of the problems that affect bulk normalization methods, each suffers from obvious limitations with respect to their applicability across diverse study designs and experimental protocols. Global-scaling methods infer a single factor per cell, and thus are unable to account for complex batch effects. Explicit regression on known nuisance factors (e.g., number of reads in a library) may miss unknown, yet unwanted variation, which may still confound the data [12]. Unsupervised normalization by regression on unknown unwanted factors may perform poorly with default parameters (e.g., number of factors adjusted for) and require tuning, while ERCC-based methods may suffer from differences between endogenous and spiked-in transcripts [4, 12]. Finally, it is important to note that protocols using unique molecular identifiers (UMIs) still require normalization; while UMIs remove amplification biases, they are often sensitive to sequencing depth and differences in capture efficiency before reverse transcription [4].

Due to the large trade-offs between normalization methods and the ambiguity in the specification of associated tuning parameters, we advocate for the inspection and evaluation of a variety of normalization strategies, using multiple data-driven metrics, in order to select suitable procedures for a given dataset. To address this task, we have developed the *scone* framework for implementing and assessing the performance of a range of normalization workflows, each consisting of multiple steps, such as gene and sample filtering, scaling, and supervised or unsupervised regression-based procedures. *scone* evaluates the performance of each workflow and ranks them by aggregating over a set of performance metrics that consider different aspects of a desired normalization outcome, covering both removal of unwanted variation and retainment of wanted variation.

We demonstrate that the *scone* methodology and software are general and applicable to different scRNA-seq protocols, including microfluidic (e.g., Fluidigm C1), plate (e.g., SMART-Seq2), and droplet (e.g., 10x Genomics Chromium) methods. The modularity of the scone software package allows researchers to compare a set of default normalizations as well as to include user-defined normalization methods, providing a useful framework for both practitioners and method developers. While our focus here is normalization, the *scone* framework is more general, facilitating the comparison of both imputation methods [21, 22] and dimensionality reduction techniques [2, 23, 24].

Results

scRNA-seq data are affected by batch effects and other unwanted variation

Our primary example is a set of 420 mouse Th17 T-cell SMART-Seq2 libraries published in Gaublomme *et al.* [25]. We focused on a subset of cells harvested after *in vitro* differentiation of CD4+ naive T-cells under 48 hours of pathogenic (IL-1β+IL-6+IL-23) or non-pathogenic (TGF-β1+IL-6) conditioning from two mouse strains (wild-type B6 and transgenic B6 mice with an IL17a GFP reporter). Reads were aligned to the mouse genome and subsequently counted over RefSeq gene intervals to generate a cells x genes count matrix. Following sample and gene filtering, 337 libraries and 7,590 genes were retained for normalization and downstream analysis (see Methods).

Even after scaling the gene-level read counts by total counts (TC), mouse-specific effects are prominently featured in *principal component analysis* (PCA; Fig. 1a). In the space defined by the first two *principal components* (PCs), the Euclidean distances between cells from mouse 7 and mouse 8 for a given biological condition are larger than the distances between pathogenic and non-pathogenic cells collected from the same mouse. These batch effects can result from multiple sources, including (i) true biological differences between mice or (ii) mouse-specific technical biases. We can begin to account for these effects by correlating the read count PCs to RNA-seq library quality control (QC) metrics (Fig. 1b; see Tables 1 and 2 and Methods).

The first three count PCs exhibit large correlations with measures of genomic alignment rate, primer contamination, intronic alignment rate, and 5' bias. The correlation structure between these QC measures reflects constraints on library quality in the study (Fig. 1c). While some of the associations solely represent natural dependencies of similar QC measures (e.g., total number of reads and total number of aligned reads), others may reflect plausible mouse-specific technical biases in the study. Applying PCA to the matrix of QC measures, we can see how these metrics provide a candidate basis for adjusting batch (i.e., mouse) effects (Fig. 1d). Inter-batch differences are relatively large as in Figure 1a, while intra-batch differences between pathogenic and non-pathogenic cells are noticeably smaller. It is important to emphasize the possibility that some of the observed associations between counts and QC measures may result from biological confounding, i.e., a cell's biological state may impact transcriptome integrity and sequencing viability [4, 26]. However, unlike factors such as mouse-of-origin, there exists a clear mechanistic interpretation for correlations between quantified expression measures and library alignment statistics. As in many single-cell studies, the partial *confounding* resulting from the experimental design impedes efforts to infer independent contributions of biological condition and batch effects (Fig. 1e). However, by monitoring the relationship between read counts and QC measures we can begin to take rational steps to lessen the effect of batches on expression measures. For example, we may normalize expression measures by regressing read counts on principal components of QC measures.

scone: A framework for the normalization of scRNA-seq data

As illustrated in Figure 1, simple global scaling alone is insufficient to correctly normalize scRNA-seq data, and more complex, regression-based normalization methods aimed at removing unwanted variation (e.g., batch effects) may be beneficial. In addition, the ability to identify and remove outlying cells, which may skew the overall distribution of the data, and the proper handling of *dropout genes* (i.e., expressed but undetected genes, with zero read counts) are essential steps in scRNA-seq data analysis.

Here, we present a general framework for implementing and evaluating preprocessing (e.g., gene and sample filtering, imputation of zero counts), normalization, and dimensionality reduction of scRNA-seq data. As illustrated in Figure 2, the *scone* pipeline consists of several additional steps prior to normalization. At an early stage we use a set of housekeeping genes to estimate the fraction of gene dropouts and compute a false negative rate (FNR) curve for each sample (see Methods). These curves may be used both for low-quality sample filtering and for the optional imputation of dropouts. The filtering step also includes data-adaptive thresholds on the total number of sequenced reads and percentage of aligned reads. Note that, as the focus of this paper is normalization, we leave the imputation option disabled in all subsequent analyses.

Following the selection of good-quality cell libraries, *scone* uses a two-step normalization template: (i) scaling of the counts to account for sequencing depth and other distributional differences among samples; (ii) regression-based adjustment for unwanted factors, such as processing batches and other complex, possibly unknown effects. As illustrated in Figure 2, many different approaches are employed in both steps of our normalization pipeline. For instance, one can simply scale the data by the total number of reads (TC) or apply more robust procedures designed to reduce the effect of outliers (such as TMM [9] or DESeq [10]). Analogously, confounding factors can be adjusted for either by directly regressing out quantities known to influence read counts (e.g., batches or the QC measures shown in Figure 1) or by an unsupervised procedure that aims at estimating hidden unwanted factors (e.g., using negative control genes as in RUV [12]). All possible choices of normalization workflow are considered in the evaluation stage. Before comparing different normalizations, we re-zero matrix elements that were originally zero and floor expression measures at zero. Given the high frequency of zero measurements in single-cell data, our choice to re-zero unmeasured transcript abundances prevents an evaluation dominated by effects on unobserved zero measurements.

The final evaluation step of *scone* is the ranking of all normalization workflows under consideration to identify the top performing methods. To achieve this, *scone* calculates a set of eight performance metrics, aimed at capturing different aspects of successfully normalized data. We classify these metrics into three broad categories: (i) clustering of samples according to factors of wanted and unwanted variation; (ii) association of expression principal components with factors of wanted and unwanted variation; (iii) between-sample distributional properties of the expression measures (see Methods). Overall, these metrics capture the trade-offs between the ability of a normalization method to remove unwanted variation and its ability to preserve biological variation of interest. The final ranking is obtained by averaging ranks based on the eight individual metrics.

Below we provide evidence that no single normalization method is uniformly optimal and that performance depends on the design of the experiment and on other characteristics of the data. This suggests an alternative role for *scone* beyond benchmarking: Rather than choosing a fixed normalization pipeline, practitioners may apply many combinations of methods and parameters and compare their performance on a given dataset, ranking each strategy by its ability to improve up to eight performance metrics.

scone removes unwanted variation while preserving biological variation of interest

The *scone* biplot (Fig. 3a) for the normalization of the Th17 dataset presented in Figure 1 demonstrates natural trade-offs between batch effect removal, unwanted variation (negative control genes, "UV", or library QC measures, "QC"), biological clustering, and wanted variation (positive control genes, "WV"). Compared to no normalization, global-scaling and full-quantile normalization primarily improve distributional properties of the data, reducing the amount of global differential expression (DE) between samples (captured by the RLE metrics; see Methods). Regression-based normalization, including batch regression and RUV, removed unwanted variation at the expense of wanted variation; the biplot can help identifying those normalization that balance the trade-off between removing too much and too little variation. The factors of unwanted variation computed from negative control genes (UV) and the QC measures were not always correlated (Supplementary Fig. 1), suggesting that these are complementary approaches.

The existence of the trade-off between removing unwanted variation and preserving wanted variation suggests that wanted variation is confounded with measurement artifacts. This is at least partly due to the imperfect nested experimental design (Fig. 1e), but may also reflect how different cell types/states are more prone to stress or RNA degradation.

Global-scaling normalization methods were ranked similarly by the *scone* aggregate score (see Methods), outperforming methods that did not include a scaling step and underperforming the more aggressive full-quantile normalization (Fig. 3b). Importantly, single-cell-specific methods, such as those implemented in the R packages scran and SCnorm, did not outperform methods developed for bulk RNA-seq. The inclusion of a batch regression step in the normalization strategy was desirable in this case; methods including QC or RUV factors without batch normalization performed poorly (Fig. 3c). This result indicate that for this study, preexisting batch classifications are better proxies of inter-batch effects than QC or RUV factors, despite their problematic associations with biological condition.

We validated the *scone* ranking by comparing normalized expression measures to data from a separate study of bulk differential expression between pathogenic and non-pathogenic Th17 cells [27]. We considered the 1,000 most and 1,000 least differentially expressed genes from that study (according to limma p-value [28]) and assessed our ability to discriminate between these gene sets based on the normalized single-cell measures. Specifically, for each normalization procedure, we used limma to test for differences in expression between Th17-positive pathogenic cells and unsorted non-pathogenic cells, generating receiver operating characteristic (ROC) curves (see Methods). We observed a relatively low correlation between the ROC area under the curve (AUC) and the *scone* aggregate score (Fig. 3d; Spearman correlation coefficient of 0.4). When considering many normalization workflows, we are mostly interested in the top ranking ones to provide a solid basis for further exploration and downstream analysis. We found that the top ten normalizations as ranked by *scone* consistently performed well in terms of ROC AUC, and better than workflows that consisted of scaling only (Fig. 3e). Furthermore, the bottom ten workflows, tended to have low AUC values. These results indicate that the *scone* ranking is a good way of detecting suitable normalization workflows for a given dataset. Interestingly, while the *scone* ranking is not necessarily consistent with the AUC-based ranking across the whole performance range (e.g., under-ranking QC-only methods), we found an overall high level of agreement between the two rankings at the level of method classes (Fig. 3d). For instance, it is evident that both rankings agreed that normalization workflows that did not adjust for unwanted variation (e.g., simple scaling normalization) performed poorly (Fig. 3e). The same was true for methods using RUV without adjusting explicitly for batch effects. The *scone* and AUC rankings also agreed on the good performance of all methods that included a batch adjustment step (Fig. 3d). Moreover, the AUC confirmed that full-quantile outperformed global-scaling normalization in this dataset and that adjusting for batch and other unwanted variation was more important than the choice of scaling method (Fig. 3d).

We performed a similar analysis on a set of developing cortical neurons assayed using the Fluidigm C1 platform [29] and on a set of peripheral blood mononuclear cells (PBMCs) assayed using the 10x Genomics Chromium platform [30] (Supplementary Fig. 2 and 3). We likewise found that *scone* was able to identify normalization methods that led to good agreement with external data (Supplementary Fig. 4 and 5). This confirmed the ability of *scone* to correctly identify appropriate normalization methods on different platforms (Fluidigm C1, 10x Genomics, and plate-sorted SMART-Seq2). Specifically, full-quartile (FQ) normalization outperformed global-scaling normalization methods for the Fluidigm C1 cortex dataset (Supplementary Fig. 4), while scaling by the DESeq size factor outperformed other scaling normalizations for the 10x Genomics PBMC dataset (Supplementary Fig. 5). Importantly, external validation data confirmed that the top 10 normalization workflows as ranked by *scone* outperformed the bottom ten both for the cortex dataset (Fig. 3f) and for the PBMC dataset (Fig. 3g). In contrast with our results above, QC-based regression normalization outperformed control-gene RUV methods for both of these datasets. Taken together, these observations suggest that there is no single normalization strategy that uniformly outperforms the others and that *scone* is able to identify the best normalization workflows in a data-dependent fashion.

