RESEARCH ARTICLE

# Deterministic column subset selection for single-cell RNA-Seq

**Shannon R. McCurdy**[1]*, **Vasilis Ntranos**[2], **Lior Pachter**[3]

**1** California Institute for Quantitative Biosciences, University of California Berkeley, Berkeley, California, United States of America, **2** Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, California, United States of America, **3** Division of Biology and Biological Engineering, Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, California, United States of America

* smccurdy@berkeley.edu

**Citation:** McCurdy SR, Ntranos V, Pachter L (2019) Deterministic column subset selection for single-cell RNA-Seq. PLoS ONE 14(1): e0210571. https://doi.org/10.1371/journal.pone.0210571

**Editor:** Peng Yu, Texas A&M University, UNITED STATES

**Received:** March 27, 2018

**Accepted:** December 26, 2018

**Published:** January 25, 2019

**Data Availability Statement:** All the single-cell gene expression files are available from the NCBI Sequence Read Archive (mouse brain: accession number SRA SRP045452, mouse bone marrow: accession number SRA SRP063520). The Python package containing code to perform the methods described in the article can be found at https://github.com/srmcc/dcss_single_cell.git. The package also contains code to download the datasets used as examples in the article" in your manuscript.

**Funding:** SRM is funded by Award Number F32HG008713 from the National Human Genome

## Abstract

Analysis of single-cell RNA sequencing (scRNA-Seq) data often involves filtering out uninteresting or poorly measured genes and dimensionality reduction to reduce noise and simplify data visualization. However, techniques such as principal components analysis (PCA) fail to preserve non-negativity and sparsity structures present in the original matrices, and the coordinates of projected cells are not easily interpretable. Commonly used thresholding methods to filter genes avoid those pitfalls, but ignore collinearity and covariance in the original matrix. We show that a deterministic column subset selection (DCSS) method possesses many of the favorable properties of common thresholding methods and PCA, while avoiding pitfalls from both. We derive new spectral bounds for DCSS. We apply DCSS to two measures of gene expression from two scRNA-Seq experiments with different clustering workflows, and compare to three thresholding methods. In each case study, the clusters based on the small subset of the complete gene expression profile selected by DCSS are similar to clusters produced from the full set. The resulting clusters are informative for cell type.

## Introduction

Advances in RNA sequencing technology have made it possible to measure the genome-wide expression profile of single cells [1]. This technology is not without computational and analytical challenges, some of which include quality control, quantification, normalization, technical variability, and other confounding factors such as batch effects [2, 3]. More general challenges stem from the high dimensionality of the expression profiles: for example, the challenge of selecting informative features from within the expression profiles.

One use for single-cell RNA sequencing (scRNA-Seq) data is the characterization of heterogeneity of expression within a population of cells for the discovery of new cell types through clustering of expression profiles [4]. This note explores the following question: is it possible reduce the number of features in the expression profile without a large effect on clustering and classification? This question is inspired by the quality control and technical variability

Research Institute. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

**Competing interests:** The authors have declared that no competing interests exist.

challenges of scRNA-Seq. Common techniques for quality control and technical variability reduction include simple thresholding schemes and principal components analysis (PCA).

One commonly used technique to reduce the number of features in the data matrix involves selecting columns from the original data matrix **A**, to form a column submatrix **C**, by thresholding the individual columns based on a score. Workflows for scRNA-Seq such as *Seurat* [5], *Monocle* [6], *MAST* [7], *Cell Ranger* [8], *scater* [9], *scran* [10], and *SCANPY* [11] all include at least one such filtering steps. Frequently used scores are based on abundance or expression level [5, 6, 8–11], detection rates related to frequencies of zero values [6, 9–11], [7] (filtering described in supplement of [12]), and variance [6, 9]. We call these methods *simple column* thresholding methods, because the score for each column $i$ depends only on column $i$. Furthermore, within each column $i$, covariance between the rows (cells) of that column is not taken into account. By selecting columns using simple column thresholding, and not linear combinations of columns from **A** as with PCA, the elements of **C** will maintain the properties of non-negativity, sparsity structure (e.g. the patterns of zeros in the retained columns), and interpretability. This is an advantage over PCA, but there are no guarantees that **C** will have other properties similar to the original data matrix **A** (e.g. a similar spectrum).

The rational behind these thresholding steps is that is that the most variable genes are responsible for the important differences between cells, and low-abundance genes and or genes with high dropout rates should be filtered out [10]. Currently there does not appear to be consensus in the literature on the best way to define a score for highly variable genes (see discussion in [10]). Scores to identify highly variable genes are based upon the coefficient of variation [13] or dispersion [5, 8, 11]. Some analysis workflows arrive at highly variable gene scores using heuristics such as binning in concert with thresholding [5, 8, 11], or modeling relationships between quantities such as the mean and variance [10]. It has also been argued that genes should be selected based on modeling the relationship between mean and dropout-rate [14]. These heuristics do not qualify as simple column thresholding and also do not come with guarantees.

Replacing the original data matrix of scRNA-Seq expression profiles with a rank-$k$ PCA truncation of the profiles is another commonly used technique to reduce the number of features and the technical variability [3]. To understand the PCA truncation, we must establish some matrix notation that we will use throughout this note. We orient the original data matrix **A** so that the $n$ rows are cells and $d$ columns are features, where $n < d$. For PCA, singular value decomposition (SVD) is performed on the column-mean centered matrix $\tilde{\mathbf{A}} = \mathbf{A} - \mathbf{1}\boldsymbol{\mu}^T$, where $\mathbf{1}$ is an $n \times 1$ column vector and $\boldsymbol{\mu} = \frac{1}{n}\mathbf{A}^T\mathbf{1}$ is a $d \times 1$ column vector of column-means. The sum of the spectrum of eigenvalues of $\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T$ is proportional to the total empirical variance of **A**. The rank-$k$ PCA truncation of **A**, which we call $\tilde{\mathbf{T}}_k$, is the rank-$k$ SVD truncation of $\tilde{\mathbf{A}}$. SVD is reviewed in Section B in S1 File, and the formula for $\tilde{\mathbf{T}}_k$ is provided there. As a consequence of the SVD, the spectrum of the square of the rank-$k$ PCA truncation $\tilde{\mathbf{T}}_k$ is identical to the spectrum of the square of the mean-centered data matrix $\tilde{\mathbf{A}}$ up to rank $k$; PCA gives a rank-$k$ approximation to the mean-centered data $\tilde{\mathbf{A}}$ that preserves the maximum empirical variance of **A**. PCA is performed to reduce technical variability under the assumption that the technical variation is primarily captured by the non-leading eigenvalues and eigenvectors of $\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T$. The technical variability due to dropout requires additional sophistication to address, such as *SCDE/PAGODA* [15], *ZIFA* [16], or *CIDR* [17]. The drawback of replacing the original data matrix with the rank-$k$ PCA truncation of the data that it fails to preserve non-negativity and sparsity structures present in the original data matrix, and the coordinates of projected cells are not interpretable in terms of single features. PCA alone does not provide a column subset.

