Analytic Pearson residuals for normalization of single-cell RNA-seq UMI data

Jan Lause[1], Philipp Berens[1,2,3,4], and Dmitry Kobak[1]

[1] *Institute for Ophthalmic Research, University of Tübingen, Germany*
[2] *Institute for Bioinformatics and Medical Informatics, University of Tübingen, Germany*
[3] *Bernstein Center for Computational Neuroscience, University of Tübingen, Germany*
[4] *Center for Integrative Neuroscience, University of Tübingen, Germany*
`name.surname@uni-tuebingen.de`

April 27, 2021

Abstract

Standard preprocessing of single-cell RNA-seq UMI data includes normalization by sequencing depth to remove this technical variability, and nonlinear transformation to stabilize the variance across genes with different expression levels. Instead, two recent papers propose to use statistical count models for these tasks: Hafemeister and Satija (2019) recommend using Pearson residuals from negative binomial regression, while Townes et al. (2019) recommend fitting a generalized PCA model. Here, we investigate the connection between these approaches theoretically and empirically, and compare their effects on downstream processing. We show that the model of Hafemeister and Satija (2019) produces noisy parameter estimates because it is overspecified (which is why the original paper employs post-hoc regularization). When specified more parsimoniously, it has a simple analytic solution equivalent to the rank-one Poisson GLM-PCA of Townes et al. (2019). Further, our analysis indicates that per-gene overdispersion estimates in Hafemeister and Satija (2019) are biased, and that the data are in fact consistent with constant overdispersion parameter across genes. We then use negative control data without biological variability to estimate the technical overdispersion of UMI counts, and find that across several different experimental protocols, the data suggest very moderate overdispersion. Finally, we demonstrate that analytic Pearson residuals (or, equivalently, rank-one GLM-PCA or negative binomial regression after regularization) strongly outperform other methods for identifying biologically variable genes, and capture more of the biologically meaningful variation when used for dimensionality reduction.

1 Introduction

The standard preprocessing pipeline for single-cell RNA-seq data includes sequencing depth normalization followed by log-transformation (Luecken and Theis, 2019; Amezquita et al., 2020). The normalization aims to remove technical variability associated with cell-to-cell differences in sequencing depth, whereas the log-transformation is supposed to make the variance of gene counts approximately independent of the mean expression. Two recent papers argue that neither step works very well in practice (Hafemeister and Satija, 2019; Townes et al., 2019). Instead, both papers suggest to model UMI (unique molecular identifier) data with count models, explicitly accounting for the cell-to-cell variation in sequencing depth (defined here as the total UMI count per cell). Hafemeister and Satija (2019) (`scTransform` package in R) use a negative binomial (NB) regression model, while Townes et al. (2019) propose Poisson generalized principal component analysis (GLM-PCA). These two models are seemingly very different.

Here we show that the model used by Hafemeister and Satija (2019) has a too flexible parametrization, resulting in noisy parameter estimates. As a consequence, the original paper employs post-hoc smoothing to correct for that. We show that a more parsimonious model produces stable estimates even without regularization and is equivalent to a special case of GLM-PCA. We then demonstrate that the estimates of gene-specific overdispersion in the original paper are strongly biased, and further argue that UMI data do not require gene-specific overdispersion parameters. Rather, the data are consistent with the

same overdispersion parameter shared between all genes. We use available negative control datasets to estimate this technical overdispersion. Furthermore, we compare Pearson residuals, GLM-PCA, and variance-stabilizing transformations for highly variable gene selection and as data transformation for downstream processing.

Our code in Python is available at http://github.com/berenslab/umi-normalization.

2 Results

2.1 Analytic Pearson residuals

A common modeling assumption for UMI or read count data without biological variability is that each gene g takes up a certain fraction p_g of the total amount n_c of counts in cell c (Love et al., 2014; Eling et al., 2018; Lopez et al., 2018; Townes et al., 2019; Svensson et al., 2020). The observed UMI counts X_{cg} are then modelled as Poisson or negative binomial (NB) samples with expected value $\mu_{cg} = p_g n_c$ without zero-inflation (Svensson, 2020):

$$X_{cg} \sim \text{Poisson}(\mu_{cg}) \text{ or } \text{NB}(\mu_{cg}, \theta), \tag{1}$$

$$\mu_{cg} = n_c p_g. \tag{2}$$

The Poisson model has a maximum likelihood solution (see Methods) that can be written in closed form as $\hat{n}_c = \sum_g X_{cg}$ (sequencing depths), $\hat{p}_g = \sum_c X_{cg} / \sum_c \hat{n}_c$, or, put together,

$$\hat{\mu}_{cg} = \frac{\sum_j X_{cj} \cdot \sum_i X_{ig}}{\sum_{ij} X_{ij}} \tag{3}$$

For the negative binomial model this holds only approximately. Using this solution, the Pearson residuals are given by

$$Z_{cg} = \frac{X_{cg} - \hat{\mu}_{cg}}{\sqrt{\hat{\mu}_{cg} + \hat{\mu}_{cg}^2/\theta}}, \tag{4}$$

where $\mu_{cg} + \mu_{cg}^2/\theta$ is the NB variance and $\theta \to \infty$ gives the Poisson limit. The variance of Pearson residuals is, up to a constant, equal to the Pearson χ^2 goodness-of-fit statistic (Agresti, 2015).

Hafemeister and Satija (2019) suggested using Pearson residuals from a related NB regression model for highly variable gene (HVG) selection and also as a data transformation for downstream processing. In parallel, Townes et al. (2019) suggested using deviance residuals (see Methods) from the same Poisson model as above for HVG selection and also for PCA as an approximation to their GLM-PCA. In the next sections we discuss the relationships between these approaches.

2.2 The regression model in `scTransform` is overspecified

Hafemeister and Satija (2019) used the 33k PBMC (peripheral blood mononuclear cells, an immune cell class that features several distinct subpopulations) dataset from 10X Genomics in their work on normalization of UMI datasets. For each gene g in this dataset, the authors fit an independent NB regression

$$X_{cg} \sim \text{NB}(\mu_{cg}, \theta_g) \tag{5}$$

$$\ln(\mu_{cg}) = \beta_{0g} + \beta_{1g} \log_{10}(\hat{n}_c). \tag{6}$$

Here θ_g is the gene-specific overdispersion parameter, $\hat{n}_c$ are observed sequencing depths as defined above, and β_{0g} and β_{1g} are the gene-specific intercept and slope. The natural logarithm follows from the logarithmic link function that is used in NB regression by default. The original paper estimates β_{0g} and β_{1g} using Poisson regression, and then uses the obtained estimates to find the maximum likelihood estimate of θ_g. The resulting estimates for each gene are shown in Figure 1a–c, reproducing Figure 2A from the original paper.

The authors observed that the estimates $\hat{\beta}_{0g}$ and $\hat{\beta}_{1g}$ were unstable and showed high variance for genes with low average expression (Figure 1a–b). They addressed this by a 'regularization' procedure that re-set all estimates to the local kernel average estimate for a given expression level (similar to Love et al., 2014; Eling et al., 2018). The kernel smoothing they employed resulted in an approximately

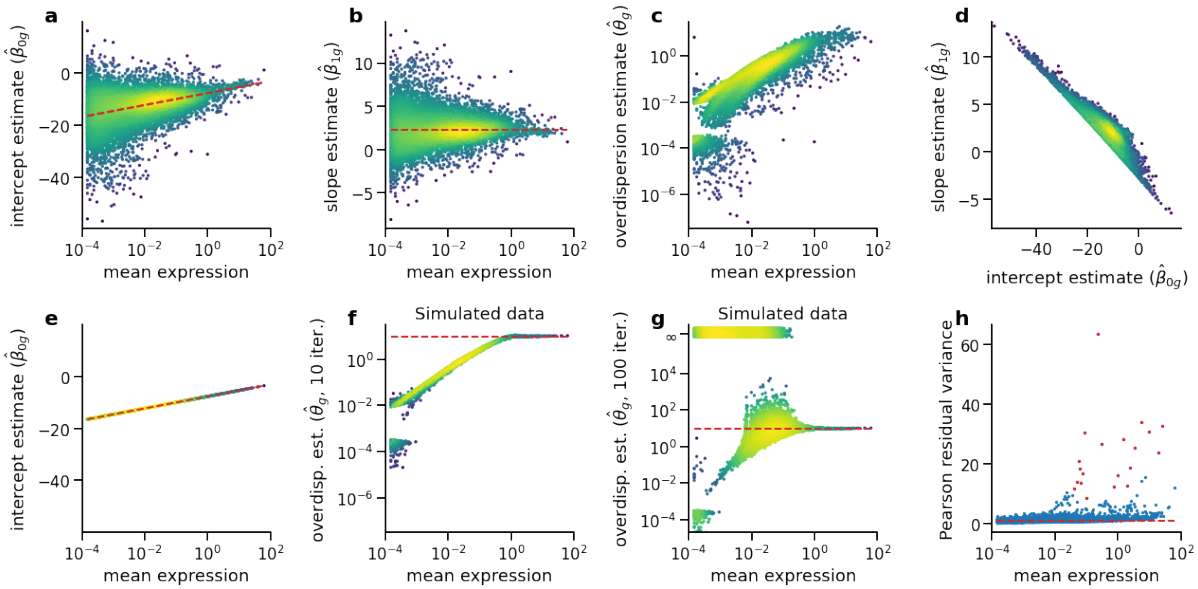

Figure 1: Regression model of Hafemeister and Satija (2019) compared to the offset model. Each dot corresponds to a model fit to the counts of a single gene in the 33k PBMC dataset (10x Genomics, $n = 33\,148$ cells). Following Hafemeister and Satija (2019), we included only the 16 809 genes that were detected in at least five cells. Color denotes the local point density from low (blue) to high (yellow). Expression mean was computed as $\frac{1}{n}\sum_c X_{cg}$. **a:** Intercept estimates $\hat{\beta}_{0g}$ in the original regression model. Dashed line: Analytical solution for $\hat{\beta}_{0g}$ in the offset model we propose. **b:** Slope estimates $\hat{\beta}_{1g}$. Dashed line: $\beta_{1g} = \ln(10) \approx 2.3$. **c:** Overdispersion estimates $\hat{\theta}_g$. **d:** Relationship between slope and intercept estimates ($\rho = -0.91$). **e:** Intercept estimates in the offset model, where the slope coefficient is fixed to 1. Dashed line shows the analytical solution, which is a linear function of gene mean. **f:** Overdispersion estimates $\hat{\theta}_g$ on simulated data with true $\theta = 10$ (dashed line) for all genes. **g:** Overdispersion estimates $\hat{\theta}_g$ on the same simulated data as in panel (f), but now with 100 instead of 10 iterations in the `theta.ml()` optimizer (R, `MASS` package). Cases for which the optimization diverged to infinity or resulted in spuriously large estimates ($\hat{\theta}_g > 10^6$) are shown at $\hat{\theta}_g = \infty$ with some jitter. Dashed line: true value $\theta = 10$. **h:** Variance of Pearson residuals in the offset model. The residuals were computed analytically, assuming $\theta = 100$ for all genes. Following Hafemeister and Satija (2019), we clipped the residuals to a maximum value of $\sqrt{n}$. Dashed line indicates unit variance. Red dots show the genes identified in the original paper as most variable.

linear increase of the intercept with the logarithm of the average gene expression (Figure 1a) and an approximately constant slope value of $\hat{\beta}_{1g} \approx 2.3$ (Figure 1b). The nature of these dependencies was left unexplained. Moreover, we found that $\hat{\beta}_{0g}$ and $\hat{\beta}_{1g}$ were strongly correlated ($\rho = -0.91$), especially for weakly expressed genes (Figure 1d). Together, these clear symptoms of overfitting suggest that the regression model was overspecified.

