nature methods

SUPPLEMENTARY INFORMATION

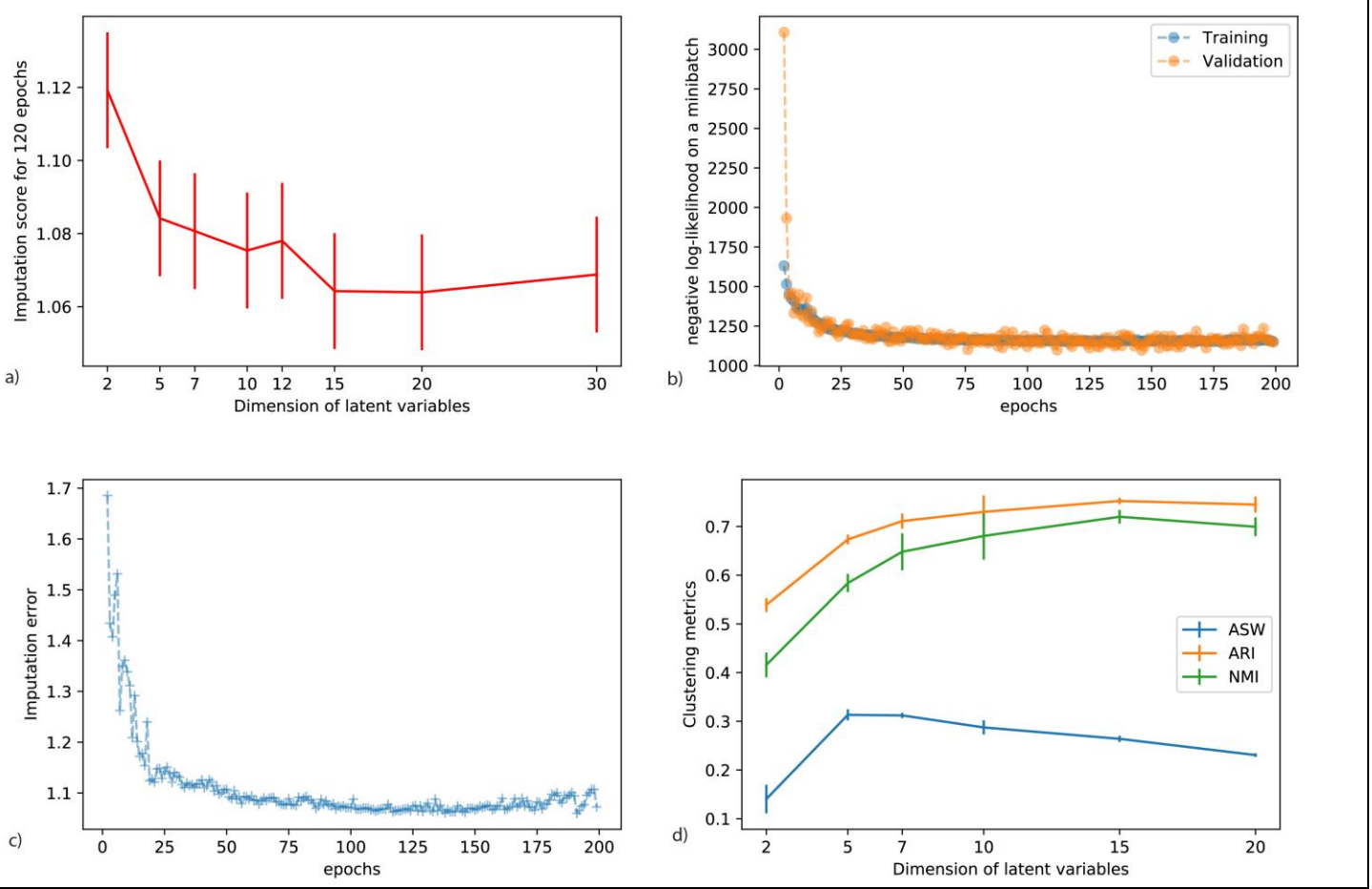
https://doi.org/10.1038/s41592-018-0229-2

In the format provided by the authors and unedited.

Deep generative modeling for single-cell transcriptomics

Romain Lopez[1], Jeffrey Regier[1], Michael B. Cole[2], Michael I. Jordan[1,3] and Nir Yosef[1,4,5]*

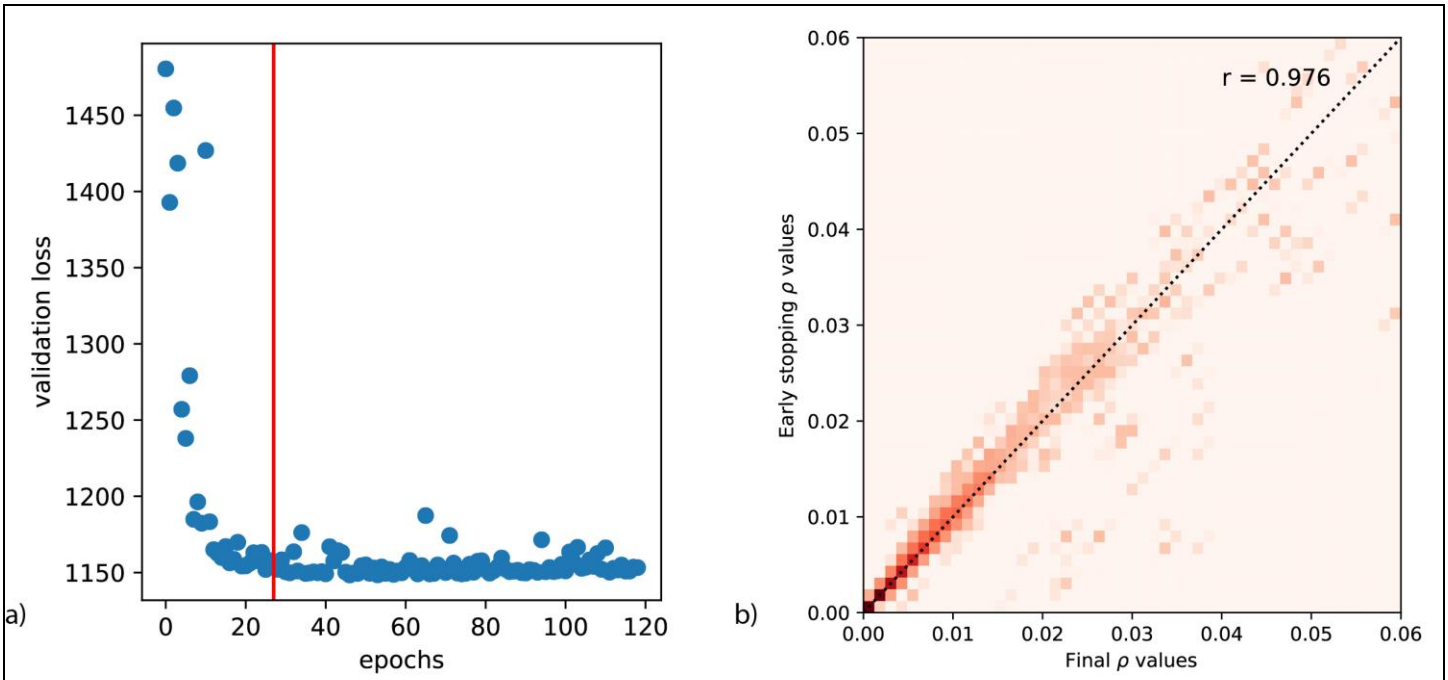
[1]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA, USA. [2]Department of Physics, University of California, Berkeley, Berkeley, CA, USA. [3]Department of Statistics, University of California, Berkeley, Berkeley, CA, USA. [4]Ragon Institute of MGH, MIT, and Harvard, Cambridge, MA, USA. [5]Chan Zuckerberg BioHub, San Francisco, CA, USA. *e-mail: niryosef@berkeley.edu



Supplementary Figure 1

Robustness analysis for scVI.

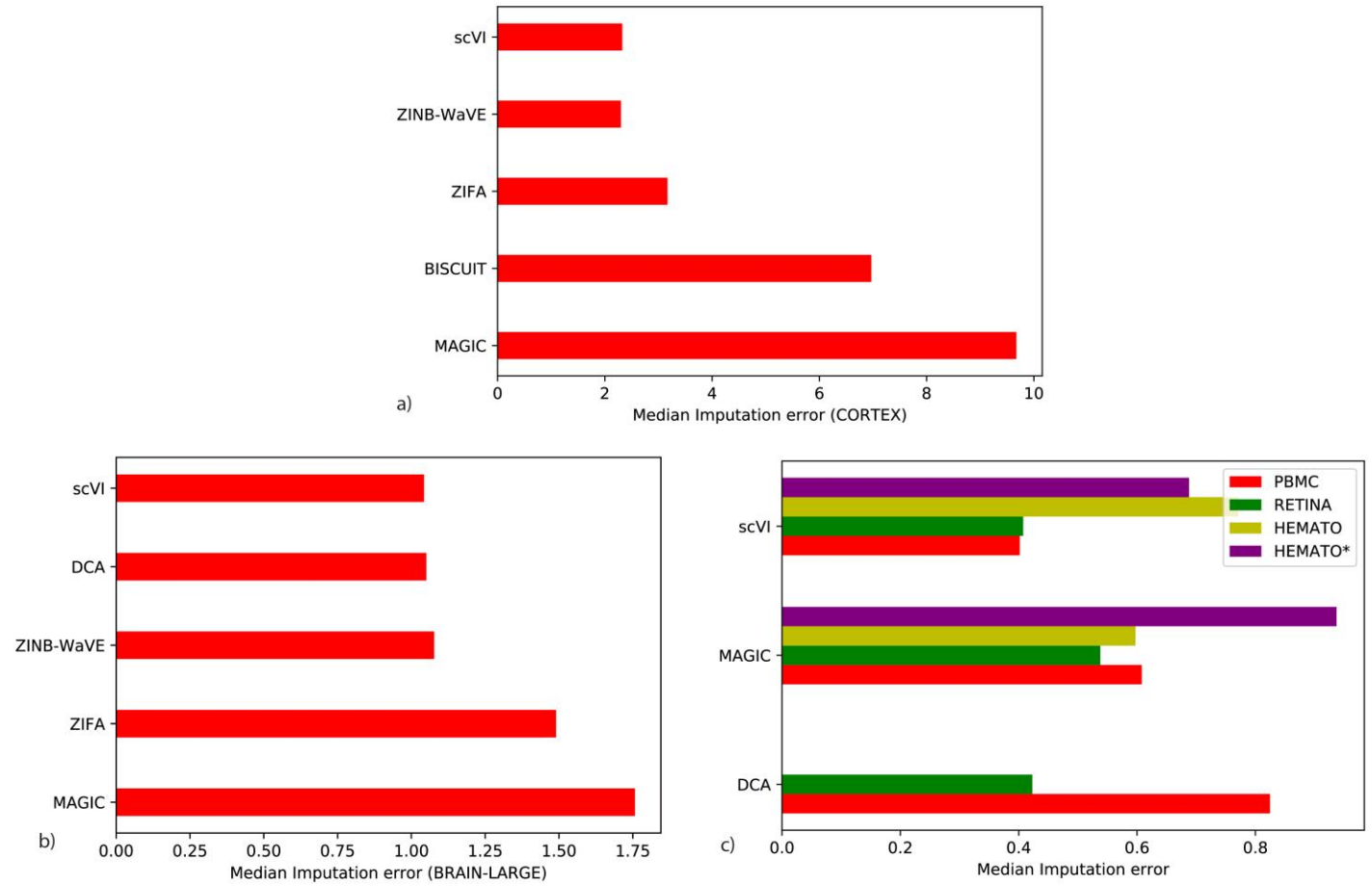
(a) Imputation score on the BRAIN-LARGE dataset across multiple random initializations (n = 5) for different dimensions of the latent space (x-axis). The center of the error bars denotes the median, and the extrema denote the s.d. (b) Visualization of scVI numerical objective function during training on the BRAIN-LARGE dataset. This shows that our model does not overfit and has a stable training procedure. (c) Imputation score as a function of the number of epochs on the BRAIN-LARGE dataset. This figure also shows stability across posterior sampling, as there is not much change in the parameters between two subsequent epochs. (d) Clustering metrics on the CORTEX dataset across multiple initializations (n = 5) and dimensions for the latent space. The center of the error bars denotes the median, and the extrema denote the s.d.