The goal of column subset selection (CSS) is to extract from a matrix **A** a column submatrix **C** that conserves favorable properties, such as conditions on the spectrum of the column submatrix **C** [18]. Like the simple column thresholding methods, CSS maintains the properties of non-negativity, sparsity structure, and interpretability, and like PCA, CSS conserves favorable matrix properties. Similar to the simple column thresholding methods discussed above, each column has a score, however in CSS algorithms, the score for each column *i* also depends on all of the other columns. This dependence is what allows **C** to retain favorable properties not guaranteed by simple column thresholding or other heuristics. We will consider rank-*k* subspace leverage scores in this note. Leverage scores have been considered for regression diagnostics and outlier detection in statistics [19, 20] and were brought to prominence more recently in the context of randomized matrix algorithms [21]. The rank-*k* subspace leverage score $\tau_i(\mathbf{A}_k)$ for the $i^{th}$ column of **A** is,

$$\tau_i(\mathbf{A}_k) = \mathbf{a}_i^T(\mathbf{A}_k\mathbf{A}_k^T)^+\mathbf{a}_i, \tag{1}$$

where the $i^{th}$ column of **A** is an $(n \times 1)$-vector denoted by $\mathbf{a}_i$, $\mathbf{M}^+$ denotes Moore-Penrose pseudoinverse of **M**, and $\mathbf{A}_k$ is the rank-*k* SVD approximation to **A**, defined in Section B in S1 File. The leverage score $\tau_i(\mathbf{A}_k)$ can also be written as the solution to the following optimization problem,

$$\tau_i(\mathbf{A}_k) = \min ||\hat{\mathbf{x}}||_2^2 \quad \text{s.t.} \quad \hat{\mathbf{x}} = \text{argmin}||\mathbf{A}_k\mathbf{x} - \mathbf{a}_i||_2^2, \quad \mathbf{x} \in \mathbb{R}^d. \tag{2}$$

where $||\mathbf{x}||_2^2$ refers to the Euclidean ($L_2$) norm of the vector **x** (see Section C in S1 File for the proof). The vector **x** measures how easily the column $\mathbf{a}_i$ can be written as a linear combination of the columns of $\mathbf{A}_k$. Eq 2 shows that leverage scores capture the importance of each column $\mathbf{a}_i$ in the column space of $\mathbf{A}_k$ and are sensitive to collinearity between columns. We illustrate this point with a toy example in Section The DCSS algorithm [22].

CSS algorithms select columns either with a random sampling procedure (such as in [21]) or a deterministic procedure. We showcase the deterministic CSS (DCSS) algorithm introduced by [22]. [22] show that for datasets with power-law decay in $\tau_i(\mathbf{A}_k)$, DCSS will select a least-squares approximation for **A**, $\mathbf{CC}^+\mathbf{A}$, requiring fewer columns with the same accuracy than random sampling methods. One of the contributions of this note is a new bound for the spectrum of the square of **C** selected by DCSS projected onto the rank-*k* subspace that best approximates **A** (Eq 9). This bound means that, once both **C** and **A** are projected onto the rank-*k* subspace that best approximates **A**, $\mathbf{CC}^T$ is "close" to $\mathbf{AA}^T$. Or, in other words, $\mathbf{CC}^T$ can be thought of as an approximation to $\mathbf{A}_k\mathbf{A}_k^T$, in a way made precise in Section New bounds for DCSS. Another consequence is that the Frobenius norm of **C** is bounded (Eq 10). The Frobenius norm is a measure of the "size" of a matrix, so this bound provides confidence that the DCSS column matrix **C** is also similar in "size" to **A** and $\mathbf{A}_k$. In the event that DCSS is performed on a mean-centered matrix $\tilde{\mathbf{A}}$, the Frobenius norm provides a measure of empirical variance. We also show a similar bound holds for random sampling (Eq 11), and under the assumption of power-law decay, DCSS requires fewer columns for the same error than random sampling.

In addition to the spectral bound, we present two case studies on two different scRNA-Seq experimental and analysis workflows to illustrate empirically the effect of thresholding features with DCSS compared to read count, variance, and index of dispersion (empirical variance/mean) on clustering and classification. The comparison to these three interpretable simple column thresholding schemes provides insight into the features selected by DCSS. To the best of our knowledge, this is the first time DCSS has been applied to scRNA-Seq data. Our interest is in principled column subset selection, so we do not empirically compare DCSS to SVD, PCA, or other commonly used dimensionality reduction techniques such as t-distributed stochastic neighbor embedding (t-SNE) [23] because, while these methods do provide dimensionality reduction, they do not select column submatrices.

The first case study is the genome-wide expression profiles of 3, 005 cells from the mouse cortex and hippocampus [4] and the clustering workflow of [24]. The second case is the genome-wide expression profiles of 4, 423 cells from mouse bone marrow [25] and the trajectory workflow of [26]. We showcase our method on two different experimental and analysis workflows to illustrate the general utility of our method. The expression profile features in the first case study are derived from a partition of reads intermediate to gene expression quantification; the expression profile features in the second case study are gene unique molecular identifier (UMI) counts (see Section Results for DCSS for further discussion). These two different expression profile feature types have different collinearity and covariance properties, and so we illustrate the DCSS method with both types. For this note, we use previously studied analysis and clustering workflows that were tailored for the datasets at hand, since our purpose is to study the effects of DCSS and thresholding on clustering, and not to evaluate many different clustering algorithms or analysis workflows. Clustering is both an art and a science, and there are many different algorithms and workflows for clustering single-cell data; others not considered here include *CIDR* [17] and *SIMLR* [27].

In both case studies, DCSS reduces the low abundance genes and maintains many of the most variable and over-dispersed genes. This shows that DCSS shares the best features of the simple column thresholding methods and, like PCA, comes with additional bounds on the spectrum. This supports our conclusion that DCSS can be used instead of the simple column thresholding methods for quality control and to reduce technical variability, in addition to selecting informative features. In both case studies, only a small fraction of the features are necessary to obtain clusters reflecting cell types, consistent with results in [28]. We show that there is high similarity between the clustering assignments computed with the complete expression profile and the reduced expression profile.

## Methods

The aim of this note is to explore the effect of thresholding features (in our setting, measurements of gene expression) with DCSS. We compare DCSS to simple column thresholding methods and also to the complete data. These thresholding methods are often the first step in the pre-processing workflow. In this section, we include the DCSS algorithm for completeness, and we describe the new bounds for DCSS.