Indeed, the theory calls for a less flexible model. As explained above, a common modeling assumption (Eq. 2) is that $\mu_{cg} = p_g n_c$, or equivalently

$$\ln(\mu_{cg}) = \ln(p_g) + \ln(n_c) = \beta_{0g} + \ln(n_c). \tag{7}$$

We see that under this assumption, the slope β_{1g} does not need to be fit at all and should be fixed to 1, if $\ln(n_c)$ is used as predictor. Not only does this suggest an alternative, simpler parametrization of the model, but it also explains why Hafemeister and Satija (2019) found that $\hat{\beta}_{1g} \approx 2.3$: they used $\log_{10}(n_c) = \ln(n_c)/\ln(10)$ instead of $\ln(n_c)$ as predictor, and so obtained $\ln(10) \approx 2.3$ as the average slope.

Under the assumption of Eq. 7, a Poisson or NB regression model should be specified using $\ln(n_c)$ as predictor with a fixed slope of 1, a so-called *offset* (Eqs. 5 and 7). This way, the resulting model has only one free parameter and is not overspecified. Moreover, the Poisson offset model is equivalent to Eqs. 1–2 and so, as explained above, has an analytical solution

$$\hat{\beta}_{0g} = \ln\left(\sum_c X_{cg} / \sum_c n_c\right) = \ln\left(\tfrac{1}{N}\sum_c X_{cg}\right) - \ln\left(\tfrac{1}{N}\sum_c n_c\right), \tag{8}$$

which forms a straight line when plotted against the log-transformed average gene expression $\frac{1}{n}\sum_c X_{cg}$

(Figure 1e). This provides an explanation for the linear trend in $\hat{\beta}_{0g}$ in the original two-parameter model (Figure 1a).

In practice, our one-parameter offset model and the original two-parameter model after regularization arrive at qualitatively similar results (Figure 1h). However, we argue that the one-parameter model is more appealing from a theoretical perspective, has an analytic solution, and does not require post-hoc averaging of the coefficients across genes.

2.3 The offset regression model is equivalent to the rank-one GLM-PCA

The offset regression model turns out to be a special case of GLM-PCA (Townes et al., 2019). There, the UMI counts are modeled as

$$X_{cg} \sim \text{Poisson}(\mu_{cg}) \text{ or } \text{NB}(\mu_{cg}, \theta_g), \tag{9}$$

$$\mu_{cg} = n_c \exp\big(\sum_{l=0}^{k} U_{cl} V_{lg}\big) = n_c \exp\big(V_{0g} + \sum_{l=1}^{k} U_{cl} V_{lg}\big), \tag{10}$$

assuming $k+1$ latent factors, with U and V playing the role of principal components and corresponding eigenvectors in standard PCA. Importantly, the first latent factor is constrained to $U_{c0} = 1$ for all cells c, such that V_{0g} can be interpreted as gene-specific intercepts. If the data are modeled without any further latent factors, Eq. (10) reduces to

$$\ln(\mu_{cg}) = V_{0g} + \ln(n_c), \tag{11}$$

which is identical to Eq. (7) with $V_{0g} = \beta_{0g}$. This shows that the proposed one-parameter offset regression model is exactly equivalent to the intercept-only rank-one GLM-PCA.

2.4 Overdispersion estimates in `scTransform` are biased

After discussing the overparametrization of the systematic component of the `scTransform` model, we now turn to the NB noise model employed by Hafemeister and Satija (2019). The $\hat{\theta}_g$ estimates in the original paper are monotonically increasing with the average gene expression, both before and after kernel smoothing (Figure 1c). This suggests that there is a biologically meaningful relationship between the expression strength and the overdispersion parameter θ_g. However, this conclusion is in fact unsupported by the data.

To demonstrate this, we simulated a dataset with NB-distributed counts $\widetilde{X}_{cg} \sim \text{NB}(\mu_{cg}, \theta = 10)$ with μ_{cg} given by Eq. 3 using X_{cg} of the PBMC dataset. Applying the original estimation procedure to this simulated dataset showed the same positive correlation of $\hat{\theta}_g$ with the average expression as in real data (Figure 1f), strongly suggesting that it does not represent an underlying technical or biological cause, but only the estimation bias. Low-expressed genes had a larger bias and only for genes with the highest average expression was the true $\theta = 10$ estimated correctly.

Moreover, the $\hat{\theta}_g$ estimates strongly depended on the exact details of the estimation procedure. Using the `theta.ml()` R function with its default 10 iterations (as Hafemeister and Satija (2019) did) led to multiple convergence warnings for the simulated data in Figure 1f. Increasing this maximum number of iterations to 100 eliminated most convergence warnings, but caused 49.9% of the estimates to diverge to infinity or above 10^{10} (Figure 1g). These instabilities are likely due to shallow maxima in the NB likelihood w.r.t. θ (Willson et al., 1986).

The above arguments show that the overdispersion parameter estimates in Hafemeister and Satija (2019) for genes with low expression were strongly biased. In practice, however, the predicted variance $\mu + \mu^2/\theta$ is only weakly affected by the exact value of θ for low expression means μ, and so the bias reported here does not substantially affect the Pearson residuals (see below). We note that large errors in NB overdispersion parameter estimates have been extensively described in other fields, with simulation studies showing that estimation bias occurs especially for low NB means, small sample sizes, and large true values of θ (Clark and Perry, 1989; Lord, 2006; Lord and Miranda-Moreno, 2008), i.e. for samples that are close to the Poisson distribution. Note also that post-hoc smoothing (Hafemeister and Satija, 2019) can reduce the variance of the $\hat{\theta}_g$ estimates, but does not reduce the bias.

2.5 Negative control datasets suggest low overdispersion

To avoid noisy and biased estimates, we suggest to use one common θ value shared between all genes. Of course, any given dataset would be better fit using gene-specific values θ_g. However, our goal is not

the best possible fit: We want the model to account only for technical variability, but not biological variability e.g. between cell types; this kind of variability should manifest itself as high residual variance.

Rather than estimating the θ value from a biologically heterogeneous dataset such as PBMC, we think it is more appropriate to estimate the technical overdispersion using negative control datasets, collected without any biological variability (Svensson, 2020). We analyzed several such datasets spanning different droplet- and plate-based sequencing protocols (10x Genomics, inDrop, MicrowellSeq) and compared the $\hat{\theta}_g$ estimates to the estimates obtained using simulated NB data with various known values of $\theta \in \{10, 100, 1000, \infty\}$. For the simulations, we used the empirically observed sample sizes and sequencing depths. We found that across different protocols, negative control data were consistent with overdispersion $\theta \approx 100$ or larger (Figure S1). The plateau at $\theta \approx 10$ in the PBMC data visible in Figure 1c could reflect biological and not technical variability. At the same time, negative control data were not consistent with the Poisson model ($\theta = \infty$), but likely overdispersion parameter values ($\theta \approx 100$) are large enough to make the Poisson model acceptable in practice.

2.6 Analytic Pearson residuals select biologically relevant genes

Both Hafemeister and Satija (2019) and Townes et al. (2019) suggested to use Pearson/deviance residuals based on models that only account for technical variability, in order to identify biologically variable genes. Indeed, genes showing biological variability should have higher variance than predicted by such a model. As explained above, Pearson residuals in the model given by Eqs. 1–2 (or, equivalently, offset regression model or rank-one GLM-PCA) can be conveniently written in closed form:

$$Z_{cg} = \frac{X_{cg} - \hat{\mu}_{cg}}{\sqrt{\hat{\mu}_{cg} + \hat{\mu}_{cg}^2/\theta}}, \quad \hat{\mu}_{cg} = \frac{\sum_j X_{cj} \cdot \sum_i X_{ig}}{\sum_{i,j} X_{ij}}, \quad \theta = 100. \tag{12}$$

For most genes in the PBMC data, the variance of the Pearson residuals was close to 1, indicating that this model predicted the variance of the data correctly and suggesting that most genes did not show biological variability (Figures 1h). Using $\theta = 100$ led to several high-expression genes selected as biologically variable that would not be selected with a lower θ (e.g. *Malat1*), but overall, using $\theta = 10$, $\theta = 100$, or even the Poisson model with $\theta = \infty$ led to only minor differences (Figure S2). Using analytical Pearson residuals for HVG selection yielded a very similar result compared to using Pearson residuals from the regularized regression presented in Hafemeister and Satija (2019), with almost the same set of genes identified as biologically variable (Figures 1h, S2). This suggests that our model is sufficient to identify biologically relevant genes.

It is instructive to compare the variance of Pearson residuals to the variance that one gets after explicit sequencing depth normalization followed by a variance-stabilizing transformation. For Poisson data, the square root transformation $\sqrt{x}$ is approximately variance-stabilizing, and several modifications exist in the literature (Bar-Lev and Enis, 1988), such as the Anscombe transformation $2\sqrt{x + 3/8}$ (Anscombe, 1948) and the Freeman-Tukey transformation $\sqrt{x} + \sqrt{x+1}$ (Freeman and Tukey, 1950). Normalizing UMI counts X_{cg} by sequencing depths n_c (and multiplying the result by the median sequencing depth $\langle n_c \rangle$ across all cells; 'median normalization') followed by one of the square-root transformations has been advocated for UMI data processing (Wagner, 2019, 2020).

Comparing the gene variances after the square-root transformation (Figure 2a) with those of Pearson residuals (Figure 2b) in the PBMC dataset showed that the square-root transformation is not sufficient for variance stabilization. Particularly affected are low-expression genes that have variance close to zero after the square-root transform (Warton, 2018). For example, platelet gene markers such as *Tubb1* have low average expression (because platelets are a rare population in the PBMC dataset) and do not show high variance after any kind of square-root transform (another example was given by the B-cell marker *Cd79a*). At the same time, Pearson residuals correctly indicate that these genes have high variance and are biologically meaningful (Figure 2c). For the genes with higher average expression, some differentially expressed genes like the monocyte marker *Lyz* or the above-mentioned *Malat1* showed high variance in both approaches. However, the selection based on the square-root transform also included high-expression genes like *Fos*, which showed noisy and biologically unspecific expression patterns (Figure 2c). Similar patterns were observed in the retinal dataset of Macosko et al. (2015) (Figure S3).