Supplementary Figure 2

Stability of the early-stopping criterion on the 1-million-cell sample of the BRAIN-LARGE dataset.

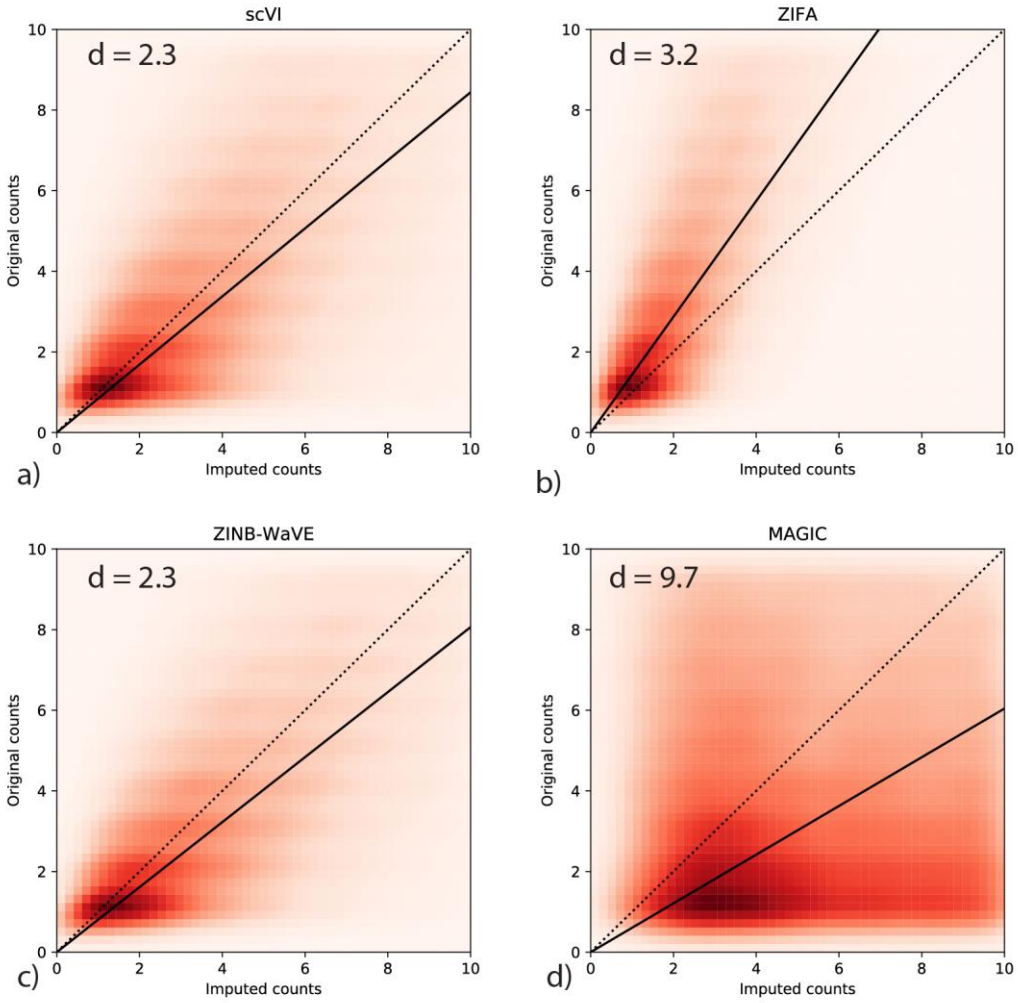
(a) Evolution of the loss function (*y*-axis) value on a validation set (*n* = 10,000 cells) with the number of epochs (*x*-axis). (b) Contrast of the expected frequency *ρ* values between the model trained with early stopping (*x*-axis) and the model trained without early stopping (*y*-axis) on a random subset of *n* = 100 cells and all 720 genes. We also report the Pearson correlation, r.



Supplementary Figure 3

Imputation results for scVI.

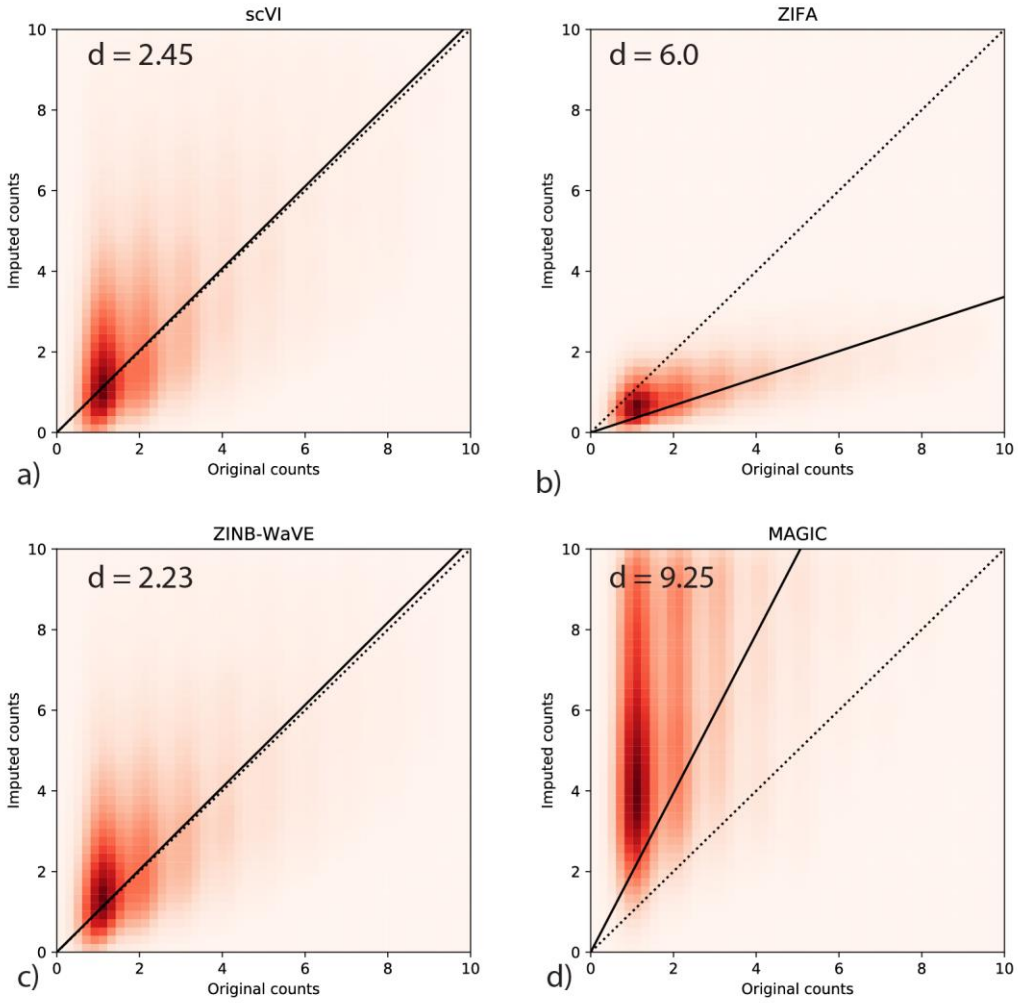
(a–c) We investigate how scVI latent space can be used to impute the data (with the uniform perturbation scheme) and report benchmarking across datasets for state-of-the-art methods.



Supplementary Figure 4

Imputation of scVI on the CORTEX dataset.