### The DCSS algorithm [22]

**Algorithm 1**. *The DCSS algorithm selects for the submatrix* $\mathbf{C}$ *all columns i with a rank-k subspace leverage score* $\tau_i(\mathbf{A}_k)$ *above a threshold* $\theta$*, determined by the error tolerance* $\epsilon$ *and the rank, k. The algorithm is as follows.*

1. *Choose the rank, k, and the error tolerance,* $\epsilon$.
2. *For every column i, calculate the rank-k subspace leverage scores* $\tau_i(\mathbf{A}_k)$ (Eq 1).
3. *Sort the columns by* $\tau_i(\mathbf{A}_k)$*, from largest to smallest. The sorted column indices are* $\pi_i$.

4. *Define an empty set* $\Theta = \{\}$. *Starting with the largest sorted column index* $\pi_0$, *add the corresponding column index* $i$ *to the set* $\Theta$, *in decreasing order, until,*

$$\sum_{i \in \Theta} \tau_i(\mathbf{A}_k) > k - \epsilon, \tag{3}$$

*and then stop. Note that* $k = \sum_{i=1}^{d} \tau_i(\mathbf{A}_k)$. *It will be useful to define* $\tilde{\epsilon} = \sum_{i \notin \Theta} \tau_i(\mathbf{A}_k)$. Eq 3 *can equivalently be written as* $\epsilon > \tilde{\epsilon}$.

5. *If the set size* $|\Theta| < k$, *continue adding columns in decreasing order until* $|\Theta| = k$.

6. *The leverage score* $\tau_i(\mathbf{A}_k)$ *of the last column* $i$ *included in* $\Theta$ *defines the leverage score threshold* $\theta$.

7. *Introduce a rectangular selection matrix* $\mathbf{S}$ *of size* $d \times |\Theta|$. *If the column indexed by* $(i, \pi_i)$ *is in* $\Theta$, *then* $\mathbf{S}_{i,\pi_i} = 1$. $\mathbf{S}_{i,\pi_i} = 0$ *otherwise. The DCSS submatrix is* $\mathbf{C} = \mathbf{AS}$.

*This algorithm requires* $\mathcal{O}(\min(n, d)nd)$ *arithmetic operations, and there are modifications to the algorithm that improve the runtime* [22].

Theorem 3 of [22] states that when the rank-$k$ subspace leverage scores exhibit a power-law decay in the sorted column index $\pi_i$,

$$\tau_{\pi_i}(\mathbf{A}_k) = \pi_i^{-a} \tau_{\pi_0}(\mathbf{A}_k) \qquad a > 1, \tag{4}$$

the number of sample columns selected by DCSS is,

$$|\Theta| = \max\left( \left(\frac{2k}{\epsilon}\right)^{\frac{1}{a}} - 1, \left(\frac{2k}{(a-1)\epsilon}\right)^{\frac{1}{a-1}} - 1, k \right). \tag{5}$$

[22] demonstrate the power-law decay behavior of many real-world datasets; we show that this behavior is a reasonable assumption for the scRNA-Seq applications in Section Results.

For a statistical interpretation of DCSS, consider the data $\mathbf{a}_i$, $i = 1, \ldots, d$ to be identically and independently distributed (i.i.d.) according to the degenerate multivariate distribution $\mathcal{N}(0, \mathbf{A}_k\mathbf{A}_k^T)$. See [29] pg. 527-528 for a discussion of the degenerate multivariate distribution. Then the total likelihood of the data matrix $\mathbf{A}$ is,

$$\begin{aligned} \mathcal{L}(\mathbf{A}) &= \frac{1}{(2\pi)^{\frac{1}{2}kd} \prod_{j=1}^{k} |\sigma_j|^d} \exp\left( -\frac{1}{2} \sum_{i=1}^{d} \mathbf{a}_i^T (\mathbf{A}_k \mathbf{A}_k^T)^+ \mathbf{a}_i \right) \\ &= \frac{1}{(2\pi)^{\frac{1}{2}kd} \prod_{i=j}^{k} |\sigma_j|^d} \exp\left( -\frac{1}{2} \sum_{i=1}^{d} \tau_i(\mathbf{A}_k) \right), \end{aligned} \tag{6}$$

where $|\sigma_j|$ are the $k$ largest singular values of $\mathbf{A}_k$. In contrast, the total likelihood of the DCSS matrix **C** is,

$$\begin{aligned}\mathcal{L}(\mathbf{C}) &= \frac{1}{(2\pi)^{\frac{1}{2}k|\Theta|}\prod_{j=1}^{k}|\sigma_j|^{|\Theta|}}\exp\left(-\frac{1}{2}\sum_{i\in\Theta}\tau_i(\mathbf{A}_k)\right)\\ &= \frac{1}{(2\pi)^{\frac{1}{2}k|\Theta|}\prod_{j=1}^{k}|\sigma_j|^{|\Theta|}}\exp\left(-\frac{1}{2}\sum_{i\in\Theta}\tau_i(\mathbf{A}_k)-\frac{1}{2}\sum_{i\notin\Theta}\tau_i(\mathbf{A}_k)+\frac{1}{2}\tilde{\epsilon}\right)\\ &= \mathcal{L}(\mathbf{A})\exp\left(\tfrac{1}{2}\tilde{\epsilon}\right)(2\pi)^{\frac{1}{2}k(d-|\Theta|)}\prod_{j=1}^{k}|\sigma_j|^{d-|\Theta|}.\end{aligned} \tag{7}$$

This shows that the DCSS matrix **C** preserves the total likelihood of the data up to a factor of $\exp\left(\frac{1}{2}\tilde{\epsilon}\right) < \exp\left(\frac{1}{2}\epsilon\right)$ and a normalization constant, under the assumption that the data is i.i.d. according to $\mathcal{N}(0, \mathbf{A}_k\mathbf{A}_k^T)$. Any other selection set $\Theta'$ of the same number of columns ($|\Theta'| = |\Theta|$) will have equal or greater error ($\epsilon \leq \epsilon'$). This interpretation illustrates that DCSS accounts for covariance $\mathbf{A}_k\mathbf{A}_k^T$ between rows (cells).

The DCSS method has two parameters, $k$, $\epsilon$ which jointly determine the number of columns $|\Theta|$ in the DCSS column submatrix **C**. The parameter $k$ determines the rank of interest of the SVD approximation to **A**. The tuning parameter $\epsilon$ is a measure of the error tolerance in the "size" of **C** compared to $\mathbf{A}_k$. Given $k$, the desired error tolerance $\epsilon$ determines the number of columns $|\Theta|$; completely equivalently, one could instead select the desired number of columns $|\Theta|$ and determine the resulting error tolerance $\epsilon$. The parameters $k$, $\epsilon$, and $|\Theta|$ will be different for different datasets and workflows. The selection of these parameters is a model selection problem, and in concert with a loss function, one could select these parameters using one's preferred model selection method (e.g. cross-validation). The aim of this note, to compare clustering performed with the complete data matrix and a column submatrix, does not have a well-defined loss function, and so we use the heuristic "elbow" method for selecting $k$ [30], and we choose $\epsilon$ to be 0.1 or 0.05, depending on the biological application.