Using contrasts to adjust for batch effects in nested designs

The *scone* analysis of the Th17 dataset (Fig. 3) demonstrated the importance of correcting for batch effects. However, depending on the experimental design, simply including a batch indicator variable in the model is not always a viable option. As an extreme case, imagine a completely *confounded design*, in which each biological condition is assayed in a distinct batch. In such a case, regressing out the batch indicator from the expression measures will result in the removal of biological effects; conversely, not accounting for batch will make it impossible to attribute the observed differences in expression measures to biological differences between conditions or technical differences between batches. Note that this is not just a thought experiment: Several examples of datasets with such poor designs are discussed in Hicks *et al.* [6].

On the opposite end of the spectrum are experiments designed in such a way that each batch contains cells from each biological condition. Such *factorial designs* are the optimal choice, when possible. Hicks *et al.* [6] and Tung *et al.* [31] discuss practical aspects of designing factorial experiments in the context of scRNA-seq.

Although optimal, factorial experiments are not always possible or practical. An alternative strategy is to collect multiple batches of cells from each biological condition of interest, in what we refer to as a *nested design*. A good example of nested design is given by the iPSC dataset analyzed in Tung *et al.* [31] (Fig. 4). After scaling normalization, the cells clearly cluster by individual, but the cells for each individual are further clustered by batch (Fig. 4a). Blindly removing these batch effects with a usual batch correction method, such as ComBat, removes the biological effects of interest along with the batch effects (Supplementary Fig. 6). Moreover, the library QC measures collected as part of the *scone* pipeline are not able to completely capture the batch effects (Fig. 4b), as the space of the first two PCs of the QC measures is dominated by the difference between a subset of low-quality cells and the rest of the cells. Explicitly accounting for the nested nature of the design while adjusting for batch effects is the only strategy that effectively removes the unwanted technical variation and preserves the biological signal of interest (Fig. 4c; see Methods for details on our nested batch effect correction). Importantly, the scone package is able to detect nested designs by examining the cross-tabulation of the biological and batch factors. Given a nested design, the nested batch effect adjustment based on Equations (3) and (4) is automatically applied as one of the different normalization strategies to be compared (see Methods). Nested designs are common in single-cell studies due to various practical constraints (e.g., processing material from different tissues separately). The aggregate ranking (Fig. 4d) shows how only methods that remove the nested batch effects rank high in the *scone* evaluation step. This holds when removing the two performance metrics that directly involve the batch indicator (BIO_SIL and BATCH_SIL; see Methods), suggesting that the result is not driven by these two metrics alone (Supplementary Fig. 6).

The scone package's user interface facilitates the exploration of normalized data

As part of the Bioconductor R package scone, we developed a Shiny app [32] that allows users to interactively explore the data at various stages of the *scone* workflow. In Figure 5, we use the cortical neurons dataset of Pollen *et al.* [29] to illustrate the app's functionality.

As shown in Figure 3a, a useful representation of the global properties of a dataset is the *biplot* [33], in which each point corresponds to a normalization strategy and the dimensions of variation, represented by the red arrows, correspond to the performance metrics. The scone package provides a function to display an interactive version of the biplot and the user can select a group of normalizations for further exploration with the app (Fig. 5a).

First, the app provides a hierarchical overview of all the compared normalization strategies (Fig. 5b). The hierarchy is based on the series of algorithmic choices that constitute a given normalization strategy: In the example, the first level of the hierarchy represents scaling (e.g., FQ, TMM), while the second represents regression-based methods (QC or RUV). In general, additional levels are present for the optional imputation and batch correction steps. Alternatively, the user can select a normalization strategy using the drop-down menu in the left panel of the app or using the interactive table at the bottom of the screen; this table can be sorted by the *scone* aggregate score or by any individual performance metric, making it easy to select, for instance, the method that maximizes the preservation of wanted variation (as measured by the EXP_WV_COR metric; see Methods).

Once a normalization approach has been selected for inspection, the shiny app provides six exploratory tabs that provide an extended view of the normalized data and should guide the selection of the final method. Here, we focus on three examples (see Methods for details). The "Silhouette" tab (Fig. 5c) shows the silhouette width for each sample, for clustering based on partitioning around medoids (PAM), clustering by batch, or clustering by biological condition (if available). If the user provides a set of positive and negative control genes, these are visualized in the "Control Genes" panel (Fig. 5d) in a heatmap that includes batch, biology, and PAM clustering information. Similarly, the "Relative Log-Expression" tab displays boxplots of the relative log-expression (RLE) measures for the normalized data (Fig. 5e).

Discussion

Many different normalization schemes are available, either specifically designed for scRNA-seq or borrowed from the bulk RNA-seq literature. Here, we have demonstrated that simple global-scaling normalization is not always sufficient to correctly normalize the data and that more sophisticated strategies may be needed. However, different normalization strategies may perform differently across datasets, depending on the experimental design, protocol, and platform.

The main idea behind *scone* is to use a data-driven approach to select an appropriate normalization strategy for the data at hand. Although it may be infeasible to select a "best" normalization, as this would depend on a somewhat subjective definition of optimality, *scone* provides a set of performance metrics (and clustering via the biplot) that can be used to reduce the number of normalization methods to further explore in the selection of a suitable strategy. We have shown using real data spanning different labs and technologies that *scone* is able to reliably rank normalizations by summarizing multiple dimensions of data quality.

Given the small amount of RNA present in a single cell, scRNA-seq data are characterized by a large fraction of dropouts, i.e., genes that are expressed but not detected. To account for the resulting false negatives, *scone* computes false negative rates that can be used to filter out low-quality samples and optionally to impute the expression measure of dropout genes. Zero imputation is still in its infancy; only a handful of methods exist [21, 22] and it is unclear whether their promised advantages outweigh their limitations. Although we focused on normalization, *scone* can be used to compare different imputation methods (including no imputation). An alternative approach to imputation is to model the zeros as part of dimensionality reduction. An example of such a method, ZINB-WaVE [2], has the ability to include additional covariates to produce a low-dimensional representation of the data that is not influenced by unwanted variation. In principle, the covariates (e.g., QC or RUV factors) selected by *scone* as important for normalization can be included in ZINB-WaVE to provide a more robust projection of the data.

We wish to emphasize that although *scone* has demonstrated its usefulness in published studies (e.g., [34, 35]) and normalization can help remove unwanted variation from data, a careful experimental design is the most important aspect of a successful analysis of scRNA-seq data. In fact, if the biological effects of interest are completely confounded with unwanted technical effects, no statistical method will be able to extract meaningful signal from the data [6].

The present article focuses on scRNA-seq, but the methodology and software are general and applicable to other types of assays, including microarray, bulk RNA-seq, and adductomic and metabolomic assays. In particular, the user could extend the package by adding different metrics for scRNA-seq, as well as metrics specific to other assays. The scone package implementation leverages core Bioconductor packages for efficient parallel computation and on-disk data representation, both essential when analyzing large datasets [36–38]. The computational complexity of *scone* is directly related to the complexity of the normalization methods included in the comparisons. In particular, all scaling methods, RUV, and regression-based methods are very efficient, leading to reasonable computing time (e.g., 60 hours with 1 CPU for the 10x Genomics PBMC dataset of 12,039 cells). With parallelization this computation can be sped up considerably (e.g., 11 hours with 10 processors). For very large dataset, sub-sampling can be used to decrease computation: One or more random subsets of the samples can be used to run *scone* and only the selected normalization can be subsequently applied to the full dataset.

Overall, *scone* provides a flexible and modular framework for the preprocessing of scRNA-seq data that can be used by practitioners to evaluate the impact of the statistical design of a given study and select an appropriate normalization, as well as by method developers to systematically compare a proposed strategy to state-of-the art approaches.

Methods

The *scone* workflow

The *scone* workflow (Fig. 2) consists of six steps: (i) Inference of dropout genes; (ii) gene and sample filtering; (iii) optional imputation or weighting of zeros; (iv) global-scaling or full-quantile normalization; (v) regression-based normalization for batch effects or other known or unknown factors of unwanted variation; (vi) performance assessment and selection of normalization procedures.

We provide details on each of these steps below, but we first introduce two key concepts that will be used throughout the workflow: Sample-level quality measures and zero inflation.

Sample-level quality control measures

For assessing scRNA-seq data quality at the sample level, we rely on over a dozen *quality control* (QC) metrics provided by software packages such as `FastQC` (`http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`), `Picard` (`http://broadinstitute.github.io/picard/`), and `Cell Ranger` (`https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/what-is-cell-ranger`). QC measures relate to the alignment of the reads to a reference sequence (e.g., genome, transcriptome) and the nucleotide composition of the reads, including, for instance, the total number of reads (NREADS), the percentage of mapped reads (RALIGN), and the percentage of intronic bases (PCT_INTRONIC_BASES). A full description of general QC measures and 10x Genomics-specific QC measures is presented in Tables 1 and 2, respectively.

QC measures can vary substantially within and between batches (e.g., C1 run) and point to low-quality samples (e.g., low percentage of mapped reads) whose inclusion in downstream analyses could seriously bias results. Additionally, QC measures can be strongly associated with read counts (Fig. 1). As detailed below, it is therefore advisable to filter samples based on QC and consider QC measures in the normalization process.

Zero inflation

Single-cell RNA-seq data have many more genes with zero read counts than bulk RNA-seq data. In some cases, the number of zeros far exceeds predictions from standard low-mean count distributions; this *zero inflation* (ZI) can occur for biological reasons (i.e., the gene is simply not expressed) or technical reasons (e.g., low capture efficiency and amplification).

Zero inflation is problematic for normalization, as many methods involve scaling and perform poorly when a large proportion of genes have zero counts. In particular, the global-scaling method of Anders & Huber [10], implemented in the Bioconductor R package DESeq, discards any gene having zero count in at least one sample. In practice, the scaling factors are therefore estimated based on only a handful of genes. Quantile-based methods [39] also do not behave properly due to ties from the large number of zeros.

In the context of zero inflation, it is informative to examine the distribution of zero counts, as this might reveal problematic cells or genes. For instance, cells with unusually high proportion of genes with zero counts might have encountered PCR or read alignment problems. Additionally, in an attempt to determine whether zeros are technical, one can employ *housekeeping genes* which are expected to be expressed at fairly high and constant levels across cells.

Specifically, let Y_{ij} denote the log-read count of gene $j \in \{1, \ldots, J\}$ in cell/sample $i \in \{1, \ldots, n\}$ and let $\mathcal{J}_0 \subseteq \{1, \ldots, J\}$ denote a set of housekeeping genes. For each cell i, fit the following *logistic regression* model to the housekeeping genes only:

$$\operatorname{logit} \mathrm{E}[\tilde{Y}_{ij} | \bar{Y}_{\cdot j}] = \beta_{i0} + \beta_{i1} \bar{Y}_{\cdot j}, \qquad j \in \mathcal{J}_0, \tag{1}$$

where $\tilde{Y}_{ij} \equiv \mathrm{I}(Y_{ij} > \epsilon)$ is an indicator variable equal to 1 if gene j is "detected" and zero otherwise (as determined by a detection threshold $\epsilon \geq 0$), $\bar{Y}_{\cdot j}$ is the median log-read count of gene j across all non-zero samples, $\beta_i = (\beta_{i0}, \beta_{i1}) \in \mathbb{R}^2$ are cell-specific regression parameters, and the logit function is defined as $\operatorname{logit}(x) \equiv \log(x/(1-x))$ for $x \in [0, 1]$. The motivation behind this model is that the detection rate $\Pr(Y_{ij} > \epsilon | \bar{Y}_{\cdot j})$ of gene j in cell i should be related to the "baseline" expression measure $\bar{Y}_{\cdot j}$ of the gene in a cell-specific manner (i.e., cell-specific regression parameters β_i) and that undetected housekeeping genes should constitute "false negatives". In practice, we set $\epsilon = 0$.

For a given detection threshold ϵ, Equation (1) yields *false negative rate* (FNR) curves for each cell, where each point on the curve corresponds to a gene's "failure probability" $\Pr(Y_{ij} \leq \epsilon | \bar{Y}_{\cdot j})$ at a baseline expression $\bar{Y}_{\cdot j}$. The higher the curve, the lower the quality of the sample. Samples can then be compared based on the *area under the curve* (AUC) for their respective FNR curves.

Note that we assume that the housekeeping genes are consistently expressed across all samples. This assumption allows us to treat the zeros as "dropouts" rather than as biologically meaningful zeros, hence the interpretation of $\Pr(Y_{ij} \leq \epsilon | \bar{Y}_{\cdot j})$ as failure probability.

Gene and sample filtering

We perform preliminary gene and sample filtering to deal with zero inflation and poor data quality. Sample filtering was not applied in the case of Pollen *et al.* [29] due to the small sample size. Filtering occurs in three steps, where the third step ensures that a gene is detected in a sufficient number of samples after sample filtering.

1. Define *common genes* based on read counts: Genes with more than n_r reads in at least f_s of samples, where n_r is the upper-quantile of the non-zero elements of the samples x genes count matrix. In practice, we set $f_s = 25\%$.