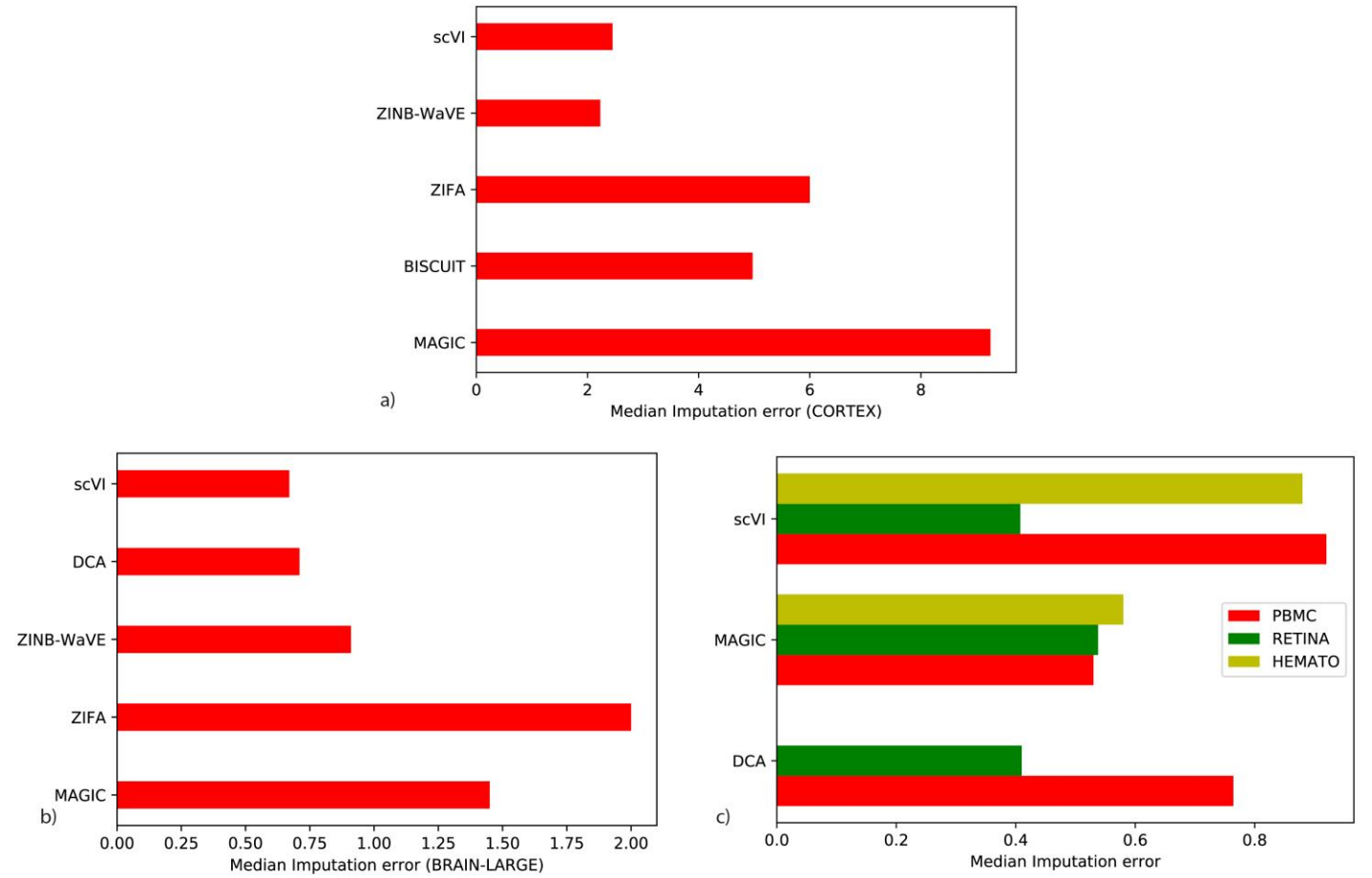
(a–d) The heat maps represent density plots of imputed values (by scVI, ZIFA, ZINB-WaVE, and MAGIC respectively) on a downsampled version versus the original (nonzero) values prior to downsampling. The reported score d is the median imputation error across all the hidden entries (lower is better; see Methods). Each density plot was computed using n = 55,932 independently perturbed entries from the original matrix.



Supplementary Figure 5

Imputation of scVI on the CORTEX dataset.

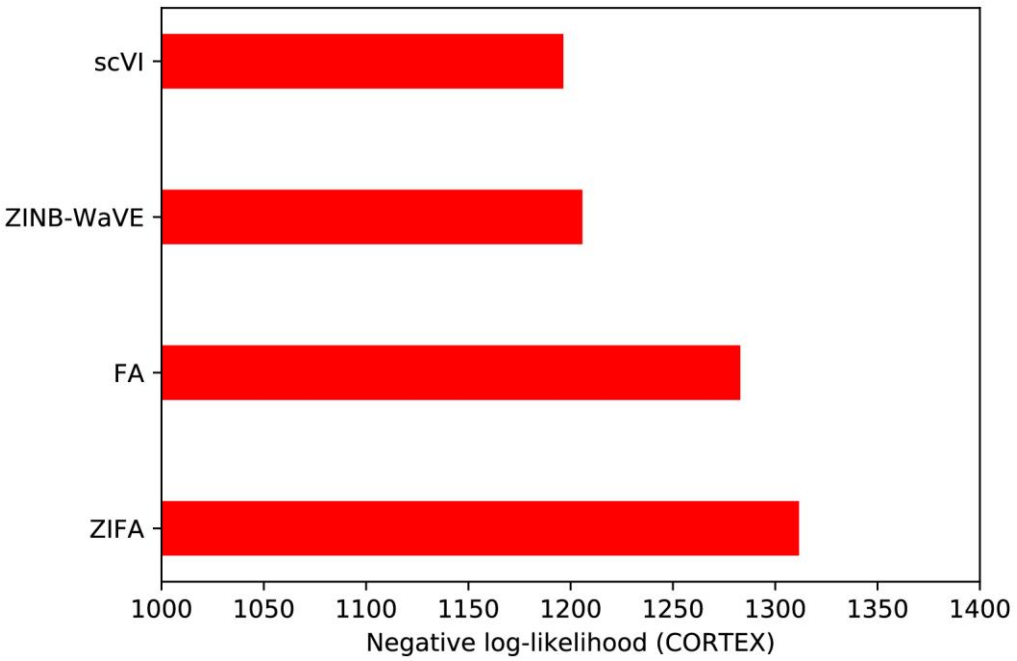
(a–d) Models were trained on a binomially corrupted dataset (see Methods). The heat maps denote density plots of imputed values scVI, ZIFA, ZINB-WaVE, and MAGIC on a downsampled version versus original values prior to downsampling. The reported score d is the median imputation error across all the hidden entries (lower is better; see Methods). Each density plot was computed using n = 55,932 independently perturbed entries from the original matrix.



Supplementary Figure 6

Imputation results for scVI.

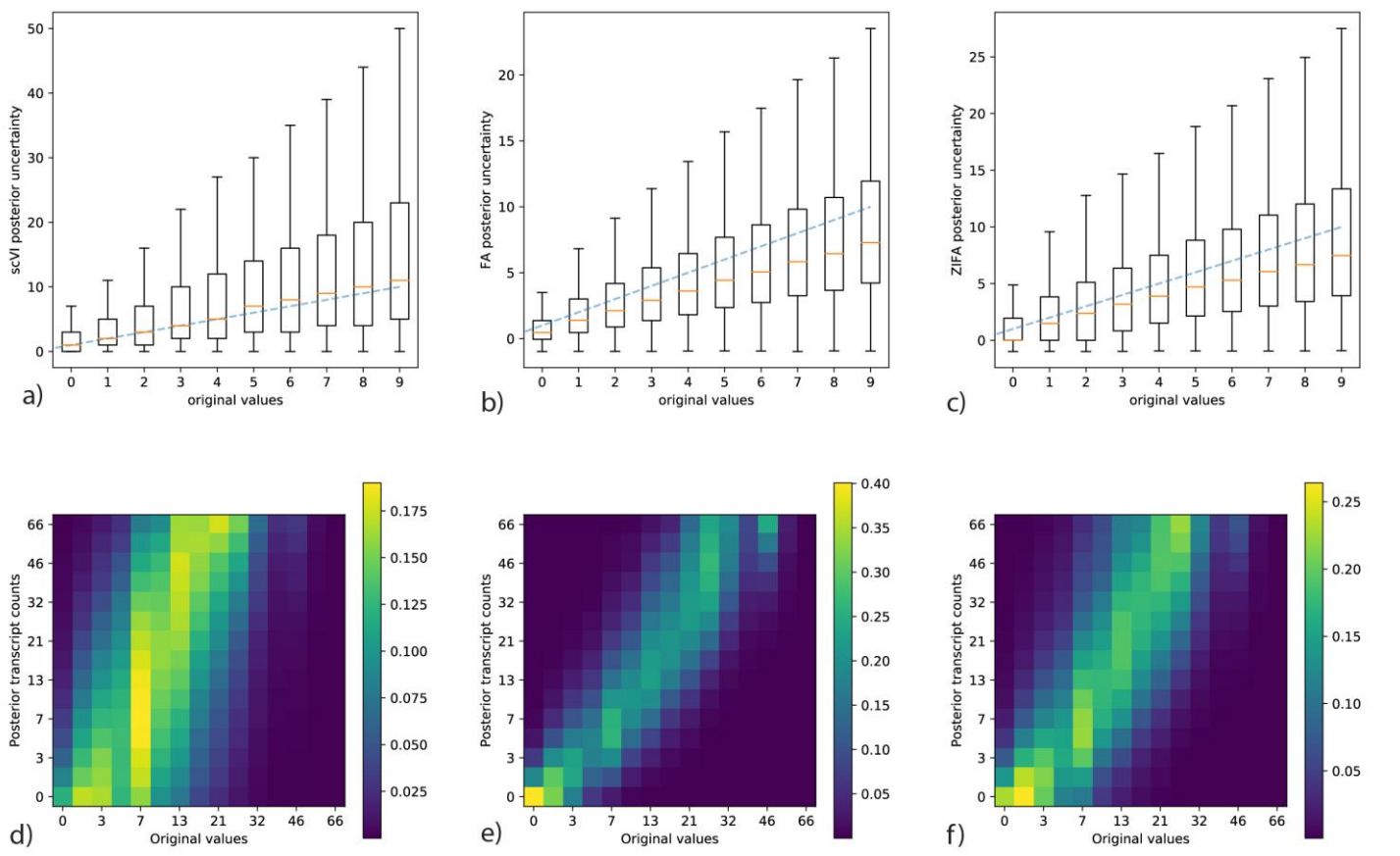
(a–c) We investigate how scVI latent space can be used to impute the data (with the binomial perturbation scheme) and report benchmarking across datasets for state-of-the-art methods.



Supplementary Figure 7

Preliminary results for scVI.