The gene with the highest average expression in the PBMC dataset, *Malat1*, showed clear signs of biologically meaningful variability: e.g. it is not expressed in platelets (Figure 2c). While this gene is selected as biologically variable based on Pearson residuals with $\theta \approx 100$ as we propose (Figure 2b), it was not selected by Hafemeister and Satija (2019) who effectively used $\theta \approx 10$ (Figures 1c,h, S2). This

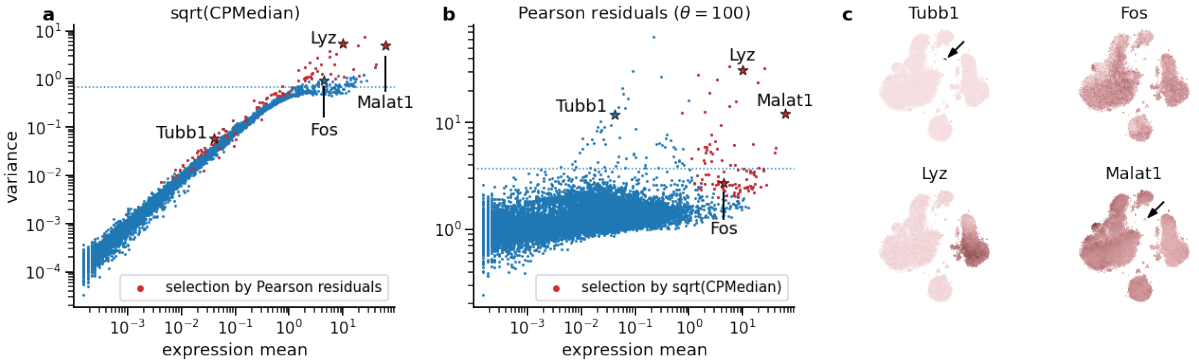

Figure 2: Selection of variable genes. In the first two panels, each dot shows the variance of a single gene in the PBMC dataset after applying a normalization method. The dotted horizontal line shows a threshold adjusted to select 100 most variable genes. Red dots mark 100 genes that are selected by the other method, i.e. that are above the threshold in the other panel. Stars indicate genes shown in the last panel. **a:** Gene variance after sequencing depth normalization, median-scaling, and the square-root transformation. **b:** Variance of Pearson residuals (assuming $\theta = 100$). **c:** t-SNE of the entire PBMC dataset (see Figure S4), colored by expression of four example genes (after sequencing depth normalization and square-root transform). Platelet marker *Tubb1* with low average expression is only selected by Pearson residuals. Arrows indicate the platelet cluster. *Fos* is only selected by the square root-based method, and does not show a clear type-specific expression pattern. *Malat1* (expressed everywhere apart from platelets) and monocyte marker *Lyz* with higher average expression are selected by both methods.

again suggests that $\theta \approx 100$ is more appropriate than $\theta \approx 10$ to model technical variability of UMI counts.

Pearson residuals may be too sensitive in that genes that are only expressed in a handful cells may get very large residual variance. Hafemeister and Satija (2019) suggested clipping residuals to $[-\sqrt{n}, \sqrt{n}]$. We found that this step avoids large residual variance in very weakly expressed genes (Figure S2 see Methods for more details). The variance of unclipped Pearson residuals under the Poisson model ($\theta = \infty$) was very similar to the Fano factor of counts after median normalization (Figure S2) and less useful for HVG selection compared to the clipped residuals.

Lastly, gene selection by the widely-used $\log(1 + x)$-transform as well as by the variance of deviance residuals as suggested by Townes et al. (2019) led to very similar results as described above for the square-root transform: many biologically meaningful genes were not selected, as all three methods overly favored high-expression genes (Figure S2). In practice, many existing HVG selection methods take the mean-variance relationship into account when performing the selection (e.g. `seurat` and `seurat_v3` methods (Satija et al., 2015; Stuart et al., 2019) as implemented in `Scanpy` (Wolf et al., 2018)). We benchmarked their performance in the next section.

2.7 Analytic Pearson residuals separate cell types better than other methods

Next, we studied the effect of different normalization approaches on PCA representations and t-SNE embeddings. The first approach is median normalization, followed by the square-root transform (Wagner, 2019, 2020). We used 50 principal components of the resulting data matrix to construct a t-SNE embedding. The second approach is computing Pearson residuals according to Eq. (12) with $\theta = 100$, followed by PCA reduction to 50 components. The third approach is computing 50 components of negative binomial GLM-PCA with $\theta = 100$ (Townes et al., 2019). We used the same initialization to construct all t-SNE embeddings to ease the visual comparison (Kobak and Berens, 2019).

We applied these methods to the full PBMC dataset (Figure S4), three retinal datasets (Macosko et al., 2015; Shekhar et al., 2016; Tran et al., 2019) (Figure 3), and a large organogenesis dataset with $n = 2$ million cells (Cao et al., 2019) (Figure 4). For smaller datasets, the resulting embeddings were mostly similar, suggesting comparable performance between methods. Hafemeister and Satija (2019) argued that using Pearson residuals reduces the amount of variance in the embedding explained by the sequencing depth variation, compared to sequencing depth normalization and log-transformation. We argue that this effect was mostly due to the large factor that the authors used for re-scaling the counts after normalization (Figure S5): large scale factors and/or small pseudocounts (ϵ in $\log(x + \epsilon)$) are known to introduce spurious variation into the distribution of normalized counts (Lun, 2018; Townes

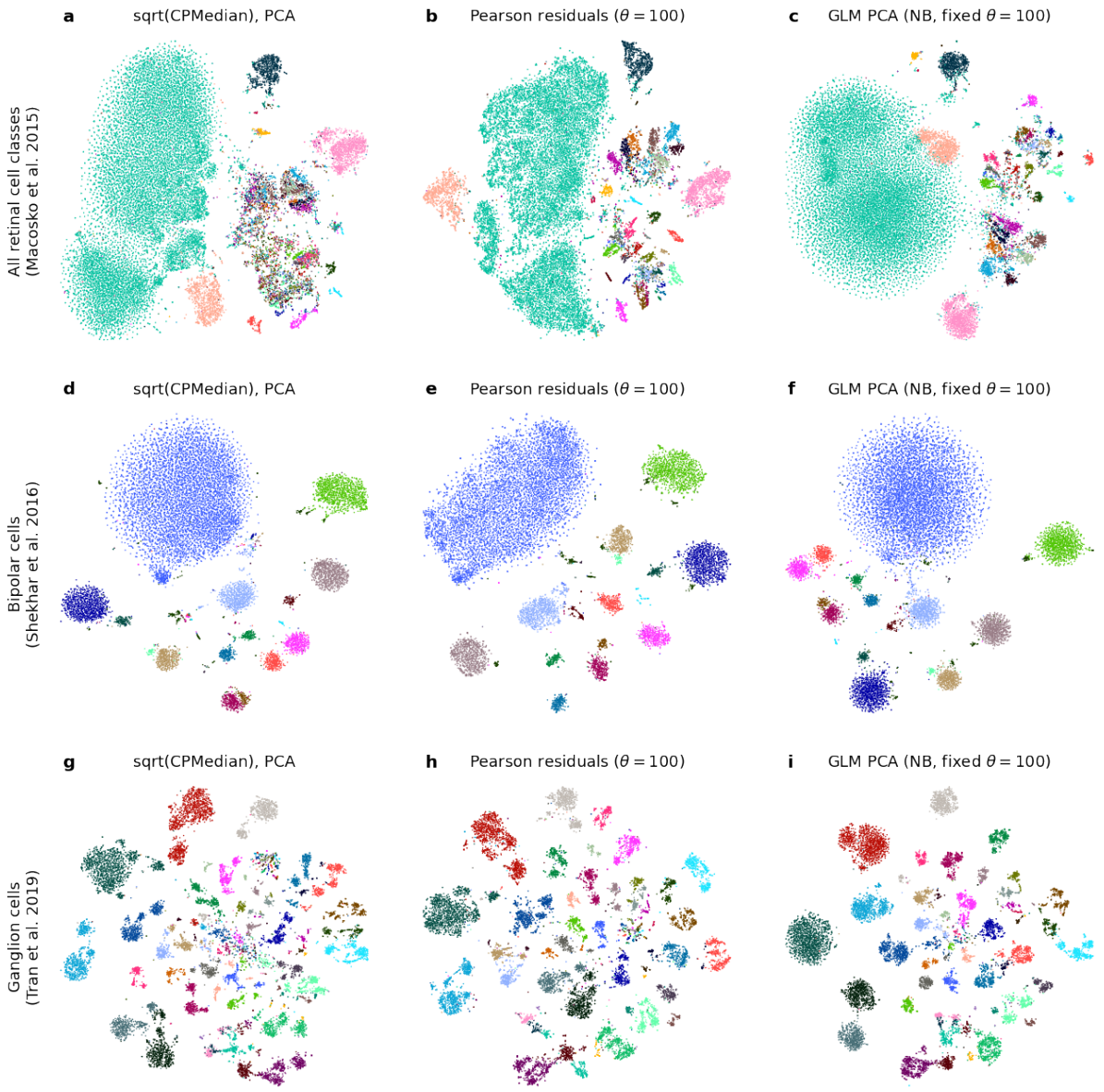

Figure 3: t-SNE embeddings of three retinal datasets. Panels in each column are based on a different data transformation method with PCA or GLM-PCA reduction to 50 dimensions (see Methods), and each row shows a different retinal dataset. We did not perform any gene selection here. Colors correspond to cell type labels provided by the original papers. **a–c:** Full-retina dataset (DropSeq) from Macosko et al. (2015), containing all retinal cell types (including glia and vascular cells). 24 769 cells. **d–f:** Bipolar cell dataset (DropSeq) from Shekhar et al. (2016). 13 987 cells. **g–i:** Retinal ganglion cell dataset (10X v2) from Tran et al. (2019). 15 750 cells.