2. Sample filtering based on QC measures: Filter out samples with low number of reads, low proportion of mapped reads (not for 10x Genomics datasets), low number of detected common genes, or high FNR AUC as defined above. The thresholds for each measure are computed in a data-adaptive fashion: A sample could fail any of these criteria if the associated metric underperforms by z_{cut} standard deviations from the mean metric or by z_{cut} median absolute deviations from the median metric. For all filtered datasets, we set $z_{cut} = 2$.

3. Primary gene filtering based on read counts: Filter out genes with fewer than n_r reads in at least n_s samples, where n_r is the upper-quantile of the non-zero elements of the samples x genes count matrix, for samples that passed the filtering described in the previous step. In practice, we set $n_s = 5$ to accommodate markers of rare populations.

Note that for the Th17 dataset, due to the small number of negative control genes (see below), we force inclusion of all detected (i.e., non-zero count) negative controls in Step 3.

Between-sample normalization

A variety of gene- and sample-specific unwanted effects can bias gene expression measures and thus require normalization. Accordingly, we distinguish between two types of normalization: Within-sample normalization [40], which adjusts for gene-specific (and possibly sample-specific) effects, e.g., related to gene length and GC-content, and between-sample normalization, which adjusts for effects related to distributional differences in read counts between samples, e.g., sequencing depth, C1 run, library preparation. Here, we focus on between-sample normalization.

Most between-sample normalization methods proposed to date are adaptations of methods for bulk RNA-seq and microarrays and range, as described next, from simple global-scaling to regression on gene- and sample-level covariates.

Global-scaling normalization

For the simplest *global-scaling normalization* procedures, gene-level read counts are scaled by a single factor per sample.

Total-count (TC). The scaling factor is the sum of the read counts across all genes, as in the widely-used reads per million (RPM), counts per million (CPM), and reads per kilobase of exon model per million mapped reads (RPKM) [8].

Single-quantile. The scaling factor is a quantile of the gene-level count distribution, e.g., upper-quartile (UQ) [39].

Trimmed mean of M values (TMM) [9]. The scaling factor is based on a robust estimate of the overall expression fold-change between the sample and a reference sample. TMM is implemented in the Bioconductor R package edgeR.

DESeq [10]. The scaling factor for a given sample is defined as the median fold-change between that sample and a synthetic reference sample whose counts are defined as the geometric means of the counts across samples. The method is implemented in the Bioconductor R packages DESeq and edgeR (as "RLE"). Note that the method discards any gene having zero count in at least one sample; as a result of zero inflation, the scaling factors are often based on only a handful of genes for single-cell RNA-seq data.

scran [15]. To reduce the effect of zero inflation on normalization, the scaling factors are computed on summed expression measures from pools of cells and then deconvolved to obtain cell-specific factors. Optionally, cells can be clustered based on their expression profiles prior to normalization to relax the assumption that the majority of genes are not differentially expressed across groups of cells. The method is implemented in the Bioconductor R package scran.

While not strictly necessary, our implementation of global-scaling procedures typically preserves the original expression scale by further rescaling the normalized measures by a single value across all samples.

Non-linear normalization

In some cases, a single scaling factor per sample may not be sufficient to capture the non-linear effects that affect gene expression measures. Hence, some authors have proposed non-linear normalization methods. Although, these are not technically "scaling" methods, they belong to the first step of the *scone* workflow, in that they are aimed at making the between-sample distributions of expression measures more similar, rather than explicitly correcting for batch or other confounding factors.

Full-quantile normalization (FQ). All quantiles of the read count distributions are matched between samples [39]. Specifically, for each sample, the distribution of sorted read counts is matched to a reference distribution defined in terms of a function of the sorted counts (e.g., median) across samples. This approach, inspired from the microarray literature [41], is implemented in the Bioconductor R package EDASeq (`http://www.bioconductor.org/packages/EDASeq`).

SCnorm [18]. Bacher *et al.* [18] noted that a single scaling factor per sample is not enough to account for the systematic variation in the relationship between gene-specific expression measures and sequencing depth. To address this problem, they use quantile regression to estimate the dependence of gene expression measures on sequencing depth, group genes with similar dependence, and use a second quantile regression to estimate scaling factors within each group. In this way, gene expression measures are normalized differently across the range of expression (i.e., highly-expressed genes are scaled differently than lowly-expressed genes). The method is implemented in the Bioconductor R package SCnorm (`http://www.bioconductor.org/packages/SCnorm`).

Regression-based removal of factors of unwanted variation

We consider the following *generalized linear model* (GLM), which allows adjustment for known and unknown factors of "unwanted variation":

$$g(\mathrm{E}[Y|X,U,W]) = X\beta + U\gamma + W\alpha, \tag{2}$$

where Y is the $n \times J$ matrix of gene-level read counts, X is an $n \times M$ design matrix corresponding to the M covariates of interest/factors of "wanted variation" (e.g., treatment) and β its associated $M \times J$ matrix of parameters of interest, U is an $n \times H$ matrix corresponding to known factors of unwanted variation (e.g., batch, sample QC measures) and γ its associated $H \times J$ matrix of nuisance parameters, W is an $n \times K$ matrix corresponding to unknown factors of unwanted variation and α its associated $K \times J$ matrix of nuisance parameters, and g is a link function, such as the logarithm in Poisson/log-linear regression.

The $U\gamma$ and $W\alpha$ terms correspond, respectively, to supervised and unsupervised removal of unwanted variation. A fully supervised version of Equation (2), without W, reduces to a simple GLM fit. The remove unwanted variation (RUV) model of Risso *et al.* [12] arises as a special case of Equation (2), when one omits the known unwanted factors U. In many cases, the data-driven unsupervised version of Equation (2), without U, captures effects related to U; for instance, W is often associated with QC measures (Supplementary Fig. 1). However, in many cases, U should still include known batches (e.g., set of samples processed at the same time), as W could capture effects related to sample quality within batches that are important to remove in addition to the batch effects captured by U. We also find that, in practice, the computationally simpler approach of fitting a linear model to log-counts Y yields good results. Hence, *scone* relies on a linear model version of (2) with identity link function ($g(x) = x$).

As detailed in Risso *et al.* [12] and implemented in the Bioconductor R package RUVSeq, the RUV unknown unwanted factors W can be estimated by singular value decomposition (SVD) using three main approaches. Here, we only use RUVg, which estimates the factors of unwanted variation based on *negative control genes*, assumed to have constant expression across all samples ($\beta = 0$).

In the current implementation of scone, the only covariates that can be included in U are those related to known experimental batches and to sample-level QC. Because QC measures are often highly correlated (Fig. 1), scone transforms the metrics using principal component analysis prior to fitting the model. The user can choose how many components to include in the model.

Nested experimental designs

The model of Equation (2) should be applied with caution and only after a careful examination of the experimental design. In particular, a common limitation of scRNA-seq datasets is the *nesting* of unwanted technical effects within the biological effects of interest. For instance, the iPSC dataset of Tung *et al.* [31] contains samples derived from three individuals, each processed in three batches. Regressing read counts on the batch covariate in U without adjusting for the covariate of interest in X would then remove the effect of interest (effect of individual donor). Additionally, to avoid collinearity issues between columns of U and X

due to nesting, one could either specify suitable contrasts or use a mixed effect model where technical effects are viewed as random. The scone implementation of the model in Equation (2) automatically detects nested designs and specifies the right contrasts to avoid removing the effects of interest and to avoid identifiability issues in the estimation procedure. See Tung *et al.* [31] for an alternative method that uses random effects.

Specifically, to account for nested batch effects, we consider the following model for each gene (cf. ANOVA). For illustration purposes and ease of notation, we do not include the gene subscript and the additional known or unknown covariates allowed in Equation (2), although the latter can be included in the scone package implementation. Let Y_{ijk} denote the expression measure of sample k in batch j of condition i, with $i = 1, \dots, a$ conditions, $j = 1, \dots, b_i$ batches for condition i, and $k = 1, \dots, n_{j(i)}$ samples for batch j of condition i. We fit the following model gene by gene

$$\mathrm{E}[Y_{ijk}|X, U] = \alpha + \beta_i + \gamma_{j(i)}, \tag{3}$$

where α is an intercept, β_i a biological effect of interest, and $\gamma_{j(i)}$ a nested batch effect. Given the $a + 1$ constraints

$$\sum_{i=1}^{a} \beta_i = 0, \tag{4}$$
$$\sum_{j=1}^{b_i} \gamma_{j(i)} = 0, \qquad i = 1, \dots, a,$$

the model is identifiable and can be fit using standard R functions such as `lm`. The batch-corrected gene expression measures are then given by the residuals $Y_{ijk} - \hat{\gamma}_{j(i)}$. When additional factors of unwanted variation are included in the model, their effects are similarly subtracted from the original matrix to produce the normalized matrix.

Note that although *scone* is able to handle the special case of nested designs, no adjustment method is able to remove batch effects while preserving biological effects of interest if the experimental design is completely confounded. For instance, if only one batch had been processed per individual in the iPSC dataset of Tung *et al.* [31], it would have been impossible to determine if the differences between samples were due to batch effects or biology [6].

Performance assessment and selection of normalization procedures

Normalization performance metrics

Different normalization procedures can lead to vastly different distributions of gene expression measures and hence results for downstream analyses such as clustering. In particular, in the context of bulk RNA-seq, we found that the choice of normalization procedure had a greater impact on differential expression results than the choice of DE test statistic [39]. A natural and essential question is therefore whether normalization is beneficial and, if so, which method is most appropriate for a given dataset. In order to address this question, we have developed eight normalization performance metrics that relate to three aspects of the distribution of gene expression measures.

Clustering of samples according to factors of wanted and unwanted variation. The following three metrics evaluate normalization procedures based on how well the samples can be grouped according to factors of wanted and unwanted variation: Clustering by wanted factors is desirable, while clustering by unwanted factors is undesirable.

As clustering quality measures, we use silhouette widths [42]. For any clustering of n samples, the *silhouette width* of sample i is defined as

$$sil(i) \equiv \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \in [-1, 1], \tag{5}$$

where $a(i)$ denotes the average distance (by default, Euclidean distance over first three PCs of expression measures) between the ith sample and all other samples in the cluster to which i was assigned and $b(i)$

denotes the minimum average distance between the ith sample and samples in other clusters. Intuitively, the larger the silhouette widths, the better the clustering. Thus, the *average silhouette width* across all n samples provides an overall quality measure for a given clustering.

- *BIO_SIL*: Group the n samples according to the value of a categorical covariate of interest (e.g., known cell type, genotype) and compute the average silhouette width for the resulting clustering.
- *BATCH_SIL*: Group the n samples according to the value of a nuisance categorical covariate (e.g., batch) and compute the average silhouette width for the resulting clustering.
- *PAM_SIL*: Cluster the n samples using *partitioning around medoids* (PAM) for a range of user-supplied numbers of clusters and compute the maximum average silhouette width for these clusterings. *scone* provides an option for stratified scoring, computing the PAM_SIL metric in all distinct strata defined jointly by biological and batch classification. This option is useful when prior biological classifications are poor proxies for single-cell states, i.e., when additional heterogeneity is expected. We used this option for all datasets but the 10x Genomics PBMC dataset for which the biological classification was data-derived.

Large values of BIO_SIL and PAM_SIL and low values of BATCH_SIL are desirable.

Association of expression measures with factors of wanted and unwanted variation. The next three metrics concern the association of log-count principal components (default 3) with "evaluation" principal components of wanted or unwanted variation.

- *EXP_QC_COR*: The weighted coefficient of determination $\bar{R}^2$ (details below) for the regression of log-count principal components on all principal components of user-supplied QC measures (using `prcomp` with `scale=TRUE`; QC measures described in Tables 1 or 2).
- *EXP_UV_COR*: The weighted coefficient of determination $\bar{R}^2$ for the regression of log-count principal components on factors of unwanted variation UW (default 3) derived from negative control genes (preferably different from those used in RUV). The submatrix of log-transformed (using `log1p`) unnormalized counts for negative control genes is row-centered and scaled (i.e., for each row/gene, expression measures are transformed to have mean zero and variance one across columns/cells) and factors of unwanted variation are defined as the right-singular vectors as computed by the `svds` function from the rARPACK package.
- *EXP_WV_COR*: The weighted coefficient of determination $\bar{R}^2$ for the regression of log-count principal components on factors of wanted variation WV (default 3) derived from positive control genes. The WV factors are computed in the same way as the UV factors above, but with positive instead of negative control genes.

Large values of EXP_WV_COR and low values of EXP_QC_COR and EXP_UV_COR are desirable.