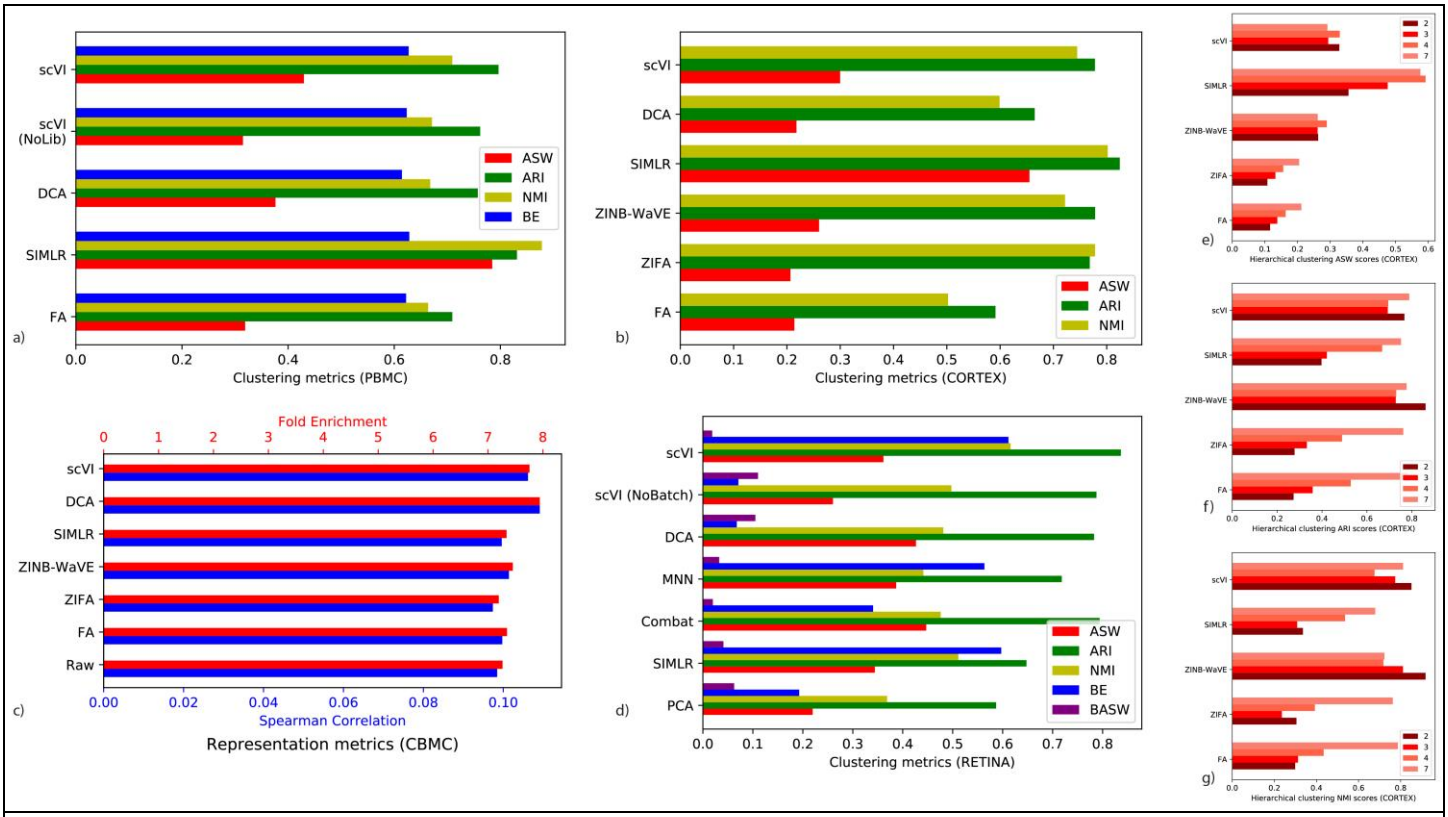
Log-likelihood results on a held-out subset of the CORTEX dataset (*n* = 751 cells).



Supplementary Figure 8

Posterior analysis of generative models on the CORTEX dataset.

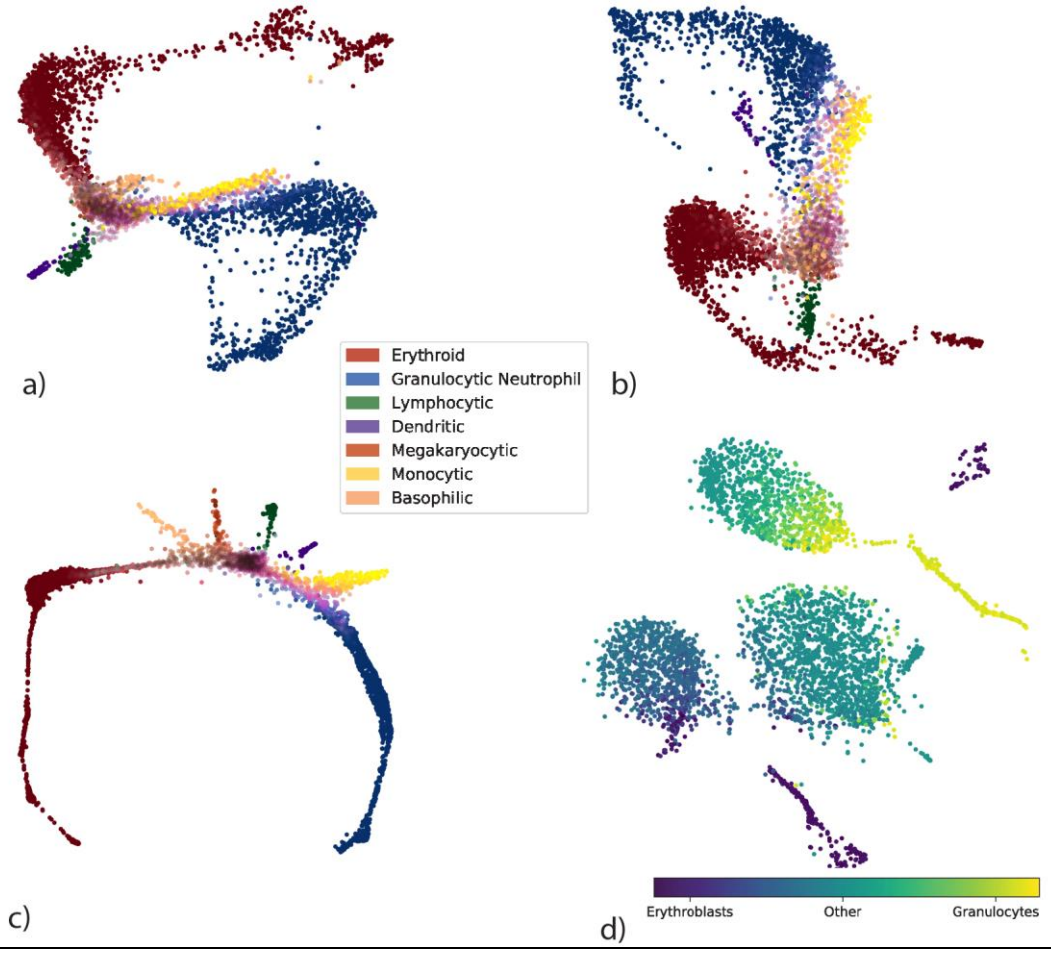
(a-c) The observed counts of n = 55,932 randomly selected entries of the data matrix (x-axis) and their posterior uncertainty (y-axis) obtained by sampling n = 500 times from the variational posterior (scVI) or the exact posterior (FA, ZIFA). The center of the box plot is the median and the hinges correspond to the interquartile range, the distance between the first and third quartiles; the whiskers represent the 5th to 95th percentiles. (d-f) The observed counts of a representative gene, *Thy1*, in the CORTEX dataset. Data are presented across all cells (n = 3,005) (x-axis) against the (n = 500) independent posterior expected counts for each cell produced by scVI, ZIFA and FA respectively (y-axis). The values in each axis have been divided into k = 20 bins and the color scale reflects the proportion of cells in each pair of bins. By definition, the uncertainty of FA is independent of the input value and tight around the observed count. ZIFA can generate zero and puts realistically more weight in this area. scVI's posterior is more complex, able to generate zero for low unique molecular identifier (UMI) values but also able to generate high UMI values when the original count observed was only of intermediate intensity.



Supplementary Figure 9

How scVI latent space can be used to cluster the data, and benchmarking across datasets for state-of-the-art methods.

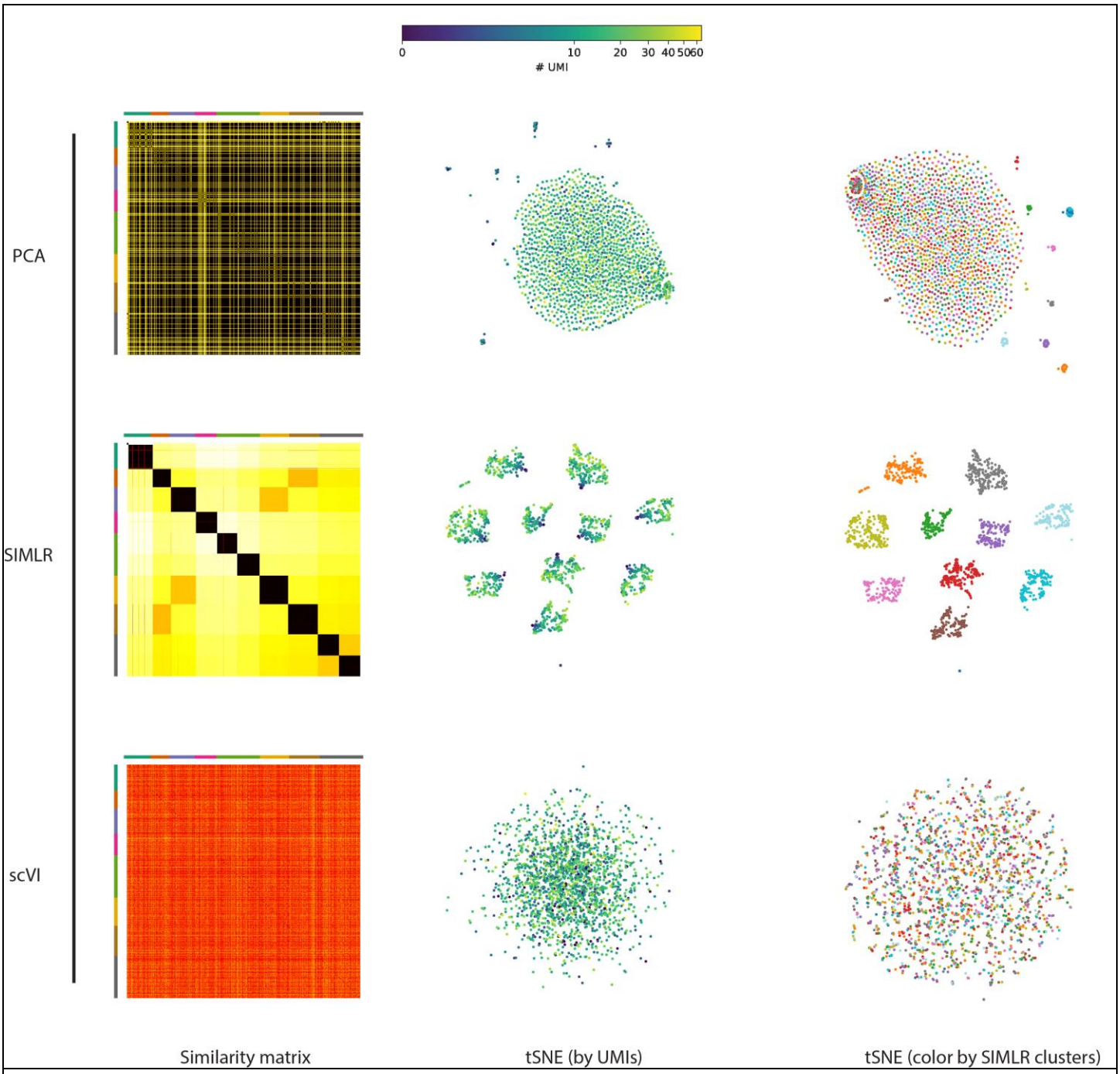
(a–d) The results for the (a) PBMC, (b) CORTEX, (c) CBMC, and (d) RETINA datasets. ASW: average silhouette width of pre-annotated subpopulations (higher is better), ARI: adjusted rand index (higher is better), NMI: normalized mutual information (higher is better), BE: batch mixing entropy (higher is better), BASW: average silhouette width of batches (lower is better). (e-g) The performance of clustering metrics for different depths of the hierarchical clustering in the CORTEX data, computed in the original publication [29]. The numbers in the legend indicate the number of clusters at the given depth.