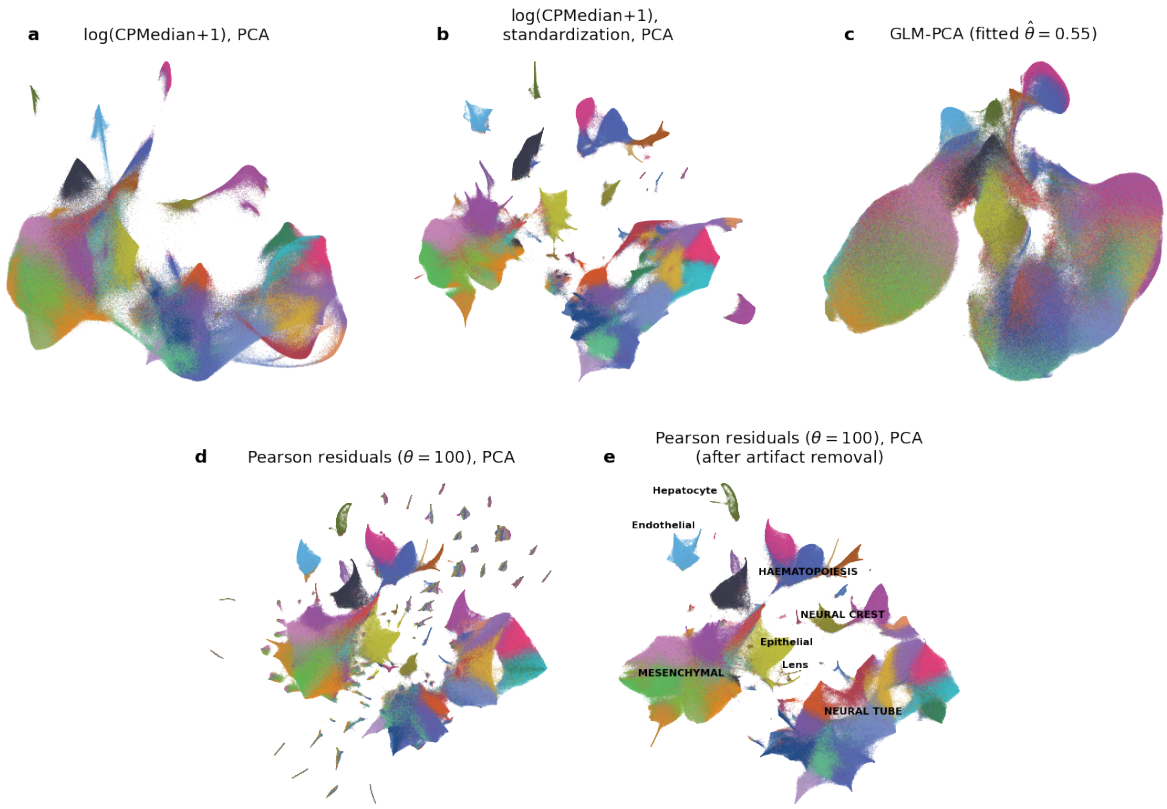

Figure 4: t-SNE embeddings of the organogenesis dataset. All panels show t-SNE embeddings of the Cao et al. (2019) dataset (2 058 652 cells), colored by the 38 main clusters identified by the original authors. All panels use 2 000 genes with the largest Pearson residual variance. Each panel shows a total of 2 026 641 cells, excluding 32 011 putative doublets identified in the original paper. All t-SNE embeddings were done with exaggeration 4 (Kobak and Berens, 2019; Böhm et al., 2020). **a:** Depth-normalization, median scaling, log-transformation and PCA with 50 principal components. **b:** Same as in (a), but with an additional standardization step that scales the normalized and log-transformed expression of each gene to mean zero and unit variance, as in Cao et al. (2019). **c:** GLM-PCA with 50 dimensions (NB model with shared overdispersion as a free parameter, estimated to be $\hat{\theta} = 0.56$). **d:** Analytic Pearson residuals with $\theta = 100$ and PCA with 50 principal components. The scattered small islands do not belong to single clusters but instead are spuriously enriched in single embryos. **e:** Same as in (d), but after removing batch-effect genes (Methods). Text labels correspond to the developmental trajectories identified by Cao et al. (2019) (uppercase: multi-cluster trajectories, lowercase: single-cluster trajectories).

et al., 2019). For the PBMC dataset, all three t-SNE embeddings showed similar amount of sequencing depth variation across the embedding space (Figure S6). Performing the embeddings on 1000 genes with the largest Pearson residual variance did not noticeably affect the embedding quality (Figure S4).

However, on closer inspection, embeddings based on Pearson residuals consistently outperformed the other two. For example, while the Pearson residual embeddings clearly separated fine cell types in the Macosko et al. (2015) dataset, the square-root embedding mixed some of them (we observed the same when using the log-transform). For the same dataset, GLM-PCA embedding did not fully separate some of the biologically distinct cell types. Furthermore, GLM-PCA embeddings often featured Gaussian-shaped blobs with no internal structure (Figure 3), suggesting that some fine manifold structure was lost, possibly due to convergence difficulties.

Embedding the organogenesis dataset (Cao et al., 2019) using Pearson residuals uncovered a strong and surprising batch artifact: hitherto unnoticed, several genes were highly expressed exclusively in small subsets of cells, with each subset coming from a single embryo. These subsets appeared as isolated islands in the t-SNE embedding (Figure 4), allowing us to uncover and remove this batch effect (Methods; Figure S7), leading to the final, biologically interpretable embedding (Figure 4). In contrast, embeddings based on log-transform or GLM-PCA did not show this batch artifact at all. GLM-PCA took days to converge (Table 1) and could recover only the coarse structure of the data. Interestingly, the final embedding based on Pearson residuals was broadly similar to the embedding obtained after log-transform and standardization of each gene, as expected given that Pearson residuals stabilize the variance by con-

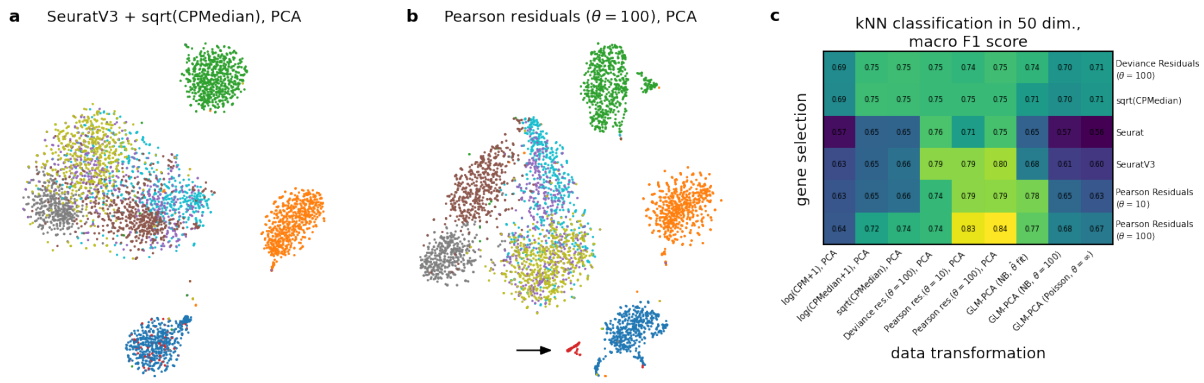

Figure 5: Benchmarking the effect of normalization on cell type separation in reduced dimensionality. We used the `Zhengmix8eq` dataset with eight ground truth FACS-sorted cell types (Zheng et al., 2017; Duò et al., 2018) (3 994 cells) and added ten pseudo-genes expressed in a random group of 50 cells from one type. All HVG selection methods were set up to select 2 000 genes, and all normalization and dimensionality reduction methods reduced the data to 50 dimensions. For details see Methods. **a:** t-SNE embedding after the `seurat_v3` HVG selection as implemented in `Scanpy`, followed by depth-normalization, median scaling, square-root transform, and PCA. Colors denote ground truth cell types, the artificially added type is shown in red. **b:** t-SNE embedding after HVG selection by Pearson residuals ($\theta = 100$), followed by transformation to Pearson residuals ($\theta = 100$), and PCA. Black arrow points at the artificially added type. **c:** Macro F1 score (harmonic mean between precision and recall, averaged across classes to counteract class imbalance) for kNN classification ($k = 15$) of nine ground truth cell types for each of the 54 combinations of HVG selection and data transformation approaches.

struction (Figure 4). Together, these qualitative observations suggest that analytical Pearson residuals can represent small, distinct subpopulations in large datasets better than other methods.

To quantify the performance of dimensionality reduction methods, we performed a systematic benchmark using the `Zhengmix8eq` dataset with known ground truth labels (Duò et al., 2018) (Figure 5). This dataset consists of PBMC cells FACS-sorted into eight different cell types before sequencing (Zheng et al., 2017), with eight types occurring in roughly equal proportions. To make the setup more challenging, we added 10 pseudo-genes expressed only in a group of 50 cells, effectively creating a ninth, rare, cell type (see Methods). We used six methods to select 2000 HVGs and nine methods for data transformation and dimensionality reduction to 50 dimensions. We assessed the resulting $6 \cdot 9 = 54$ pipelines using kNN classification of cell types. We used the macro F1 score (harmonic mean between precision and recall, averaged across classes) because this metric fairly averages classifier performance across classes of unequal size. Together, the F1 score of the kNN classifier quantifies how well each pipeline separated cell types in the 50-dimensional representation (Figure 5c). We did not include approaches that use depth normalization with inferred size factors (Lun et al., 2016) in this comparison.

The pipeline that used analytic Pearson residuals for both gene selection and data transformation outperformed all other pipelines with respect to cell type classification performance. In contrast, popular methods for HVG selection (e.g. `seurat_v3` as implemented in `Scanpy` (Wolf et al., 2018; Stuart et al., 2019)) combined with log or square-root transformations after depth normalization performed worse and in particular were often unable to separate the rare cell type (Figure 5a,b; see Figure S8 for additional embeddings). The performance of GLM-PCA was also poor, likely due to convergence issues (with 15-dimensional output spaces, GLM-PCA performed on par with Pearson residuals; data not shown), in agreement with what we reported above for the retinal datasets. Finally, deviance residuals (Townes et al., 2019) were clearly outperformed by Pearson residuals both as gene selection criterion and as data transformation. This is due to the reduced sensitivity of deviance residuals to low- or medium-expression genes (Figure S2).

2.8 Analytic Pearson residuals are fast to compute

The studied normalization pipelines differ in both space and time complexity. UMI count data are typically very sparse (e.g. in the PBMC dataset, 95% of entries are zero) and can be efficiently stored as a sparse matrix object. Sequencing depth normalization and square-root or log-transformation do not affect the zeros, preserving the sparsity of the matrix, and PCA can be run directly on a sparse matrix.

	Sqrt(CPMedian), PCA	Pearson residuals, PCA	GLM-PCA
33k PBMC	31 s	35 s	15 h
33k PBMC, 1000 HVGs	3 s	4 s	4 h
2M organogenesis, 2000 HVGs	166 s	224 s	52 h

Table 1: Runtimes for different normalization pipelines. The datasets are: the full 33k PBMC dataset, the PBMC dataset after selecting 1 000 HVGs, and the Cao et al. (2019) dataset after selecting 2 000 HVGs. Genes with largest Pearson residual variances were selected, which took 9 seconds (PBMC) and 15 minutes (organogenesis), respectively. Both Pearson residuals and GLM-PCA used fixed $\theta = 100$. All runtimes measured on a machine with 256 Gb RAM and 30 CPU threads at 2.1 GHz.

In contrast, Pearson residuals form a dense matrix without any zeros, and so can take a large amount of memory to store (4.5 Gb for the PBMC dataset). For large datasets this can become prohibitive. In contrast, GLM-PCA can run directly on a sparse matrix but takes a long time to converge (Table 1), becoming prohibitively slow for bigger datasets.

Computational complexity can be greatly reduced if gene selection is performed in advance. After selecting 1000 genes, Pearson residuals do not require a lot of memory (0.3 Gb for the PBMC dataset) and so can be conveniently used. Note that the Pearson residual variance can be computed per gene, without storing the entire residual matrix in memory. GLM-PCA, however, remained slow even after gene selection (4 h vs. 4 s for Pearson residuals for the PBMC dataset; 2 days vs. 4 m for the organogenesis dataset).

3 Conclusions

We reviewed and contrasted different methods for normalization of UMI count data. We showed that without post-hoc regularization, the negative binomial regression model of Hafemeister and Satija (2019) exhibits high variance in its parameter estimates because it is overspecified, which is why it had to be regularized in the first place. We argued that instead of regularizing an overspecified model, one should resort to a more parsimonious and theoretically motivated model specification involving an offset term. This made the model equivalent to the rank-one GLM-PCA of Townes et al. (2019) and yielded a simple analytic solution. Further, we showed that the estimates of per-gene overdispersion parameter θ_g in the original paper exhibit substantial bias and that the data are in fact consistent with a single θ value for all genes. We used negative control datasets from different experimental protocols to show that UMI counts have low overdispersion and are well described by $\theta \approx 100$ shared across genes.