As a toy example to illustrate how DCSS differs from the simple column thresholding methods, consider the following toy data matrix,

$$\mathbf{A} = \begin{pmatrix} 40 & 20 & 10 \\ 20 & 10 & 15 \end{pmatrix}. \tag{8}$$

If the goal is to select a column submatrix with two columns, it is easy to check that simple column thresholding by mean, variance, and index of dispersion all select the first and second columns. However, the resulting column submatrix is only rank 1, because the first and second columns are linearly dependent. In contrast, DCSS with ($k$ = 2, $\epsilon > 0.2$) will select the first and third columns, and the resulting DCSS column submatrix will be rank 2. Unlike the first three methods, DCSS takes into account the collinearity between columns in the selection procedure. If the DCSS error tolerance for this toy example is less than 0.2, DCSS will select all three columns.

We also mention two asides: first, in applications where the number of cells is far greater than the number of gene features ($n > d$), the method can instead be applied to $\mathbf{A}^T$ instead of **A** to filter cells instead gene features; second, the method can be modified to select columns for any rank-$k$ subspace defined by $k$ singular vectors of **A**, and not just the leading-$k$ subspace (e.g. drop component 1 but include component 2). This could be useful when some of the leading singular vectors are highly correlated with batch, dropout, or other confounding effects.

## New bounds for DCSS

We derive a new spectral approximation bound (Eq 9) for the square of the submatrix **C** selected with DCSS and projected onto the rank-$k$ subspace that best approximates **A**.

**Theorem 1**. *Let* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *be a matrix of at least rank k and* $\tau_i(\mathbf{A}_k)$ *be defined as in* Eq 1. *Construct* **C** *following the DCSS algorithm described in Section The DCSS algorithm* [22]. *Then* **C** *satisfies,*

$$(1-\epsilon)\mathbf{A}_k\mathbf{A}_k^T \quad \preceq \quad \mathbf{U}_k\mathbf{U}_k^T\mathbf{C}\mathbf{C}^T\mathbf{U}_k\mathbf{U}_k^T \preceq \mathbf{A}_k\mathbf{A}_k^T. \tag{9}$$

$\mathbf{U}_k$ *is the matrix of left singular vectors from the rank-k SVD approximation to* **A**, *defined in Section B in* S1 File. *The symbol* $\preceq$ *denotes the Loewner partial ordering which is reviewed in Section B in* S1 File. *Conceptually, the Loewner ordering is the generalization of the ordering of real numbers* (*e.g.* $1 < 1.5$) *to Hermitian matrices.*

This bound means that after projection onto the rank-$k$ subspace that best approximates **A**, $\mathbf{CC}^T$ is "close" to $\mathbf{AA}^T$ on that subspace. Statements of Loewner ordering are quite powerful; important consequences include inequalities for the eigenvalues and Euclidean distances. Some of the consequences of the Loewner ordering are reviewed in Section B in S1 File. Eq 9 and the fact that $\mathbf{CC}^T \preceq \mathbf{AA}^T$ implies a bound on the Frobenius norm of **C**, a measure of the

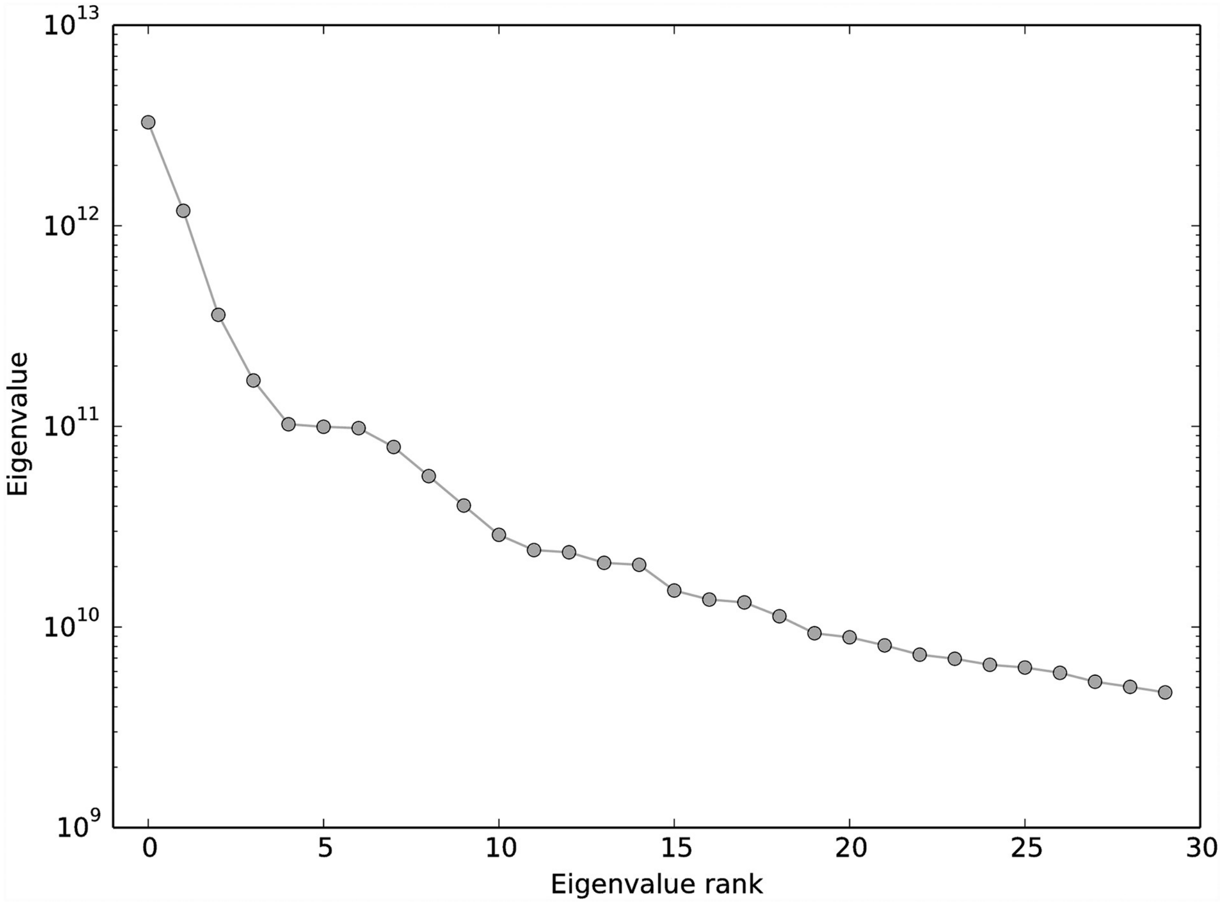

**Fig 1. Eigenvalues for $\mathbf{AA}^T$, where A is the data matrix from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24].** The first "elbow" occurs at the fifth largest eigenvalue. We choose $k = 5$ for the DCSS workflow.