Supplementary Figure 10

Complement to the analysis on the HEMATO dataset.

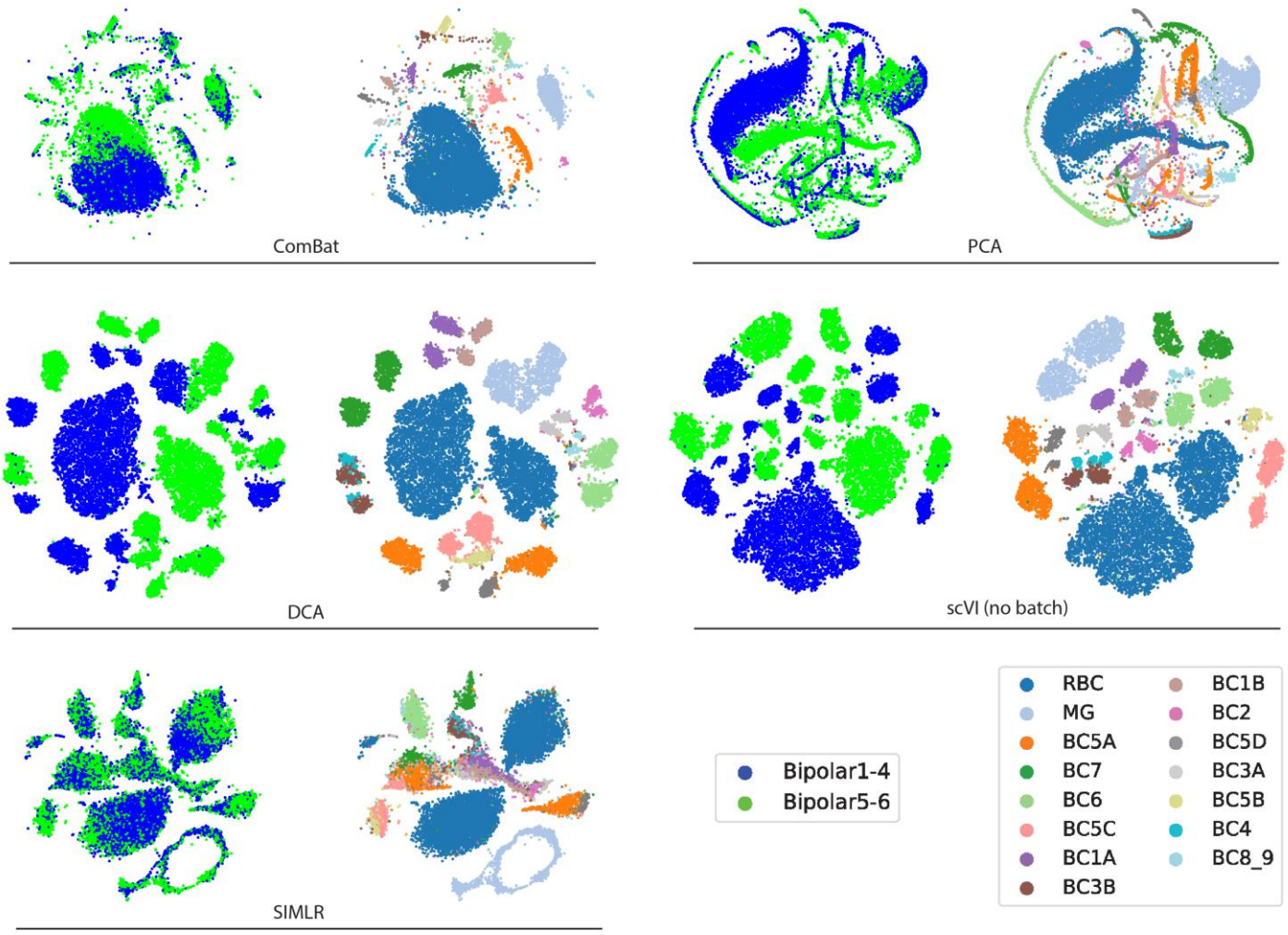
(*n* = 4,016 cells) (a–c) All scatter plots illustrate the embedding of a 5-nearest-neighbors graph of a latent space. Cell positions are computed using a force-directed layout. (a) A reduction to 60 pcs as in the original paper is shown. (b) The output of an scVI in dimension 60. As the dimension is sensibly different from other experiments, the warm-up schedule (which governs how the prior on z is enforced) was adjusted. (c) The figure from the main paper. To recover all the differentiation paths, the authors performed several operations on the *k*-nearest-neighbors graph that we did not reproduce in this analysis. We instead visualize the graph before the smoothing procedure. (d) SIMLR t-SNE on the HEMATO dataset. We prefer to visualize the SIMLR embedding on a *k*-NNs graph, as even t-SNE would lose the continuum structure of the dataset.



Supplementary Figure 11

Analysis on the ZINB random dataset.

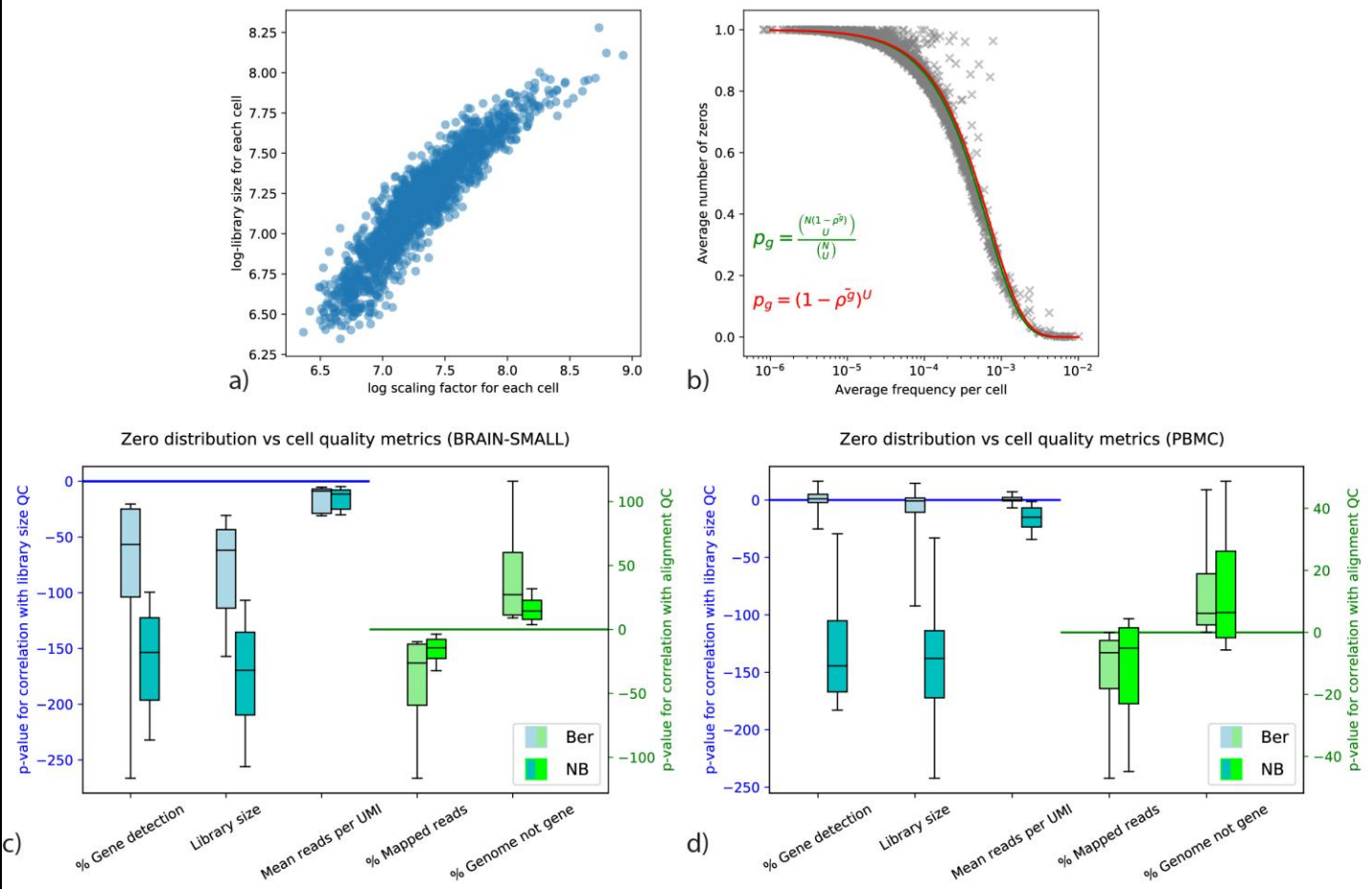
(*n* = 2,000 cells). For three algorithms (PCA, SIMLR, and scVI), we compute a latent space (we let SIMLR choose *k*; here SIMLR found 11 clusters). Then, we compute a cell–cell similarity matrix where cells are ordered by the SIMLR clusters (first column). We either apply t-SNE to the latent space (scVI, PCA) or use the two-dimensional embedding from SIMLR and color by number of UMIs (second column), or by SIMLR cluster labels (third column).



Supplementary Figure 12

Batch effect removal on the RETINA dataset.