The weighted coefficients of determination are computed as follows. For each type of evaluation criterion (i.e., QC, UV, or WV), regress each expression PC on all supplied evaluation PCs. Let SST_k, SSR_k, and SSE_k denote, respectively, the total sum of squares, the regression sum of squares, and the residual sum of squares for the regression for the kth expression PC. The *coefficient of determination* is defined as usual as

$$R_k^2 \equiv \frac{SSR_k}{SST_k} = 1 - \frac{SSE_k}{SST_k}$$

and our weighted average coefficient of determination as

$$\bar{R}^2 \equiv \frac{\sum_k SST_k R_k^2}{\sum_k SST_k} = \frac{\sum_k SSR_k}{\sum_k SST_k} = 1 - \frac{\sum_k SSE_k}{\sum_k SST_k}. \tag{6}$$

Between-sample distribution of expression measures. When comparing distributions of expression measures between samples, gene-level *relative log-expression* (RLE) measures, defined as log-ratios of read counts to median read counts across samples, are more informative than log-counts [43]:

$$RLE_{ij} \equiv \log \frac{Y_{ij}}{\text{Median}_i\, Y_{ij}}, \tag{7}$$

for gene j in cell i. For similar distributions, the RLE should be centered around zero and have have similar spread across samples.

- *RLE_MED*: Mean squared median relative log-expression.
- *RLE_IQR*: Variance of inter-quartile range (IQR) of RLE.

Low values of RLE_MED and RLE_IQR are desirable.

Combining performance metrics to rank normalization approaches. In the *scone* framework, the expression measures are normalized according to a user-specified set of procedures (including no normalization) and the eight metrics above are computed for each normalized dataset.

The performance assessment results can be visualized using biplots [33] and the normalization procedures ranked based on a function of the performance metrics. In particular, we define the *aggregate score* by resigning the metrics so that large values correspond to good performance, ranking procedures for each metric, and averaging the ranks across metrics.

Note that a careful, global interpretation of the metrics is recommended, as some metrics tend to favor certain methods over others, e.g., EXP_UV_COR naturally favors RUVg, especially when the same set of negative control genes are used for normalization and evaluation.

Evaluation of *scone* rankings

To independently evaluate the *scone* normalization performance metrics, we compared the *scone* rankings to rankings based on external data available for three of our datasets.

Plate-sorted SMART-Seq2 Th17 dataset [25]. To evaluate *scone* performance on the Th17 dataset, we analyzed independent bulk microarray data from Lee *et al.* [27], available on GEO with accession GSE39820. We computed sets of positive (DE) and negative (non DE) control genes by comparing cytokine activation IL-1beta, IL-6, IL-23 and cytokine activation TGF-beta1, IL-6 microarray samples, using the Bioconductor R package limma [28]. Specifically, we identified as positive controls the 1,000 genes with the smallest p-values and as negative controls the 1,000 genes with the largest p-values on the microarray data, excluding any *scone* controls from these sets. For each normalization procedure, we then applied limma with voom weights [44] to the scRNA-seq data to perform differential expression analysis between Th17-positive pathogenic cells and unsorted non-pathogenic cells. We used the resulting p-values to infer the positive and negative controls derived from the microarray data and produce a *receiver operating characteristic* (ROC) curve. The ranking of normalization procedures by *scone* was then compared to the ranking by the ROC *area under the curve* (AUC).

Fluidigm C1 cortex dataset [29]. We similarly processed independent bulk microarray data from Miller *et al.* [45], available from the BrainSpan atlas (`http://brainspan.org/static/download.html`). We computed sets of positive (DE) and negative (non DE) control genes by comparing CP+SP and SZ+VZ tissues, using the limma package [28] as for the Th17 dataset above. For each normalization procedure, we then applied limma with voom weights [44] to the scRNA-seq data to perform differential expression analysis between the GW16 and GW21+3 conditions. The ranking of normalization procedures by *scone* was compared to the ranking by the ROC AUC, as detailed for the Th17 dataset.

10x Genomics PBMC dataset [30]. We processed independent bulk microarray data from Nakaya *et al.* [46], available on GEO with accession GSE29618. We computed sets of positive (DE) and negative (non DE) control genes by comparing baseline B-cell and baseline dendritic cell (DC) microarray samples, using the limma package [28] as for the Th17 dataset above. For each normalization procedure, we then applied limma with voom weights [44] to the scRNA-seq data to perform differential expression analysis between the B-cell and dendritic cell clusters, as defined by Seurat [47]. The ranking of normalization procedures by *scone* was compared to the ranking by the ROC AUC, as detailed for the Th17 dataset.

Bioconductor R software package scone

Our *scone* framework is implemented in the open-source R package scone, freely available through the Bioconductor Project [36] at `https://bioconductor.org/packages/scone`.

The package implements methods for sample and gene filtering, provides wrappers for commonly-used normalization procedures, and allows the user to define custom normalization methods in the form of simple R functions.

The main function in the package, called scone, can be used to run and compare different normalization strategies. Through its integration with the BiocParallel [37] and rhdf5 [38] packages, scone can take full advantage of parallel processing and on-disk data representation, to avoid storing all the normalization results in memory and instead store the results in HDF5 files. See the package vignette at `https://bioconductor.org/packages/scone` for additional details.

Although here we used PCA to evaluate the performance of normalization methods (in particular, regarding the metrics EXP_QC_COR, EXP_UV_COR, and EXP_WV_COR), the scone package allows the choice of other dimensionality reduction techniques. scone also has an optional imputation step that can be turned on to evaluate the performance of imputation methods and their effect on normalization.

In addition to the ability to retrieve normalized expression measures, scone allows the user to retrieve the related design matrix. This is useful for downstream differential expression analysis, where rather than removing the unwanted variation and using the residuals as expression measures, one may want to include the factors of unwanted variation as additional covariates in a regression model.

R code has been provided with this manuscript to reproduce analyses presented here.

Datasets

Plate-sorted SMART-Seq2 Th17 dataset [25]. Cells were harvested from two C57BL/6J and three IL-17A $-$ GFP$^+$ mice. Unsorted non-pathogenic cells were collected from the first two mice and both IL-17A-sorted pathogenic and non-pathogenic cells were collected from the three remaining mice. Cells were sorted in plate and the SMART-Seq2 protocol was used for single-cell RNA extraction and sequencing. Following sample filtering, 337 cells were retained from 4 donor mice – one mouse was filtered out due to the small number of acceptable cells. We consider the problem of normalizing 7,590 gene features over these 337 cells. For *scone*, we provided negative and positive control genes based on Supplementary Table S6 from Yosef *et al.* [48].

Fluidigm C1 cortex dataset [29]. 65 cells from the developing cortex were assayed using the Fluidigm C1 microfluidics system; each cell was sequenced at both high and low depths. We focus on the high-coverage data. The data are available as part of the Bioconductor R package scRNAseq (`https://bioconductor.org/packages/scRNAseq`). No sample filtering was applied to this dataset and 4,706 genes were retained following gene filtering. For *scone*, we provided default negative control genes from the "housekeeping" list, and for positive controls: genes annotated in MSigDB related to neurogenesis (JEPSEN_SMRT_TARGETS and GO_NEURAL_PRECURSOR_CELL_PROLIFERATION; `http://software.broadinstitute.org/gsea/msigdb/cards/`).

Fluidigm C1 iPSC dataset [31]. Three batches of 96 libraries from each of three YRI iPSC lines were sequenced using the Fluidigm C1 microfluidics system. The full dataset, including UMI counts, read counts, and quality metrics, was obtained from `https://github.com/jdblischak/singleCellSeq`. Library-level QC measures included: (i) Proportion of reads aligning to ERCC spike-ins (matching the pattern "ÊRCC");

(ii) Number of unique molecules; (iii) Well number as reported in online metadata ("wel"); (iv) Concentration as reported in online metadata ("concentration"); (v) Number of detected molecule classes (genes with more than zero UMI). Following gene and sample filtering we retained 6,818 genes and 731 libraries; retained samples had greater than 24,546 reads, 80% of common genes detected, and FNR AUC below 0.65. For *scone*, we provided default negative control genes from the "housekeeping" list, as well as positive control genes from the "cellcycle_genes" default list. ERCC genes were used as negative controls for RUVg normalization. Patient was used as a proxy for biological condition, while batch was defined as an individual C1 run.

10x Genomics PBMC dataset [30]. We considered scRNA-seq data from two batches of peripheral blood mononuclear cells (PBMCs) from a healthy donor (4k PBMCs and 8k PBMCs). The data were dowloaded from the 10x Genomics website (`https://www.10xgenomics.com/single-cell/`) using the cellrangerRkit R package (v. 1.1.0). After filtering, 12,039 cells and 10,310 genes were retained. For *scone*, we provided default negative control genes from the "housekeeping" list, and for positive controls: the top 513 most common genes annotated in the MSigDB C7 immunological signature collection (`http://software.broadinstitute.org/gsea/msigdb/collections.jsp`). Seurat-derived clusters [47] were used as a biological condition, while batch was defined as an individual 10x run.

For the Th17 and cortex datasets, SRA files were downloaded from the Sequence Read Archive and transformed to FASTQ format using the SRA Toolkit. Reads were aligned with TopHat (v. 2.0.11) [49] to the appropriate reference genome (GRCh38 for human samples, GRCm38 for mouse). RefSeq mouse gene annotation (GCF_000001635.23_GRCm38.p3) was downloaded from NCBI on Dec. 28, 2014. RefSeq human gene annotation (GCF_000001405.28) was downloaded from NCBI on Jun. 22, 2015. `featureCounts` (v. 1.4.6-p3) [50] was used to compute gene-level read counts.

Acknowledgments

We would like to thank Russell Fletcher and Diya Das for their constant feedback on the *scone* method and R package. We are grateful to Bioconductor core developers for the thorough code review and to all the developers that contributed issue reports, bug fixes, and pull requests on GitHub. DR, JN, EP, and SD are supported by the National Institutes of Health BRAIN Initiative (grant U01 MH105979, PI: John Ngai, and U19 MH114830, PI: Hongkui Zeng). MC is supported by the National Institute of Dental and Craniofacial Research (NIDCR) (F31DE025176).

References

1. Finak, G. *et al.* MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. *Genome Biology* **16,** 1 (2015).
2. Risso, D., Perraudeau, F., Gribkova, S., Dudoit, S. & Vert, J.-P. ZINB-WaVE: A general and flexible method for signal extraction from single-cell RNA-seq data. *bioRxiv,* 125112 (2017).
3. Bacher, R. & Kendziorski, C. Design and computational analysis of single-cell RNA-sequencing experiments. *Genome Biology* **17,** 1 (2016).
4. Vallejos, C. A., Risso, D., Scialdone, A., Dudoit, S. & Marioni, J. C. Challenges in the normalization of single-cell RNA sequencing datasets. *Nature Methods* **14,** 565–571 (2017).
5. Kharchenko, P. V., Silberstein, L. & Scadden, D. T. Bayesian approach to single-cell differential expression analysis. *Nature Methods* **11,** 740–2 (2014).
6. Hicks, S. C., Townes, F. W., Teng, M. & Irizarry, R. A. Missing data and technical variability in single-cell RNA-sequencing experiments. *Biostatistics,* kxx053 (2017).
7. Ilicic, T. *et al.* Classification of low quality cells from single-cell RNA-seq data. *Genome Biology* **17,** 1 (2016).
8. Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L. & Wold, B. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature Methods* **5,** 621–628 (2008).
9. Robinson, M. D. & Oshlack, A. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biology* **11,** R25 (2010).
10. Anders, S. & Huber, W. Differential expression analysis for sequence count data. *Genome Biology* **11,** R106 (2010).
11. Gagnon-Bartsch, J. A. & Speed, T. P. Using control genes to correct for unwanted variation in microarray data. *Biostatistics* **13,** 539–552 (2012).
12. Risso, D., Ngai, J., Speed, T. P. & Dudoit, S. Normalization of RNA-seq data using factor analysis of control genes or samples. *Nature Biotechnology* **32,** 896–902 (2014).
13. Leek, J. T. & Storey, J. D. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS genetics* **3,** e161 (2007).
14. Leek, J. T. svaseq: removing batch effects and other unwanted noise from sequencing data. *Nucleic Acids Research* **42,** e161–e161 (2014).
15. Lun, A. T., Bach, K. & Marioni, J. C. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biology* **17,** 1 (2016).
16. Qiu, X. *et al.* Single-cell mRNA quantification and differential analysis with Census. *Nature Methods* **14,** 309–315 (2017).
17. Buettner, F. *et al.* Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. *Nature Biotechnology* **33,** 155–160 (2015).
18. Bacher, R. *et al.* SCnorm: robust normalization of single-cell RNA-seq data. *Nature Methods* **14,** 584–586 (2017).
19. Ding, B. *et al.* Normalization and noise reduction for single cell RNA-seq experiments. *Bioinformatics,* btv122 (2015).
20. Vallejos, C. A., Marioni, J. C. & Richardson, S. BASiCS: Bayesian analysis of single-cell sequencing data. *PLoS Computational Biology* **11,** e1004333 (2015).
21. Van Dijk, D. *et al.* MAGIC: A diffusion-based imputation method reveals gene-gene interactions in single-cell RNA-sequencing data. *bioRxiv,* 111591 (2017).
22. Li, W. V. & Li, J. J. scImpute: accurate and robust imputation for single cell RNA-seq data. *bioRxiv,* 141598 (2017).
23. Pierson, E. & Yau, C. ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome Biology* **16,** 241 (2015).