https://doi.org/10.1371/journal.pone.0210571.g001

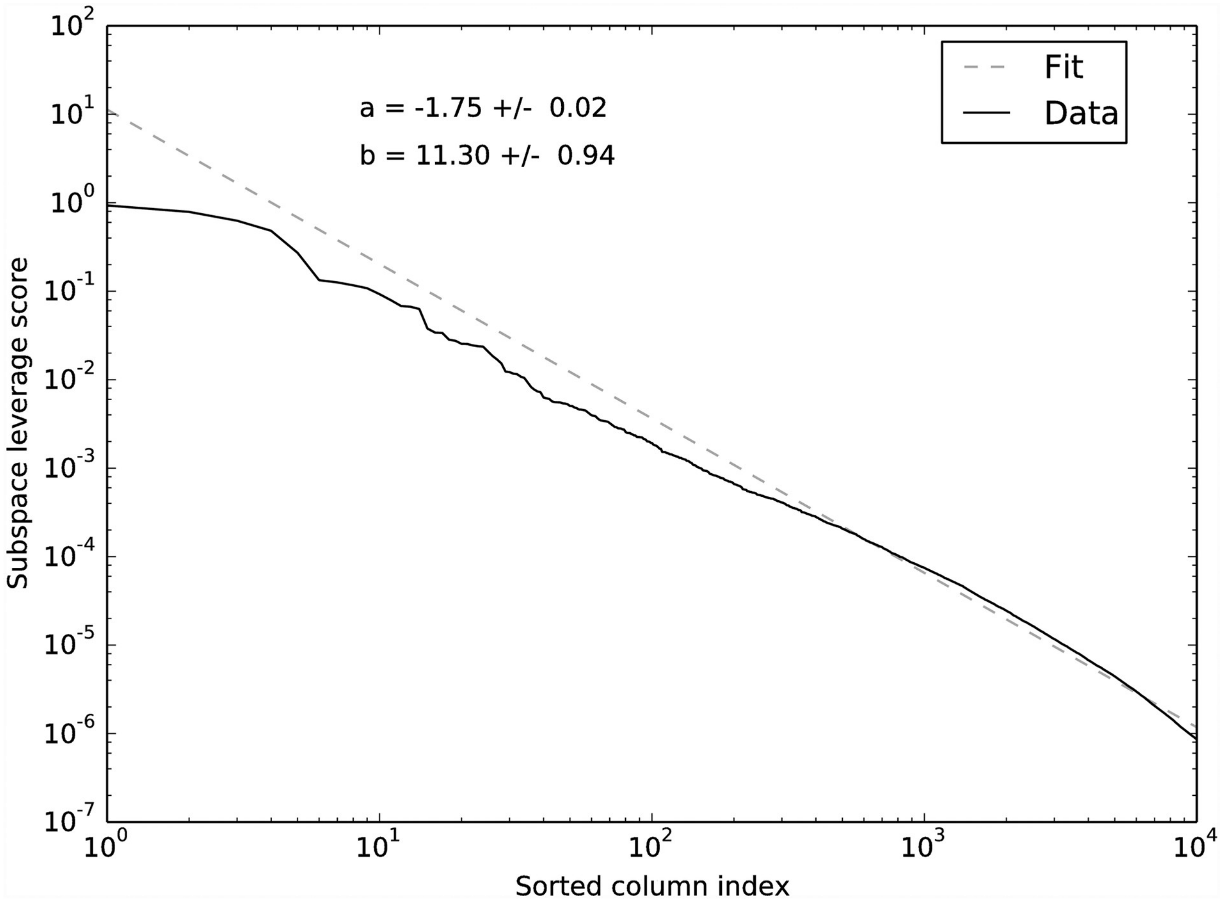

**Fig 2. Power-law decay of $k = 5$ subspace leverage scores with sorted index for each column in A, where A is the data matrix from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24].** The fit is to Score = b × (Index)$^a$. This shows that power-law decay is a reasonable assumption for this dataset.

https://doi.org/10.1371/journal.pone.0210571.g002

**Table 1. PANTHER overrepresentation test (release 20160715) with the GO ontology database (release 2016-08-22) for the submatrix C selected from the data matrix A from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24] using DCSS with $k = 5$, $\epsilon = 0.1$.** The 862 DCSS-selected ECs of **C** are mapped to 1, 642 genes for the PANTHER overrepresentation test.

| Type | Gene ontology (GO) term | Bonferroni p-value |
|---|---|---|
| Biological process | cellular component organization (GO:0016043) | 1.12E-02 |
| Biological process | cellular component organization or biogenesis (GO:0071840) | 8.01E-03 |
| Biological process | localization (GO:0051179) | 4.37E-02 |
| Cellular component | neuron projection (GO:0043005) | 4.52E-04 |
| Cellular component | neuron part (GO:0097458) | 8.24E-05 |
| Cellular component | cell projection (GO:0042995) | 8.36E-03 |
| Cellular component | cytoplasm (GO:0005737) | 1.59E-02 |
| Cellular component | intracellular part (GO:0044424) | 4.89E-02 |
| Molecular function | enzyme binding (GO:0019899) | 3.35E-02 |
| Molecular function | olfactory receptor activity (GO:0004984) | 1.30E-02 |

https://doi.org/10.1371/journal.pone.0210571.t001 "size" of a matrix,

$$(1-\epsilon)||\mathbf{A}_k||_F^2 \quad \leq \quad ||\mathbf{C}||_F^2 \leq ||\mathbf{A}||_F^2. \tag{10}$$

In the event that **A** is mean-centered, Eq 10 means that the total empirical variance of **C** is bounded from below by $(1 - \epsilon)$ the variance in $\mathbf{A}_k$ and bounded from above by the total variance of **A**, similar to PCA (discussed in Section Introduction). The proof of Eqs 9 and 10 is included in Section D in S1 File. To develop further intuition for these bounds, consider a mean-centered **A**. Consider also the PCA truncation $\tilde{\mathbf{T}}_k$ (defined in Section B in S1 File) instead of a column matrix **C** in Eqs 9 and 10. The PCA truncation $\tilde{\mathbf{T}}_k$ also satisfies both bounds.