We found that the approach developed by Hafemeister and Satija (2019) and implemented in the R package `scTransform` in practice yields Pearson residuals that are often similar to our analytic Pearson residuals with fixed overdispersion parameter (Figure S2). We argue that our model with its analytic solution is attractive for reasons of parsimony, theoretical simplicity, and computational speed. Moreover, it provides an explanation for the linear trends in the smoothed estimates in the original paper. Analytic residuals have already been added to `scTransform`. We are also in the process of integrating Pearson residuals into `Scanpy` (Wolf et al., 2018).

We showed that Pearson residuals outperform other methods for selecting biologically variable genes. They also are better than other methods for downstream processing: In a systematic benchmarking effort, we demonstrated that Pearson residuals provide a good basis for general-purpose dimensionality reduction and for constructing 2D embeddings of single-cell UMI data. In particular, they are well suited for identifying rare cell types and their genetic markers. Applying gene selection prior to dimensionality reduction reduces the computational cost of using Pearson residuals down to negligible. In contrast, GLM-PCA model can take over 100–1000 times longer to fit. We conclude that analytic Pearson residuals is a theory-based, fast, and convenient method for normalization of UMI datasets.

4 Methods

4.1 Mathematical details

Analytic solution

The log-likelihood for the model defined in Eqs. 1–2

$$X_{cg} \sim \text{Poisson}(n_c p_g) \tag{13}$$

can be, up to a constant, written as

$$\mathcal{L} = \sum_{cg} \Big[X_{cg} \ln(n_c p_g) - n_c p_g \Big], \tag{14}$$

where we used the Poisson density $p(x) = e^x e^{-\mu}/x!$. Taking partial derivatives with respect to n_c and p_g and setting them to zero, one obtains

$$\hat{n}_c = \frac{\sum_g X_{cg}}{\sum_g \hat{p}_g}, \quad \hat{p}_g = \frac{\sum_c X_{cg}}{\sum_c \hat{n}_c}. \tag{15}$$

This is a family of solutions. Setting $\sum_g \hat{p}_g = 1$, we obtain Eq. 3 and the formulas for $\hat{n}_c$ and $\hat{p}_g$ given in Section 2.1.

This derivation does not generalize for the negative binomial model with density

$$p(x) = \frac{\Gamma(x+\theta)}{x!\,\Gamma(\theta)} \Big(\frac{\theta}{\theta+\mu}\Big)^{\theta} \Big(\frac{\mu}{\theta+\mu}\Big)^{x}, \tag{16}$$

where the log-likelihood (for fixed θ), up to a constant, is

$$\mathcal{L} = \sum_{cg} \Big[X_{cg} \ln(n_c p_g) - (X_{cg} + \theta) \ln(n_c p_g + \theta) \Big]. \tag{17}$$

This does not have an analytic maximum likelihood solution. However, for large θ values Eq. 3 can be taken as an approximate solution.

Deviance residuals

Deviance is defined as the doubled difference between the log-likelihood of the saturated model and the log-likelihood of the actual model. The saturated model, in our case, is a full rank model with $\hat{\mu}^*_{cg} = X_{cg}$. For the Poisson model, the deviance can therefore be obtained from Eq. 14 and is equal to

$$\mathcal{D} = 2 \sum_{cg} \Big[X_{cg} \ln \frac{X_{cg}}{\hat{\mu}_{cg}} - (X_{cg} - \hat{\mu}_{cg}) \Big], \tag{18}$$

where the terms with $\hat{\mu}_{cg} = X_{cg}$ are taken to be zero.

Deviance residuals are defined as square roots of the respective deviance terms, such that the sum of squared deviance residuals is equal to the deviance (note that for the Gaussian case this already holds true for the raw residuals, because the saturated model has zero log-likelihood, and the deviance is simply the squared error). It follows that for the Poisson model deviance residuals (Townes et al., 2019) are given by

$$Z_{cg} = \text{sign}(X_{cg} - \hat{\mu}_{cg}) \sqrt{2 \Big[X_{cg} \ln \frac{X_{cg}}{\hat{\mu}_{cg}} - (X_{cg} - \hat{\mu}_{cg}) \Big]} \tag{19}$$

Similarly, for the negative binomial model with fixed θ, the deviance residuals follow from Eq. 17 and are given by

$$Z_{cg} = \text{sign}(X_{cg} - \hat{\mu}_{cg}) \sqrt{2 \Big[X_{cg} \ln \frac{X_{cg}}{\hat{\mu}_{cg}} - (X_{cg} + \theta) \ln \frac{X_{cg} + \theta}{\hat{\mu}_{cg} + \theta} \Big]} \tag{20}$$

It is easy to verify that this formula reduces to the Poisson case when $\theta \to \infty$. When computing deviance residuals, we estimated $\hat{\mu}_{cg}$ using Eq. 3.

dataset	protocol	cells	genes	species	description
33k PBMC	10X v1	33 148	16 809	human	peripheral blood mononuclear cells
Zheng et al. (2017)	10X v1	3 994	15 715	human	FACS-sorted PBMC cells
Svensson et al. (2017)	10X v2	2 000	13 025	human	droplets of bulk RNA solution
Klein et al. (2015)	inDrop	953	25 025	human	droplets of bulk RNA solution
Han et al. (2018)	MicrowellSeq	9 994	15 069	mouse	non-differentiating stem cells
Macosko et al. (2015)	DropSeq	24 769	17 973	mouse	retinal cells
Shekhar et al. (2016)	DropSeq	13 987	16 520	mouse	retinal bipolar cells
Tran et al. (2019)	10X v2	15 750	17 685	mouse	retinal ganglion cells
Cao et al. (2019)	sci-RNA-seq3	2 058 652	26 183	mouse	mouse embryo cells

Table 2: Overview of UMI datasets used for analysis. In the Svensson et al. (2017) dataset, we used only sample 1. In the Han et al. (2018) dataset, we used the E14 dataset. In the three retinal datasets, we only used cells from the largest batch. The FACS-sorted PBMC dataset was assembled by Duò et al. (2018) using the data from Zheng et al. (2017). Numbers of genes and cells are after batch selection (where applicable) and initial gene filtering (see Methods).

Clipping Pearson residuals

Clipping Pearson residuals to $\pm\sqrt{n}$ as suggested by Hafemeister and Satija (2019) is needed to avoid large residual variance in rarely expressed genes (Figure S2d). The intuition behind this heuristic is as follows. Consider a UMI dataset with n cells containing a biologically distinct rare population P of size $m \ll n$. Let this population have a marker gene with expression following Poisson(λ) for the cells from P, and zero expression for all $n-m$ remaining cells. For simplicity we assume the Poisson model here, and further assume that all cells have the same sequencing depth.

The expected average expression of this gene is $\lambda m/n$ and so the expected Pearson residual value for this gene for the cells from P is $(\lambda - \lambda m/n)/\sqrt{\lambda m/n} = (n-m)\sqrt{\lambda/(nm)} \approx \sqrt{\lambda n/m}$. With the clipping threshold $\sqrt{n}$, clipping will happen whenever $\lambda > m$, i.e. when the population P is either very small or has very large UMI counts. For example, a population of 10 cells having a marker gene with the within-population mean expression of 20 UMIs, will result in clipped residuals, as if the within-population mean expression were $\sim$10 UMIs.

4.2 Experimental details

Analyzed datasets and preprocessing

Used datasets are listed in Table 2. For the organogenesis dataset and the FACS-sorted PBMC dataset, we applied no further filtering. In all remaining datasets we excluded genes that were expressed in fewer than 5 cells, following Hafemeister and Satija (2019). The data were downloaded following links in original publications in form of UMI count tables. Direct links to all data sources are given in our Github repository `https://github.com/berenslab/umi-normalization`.

Data transformation and dimensionality reduction methods

We used the following abbreviations to denote data transformations: `sqrt(CPMedian)` — normalization by sequencing depth, followed by scaling by the median depth across all cells ('counts per median'), followed by the square-root transform; `log(CPMedian + 1)` — normalization by sequencing depth, followed by scaling by the median depth across all cells, followed by $\log(x+1)$ transform; `log(CPM + 1)` — normalization by sequencing depth, followed by scaling by one million ('counts per million'), followed by $\log(x+1)$ transform. Pearson residuals were computed with Eq. 12 and then clipped to $\pm\sqrt{n}$. Deviance residuals were computed with Eq. 20.

All of these methods were typically followed by dimensionality reduction by PCA to 50 dimensions using the `Scanpy` implementation (Wolf et al., 2018), unless otherwise stated.

Further, we used three variants of GLM-PCA to transform raw counts and reduce dimensionality down to 50 in a joint step: Poisson GLM-PCA, negative binomial GLM-PCA with estimation of single overdispersion parameter θ shared across genes, and negative binomial GLM-PCA with fixed shared θ. In Townes et al. (2019), the authors only used the former two methods. Whenever possible, we used the `glmpca-py` implementation. On the organogenesis dataset, the Python implementation did not converge within reasonable time, so for this dataset, we resorted to the R implementation. It uses a

different optimization method and employs stochastic minibatches. All reported GLM-PCA results for this dataset are for batchsize 10 000 as batchsizes 100 and 1 000 (default) resulted in considerably longer runtimes. Because the R implementation does not support NB GLM-PCA with fixed theta, for this dataset we used GLM-PCA with jointly fit $\hat{\theta}$.

Unless otherwise stated, all residuals and GLM-PCA with fixed θ used $\theta = 100$.

HVG selection methods

For gene selection using `sqrt(CPMedian)`, Pearson residuals, and deviance residuals, we applied the respective data transformation and used the variance after transformation as selection criterion. For `Seurat` and `Seurat_v3` methods, we used the respective `Scanpy` implementations. In brief, these two methods regress out the mean-variance relationship, and return an estimate of the 'excess' variance for each gene (Satija et al., 2015; Stuart et al., 2019). For `scTransform` we used the corresponding R package Hafemeister and Satija (2019). The Fano factor was computed after normalizing by sequencing depth and scaling by median sequencing depth.

Benchmarking cell type separation with kNN classification

We used the `Zhengmix8eq` dataset with known ground truth labels obtained by FACS-sorting of eight PBMC cell types (Zheng et al., 2017; Duò et al., 2018). There were 400–600 cells in each cell type. We created a ninth, artificial population from 50 randomly selected B-cells (marked blue in Figure 5). To mimic a separate cell type, we added 10 pseudo marker that had zero expression everywhere apart from those 50 cells. For those 50 cells, UMI values were simulated as Poisson($n_i p$), where n_i is the sequencing depth of the i-th selected cell (range: 452–9697), and expression fraction p was set to 0.001.

We then applied the 54 normalization pipelines shown in Figure 5 to this dataset. Each pipeline used one of the six methods to select 2000 HVGs, followed by one of the nine methods for data transformation and dimensionality reduction to 50 dimensions. To assess cell type separation in this output space, we used a kNN classifier with an implicit leave-one-out procedure. This resulted in a class prediction for each cell based on the majority vote of its $k = 15$ neighboring cells. We quantified the performance of this prediction by computing the macro F1 score (harmonic mean between precision and recall, averaged across classes to counteract class imbalance). We used the `sklearn` implementations for kNN classification and the F1 score (Pedregosa et al., 2011).