24. Townes, F. W., Hicks, S. C., Aryee, M. J. & Irizarry, R. A. Varying-Censoring Aware Matrix Factorization for Single Cell RNA-Sequencing. *bioRxiv,* 166736 (2017).

25. Gaublomme, J. T. *et al.* Single-Cell Genomics Unveils Critical Regulators of Th17 Cell Pathogenicity. *Cell* **163,** 1400–1412 (2015).

26. Wagner, A., Regev, A. & Yosef, N. Revealing the vectors of cellular identity with single-cell genomics. *Nature Biotechnology* **8 Nov** (2016).

27. Lee, Y. *et al.* Induction and molecular signature of pathogenic TH17 cells. *Nature Immunology* **13,** 991–999 (2012).

28. Ritchie, M. E. *et al.* limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Research* **43,** e47 (2015).

29. Pollen, A. A. *et al.* Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. *Nature Biotechnology* **32,** 1053–8 (2014).

30. Zheng, G. X. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nature Communications* **8,** 14049 (2017).

31. Tung, P.-Y. *et al.* Batch effects and the effective design of single-cell gene expression studies. *Scientific Reports* **7,** 39921 (2017).

32. Chang, W., Cheng, J., Allaire, J., Xie, Y. & McPherson, J. *shiny: Web Application Framework for R* R package version 1.0.5.9000 ().

33. Gabriel, K. R. The biplot graphic display of matrices with application to principal component analysis. *Biometrika* **58,** 453–467 (1971).

34. Fletcher, R. B. *et al.* Deconstructing Olfactory Stem Cell Trajectories at Single-Cell Resolution. *Cell Stem Cell* **20,** 817–830 (2017).

35. Afik, S. *et al.* Targeted reconstruction of T cell receptor sequence from single cell RNA-seq links CDR3 length to T cell differentiation state. *Nucleic Acids Research* (2017).

36. Gentleman, R. C., Carey, V. J., Bates, D. M., *et al.* Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology* **5,** R80 (2004).

37. Morgan, M., Obenchain, V., Lang, M. & Thompson, R. *BiocParallel: Bioconductor facilities for parallel evaluation* R package version 1.11.11 (2017).

38. Fischer, B., Pau, G. & Smith, M. *rhdf5: HDF5 interface to R* R package version 2.21.6 (2017).

39. Bullard, J. H., Purdom, E. A., Hansen, K. D. & Dudoit, S. Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. *BMC Bioinformatics* **11,** Article 94 (2010).

40. Risso, D., Schwartz, K., Sherlock, G. & Dudoit, S. GC-Content Normalization for RNA-Seq Data. *BMC Bioinformatics* **12,** Article 480 (2011).

41. Irizarry, R. A. *et al.* Exploration, Normalization, and Summaries of High Density Oligonucleotide Array Probe Level Data. *Biostatistics* **4,** 249–264 (2003).

42. Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* **20,** 53–65 (1987).

43. Gandolfo, L. C. & Speed, T. P. RLE Plots: Visualising Unwanted Variation in High Dimensional Data. *arXiv preprint arXiv:1704.03590* (2017).

44. Law, C. W., Chen, Y., Shi, W. & Smyth, G. K. Voom: precision weights unlock linear model analysis tools for RNA-seq read counts. *Genome Biology* **15,** R29 (2014).

45. Miller, J. A. *et al.* Transcriptional landscape of the prenatal human brain. *Nature* **508,** 199–206 (2014).

46. Nakaya, H. I. *et al.* Systems biology of vaccination for seasonal influenza in humans. *Nature Immunology* **12,** 786–795 (2011).

47. Satija, R., Butler, A. & Hoffman, P. *Seurat: Tools for Single Cell Genomics* R package version 2.1.0 (2017).

48. Yosef, N. *et al.* Dynamic regulatory network controlling TH17 cell differentiation. *Nature* **25 April** (2013).

49. Trapnell, C., Pachter, L. & Salzberg, S. L. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics* **25,** 1105–1111 (2009).

50. Liao, Y., Smyth, G. K. & Shi, W. FeatureCounts: An efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics* **30,** 923–930 (2014).

Tables and Figures

Table 1: *Sample-level quality control (QC) measures (non-10x Genomics).*

Name	Description	Source
NREADS	Total number of sequenced reads	Picard
NALIGNED	Total number of aligned reads	Picard
RALIGN	Percentage of mapped reads	Picard
TOTAL_DUP	Number of duplicate reads	FastQC
PRIMER	Percentage of primer sequence reads	FastQC
INSERT_SZ	Average insert size	Picard
INSERT_SZ_STD	Insert size variance	Picard
COMPLEXITY	Sequence Complexity	Picard
NDUPR	Percentage of unique reads	Picard
PCT_RIBOSOMAL_BASES	Percentage of ribosomal bases	Picard
PCT_CODING_BASES	Percentage of coding bases	Picard
PCT_UTR_BASES	Percentage of UTR bases	Picard
PCT_INTRONIC_BASES	Percentage of intronic bases	Picard
PCT_INTERGENIC_BASES	Percentage of intergenic bases	Picard
PCT_MRNA_BASES	Percentage of mRNA bases	Picard
MEDIAN_CV_COVERAGE	Median coefficient of variation of coverage	Picard
MEDIAN_5PRIME_BIAS	Mean 5' coverage bias	Picard
MEDIAN_3PRIME_BIAS	Mean 3' coverage bias	Picard
MEDIAN_5PRIME_TO_3PRIME_BIAS	Mean 5' to 3' coverage bias	Picard

Table 2: *Sample-level quality control (QC) measures (10x Genomics).*

Name	Description	Source
num_umi	Number of unique UMI sequences	CellRanger
num_reads	Total number of reads (regardless of mapping)	CellRanger
mean_reads_per_umi	The average number of reads supporting each UMI	CellRanger
std_reads_per_umi	Standard deviation of the number of reads supporting each UMI	CellRanger
mapped_reads	Proportion of reads which confidently mapped to a gene	CellRanger
genome_not_gene	Proportion of reads mapping to the genome, but not to a gene	CellRanger
unmapped_reads	Proportion of reads which did not align	CellRanger
umi_corrected	Proportion of reads whose UMI sequence was corrected by CellRanger	CellRanger
barcode_corrected	Proportion of reads whose barcode sequence was corrected by CellRanger	CellRanger

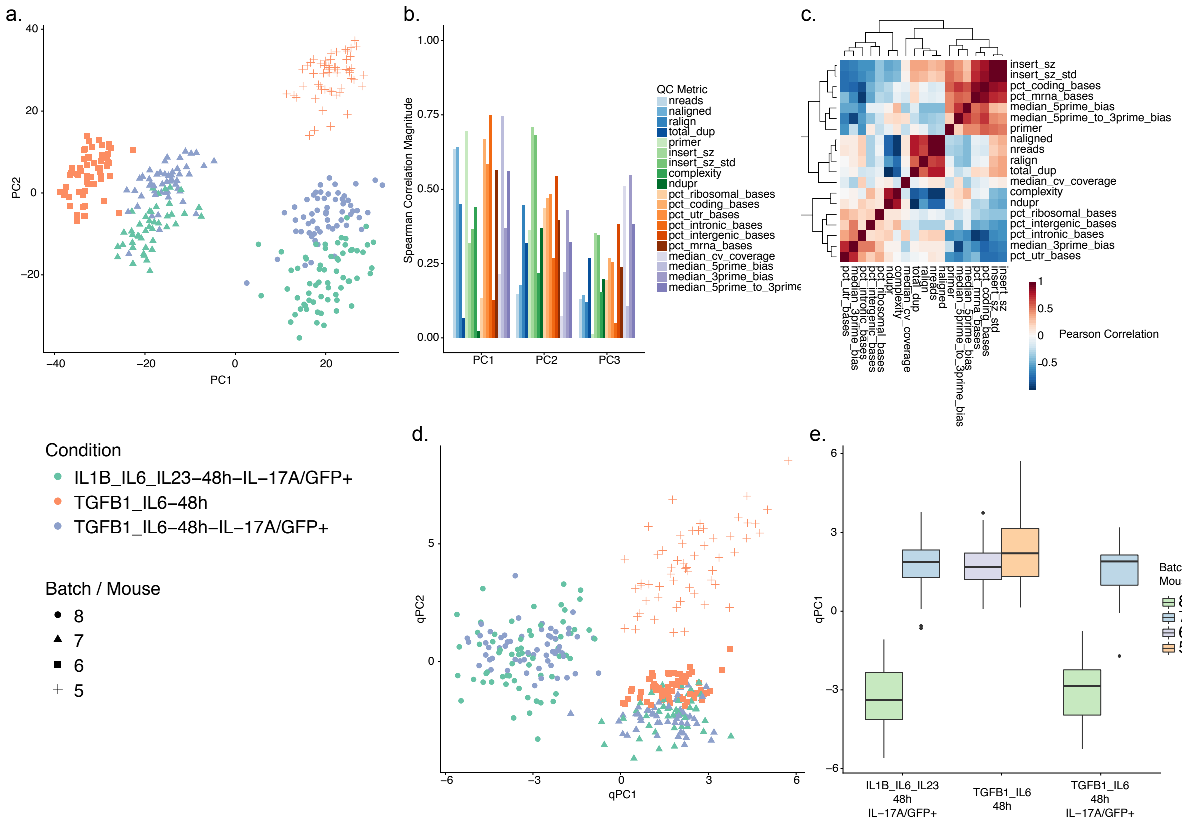


Figure 1: *Exploratory data analysis of the Th17 dataset [25].* **(a)** Principal component analysis (PCA) of the log-transformed, TC-normalized gene-level read counts. Cells are color-coded by biological condition and shape represents mouse/batch. Cells cluster by biological condition and batch, suggesting the presence of unwanted covariation. **(b)** Absolute Spearman correlation coefficient between the first three principal components of the expression measures (PCs; as computed in (a)) and a set of QC measures (Table 1). **(c)** Heatmap of the matrix of pairwise Pearson correlation coefficients between QC measures. **(d)** PCA of the QC measures. Cells are color-coded by biological condition and shape represents mouse/batch. Cells cluster by batch, suggesting that the QC measures are able to capture aspects of the batch effects. **(e)** Boxplot of the first principal component of the QC measures, stratified by biological condition and batch. This plot emphasizes the partially confounded design of the study: While two of the three conditions were both assayed in the same two batches (IL1B_IL6_IL23-48h-IL-17A/GFP+ and TGFB1_IL6-48h-IL-17A/GFP+ in batches 7 and 8), the third condition (TGFB1_IL6-48h) was assayed in two distinct batches (batches 5 and 6) (note that there are different numbers of cells in each of the batches).