One simple consequence of Eq 9 is the following bound,

$$(1-\epsilon)\mathbf{A}_k\mathbf{A}_k^T \quad \preceq \quad \mathbf{U}_k\mathbf{U}_k^T\mathbf{C}\mathbf{C}^T\mathbf{U}_k\mathbf{U}_k^T \preceq (1+\epsilon)\mathbf{A}_k\mathbf{A}_k^T. \tag{11}$$

Eq 11 also holds for **C** selected by random sampling methods with *t* columns (see Section E in S1 File for the theorem and proof). Thus, DCSS selects fewer columns with the same

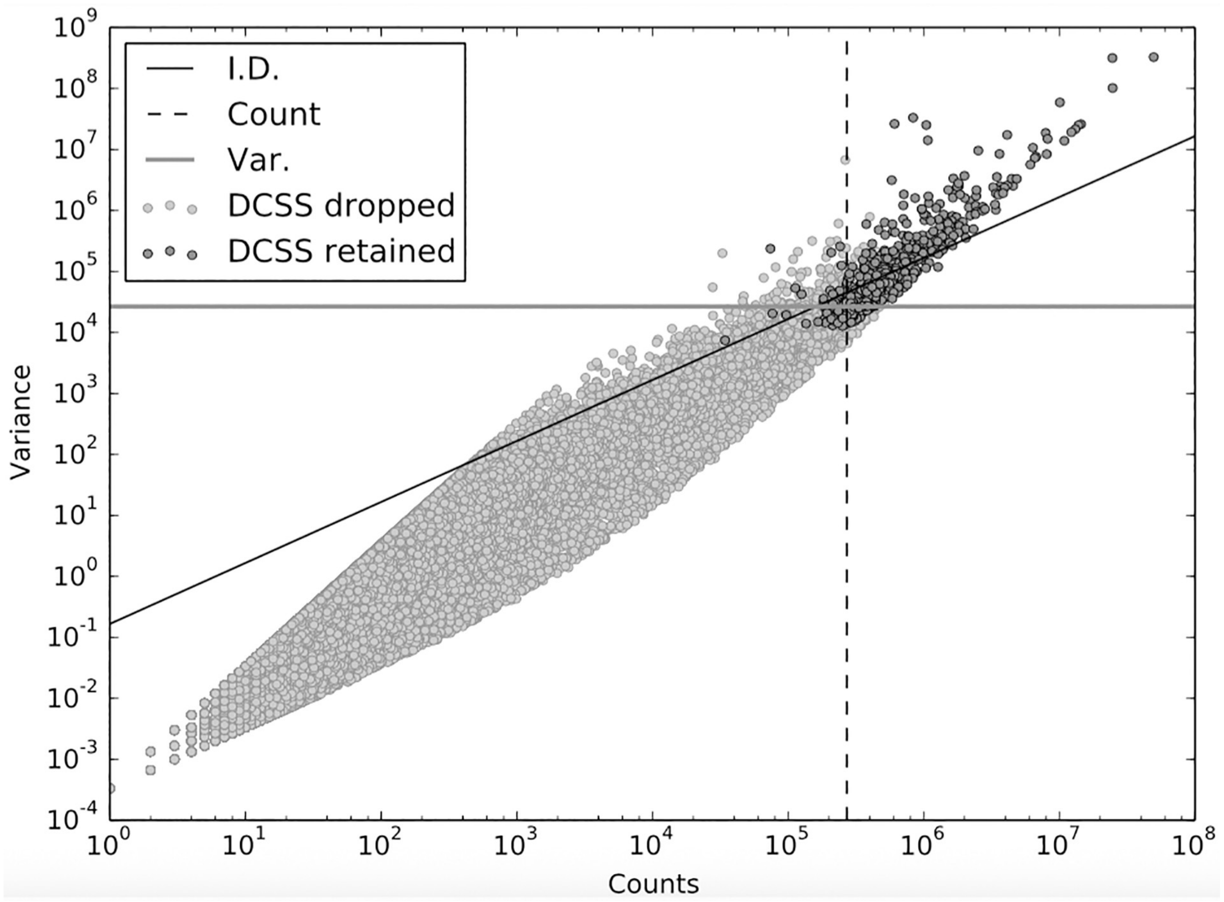

**Fig 3. Count-variance plot for each column of A, where A is the data matrix from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24].** The color for each column represents whether the column is selected or not by $k = 5$, $\epsilon = 0.1$ DCSS. The plot also shows the thresholds for count, variance, and index of dispersion with same number of selected columns as DCSS. The columns selected by DCSS are highly variable and have large counts.

https://doi.org/10.1371/journal.pone.0210571.g003

accuracy $\epsilon$ in Eq 11 for power-law decay in the rank-$k$ subspace leverage scores when,

$$|\Theta| = \max\left(\left(\frac{2k}{\epsilon}\right)^{\frac{1}{a}} - 1, \left(\frac{2k}{(a-1)\epsilon}\right)^{\frac{1}{a-1}} - 1, k\right) < \frac{2}{\epsilon^2}(k + m\gamma)\left(1 + \frac{1}{3}\epsilon\right)\ln\left(\frac{16k}{\delta}\right) \leq t. \tag{12}$$

In this expression, $m$ is the number of columns with zero rank-$k$ subspace leverage score, $\gamma$ is the minimum non-zero leverage score, and $\delta$ is the probability that Eq 11 fails to hold under random sampling.

## Results

We present two case studies where we compare DCSS to the simple column thresholding methods of variance, count, and index of dispersion. We analyze the overlap in the selected columns. We also illustrate the effect of DCSS compared to the complete data for single-cell clustering.