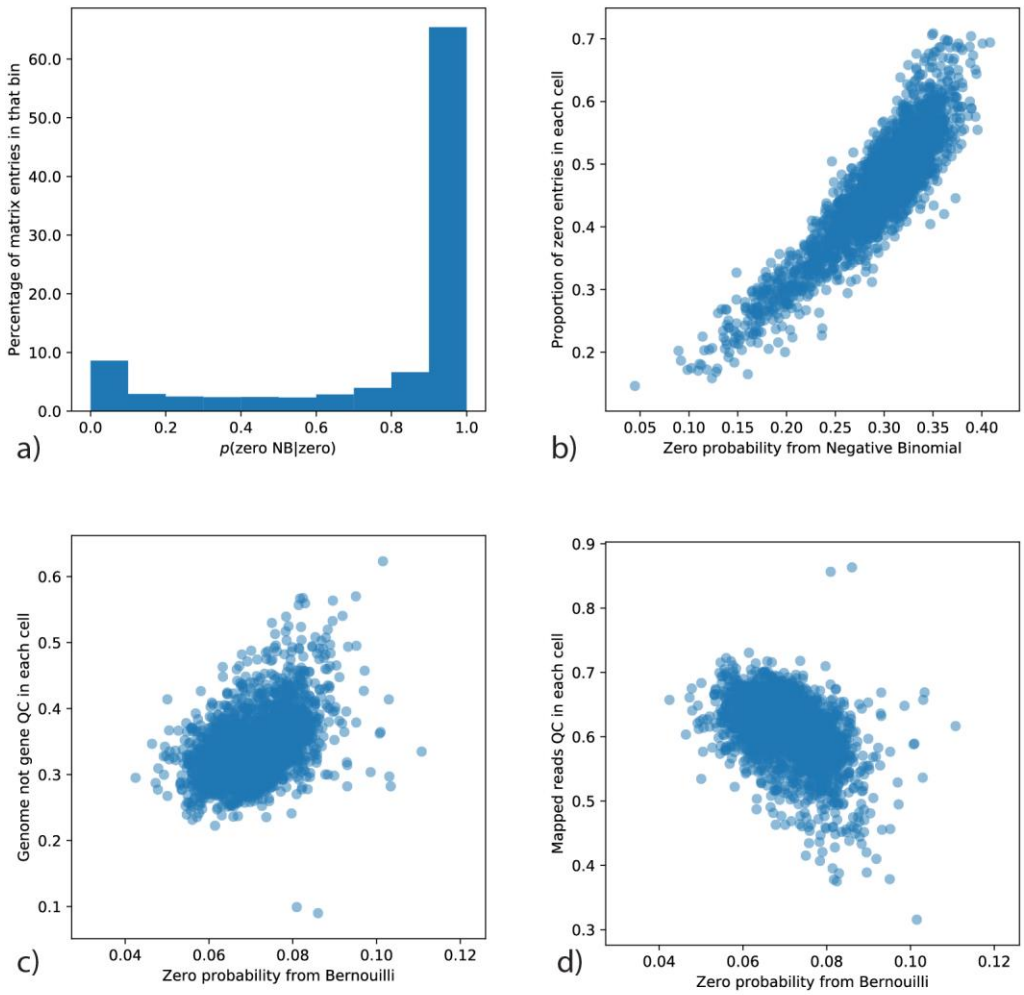
n = 27,499 cells for scVI with no batch, DCA, PCA, SIMLR, and ComBat. Embedding plots for all methods but SIMLR were generated by application of t-SNE on the respective latent space. For SIMLR, we used the t-SNE coordinates provided by the program and the number of clusters was set to the number of pre-annotated subpopulations (*n* = 15).



Supplementary Figure 13

Capturing technical variability with scVI.

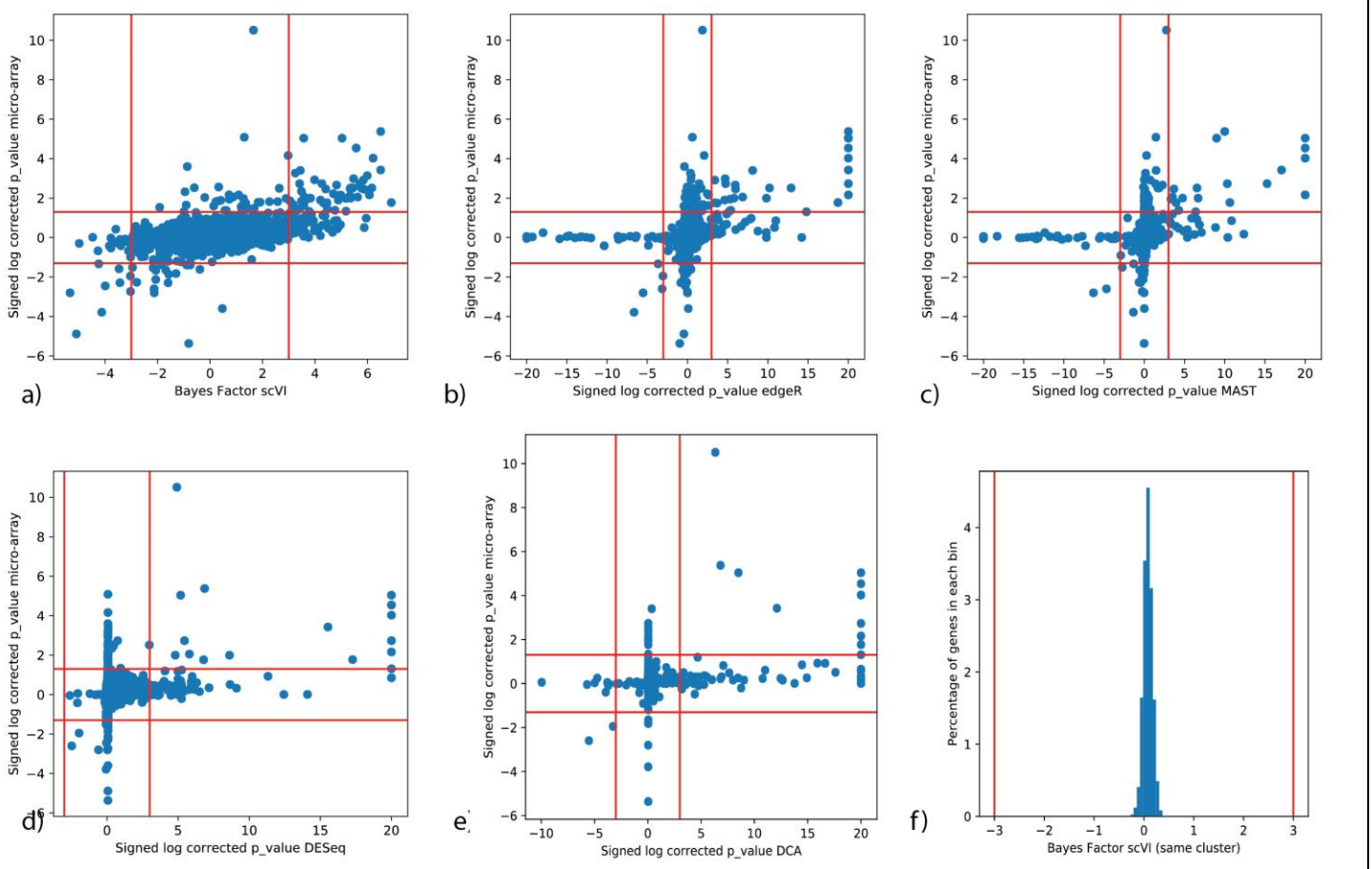
(a,b) Data are based on the CD14$^+$ cell subpopulation of the PBMC dataset. (a) Scatter plot for each cell (n = 2,237) of inferred scaling factor using scVI against library size. (b) The frequency of observed zero values versus the expected expression level, as produced by scVI. Each of the n = 3,346 points represents a gene g, where the x-axis is ρ^g, the average expected frequency per cell (for gene g, average over ρ^g for all cells c in the subpopulation), and the y-axis is the observed percentage of cells that detects the fee (UMI > 0). The green curve depicts the probability for selecting zero transcripts from every gene as a function of its frequency, assuming a simple model of sampling U molecules from a cell with N molecules at random without replacement. U = 1,398 is the average number of UMIs in the subpopulation and N is the average number of transcripts per cell (for this curve N = 10,000). Notably, the curve converges for values larger than 20,000 to the red curve, a binomial selection procedure (conform to the probabilistic limit of the sampling process when N goes to infinity). (c,d) Signed log P values for a Pearson correlation test between the zero probabilities from the two distributions (negative binomial, Bernoulli) and quality control metrics across n = 5 random initializations of scVI and all subpopulations of the PBMC (n = 12,039 cells) and BRAIN-SMALL (n = 9,128 cells) datasets. The center of the box plot is the median and the hinges correspond to the interquartile range, the distance between the first and third quartiles; the whiskers represent the 5th to 95th percentiles.



Supplementary Figure 14

The generative distributions of scVI.

This study focuses on a particular subpopulation of the BRAIN-SMALL dataset (n = 2,592). (a) To assess whether most of the zeros in the data come from the negative binomial, for each entry of the count matrix (percentage in y-axis), we plot the probability that a given zero comes from the NB conditioned on having a zero (x-axis). (b) Number of genes detected versus negative binomial zero probability averaged across all genes. (c) Genome_not_gene versus Bernoulli zero probability averaged across all genes. (d) Mapped_reads versus Bernoulli zero probability averaged across all genes.



Supplementary Figure 15

Differential expression with scVI on the PBMC dataset.

(a–e) BH-corrected *P* values from bulk analysis against BH-corrected *P* values or Bayes factor for $CD4^+/CD8^+$ comparison (microarrays, $n = 12$ in each group, scRNA-seq $n = 100$ cells). Each point is a gene ($g = 3{,}346$). In the order indicated, scVI, edgeR, MAST, DESeq2, DESeq2 on DCA imputed counts. (f) Histogram for the Bayes factor of scVI when applying differential expression to two sets of $n = 100$ random cells from the same cluster.

Deep Generative Modeling for Single-cell Transcriptomics

Supplementary Information

Romain Lopez[1], Jeffrey Regier[1], Michael B. Cole[2],
Michael I. Jordan[1,3] & Nir Yosef[1,5,6]

[1]Department of Electrical Engineering and Computer Sciences
[2]Department of Physics
[3]Department of Statistics
University of California, Berkeley
[5]Ragon Institute of MGH, MIT and Harvard
[6]Chan-Zuckerberg Biohub Investigator

Correspondence should be addressed to N.Y (niryosef@berkeley.edu)

Supplementary Tables

List of Tables

	Characteristics of the Model					Features of the Software		
	Dropout	**Count distribution**	**Library size**	**Batch Effects**	**Generative Model**	**Dimensionality Reduction**	**Imputation**	**Differential Expression**
Factor Analysis					X	X		
ZIFA	X				X	X		
SIMLR	X					X		
BISCUIT			X		X	X*		
ZINB-WaVE	X	X		X		X	X	
MAGIC	X						X	
DESeq2		X	X	X				X
MAST	X	X	X	X				X
edgeR		X	X	X				X
ComBat				X			X	
MNNs				X			X	
scvis					X	X		
DCA	X	X				X	X	
VASC	X				X	X		
scVAE	X	X			X	X	X	
scVI	X	X	X	X	X	X	X	X

Table 1: Presentation of all the papers cited in this manuscript. We describe both features of the model and the corresponding open-source software. *Dropout*: any computing solution to mitigate the dropout effect on data analysis. *Count distribution*: the underlying distribution modeling the gene expression has support in the integer set. *Library size*: the model is designed to correct for library size or other technical effects. *Batch effects* refers to a model that can account for this technical variation. *Generative Model* refers to the possibility of sampling from a posterior. The star (*) for BISCUIT indicates that this algorithm cannot perform dimensionality reduction but can still cluster cells.