Measuring runtimes

All runtimes given in Table 1 are wall times from running the code in a Docker container with an Ubuntu 18 system on a machine with 256 GB RAM and 2×24 CPUs at 2.1 Ghz (Xeon Silver 4116 Dodecacore). The Docker container was restricted to use at most 30 CPU threads. To reduce overhead, we did not use `Scanpy` for timing experiments, and instead used `numpy` for basic computations and `sklearn` for PCA with default settings.

t-SNE embeddings

All t-SNE embeddings were made following recommendations in Kobak and Berens (2019) using FIt-SNE implementation (Linderman et al., 2019). We used the PCA (or, when applicable, GLM-PCA) representation of the data as input. We used default FIt-SNE parameters, including automatically chosen learning rate. For initialization, we used the first two principal components of the data, scaled such that PC1 had standard deviation 0.0001 (as is default in FIt-SNE). The initialization was shared among all embeddings shown in the same figure, i.e. PCs of one data representation were used to initialize all other embeddings as well. For all datasets apart from the organogenesis one, we used perplexity combination of 30 and $n/100$, where n is the sample size (Kobak and Berens, 2019). For the organogenesis dataset embeddings we used exaggeration 4 and perplexity 30.

Acknowledgements We thank Christoph Hafemeister, Rahul Satija, William Townes, and Florian Wagner for discussions and helpful comments, and the `Scanpy` team for support.

This work was supported by the Deutsche Forschungsgemeinschaft (BE5601/4-1, BE5601/6-1, and EXC 2064 ML, project number 390727645), by the German Ministry of Education and Research (01GQ1601 and 01IS18039A), and by the National Institute Of Mental Health of the National Institutes of Health under Award Number U19MH114830. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

Competing interests The authors declare that they have no competing interests.

Availability of data and materials All used datasets are publicly available. The Python code to reproduce the analysis of this paper is available at http://github.com/berenslab/umi-normalization, including download instructions for the used datasets.

References

Alan Agresti. *Foundations of linear and generalized linear models*. John Wiley & Sons, 2015.

Robert A Amezquita, Aaron TL Lun, Etienne Becht, Vince J Carey, Lindsay N Carpp, Ludwig Geistlinger, Federico Marini, Kevin Rue-Albrecht, Davide Risso, Charlotte Soneson, et al. Orchestrating single-cell analysis with Bioconductor. *Nature Methods*, 17(2):137–145, 2020.

Francis J Anscombe. The transformation of Poisson, binomial and negative-binomial data. *Biometrika*, 35(3/4):246–254, 1948.

Shaul K Bar-Lev and Peter Enis. On the classical choice of variance stabilizing transformations and an application for a Poisson variate. *Biometrika*, 75(4):803–804, 1988.

Jan Niklas Böhm, Philipp Berens, and Dmitry Kobak. A unifying perspective on neighbor embeddings along the attraction-repulsion spectrum. *arXiv preprint arXiv:2007.08902*, 2020.

Junyue Cao, Malte Spielmann, Xiaojie Qiu, Xingfan Huang, Daniel M Ibrahim, Andrew J Hill, Fan Zhang, Stefan Mundlos, Lena Christiansen, Frank J Steemers, et al. The single-cell transcriptional landscape of mammalian organogenesis. *Nature*, 566(7745):496–502, 2019.

Suzanne J Clark and Joe N Perry. Estimation of the negative binomial parameter κ by maximum quasi-likelihood. *Biometrics*, pages 309–316, 1989.

Angelo Duò, Mark D Robinson, and Charlotte Soneson. A systematic performance evaluation of clustering methods for single-cell RNA-seq data. *F1000Research*, 7, 2018.

Nils Eling, Arianne C Richard, Sylvia Richardson, John C Marioni, and Catalina A Vallejos. Correcting the mean-variance dependency for differential variability testing using single-cell RNA sequencing data. *Cell Systems*, 7(3):284–294, 2018.

Murray F Freeman and John W Tukey. Transformations related to the angular and the square root. *The Annals of Mathematical Statistics*, pages 607–611, 1950.

Christoph Hafemeister and Rahul Satija. Normalization and variance stabilization of single-cell RNA-seq data using regularized negative binomial regression. *Genome Biology*, 20:296, 2019.

Xiaoping Han, Renying Wang, Yincong Zhou, Lijiang Fei, Huiyu Sun, Shujing Lai, Assieh Saadatpour, Ziming Zhou, Haide Chen, Fang Ye, et al. Mapping the mouse cell atlas by microwell-seq. *Cell*, 172 (5):1091–1107, 2018.

Allon M Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A Weitz, and Marc W Kirschner. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, 2015.

Dmitry Kobak and Philipp Berens. The art of using t-sne for single-cell transcriptomics. *Nature Communications*, 10(1):1–14, 2019.

George C Linderman, Manas Rachh, Jeremy G Hoskins, Stefan Steinerberger, and Yuval Kluger. Fast interpolation-based t-sne for improved visualization of single-cell rna-seq data. *Nature methods*, 16(3): 243–245, 2019.

Romain Lopez, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058, 2018.

Dominique Lord. Modeling motor vehicle crashes using Poisson-gamma models: Examining the effects of low sample mean values and small sample size on the estimation of the fixed dispersion parameter. *Accident Analysis & Prevention*, 38(4):751–766, 2006.

Dominique Lord and Luis F Miranda-Moreno. Effects of low sample mean values and small sample size on the estimation of the fixed dispersion parameter of Poisson-gamma models for modeling motor vehicle crashes: A bayesian perspective. *Safety Science*, 46(5):751–770, 2008.

Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12):1–21, 2014.

Malte D Luecken and Fabian J Theis. Current best practices in single-cell RNA-seq analysis: a tutorial. *Molecular Systems Biology*, 15(6), 2019.

Aaron Lun. Overcoming systematic errors caused by log-transformation of normalized single-cell rna sequencing data. *BioRxiv*, page 404962, 2018.

Aaron TL Lun, Karsten Bach, and John C Marioni. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biology*, 17(1):1–14, 2016.

Evan Z Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R Bialas, Nolan Kamitaki, Emily M Martersteck, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, 2015.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Rahul Satija, Jeffrey A Farrell, David Gennert, Alexander F Schier, and Aviv Regev. Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology*, 33(5):495–502, 2015.

Karthik Shekhar, Sylvain W Lapan, Irene E Whitney, Nicholas M Tran, Evan Z Macosko, Monika Kowalczyk, Xian Adiconis, Joshua Z Levin, James Nemesh, Melissa Goldman, et al. Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics. *Cell*, 166(5):1308–1323, 2016.

Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M Mauck III, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *Cell*, 177(7):1888–1902, 2019.

Valentine Svensson. Droplet scrna-seq is not zero-inflated. *Nature Biotechnology*, 38(2):147–150, 2020.

Valentine Svensson, Kedar Nath Natarajan, Lam-Ha Ly, Ricardo J Miragaia, Charlotte Labalette, Iain C Macaulay, Ana Cvejic, and Sarah A Teichmann. Power analysis of single-cell rna-sequencing experiments. *Nature Methods*, 14(4):381–387, 2017.

Valentine Svensson, Adam Gayoso, Nir Yosef, and Lior Pachter. Interpretable factor models of single-cell RNA-seq via variational autoencoders. *Bioinformatics*, 2020.

F William Townes, Stephanie C Hicks, Martin J Aryee, and Rafael A Irizarry. Feature selection and dimension reduction for single-cell RNA-seq based on a multinomial model. *Genome Biology*, 20:295, 2019.

Nicholas M Tran, Karthik Shekhar, Irene E Whitney, Anne Jacobi, Inbal Benhar, Guosong Hong, Wenjun Yan, Xian Adiconis, McKinzie E Arnold, Jung Min Lee, et al. Single-cell profiles of retinal ganglion cells differing in resilience to injury reveal neuroprotective genes. *Neuron*, 104(6):1039–1055, 2019.

Florian Wagner. Straightforward clustering of single-cell RNA-Seq data with t-SNE and DBSCAN. *BioRxiv*, page 770388, 2019.

Florian Wagner. Monet: An open-source Python package for analyzing and integrating scRNA-Seq data using PCA-based latent spaces. *bioRxiv*, 2020.

David I Warton. Why you cannot transform your way out of trouble for small counts. *Biometrics*, 74 (1):362–368, 2018.

LJ Willson, JL Folks, and JH Young. Complete sufficiency and maximum likelihood estimation for the two-parameter negative binomial distribution. *Metrika*, 33(1):349–362, 1986.

F Alexander Wolf, Philipp Angerer, and Fabian J Theis. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):1–5, 2018.

Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8(1):1–12, 2017.

Supplementary Figures

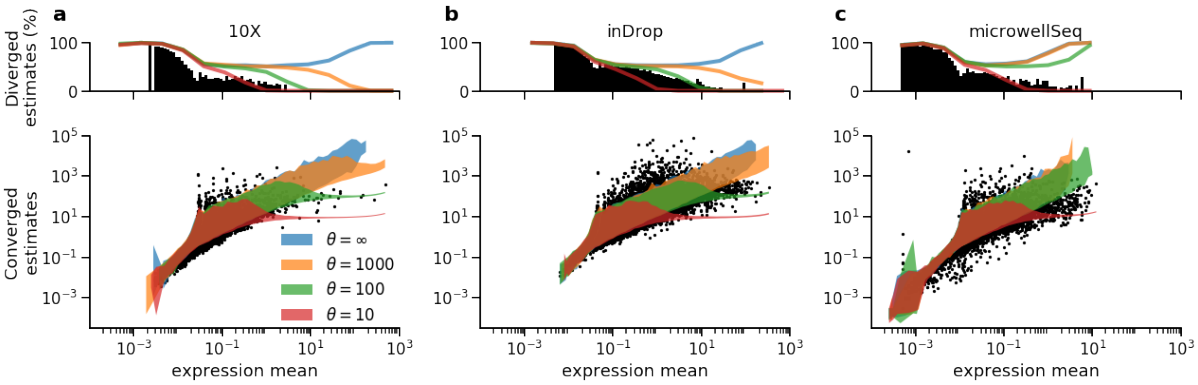

Figure S1: Overdispersion in negative control datasets. In the lower panels, each dot corresponds to one gene g. The overdispersion parameter estimates were obtained with `theta.ml()` using 100 iterations, using the expression means predicted by our model. Diverged estimates were clipped as in Figure 1g. Colors show the 5th to 95th percentile regions of the overdispersion estimates for simulated datasets with a known θ shared across genes. For each value of $\theta \in \{10, 100, 1000, \infty\}$, we simulated 1000 datasets according to Eq. (5), using means $\mu_{cg} = p_g n_c$ with empirical n_c and a series of p_g values chosen to cover the entire data range. The upper panels show the fraction of genes for which the estimate diverged (black: original data, colors: simulated data). **a:** 10x Genomics Chromium technical negative control dataset (sample 1), consisting of RNA solution split into droplets (Svensson et al., 2017). 2 000 cells. **b:** inDrop technical negative control dataset prepared as in (a) (Klein et al., 2015). 953 cells. **c:** microwellSeq biological negative control dataset from non-differentiating embryonic stem cell culture (E14 line from strain 129P2/Ola) (Han et al., 2018). 9 994 cells.