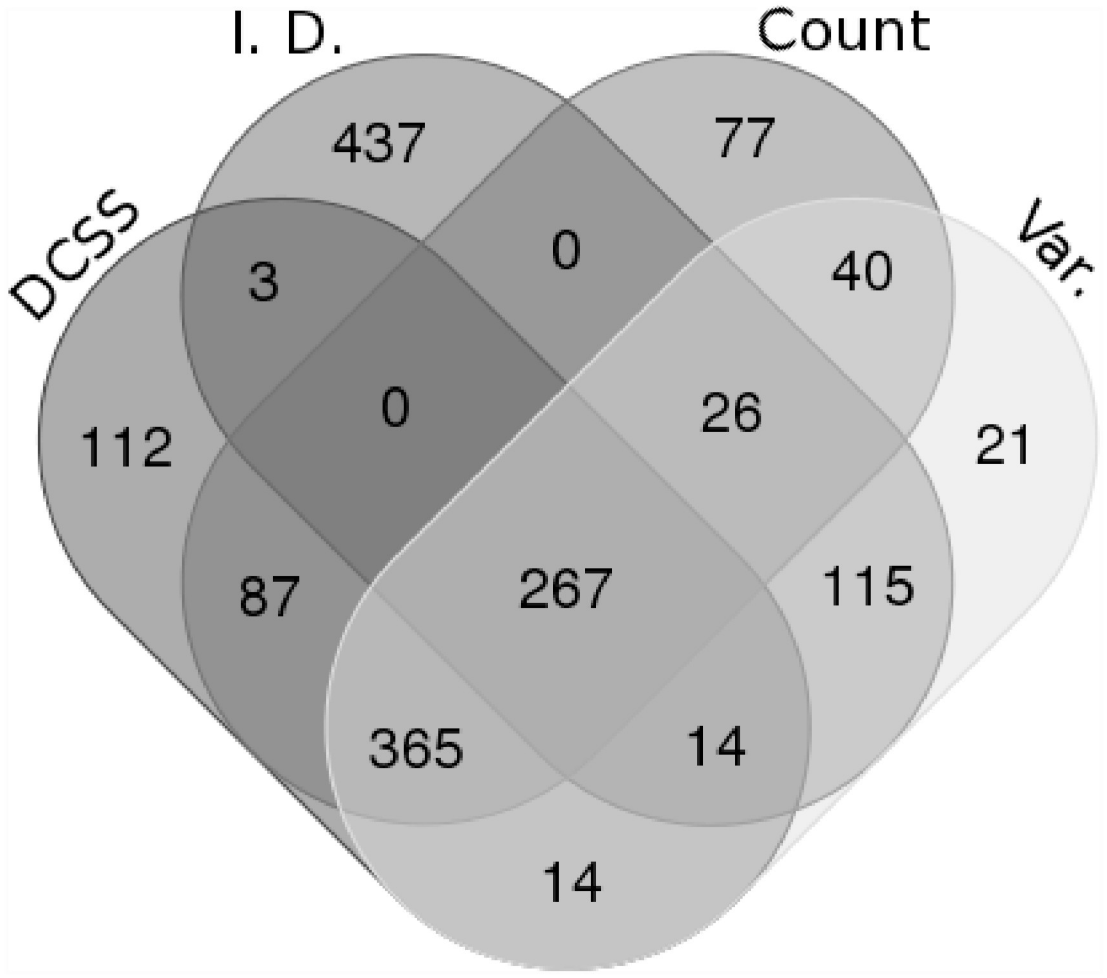

**Fig 4. Venn diagram comparing the overlap between selected columns between $k = 5$, $\epsilon = 0.1$ DCSS, count, variance, and index of dispersion thresholding on the data matrix from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24].** The four methods select many of the same columns. Recall from the toy example in Eq 8 that changing only one column can significantly change properties of the submatrix. Figure tool credit: [33].

https://doi.org/10.1371/journal.pone.0210571.g004

## Mouse cortex and hippocampus single-cell gene expression

As a concrete illustration of the DCSS method, we focus on the genome-wide expression profiles of 3005 cells from the mouse somatosensory cortex and hippocampal CA1 region [4] and the clustering workflow of [24]. The main contribution of [24] is to perform clustering directly on the partition of reads into equivalence classes (ECs) rather than on a full quantification of reads into gene expression. ECs are a partition of reads into distinct classes, such that every read in a class maps to exactly the same set of transcripts [31]. This method is computationally scalable, comparable across scRNA-Seq experiments, and can be more accurate than clustering performed on a full quantification of reads into gene expression profiles [24].

The [24] data matrix **A** is 3, 005 cells × 246, 981 EC counts. By the elbow method, we choose $k = 5$ for the DCSS workflow (Fig 1). We select an error tolerance of $\epsilon = 0.1$. The DCSS algorithm ran in less than a minute. The rank-5 subspace leverage scores and the power-law fit for the top-scored 10, 000 ECs are shown in Fig 2. The column submatrix **C** has only 862 ECs, or approximately 0.3% of the total ECs. These ECs contain 42.3% of the reads. These 862 ECs map to 2, 748 transcripts and to 1, 642 genes. Table 1 contains the gene ontology term enrichment analysis [32] on the genes corresponding to the DCSS ($k = 5$, $\epsilon = 0.1$) ECs. Enrichments relevant for the brain include neuron part, neuron projection, and olfactory receptor activity.

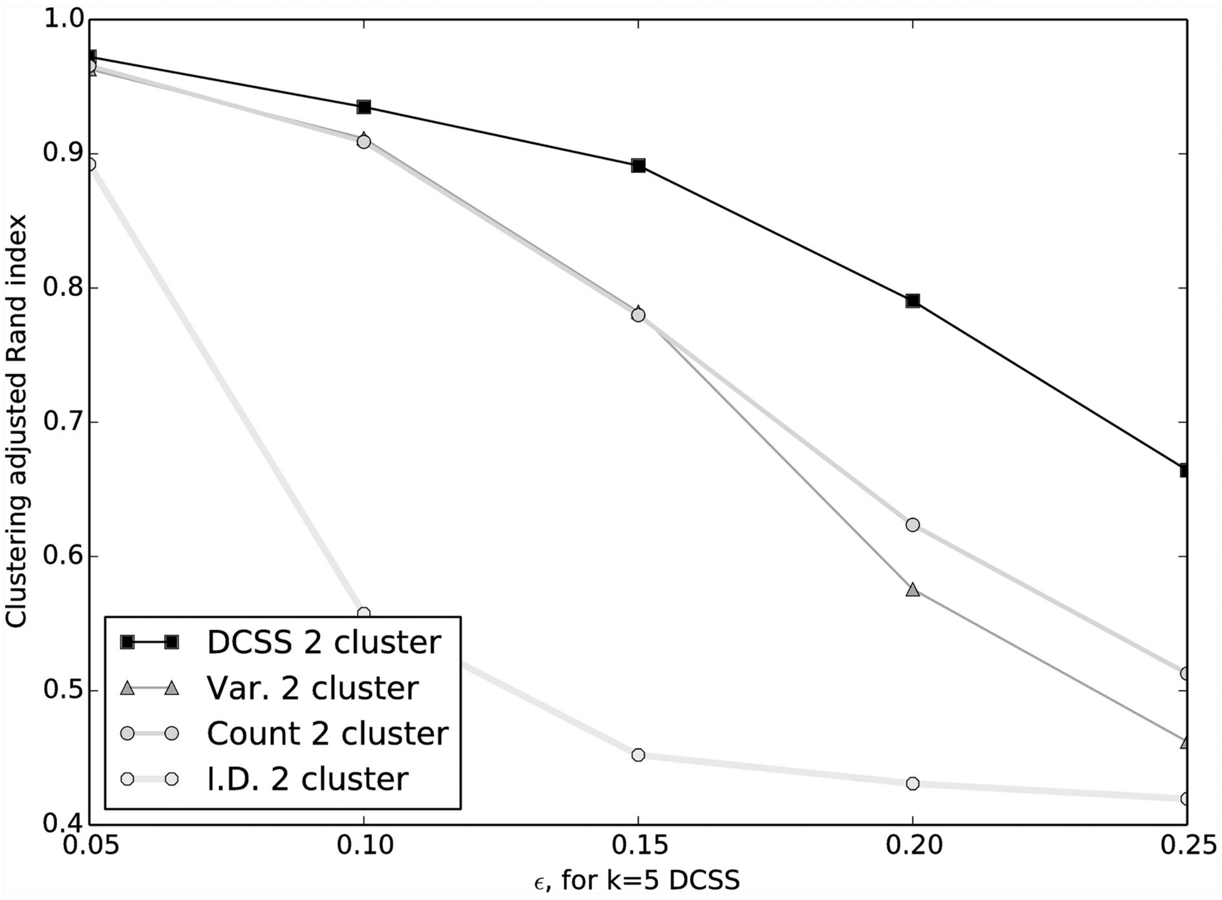

**Fig 5. Average spectral clustering ARI for two clusters for DCSS, count, variance, and index of dispersion thresholding on the data matrix from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24].** Perfect agreement between cluster assignments has an ARI of 1. We vary the error tolerance $\epsilon$ with $k = 5$ for DCSS. Increasing the error tolerance decreases the agreement between clusters.