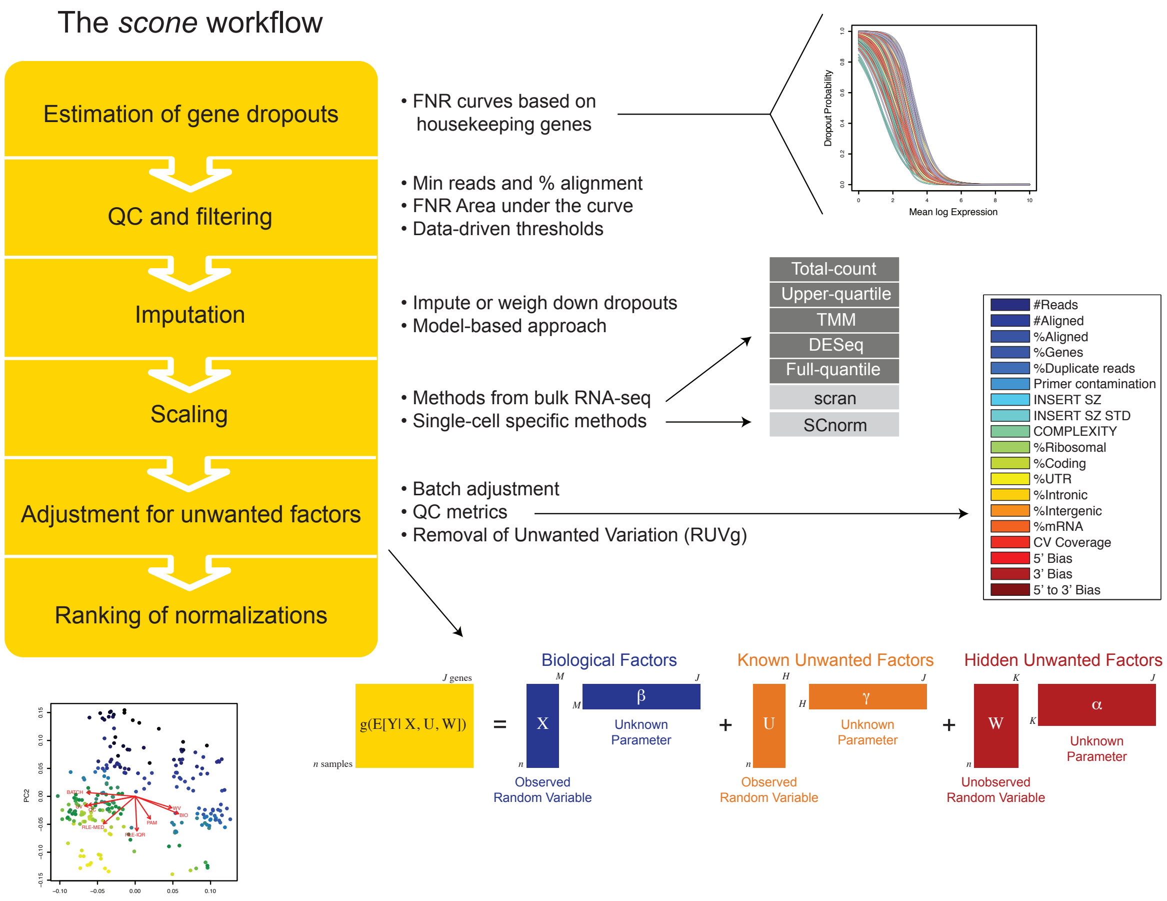


Figure 2: *Schematic view of the* scone *workflow.* The yellow box summarizes the six steps of the *scone* workflow. False negative rate (FNR) curves based on housekeeping genes are computed for each sample and used to infer dropouts. Using the FNR curves and the QC measures of Tables 1 or 2, low-quality samples and lowly-expressed genes are filtered out. Following an optional imputation step, the data are normalized via a combination of scaling normalization and regression-based normalization (using a linear model that includes both known and unknown covariates). Finally, the different normalization strategies are ranked according to eight performance metrics (see Methods).

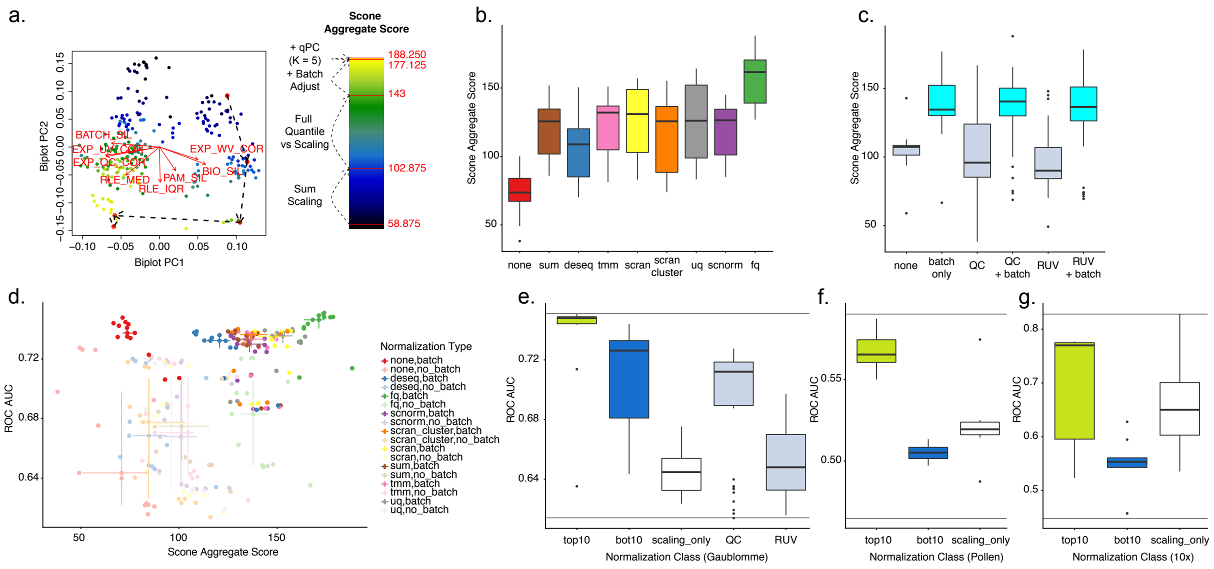


Figure 3: scone *results on three scRNA-seq datasets.* **(a-d)** Results on Th17 dataset [25]. **(a)** Biplot [33] of the *scone* performance metrics. Each point corresponds to a normalization workflow in the space of the first two principal components of the eight performance metrics, color-coded by the *scone* aggregate score (mean of eight *scone* score ranks). The red arrows correspond to the PCA loadings for each of the eight metrics. The direction and length of an arrow can be interpreted as a measure of how much of that metric contributed to the two PCs. The points further along the direction of the arrow typically have a higher value for the corresponding metric. For instance, points in the bottom-left corner of the biplot have a higher value for RLE_MED. The red circles mark the following normalization procedures (from top-right to bottom-left): No normalization, total-count (TC), full-quantile (FQ), FQ + batch adjustment, FQ + batch adjustment + 5 factors of QC. **(b)** Boxplot of *scone* aggregate score, stratified by scaling normalization method. **(c)** Boxplot of *scone* aggregate score, stratified by regression-based normalization method (batch, QC, and RUV). **(d)** ROC area under the curve (AUC) vs. *scone* aggregate score. Each point corresponds to a normalization workflow, color-coded by normalization method type. Methods in the top-right corner are deemed best both by *scone* and by our independent validation, and are all based on an initial FQ step. The horizontal and vertical lines indicate, for each method type, the IQR for each variable. **(e-g)** Boxplots of ROC AUC for the bottom 10 (bot10) and top 10 (top10) methods as ranked by *scone* and for methods with no regression-based normalization (scaling_only). The observed range of ROC AUC across all methods is delimited with horizontal black lines (note difference in AUC range across datasets). **(e)** Comparison of pathogenic and non-pathogenic cells for Th17 dataset [25]. Additional boxes (filled in gray) are included for methods with RUV or QC but no batch adjustment. **(f)** Comparison of GW16 and GW21+3 cells for cortex dataset [29]. **(g)** Comparison of B-cells and dendritic cells for PBMC dataset [30].

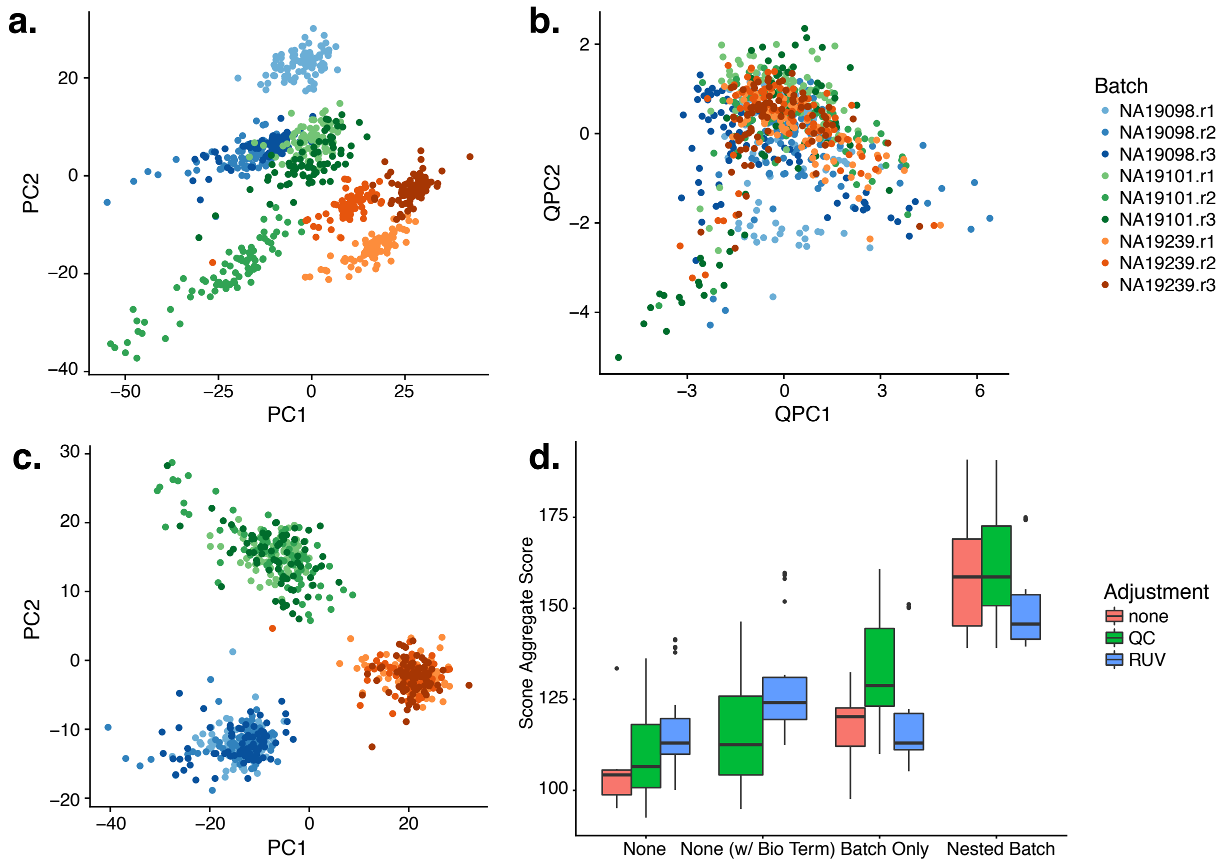
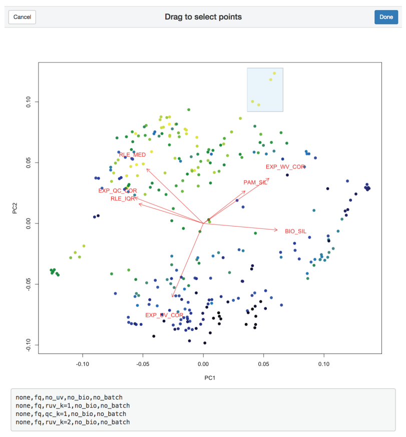


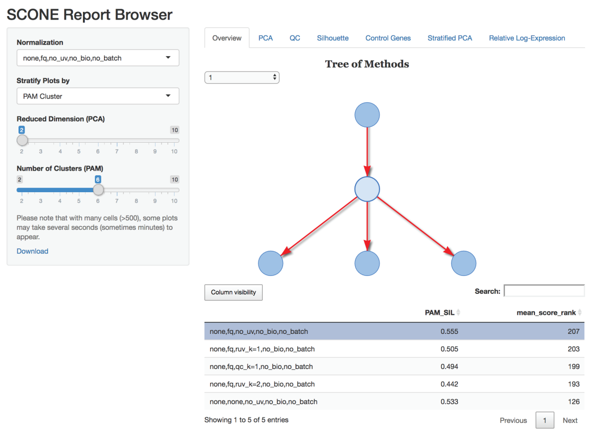
Figure 4: scone *results on the iPSC dataset [31].* **(a)** Principal component analysis (PCA) of TC-normalized measures, with points coded by donor (color) and batch (shade). The cells cluster by batch, indicating a substantial batch effect. **(b)** PCA of QC measures, with points coded by donor and batch. The QC measures do not capture batch effects, but rather intra-batch technical variation. **(c)** PCA of expression measures from full-quantile normalization followed by normalization for nested batch effects (top method identified by *scone*). As desired, cells cluster by donor, but not by batch. **(d)** Boxplot of *scone* aggregate score, stratified by regression-based normalization method. Methods including a nested design batch correction performed better than those not including such a step.

a.



```
none,fq,no_uv,no_bio,no_batch
none,fq,ruv_k=1,no_bio,no_batch
none,fq,qc_k=1,no_bio,no_batch
none,fq,ruv_k=2,no_bio,no_batch
```

b.



Normalization: none,fq,no_uv,no_bio,no_batch
Stratify Plots by: PAM Cluster
Reduced Dimension (PCA): 2
Number of Clusters (PAM): 6

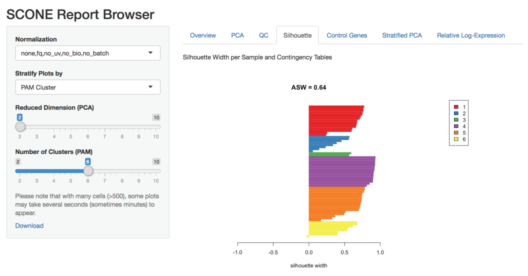
Please note that with many cells (>500), some plots may take several seconds (sometimes minutes) to appear.