	dataset properties		scVI hyperparameters			scalability to other algorithms				
	cells	**genes**	**learning rate**	**layers**	**neurons**	**FA**	**ZIFA**	**ZINB WaVE**	**MAGIC**	**SIMLR**
CORTEX	3,005	558	0.0004	1	128	_	_	_	_	_
PBMC	12,039	3,346	0.0004	1	128	_	x	x	_	_
HEMATO	4,016	7,397	0.0004	1	128	_	x	x	_	NC
HEMATO*	4,016	700	0.0004	1	128	_	x	x	_	NC
BRAIN LARGE	10,000	720	0.001	1	128	_	_	_	_	NA
BRAIN LARGE	15,000	720	0.001	2	128	_	_	_	NC	NA
BRAIN LARGE	50,000	720	0.001	3	256	_	x	x	NC	NA
BRAIN LARGE	1M	720	0.001	3	256	_	x	x	x	NA
BRAIN LARGE	1M	10,000	0.001	3	256	_	x	x	x	NA
RETINA	27,499	13,166	0.0005	1	128	_	x	x	x	_
CBMC	8,617	600	0.0005	1	128	_	_	_	NC	_
BRAIN SMALL	9,128	3,000	0.001	1	128	NC	NC	NC	NC	NC

Table 2: Presentation of the different datasets, their gene filtering and applicability of algorithms. "_" indicates combinations of algorithms and datasets that were included in this study. "x" indicates that the algorithm took more than four hours to run (ZIFA) or that the computer ran out of memory (others). "NA" indicates datasets where pre-annotated subpopulations were not available, which makes them less useful for application with SIMLR and benchmarks of clustering. "NC" indicates the remaining combinations of algorithms and datasets that were not considered for this study. For instance, BRAIN-SMALL was only used to study the correlation of scVI zero probabilities and quality parameters (Figure 6).

cells	4K	10K	15K	50K	100K
FA	-1175.36	-1177.35	-1177.27	-1171.93	-1169.86
ZIFA	-1250.44	-1250.77	-1250.59	NA	NA
ZINB-WaVE	-1166.39	-1163.91	-1163.39	NA	NA
scVI	**-1150.96**	**-1146.59**	**-1144.88**	**-1136.57**	**-1133.94**

Table 3: Marginal log likelihood for a held-out subset of the brain-cell dataset. NA indicates we could not run the given algorithm for this sample size. FA denotes factor analysis.

Cell Type	# cells	Cluster ID (original)	Cluster ID (#4)	Cluster ID (#3)	Cluster ID (#2)
astrocytes_ependymal	224	0	1	0	0
endothelial-mural	235	1	2	1	0
interneurons	290	2	3	2	1
microglia	98	3	2	1	0
oligodendrocytes	820	4	0	0	0
pyramidal CA1	939	5	3	2	1
pyramidal SS	399	6	3	2	1

Table 4: Cell types present in the CORTEX dataset and their labels in some slices of the original hierarchical clustering.

Cell Types	# cells
B cells	1625
CD14+ Monocytes	2237
CD4 T cells	5024
CD8 T cells	1452
Dendritic Cells	339
FCGR3A+ Monocytes	351
Megakaryocytes	88
NK cells	459
Other	464

Table 5: Cell types present in the PBMC dataset.

Supplementary Note 1: Related work

Independent of our work, several recent articles and preprints have also describe the use of neural networks, including variational autoencoders [25] (VAEs), to embed scRNA-seq datasets. Like scVI, these methods use stochastic optimization and therefore also scale to the size of recent data sets. scVI stands apart from this research in its focus on separating technical effects from biological signals. To this end, we explicitly model library size and batch effects as nuisance variables in a hierarchical Bayesian model. Empirical advantages of such careful modeling are underlined in [8, 22] for library size and [10, 23] for batch effects. scVI is also distinguished by the large variety of datasets it has been validated on, and the number downstream applications it supports (see **Supplementary Table 1**). In the following, we provide a brief description of each method.

scvis [18] proposes a new variational lower bound inspired by the parametric version of tSNE for single-cell visualization. Unlike scVI, scvis uses principal component analysis for preprocessing the data, and relies on t-distributed conditional distributions. These assumptions may be fruitful for visualization purposes but may not help in recovering hidden gene expression levels for either imputation or differential expression. One notable reason is that the underlying statistical assumptions for applying principal component analysis are not verified in the scRNA-seq dataset and the output of this preprocessing may have artifacts [10].

VASC [19] adds zero-inflation to a VAE with Gaussian conditional distributions and shows competitive performance on clustering. Preprocessing relies on normalizing the data by library size in order to suit their loss function. However, it is known that size scaling normalization can induce bias in the analysis [5] and that count distributions such as ZINB outperform zero-inflated Gaussian conditional distribution for modeling and downstream analysis [10, 9].

DCA [20] is a denoising auto-encoder that proposes a ZINB loss function. The output of this algorithm is a latent space as well as a point estimate for the conditional distributions (this denoising method is essentially a non-linear version of ZINB-WaVE). However, it is hard to extend this method for differential expression without the possibility of posterior sampling. Notably, library size discrepancies are also corrected via size scaling and potentially hurt downstream analysis [5].

scVAE [21] proposes to use VAEs with count conditional distributions for Bayesian modeling and clustering analysis. Notably, their work uses (among others) the NB and the ZINB as conditional distributions but do not propose an approach for normalization, which is crucial for further analysis.

Supplementary Note 2: Capturing technical variability with scVI

To further interpret the fitted models, we study the extent to which they capture technical variability. We focus on datasets that were generated by 10x, as they share the same set of cell quality metrics (generated by cell ranger; see **Online Methods**) and can thus provide reproducible insights about the relationship between our parameters and library quality. Additionally, we required our test datasets to have pre-annotated subpopulations, with the assumption that each subpopulation consists primarily of cells of the same type, thus decreasing the extent of biological heterogeneity (e.g., in total mRNA content). The two datasets that fit these requirements, which we report next, are the PBMC and BRAIN-SMALL.

In each case, we trained the model on the entire dataset and then investigated each pre-annotated subpopulation separately. As a general rule, we find that variation in library size correlates strongly with the cell-specific scaling factor, which is to be expected by the definition of the model. **Supplementary Figure 13a** depicts these results for the CD14+ monocytes subpopulation in the PBMC data (notably, the negative curvature on the plot can be explained by shrinking the values towards the mean due to the use of a prior). Another important nuisance factor to be considered is the limitation in sensitivity, which exacerbates the number of zero entries. In principle, zero entries can be captured by two different components of our model: the negative binomial and the "inflation" of zeros added to it with a Bernoulli distribution. Evidently, the expected number of zeros generated by the negative binomial for each cell correlates strongly with the library size and its proxies (e.g., the number of detected genes or the number of reads per UMI; **Supplementary Figures 13cd, and 14d**). This result can be explained by our definition of the negative binomial mean, which is the predicted frequency of expression ρ_n^g scaled by the respective library size ℓ_n.

The remaining question is therefore, what is the relationship between the expected frequency of expression ρ_n^g and the observed zeros? A simple model would consist of a random process of sampling genes from each cell, in a manner proportional to their frequency, and with no added bias (e.g., in capture efficiency). To explore this, for every gene we plot the mean expected frequency against the percent of detecting cells in a subpopulation of interest (**Supplementary Figure 13b**). The resulting trend supports this simple model as it closely fits with the zero probability of a hypergeometric distribution—namely, random sampling of molecules without replacement.

Interestingly, the number of additional zeros induced by the Bernoulli random variable in each cell is less correlated with library size, and instead correlates with metrics of alignment rate (**Supplementary Figures 13cd and 14cd**). These metrics are not necessarily coupled to size, but may reflect other technical factors such as contamination or the presence of degraded mRNA. However, we observed that most zero values in the data can be explained by the negative bi-

nomial component alone (**Supplementary Figure 14a**). Taken together, these results therefore corroborate the idea that most zeros, at least in the datasets explored here, can be explained by low (or zero) "biological" abundance of the respective transcript, which is exacerbated by limited sampling.

Supplementary Note 3: Marginalizing out the latent variables of scVI

First, take r to be the gene-specific shape parameter of a Gamma variable w and $\frac{p}{1-p}$ to be its scale parameter, use a scalar $\lambda \in \mathbb{R}^+$, then the count variable $y|w \sim \text{Poisson}(\lambda w)$ has a negative binomial marginal distribution with mean $r\lambda\frac{p}{1-p}$

$$\begin{aligned} p(y) &= \int p(y|w)p(w)dw \\ &= \int \frac{w^{r-1}e^{-w(\frac{1}{p}-1)}(1-p)^r}{p^r\Gamma(r)}\frac{e^{-\lambda w}\lambda^y w^y}{\Gamma(y+1)}dw \\ &= \frac{\Gamma(y+r)}{\Gamma(y+1)\Gamma(r)}\left(\frac{1-p}{1-p+\lambda p}\right)^r\left(\frac{p\lambda}{1-p+\lambda p}\right)^y \end{aligned} \tag{3}$$

.

Second, multiplication by zero to y_{ng} can be formally encoded as a mixture between a point-mass at zero and the original distribution of y_{ng}.

Consequently, our conditional $p(x_{ng}|z_n, \ell_n, s_n)$ is a zero-inflated negative binomial with probability mass function

$$\begin{cases} p(x_j = 0|z,l,s) = f_h(z)_j + (1 - f_h(z)_j)\left(\frac{\theta_j}{\theta_j + l f_w(z)_j}\right)^{\theta_j} \\ p(x_j = y|z,l,s) = (1 - f_h(z)_j)\frac{\Gamma(y+\theta_j)}{\Gamma(y+1)\Gamma(\theta_j)}\left(\frac{\theta_j}{\theta_j + l f_w(z)_j}\right)^{\theta_j}\left(\frac{f_w(z)_j}{\theta_j + l f_w(z)_j}\right)^y, \forall y \in \mathbb{N}^* \end{cases}$$

where $f_h(z)$ is encoding the zero probability of h and $f_w(z)$ the mean of w.

Supplementary Note 4: Negative binomial parametrization

Negative binomial PMF parametrization A choice of parametrization is crucial for optimization considerations. We could follow [10] by using a mean μ and an inverse-dispersion θ parameter:

$$p_{NB}(n;\mu,\theta) = \frac{\Gamma(n+\theta)}{\Gamma(n+1)\Gamma(\theta)}\left(\frac{\theta}{\theta+\mu}\right)^{\theta}\left(\frac{\mu}{\theta+\mu}\right)^{n}$$

We also keep in mind a more gentle parametrization with nicer form although it is still non-convex:

$$p_{NB}(n;p,r) = \frac{\Gamma(n+r)}{\Gamma(n+1)\Gamma(r)}p^n(1-p)^r$$

with $(p,r) = (\frac{\mu}{\theta+\mu},\theta)$ or $(\mu,\theta) = (\frac{rp}{1-p},r)$

Because the first parametrization has better behavior when scaling the Poisson mean (as we do in library size normalization), this is the one we retain.

Equivalence of parametrization Assume now that we want to simulate what the rate of the latent corresponding Poisson variable would have been; we have to sample from a Gamma of shape r and scale $\frac{p}{1-p}$ or rate $\frac{1-p}{p}$.