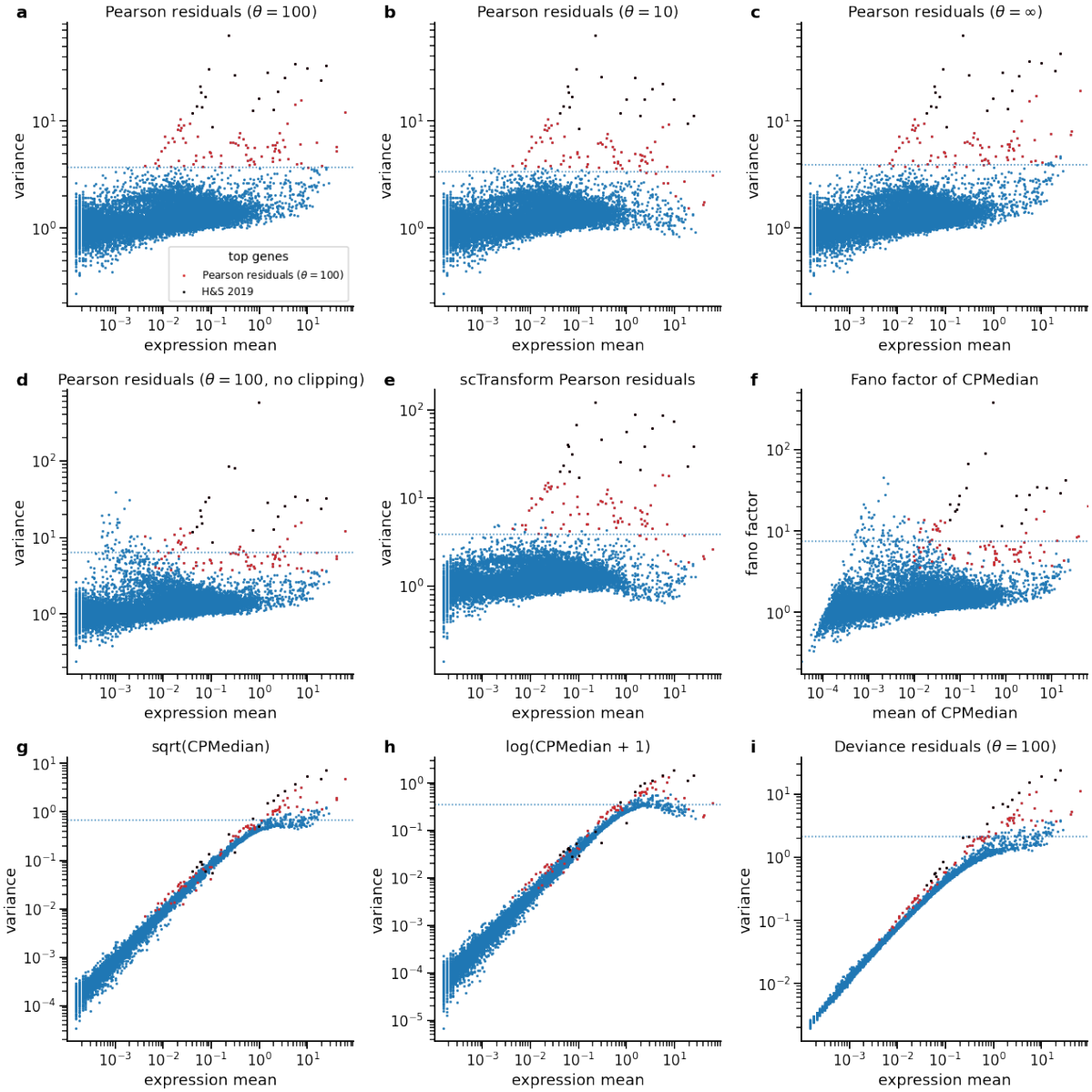

Figure S2: Comparing gene selection criteria. Each dot shows the selection criterion for a single gene in the PBMC dataset after applying a normalization method. The 100 genes with the largest criterion values in each panel lie above the dotted horizontal line. Red dots show the top 100 genes selected by the variance of Pearson residuals with $\theta = 100$. Black dots show the top 20 genes selected by `scTransform` (Hafemeister and Satija, 2019). **a:** Reproduced from Figure 2b. **b:** Analytic Pearson residuals with $\theta = 10$, roughly corresponding to `scTransform`. **c:** Analytic Pearson residuals with $\theta = \infty$, i.e. a Poisson model. **d:** Analytic Pearson residuals with $\theta = 100$ but without clipping of the residuals to $\pm\sqrt{n}$. This leads to selection of some low-expression genes. **e:** Pearson residuals of the Hafemeister and Satija (2019) model, obtained via the R implementation of `scTransform`. 87% of the top-100, and 83% of the top-1000 genes are the same as in the selection by analytic Pearson residuals with $\theta = 100$. **e:** Fano factor after sequencing depth normalization and median scaling. **g:** Variance after sequencing depth normalization, median-scaling and square-root transform. **h:** Variance after sequencing depth normalization, median-scaling and $\log(1 + x)$-transform. Note that here the variance is a non-monotonic function of the average expression. Note also that some biologically variable genes (black and red dots) in both (g) and (h) lie below the main cloud, i.e. have *lower* variance than the genes with the same average expression but without biological variability. This happens for genes that are strongly expressed but only in a small subset of cells. Square-root and log-transformations act 'too strong' on these large counts, yielding lower overall variance. **i:** Variance of deviance residuals for $\theta = 100$. Consistent with Supplementary Figure S7 in Townes et al. (2019), only high expression genes are selected.

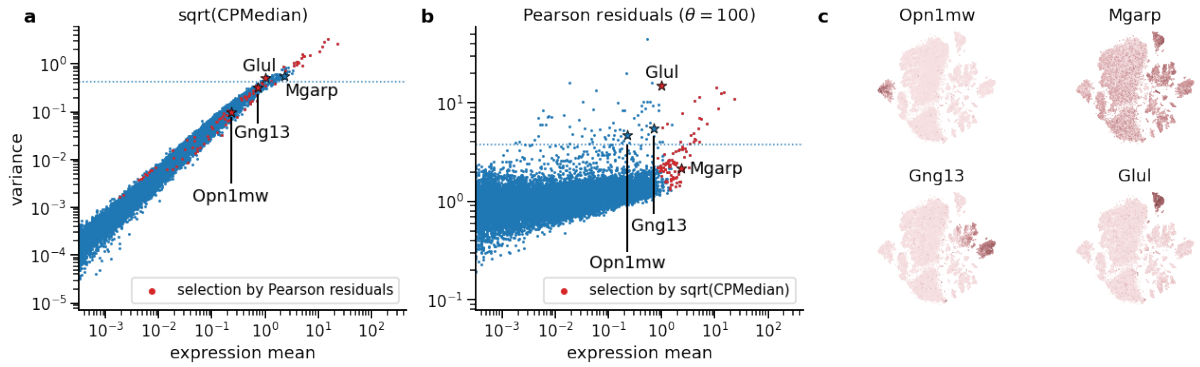

Figure S3: Selection of variable genes for a retinal dataset. All panels as in Figure 2, here showing gene selection for the largest batch of the retina dataset from Macosko et al. (2015). It consists of replicates `p1` and `r4`–`r6`, with a total of $N = 24\,769$ cells. The cone photoreceptor marker *Opn1mv* and the bipolar cell marker *Gng13* were only selected when the variance of Pearson residuals was used as criterion. The diffusely expressed gene *Mgarp* is present in many cell types, and was only selected based on the variance of the square-root transformed data. The Müller glia marker *Glul* was selected by both methods.

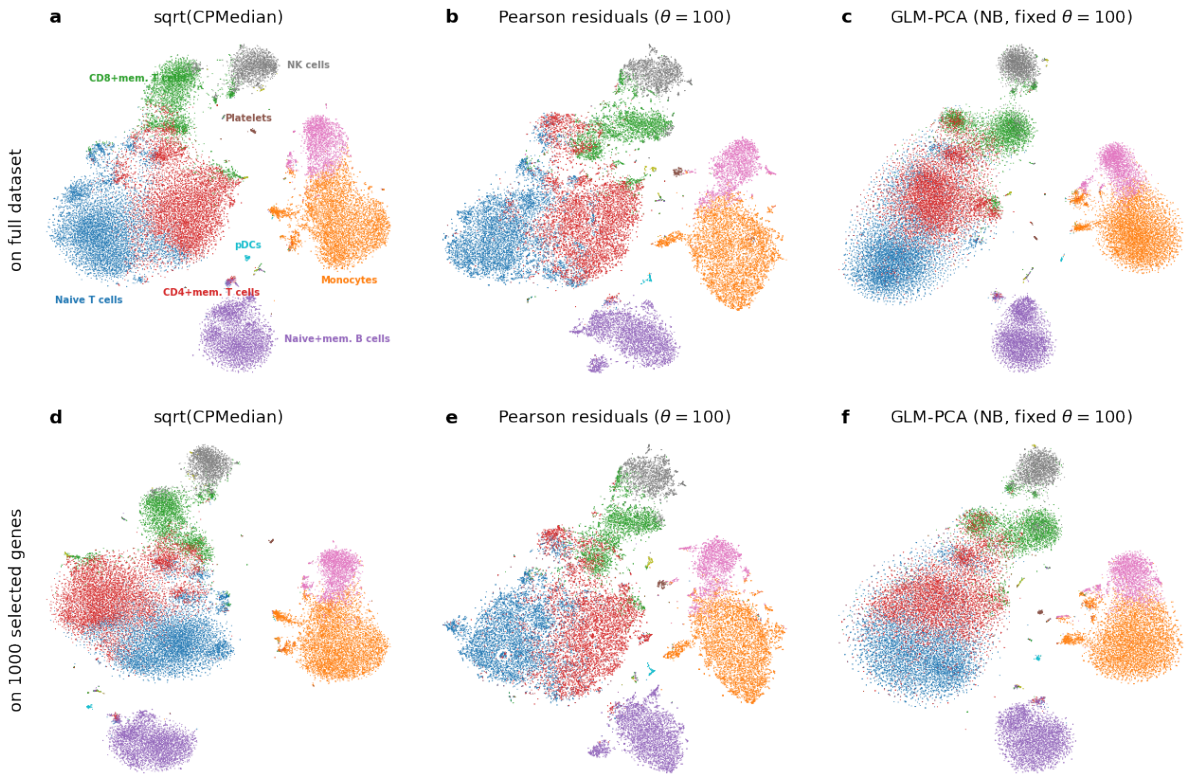

Figure S4: t-SNE embeddings based on different normalization approaches. Each panel shows a t-SNE of the PBMC dataset, based on a different normalization method. Colors correspond to 10 k-means clusters provided by 10x Genomics together with the PBMC dataset. We annotated eight clusters based on known marker genes (taken from Wagner (2019)). **a:** The UMI counts after sequencing depth normalization and square-root transform, reduced to 50 principal components. **b:** Pearson residuals assuming $\theta = 100$, reduced to 50 principal components. **c:** 50 components of negative binomial GLM-PCA with $\theta = 100$ (Townes et al., 2019). **d–f:** Same as a–c, but now we reduced the dimensionality of the dataset by keeping only the 1000 genes with the highest Pearson residual variance. For shared initialization, we used the first two principal components of the square-root transformed data matrix after gene selection. GLM-PCA on the reduced dataset did not converge with default penalty 1, so we increased it to 5, following the automatic tuning procedure used in the authors' `R` implementation.