Tree of Methods

	PAM_SIL	mean_score_rank
none,fq,no_uv,no_bio,no_batch	0.555	207
none,fq,ruv_k=1,no_bio,no_batch	0.505	203
none,fq,qc_k=1,no_bio,no_batch	0.494	199
none,fq,ruv_k=2,no_bio,no_batch	0.442	193
none,none,no_uv,no_bio,no_batch	0.533	126

Showing 1 to 5 of 5 entries

c.

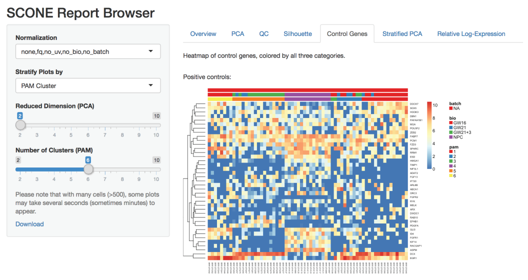


Normalization: none,fq,no_uv,no_bio,no_batch
Stratify Plots by: PAM Cluster
Reduced Dimension (PCA): 2
Number of Clusters (PAM): 6

Please note that with many cells (>500), some plots may take several seconds (sometimes minutes) to appear.

Silhouette Width per Sample and Contingency Tables

d.



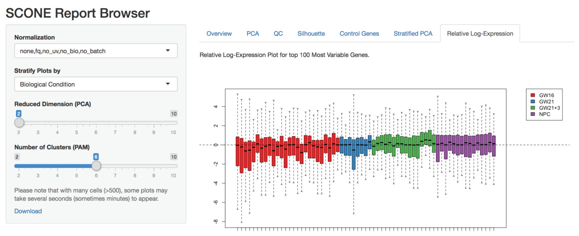
Normalization: none,fq,no_uv,no_bio,no_batch
Stratify Plots by: PAM Cluster
Reduced Dimension (PCA): 2
Number of Clusters (PAM): 6

Please note that with many cells (>500), some plots may take several seconds (sometimes minutes) to appear.

Heatmap of control genes, colored by all three categories.

Positive controls:

e.

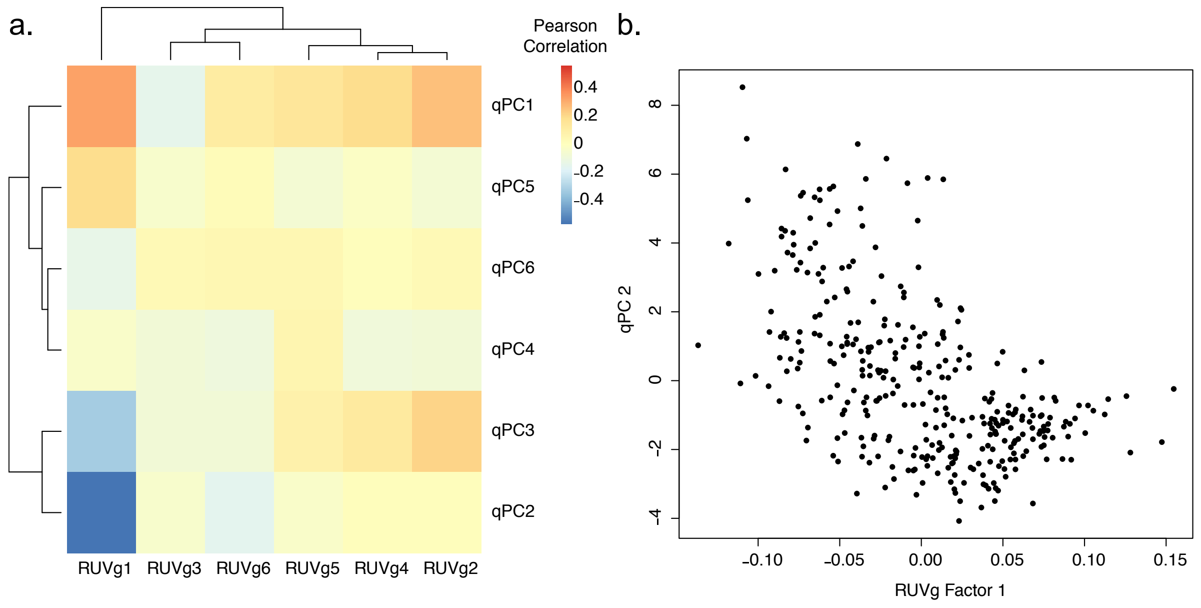


Normalization: none,fq,no_uv,no_bio,no_batch
Stratify Plots by: Biological Condition
Reduced Dimension (PCA): 2
Number of Clusters (PAM): 6

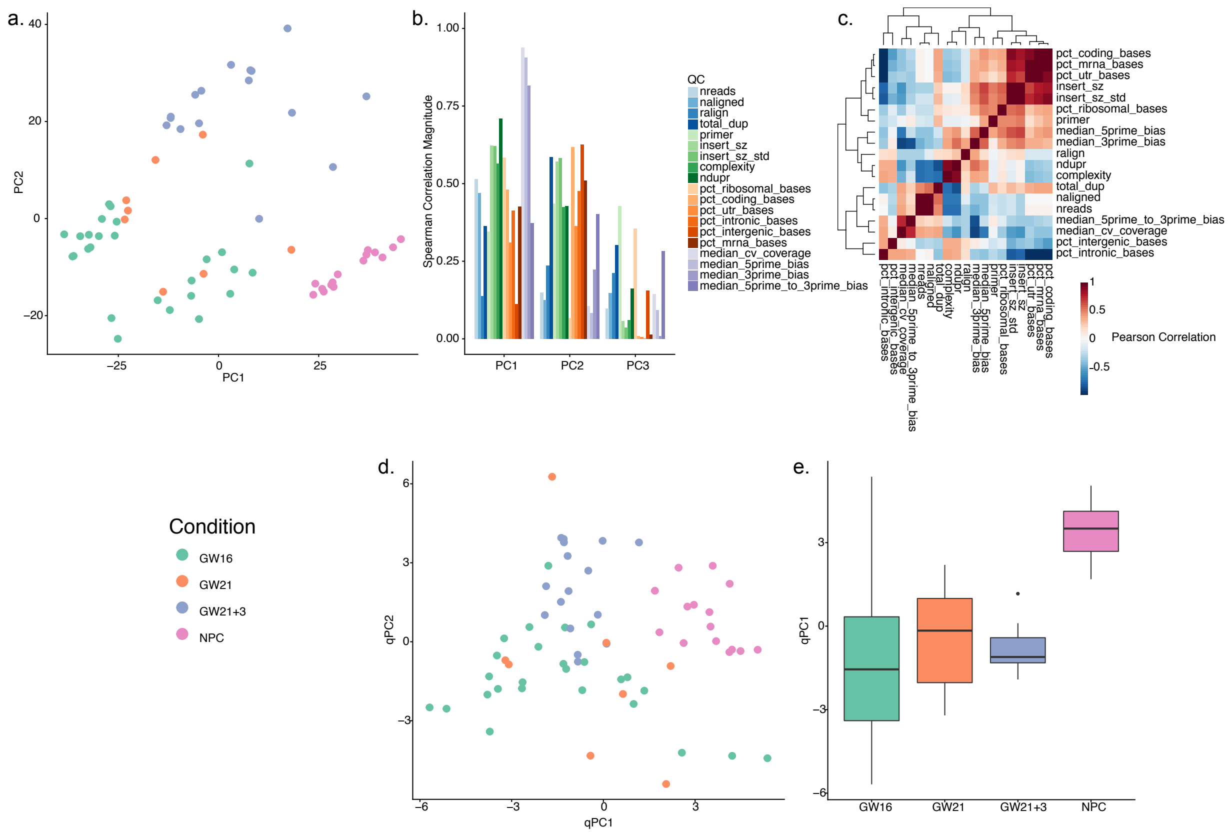
Please note that with many cells (>500), some plots may take several seconds (sometimes minutes) to appear.

Relative Log-Expression Plot for top 100 Most Variable Genes.

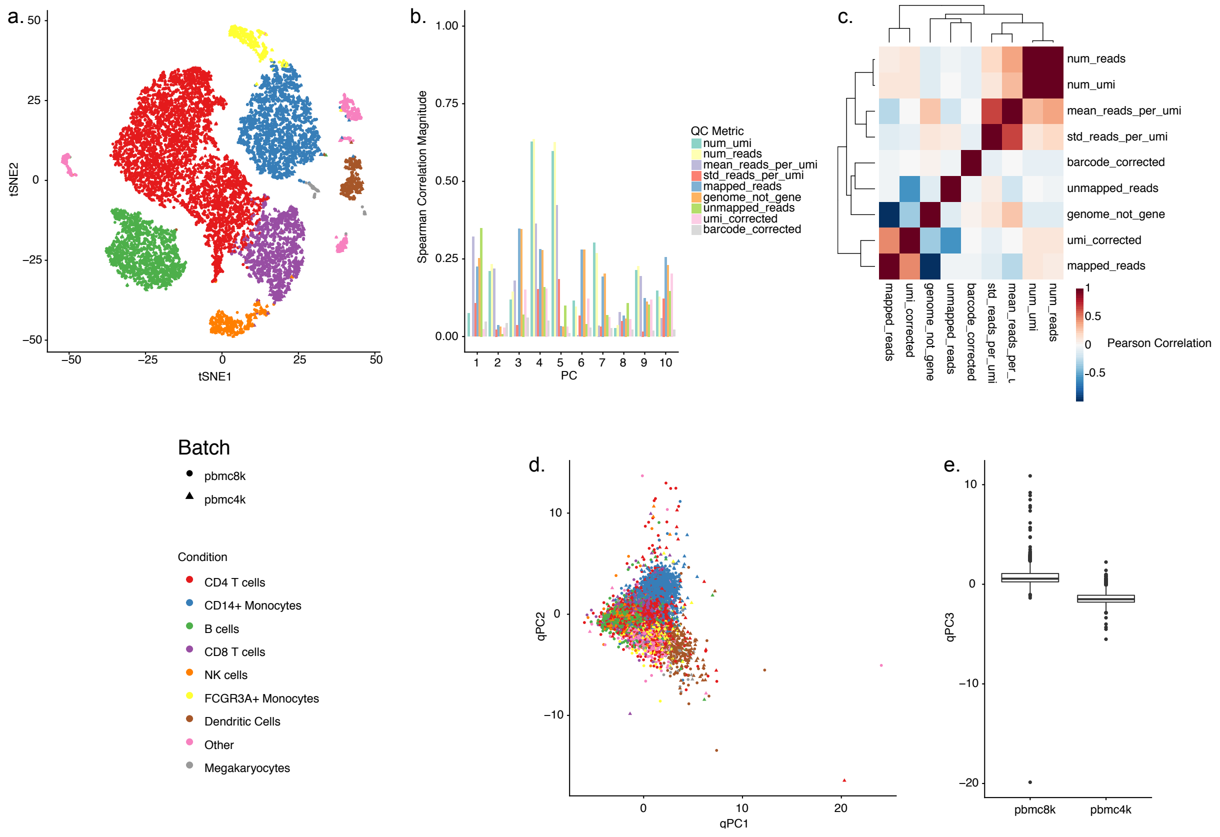
Figure 5: *scone Report Browser user interface.* **(a)** Selecting the workflows to further explore using the interactive biplot and its drag-and-drop window selection tool. This allows to select methods that perform similarly according to the eight performance metrics. **(b)** Method selection: The normalization workflows to further explore can be selected using the interactive tree representation (top-right panel), the sortable table (bottom-right panel), or the drop-down menu (side panel). **(c)** "Silhouette" tab: For the selected method, the silhouette width of each sample is computed when grouping samples by biological condition, batch, or PAM clustering. The drop-down menu in the left bar allows the user to switch between the three groupings; the slider in the left panel allows the user to select the number of clusters for PAM. **(d)** "Control Gene" tab: If the user provides positive and negative controls, the expression measures of these genes are visualized using heatmaps, which include at the top information on the samples (biological condition, batch, PAM clustering). **(e)** "Relative Log-Expression" tab: A boxplot of the relative log-expression (RLE) measures is shown for the selected normalization. The samples are color-coded by biological condition, batch, or PAM clustering (drop-down selection in the left panel). The RLE distributions of the samples should be as similar as possible to each other and centered around zero, if the majority of genes are not expected to be differentially expressed.