Numerical considerations We transformed the expression to incorporate logits and use Tensorflow numerically stable functions. Instead of explicitly writing a sigmoid non-linearity, the probability of zero in the mixture is given by

$$f_\pi(z) = \frac{1}{1+e^{-F_\pi^z}}$$

,

where F_π^z is the output of the neural network without non-linearity. We then write the log-likelihood as a function of F_π^z.

The use of an r can either be parametrized by a neural net or constant for each gene, and will be noted r for simplicity. $\mathcal{S}$ denotes the softplus function $x \mapsto \log(1+e^x)$.

$$\begin{aligned}\log p(y|z) = {} & \mathbb{1}_{y=0}\left[\mathcal{S}(-F_\pi^z + f_\theta^z \log\frac{f_\theta^z}{f_\theta^z+f_\mu^z}) - \mathcal{S}(-F_\pi^z)\right] \\ & + \mathbb{1}_{y>0}\left[-F_\pi^z - \mathcal{S}(-F_\pi^z) + f_\theta^z\log\frac{f_\theta^z}{f_\theta^z+f_\mu^z} + y\log\frac{f_\mu^z}{f_\theta^z+f_\mu^z} + \log\frac{\Gamma(y+\theta)}{\Gamma(\theta)\Gamma(y+1)}\right]\end{aligned}$$

Supplementary Note 5:
Details on algorithms used for benchmarking

Below we describe the algorithms used whenever applicable (the complete scalability table for each benchmark is in **Supplementary Table 2**). A star after the algorithm name indicates that we used it in order to make a specific point in the paper but not for general benchmarking.

Factor analysis We used the factor analysis (FA) method from the scikit-learn python package. FA is always applied to log-data.

ZIFA We used the zero-inflated factor analysis method (ZIFA) from `https://github.com/epierson9/ZIFA` with default parameters. We always apply ZIFA to log-data.

SIMLR We used the large scale version of the Single-cell Interpretation via Multi-kernel LeaRning (SIMLR) algorithm from Bioconductor with parameters recommended by the authors (k=30, kk=200). We always apply SIMLR to log-data as recommended in the original paper. We used the same number of clusters as the number of cell types, except for the HEMATO data and the random ZINB dataset, where we used the procedure SIMLR_Estimate_Number_of_Clusters.

BISCUIT We applied the BISCUIT algorithm from `https://github.com/sandhya212/BISCUIT_SingleCell_IMM_ICML_2016` with default parameters. As the code to recover the exact results from [8] was not available, we did not consider BISCUIT in the clustering benchmark. In addition, we checked different parameters for the number of iterations and the spin parameter without being able to generate a configuration that would provide a better score for imputation.

ZINB-WaVE We applied the ZINB-WaVE procedure from the R package zinbwave with both the gene-level covariates and the cell-level covariates set to ones vectors. We always apply ZINB-WaVE to count-data.

PCA We used the Principal Component Analysis method from the scikit-learn python package. We always apply PCA to log-data.

MAGIC We used the Python3 code of the Markov Affinity-based Graph Imputation of Cells algorithm from `https://github.com/KrishnaswamyLab/magic`. We used the parameters from their iPython notebook on the github repo.

MAST We used the R package MAST on log-counts to provide our differential expression analysis.

DESeq2 We used the R package DESeq2 on raw counts to provide our differential expression analysis.

edgeR We used the R package edgeR on raw counts to provide our differential expression analysis.

ComBat We used the R package sva on the bipolar dataset following the original preprocessing steps of the bipolar paper [31].

tSNE We used the TSNE class from scikit-learn with a default perplexity parameter of 30.

Force-directed Layout We used the spring_layout module from the networkx package by working with the raw 5-nearest neighbors adjacency matrix.

Mutual Nearest Neighbors* We used the mnnCorrect package from `https://rdrr.io/bioc/scran/man/mnnCorrect.html` with default parameters. MNNs was specifically used on the RETINA dataset for the batch removal benchmark.

DCA* We used the Deep Count Autoencoder package from `https://github.com/theislab/dca` with default parameters. DCA was specifically used on the PBMC, RETINA and BRAIN-LARGE dataset for the imputation, batch-removal and differential expression benchmark.

Supplementary Note 6: Comparing log-likelihood across models

The common method of evaluation for generative models is to evaluate the model on data it has never seen. Let us start with a complete dataset X. From this dataset, we randomly selected a training set X_{train} and a held-out testing set X_{test}. Each model gives and underlying probability measure p that we will specify. We define the log-likelihood on held-out data by integrating in the following way:

$$\int_{\Theta} p(X_{test}|\Theta) \cdot dp(\Theta|X_{train})$$

where $dp(\Theta|X_{train})$ designates the posterior parameters of the model after having fitted the training data and where $p(X_{test}|\Theta)$ is assessing the goodness-of-fit of the held-out data under the chosen parameter Θ.

In the case of a fully generative model like BISCUIT, the posterior parameter $dp(\Theta|X_{train})$ is designated by a probability measure we have to sample from. For technical reasons, we did not include BISCUIT in this analysis. Specifically, the original code does not provide a straightforward way to evaluate posterior likelihood on unseen data. In most generative models, we take a point estimate over these parameters (parameters of a neural network in scVI, factor loading matrix or decay rate in ZIFA) and the former measure is a Dirac centered on the parameters fixed at testing time.

Now, we focus on the value $p(X_{test}|\Theta)$ itself. First, because some algorithms are run on log transformed data and some on raw data, we take into account the distortion for the log-likelihood scores. Second, this quantity can often be intractable because of latent variables we have to marginalize out. In that case, we can take lower bounds from the Expectation Maximization algorithm for ZIFA, exact value for FA and the variational lower bound for scVI. In the case of an algorithm where the latent variables are actual parameters to optimize (as in ZINB-WaVE), we need to re-run this partial optimization at testing-time.

Log and non-log data Let X be a positive random variable and let us note $Y = \log(1+X)$ and suppose we have a model for Y written $\mathbb{P}_Y$. The likelihood score on the raw data is given by evaluating the density $\mathbb{P}_X$, which is

$$\forall x = (x_1, ..., x_d) \in \mathbb{R}^d, d\mathbb{P}_X(x) = d\mathbb{P}_Y(\log(1+x)) \prod_{i=1}^{d} \frac{1}{1+x_i}$$

So, for the likelihood scores, this yields

$$\log \mathbb{P}_X(X = x) = \log \mathbb{P}_Y(Y = \log(1+x)) - \sum_{i=1}^{d} \log(1+x_i)$$

Log-likelihood for ZIFA The EM algorithm naively provides a lower bound on the log-likelihood under ZIFA:

$$\log p(Y|\Theta) \geq \mathbb{E}_{p(Z,X,H|Y,\Theta)} \log p(Z, X, H, Y|\Theta)$$

The complete log-likelihood has a simple expression:

$$\begin{aligned} LL = \log p(z_i, x_i, h_i, y_i|\Theta) = &-\frac{1}{2} z_i^T z_i - \sum_j \log(\sigma_j) \\ &+ \sum_{j|y_{i,j}=0} -\frac{(x_{i,j} - (Az_i)_j - \mu_j)^2}{w\sigma_j^2} - \lambda_j x_{i,j}^2 \\ &+ \sum_{j|y_{i,j}>0} -\frac{(y_{i,j} - (Az_i)_j - \mu_j)^2}{w\sigma_j^2} + \log(1 - e^{-\lambda_j y_{i,j}^2}) \end{aligned}$$

and the prior distribution is close to Gaussian; therefore, we can modify ZIFA code and use an E-step to compute the desired value. The E-step gives us the following values:

- $\mathbb{E}(x_i \odot x_i) = EX^2$
- $\mathbb{E}(x_i z_i^T) = EXZ$
- $\mathbb{E}(z_i z_i^T) = EZZ^T$
- $\mathbb{E}(x_i) = EX$
- $\mathbb{E}(z_i) = EZ$

Then we have

$$\begin{aligned} LL = &-\frac{1}{2} Tr(EZZ^T) - \frac{d}{2}\log(2\pi) - \sum_j [\frac{\log(2\pi\sigma_j^2)}{2} + \frac{\mu_j^2}{2\sigma_j^2} + \frac{(AEZZ^TA^T)_{j,j}}{2\sigma_j^2} + \frac{(AEZ \odot \mu)_j}{\sigma_j^2}] \\ &+ \sum_{j|y_{i,j}=0} \frac{1}{2\sigma_j^2}[-EX_j^2 + 2(EXZA^T)_{j,j} + 2(EX \odot \mu)_j] - \lambda EX_j^2 \\ &+ \sum_{j|y_{i,j}>0} \frac{1}{2\sigma_j^2}[-y_{i,j}^2 + 2(y_i \odot AEZ)_{j,j} + 2(y_i \odot \mu)_j] + \log(1 - e^{-\lambda y_{i,j}^2}) \end{aligned}$$

Log-likelihood for scVI Our variational inference procedure provides us with a lower bound on the log-likelihood of held-out data:

$$\log p(x) \geq \mathbb{E}_{q(z,l|x)} \log p(x|z, l) - KL(q(z|x)||p(z)) - KL(q(l|x)||p(l)) \qquad (4)$$

The lower-bound is tight whenever $q(z|x) = p(z|x)$. With the generative model's parameters held fixed, we can optimize our inference network at test-time to have a better lower-bound of the held-out log-likelihood and report the best value. This is equivalent to assessing the marginal likelihood of held-out data, conditioned on a latent representation learned for the held-out data.

Log-likelihood for ZINB-WaVE The function to be optimized for ZINB-WAVE is essentially penalized likelihood. We can thus run the full optimization function on a training set once, as follow:

$$\max_{\beta,\gamma,W,\alpha,\zeta} \mathcal{L}_{\text{train}}(\beta,\gamma,W,\alpha,\zeta) - \text{Pen}(\beta,\gamma,W,\alpha,\zeta)$$

This optimization is performed by alternating minimization. By fixing the variables β, α, ζ learned from the training set, we can compute a likelihood of a validation set by performing inference over the latent variables γ, W, which is a simple Ridge that can be solved in parallel through a simple modification of their code.

$$\max_{\gamma,W} \mathcal{L}_{\text{val}}(\beta^*,\gamma,W,\alpha^*,\zeta^*) - \text{Pen}(\beta^*,\gamma,W,\alpha^*,\zeta^*)$$

Results In this test we evaluate how likely is the decoding part of the model to generate the unseen data by first sampling from the latent space. We used the BRAIN-LARGE dataset as above and explored a range of training set sizes from a few thousand cells to a hundred thousand cells. We also analyzed a smaller data set of 3k mouse cortex cells [29] (referred to henceforth as CORTEX), leaving out one third of the cells. Overall, we find that methods that are based on ZINB distribution (ZINB-WAVE, scVI) are more accurate than ones based on log-normal or a zero-inflated-log-normal (ZIFA or a standard factor analysis (FA)), thus corroborating that scRNA-seq data is better approximated by the former distribution. Furthermore, we find that scVI provides the most likely models and that its accuracy increases as the training dataset size grows.