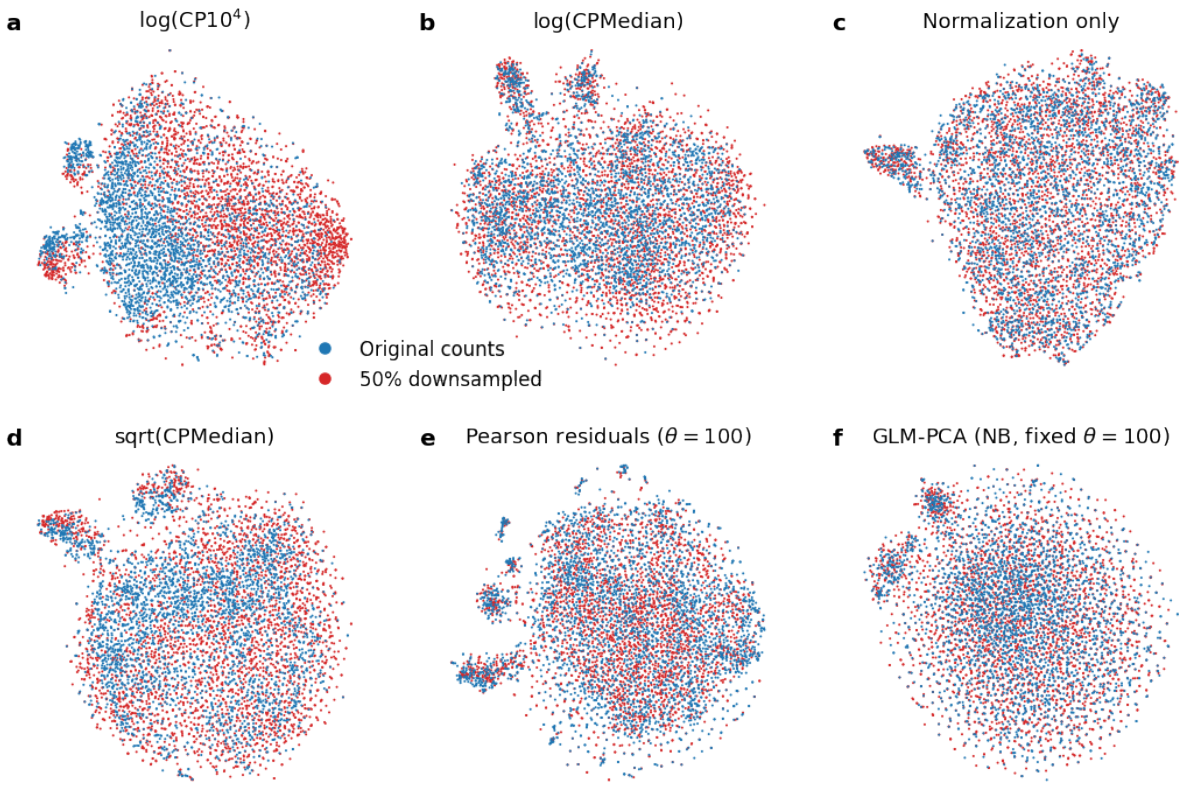

Figure S5: t-SNE embeddings of the monocyte cluster with downsampling. We analyzed the monocyte cluster from the PBMC dataset (marked orange in Figure S4, $n = 6\,307$ cells). Half of the cells (randomly selected) were downsampled to 50% of their original sequencing depth, by simulating new counts $\tilde{X}_{cg} = \text{Binomial}(n = X_{cg}, p = 0.5)$, following the analysis in Hafemeister and Satija (2019) (Figure 6C). We then compared how well different normalization strategies remove the simulated batch-effect. PCA and t-SNE were applied after normalization as in Figure S4. **a:** Depth-normalized counts, scaled with 10^4 and then log-transformed, as in Hafemeister and Satija (2019). Note the strong batch effect due to the large scale. **b:** Depth-normalized counts, scaled with median depth (≈ 1500) and then log-transformed. Note the smaller scale led to a much weaker batch effect compared to (a). **c:** Depth-normalized counts. At the cost of not applying a variance-stabilizing transform, no batch effect remains. **d:** Depth-normalized counts, square-root transformed. A weak batch effect remains. **e:** Pearson residuals, computed with $\theta = 100$. **f:** Negative binomial GLM PCA, computed with fixed $\theta = 100$.

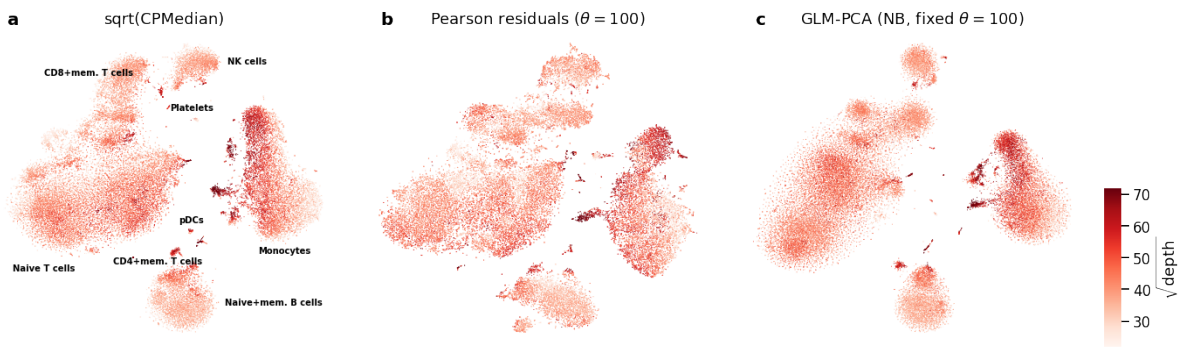

Figure S6: Sequencing depth in t-SNE embeddings based on different normalization approaches. All panels are as in Figure S4 a–c, but overlayed with square-root-transformed sequencing depth. Values above 70 were clipped to 70.

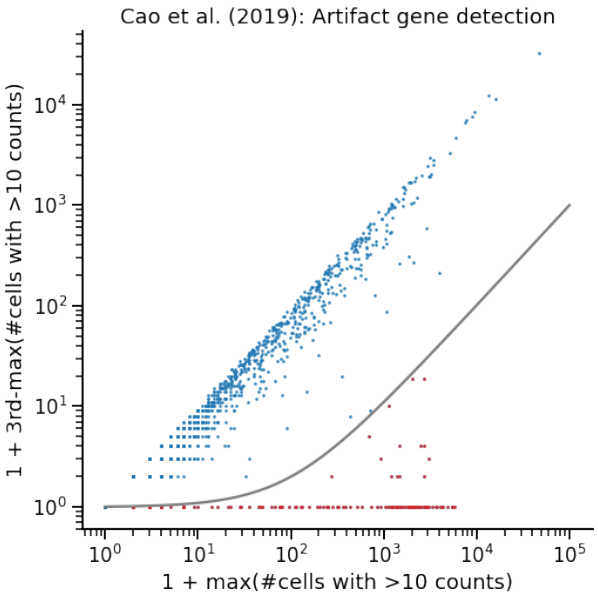

Figure S7: Artifact genes in the organogenesis dataset. Scatter plot over the 2 000 highly variable genes that we selected using Pearson residuals. For each gene and each of the 61 embryos in the data, we computed how many cells in that embryo have over 10 UMI counts of that gene. The plot compares the largest number across embryos with the third largest number. For the majority of the genes, these values were very similar, and therefore lie on the diagonal of the plot. In contrast, genes with spurious enrichment in one or two embryos lie below the diagonal. The gray line shows our exclusion criterion: 100 times difference between the largest and the third-largest number of cells with >10 UMI counts. The red dots mark 249 genes excluded for Figure 4e.

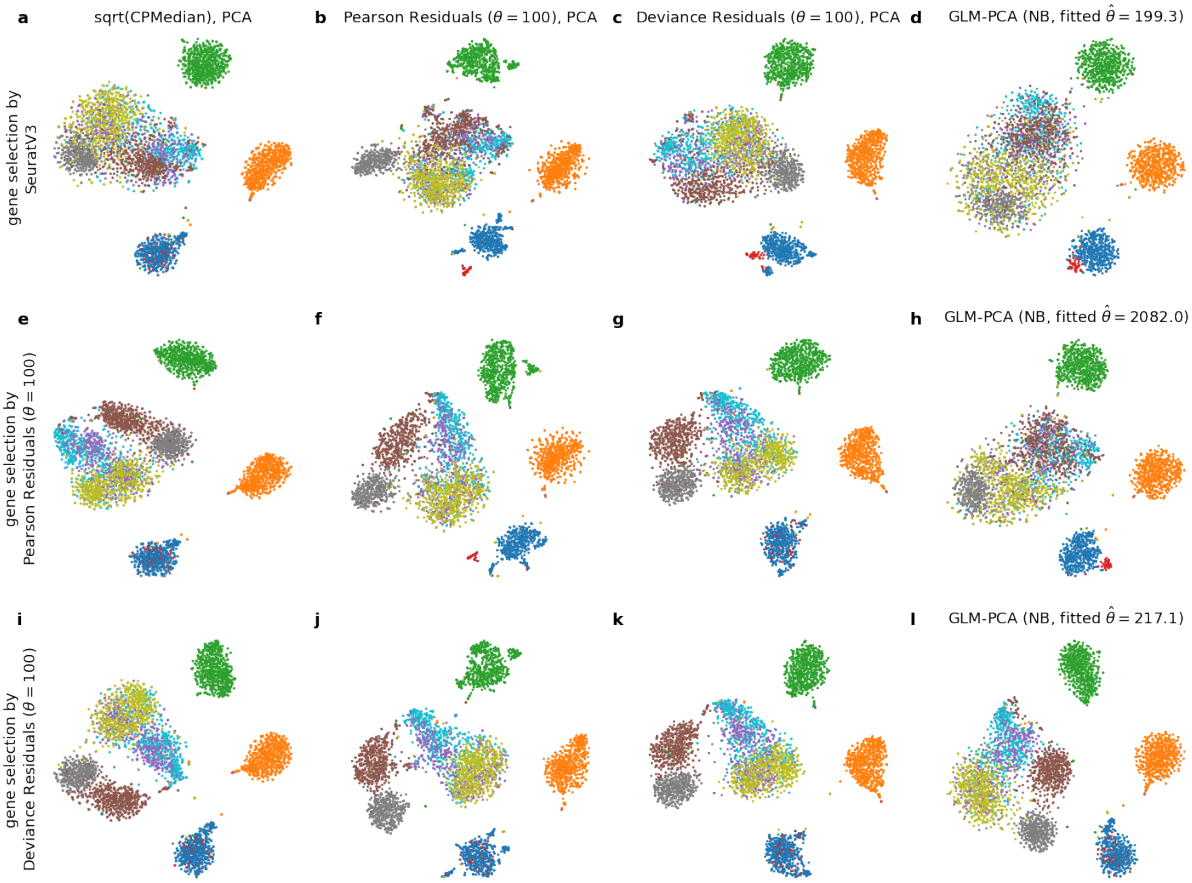

Figure S8: t-SNE embeddings of the benchmarking dataset for selected normalization pipelines. Rows show three HVG selection approaches, columns show four normalization and dimensionality reduction approaches. Each panel corresponds to one of the entries in Figure 5c.