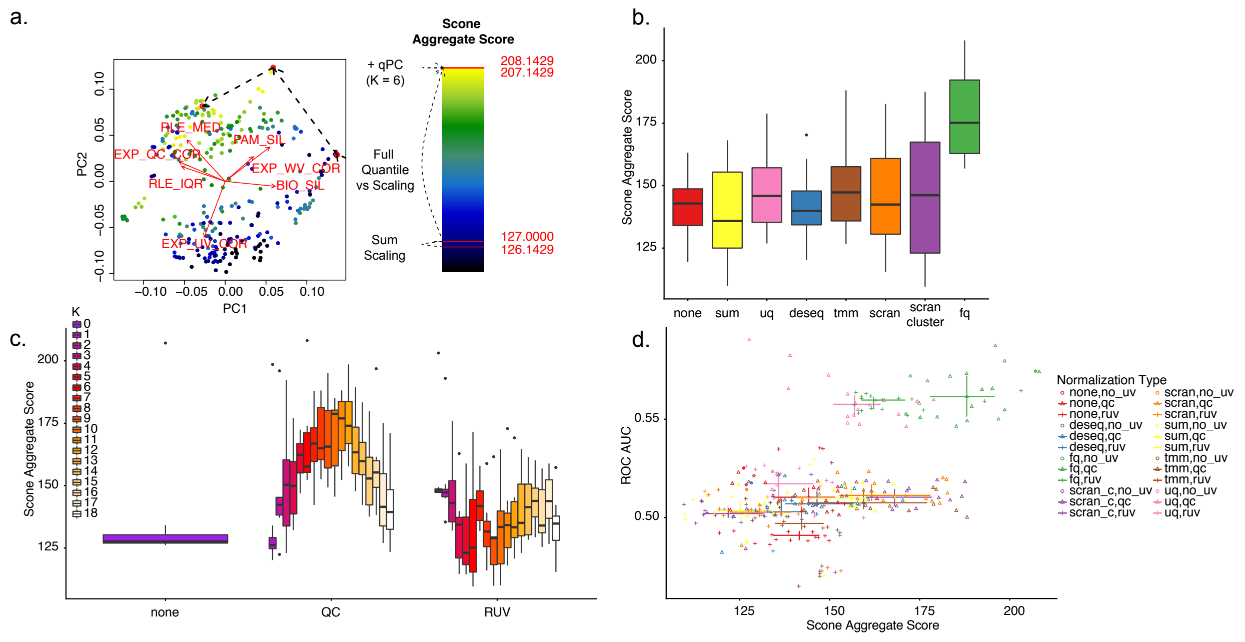
Supplementary Figure 1: *Factors of unwanted variation for the Th17 dataset [25].* **(a)** Heatmap of matrix of Pearson correlation coefficients between RUVg factors of unwanted variation and PCs of QC measures. Rows and columns are sorted by the default ordering from the R `hclust` function. **(b)** Scatterplot of a pair of anti-correlated RUVg factor and QC PC, selected based on the correlation matrix displayed in (a).



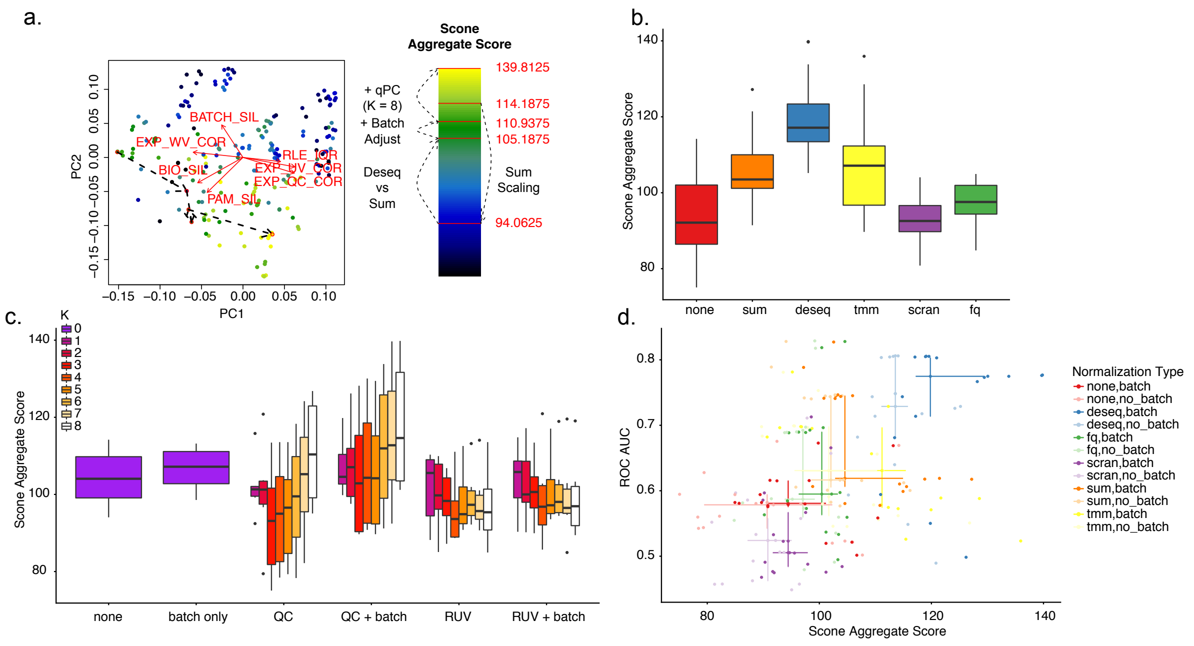
Supplementary Figure 2: *Exploratory data analysis of the cortex dataset [29].* **(a)** Principal component analysis (PCA) of the log-transformed, TC-normalized gene-level read counts. Cells are color-coded by biological condition. **(b)** Absolute Spearman correlation coefficient between the first three principal components of the expression measures (PCs; as computed in (a)) and a set of QC measures (Table 1). **(c)** Heatmap of the matrix of pairwise Pearson correlation coefficients between QC measures. **(d)** PCA of the QC measures. Cells are color-coded by biological condition. **(e)** Boxplot of the first principal component of the QC measures, stratified by biological condition. QC measures differ significantly between biological conditions (e.g., NPCs vs. GW16). However, due to the fully confounded design (one batch per biological condition), batch adjustment is not advisable here, as it would also remove the biological effects of interest.



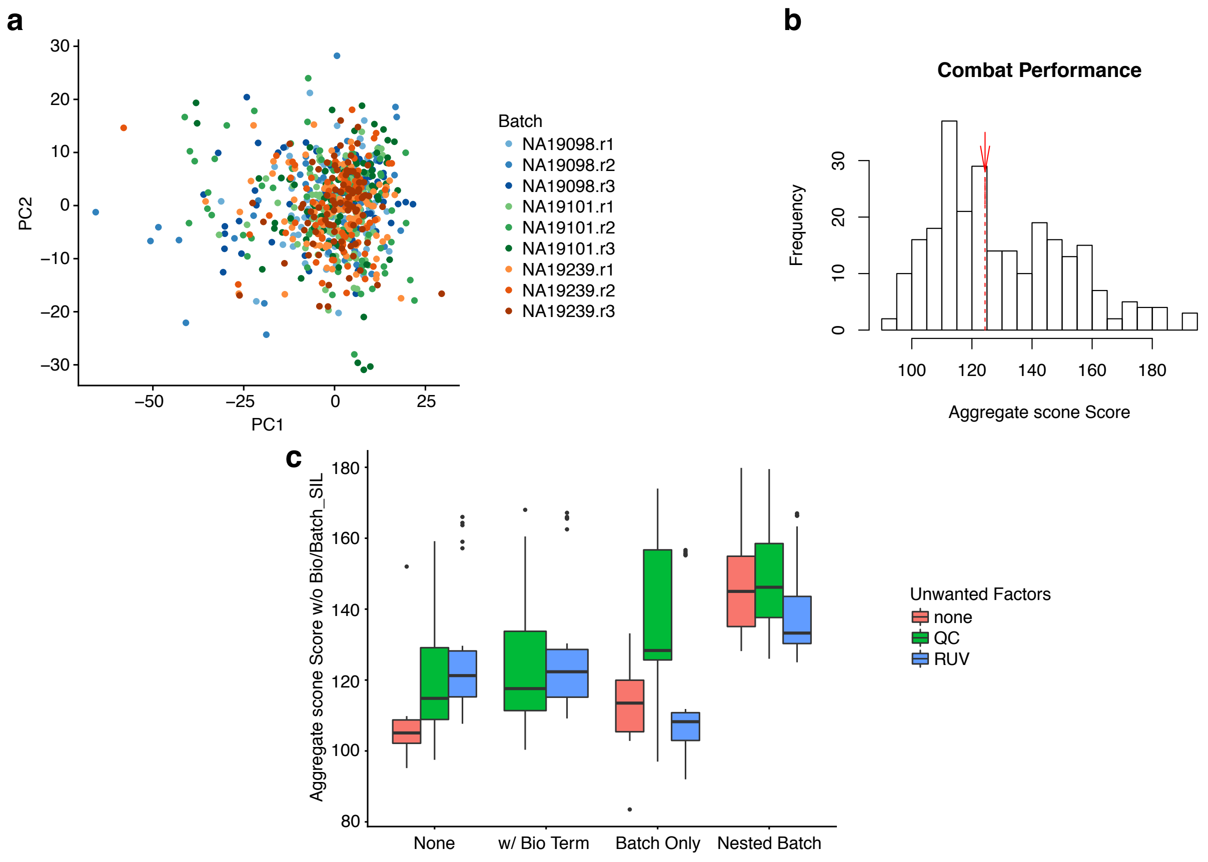
Supplementary Figure 3: *Exploratory data analysis of the PBMC dataset [30].* **(a)** t-distributed stochastic neighbor embedding (tSNE) of first 10 principal components (PCs) of the log-transformed, TC-normalized gene-level UMI counts. Cells are color-coded by merged Seurat clusters and shape represents batch. **(b)** Absolute Spearman correlation coefficient between the first ten principal components of the expression measures (PCs; as computed in (a)) and a set of QC measures (Table 2). **(c)** Heatmap of the matrix of pairwise Pearson correlation coefficients between QC measures. **(d)** PCA of the QC measures. Cells are color-coded by biological condition and shape represents batch. **(e)** Boxplot of the third principal component of the QC measures, stratified by batch (the third QC PC is the PC with the highest correlation with batch).



Supplementary Figure 4: scone *results on the cortex dataset [29].* **(a)** Biplot [33] of the *scone* performance metrics. Each point corresponds to a normalization workflow in the space of the first two principal components of the seven performance metrics, color-coded by the *scone* aggregate score (mean of seven *scone* score ranks; there is no batch covariate for this dataset). The red arrows correspond to the PCA loadings for each of the eight metrics. The direction and length of an arrow can be interpreted as a measure of how much of that metric contributed to the two PCs. The points further along the direction of the arrow typically have a higher value for the corresponding metric. For instance, points in the bottom of the biplot have a higher value for EXP_UV_COR. The red circles mark the following normalization procedures (from right to left): No normalization, total-count (TC), full-quantile (FQ), FQ + 6 factors of QC. **(b)** Boxplot of *scone* aggregate score, stratified by scaling normalization method. **(c)** Boxplot of *scone* aggregate score, stratified by regression-based normalization method (batch, QC, and RUV). **(d)** ROC area under the curve (AUC) vs. *scone* aggregate score. Each point corresponds to a normalization workflow, color-coded by normalization method type. Methods in the top-right corner are deemed best both by *scone* and by our independent validation, many of which include an initial FQ step. The horizontal and vertical lines indicate, for each method type, the IQR for each variable.



Supplementary Figure 5: scone *results on the PBMC dataset [30].* **(a)** Biplot [33] of the *scone* performance metrics. Each point corresponds to a normalization workflow in the space of the first two principal components of the eight performance metrics, color-coded by the *scone* aggregate score (mean of eight *scone* score ranks). The red arrows correspond to the PCA loadings for each of the eight metrics (RLE_MED has zero length). The direction and length of an arrow can be interpreted as a measure of how much of that metric contributed to the two PCs. The points further along the direction of the arrow typically have a higher value for the corresponding metric. For instance, points in the upper-left of the biplot have a higher value for BATCH_SIL. The red circles mark the following normalization procedures (from left to right): No normalization, total-count (TC), DESeq scaling, DESeq + batch adjustment, DESeq + batch adjustment + 8 factors of QC. **(b)** Boxplot of *scone* aggregate score, stratified by scaling normalization method. **(c)** Boxplot of *scone* aggregate score, stratified by regression-based normalization method (batch, QC, and RUV). **(d)** ROC area under the curve (AUC) vs. *scone* aggregate score. Each point corresponds to a normalization workflow, color-coded by normalization method type. Methods in the top-right corner are deemed best both by *scone* and by our independent validation. The horizontal and vertical lines indicate, for each method type, the IQR for each variable.



Supplementary Figure 6: *Batch adjustment for the iPSC dataset [31].* **(a)** PCA of ComBat-normalized data. Donor-specific effects are removed. **(b)** Histogram of *scone* aggregate scores recomputed to include ComBat (red arrow). **(c)** Boxplot of *scone* aggregate scores for various normalization strategies, excluding BIO_SIL and BATCH_SIL from the aggregate score calculation.