https://doi.org/10.1371/journal.pone.0210571.g005

The enrichment analysis has an important caveat: because we map ECs to transcripts without positing an error model, there could be a high rate of false positives in the resulting transcripts and genes.

We are interested in how differently selected subsets of columns of the same size compare, so we compare DCSS to the three simple column thresholding methods with the same number of columns in Figs 3 and 4. These figures show the similarities and differences in columns selected by the four thresholding methods. The simple column thresholding methods have sharp boundaries in Fig 3, while the DCSS boundary is not linearly separable. The DCSS boundary approximately interpolates between the count and variance boundaries, and is most distinct from the index of dispersion boundary. Fig 4 summarizes the overlap between selected columns in a Venn diagram. These figures illustrate that the DCSS method selects columns that are highly variable, have large counts, and frequently are over-dispersed; as such, the DCSS method is prescribed for quality control and to control technical variability.

The [24] workflow for the [4] dataset is to perform spectral clustering on pairwise Jensen-Shannon (JS) distances derived from the partition of reads into ECs. The spectral clustering algorithm used is standard; the algorithm is to perform $k$-means clustering on the $k$-dimensional SVD projection of the normalized Laplacian of the symmetric similarity matrix **S**. The

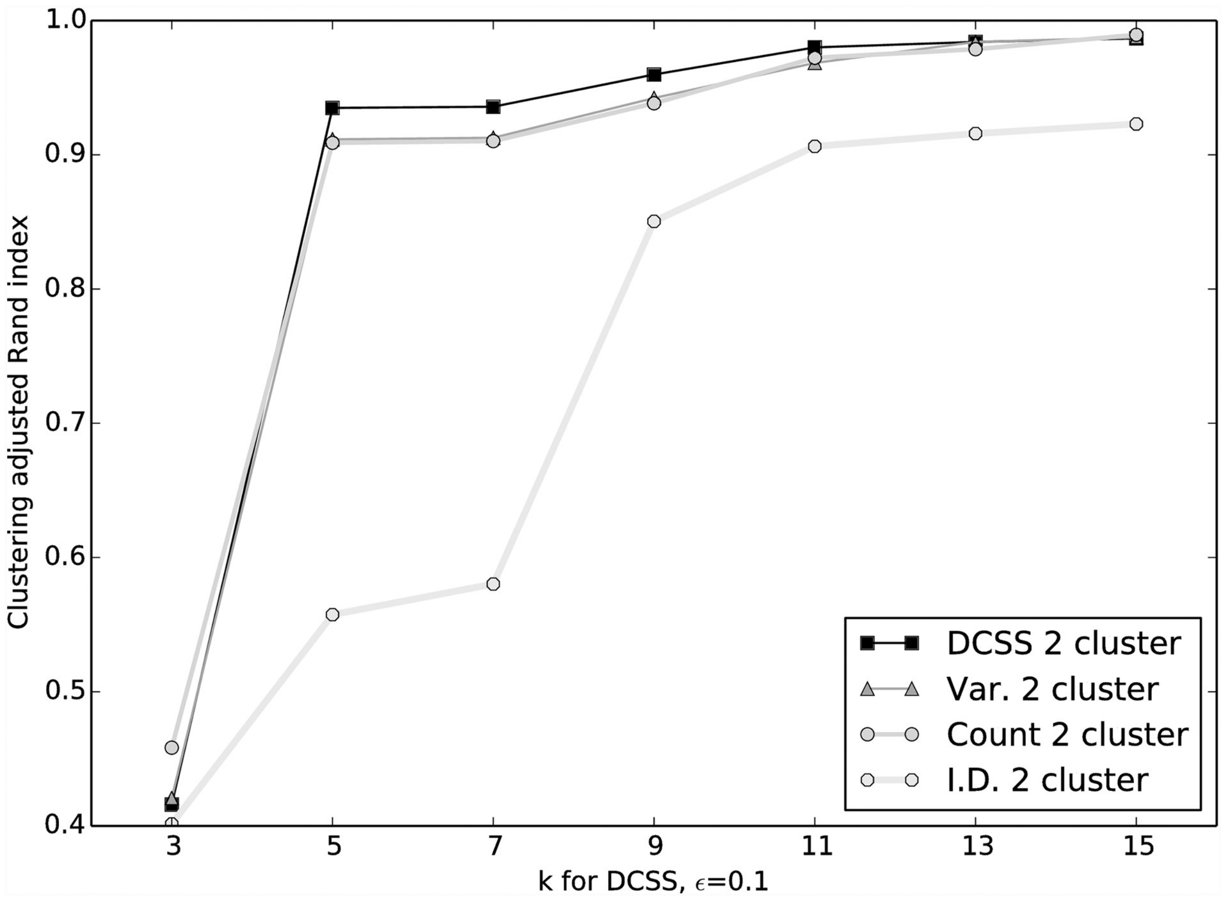

**Fig 6. Average spectral clustering ARI for two clusters for DCSS, count, variance, and index of dispersion thresholding on the data matrix from the mouse cortex scRNA-Seq experiment [4] and the clustering workflow of [24].** We vary the dimension $k$ with fixed error tolerance $\epsilon = 0.1$ for DCSS. Increasing the dimension increases the agreement between clusters.

https://doi.org/10.1371/journal.pone.0210571.g006

similarity matrix used for spectral clustering is $S(\mathbf{p}, \mathbf{q}) = 1 - D_{JS}(\mathbf{p}, \mathbf{q})$, where $D_{JS}(\mathbf{p}, \mathbf{q})$ is the JS distance between two probability mass functions $\mathbf{p}, \mathbf{q} \in \mathbb{R}^d$. JS distances are well-suited to high-dimensional data, and provide more accurate clustering than $L_2$ distances on scRNA-Seq data [24]. For the [4] data, the probability mass function for each cell is the vector of EC counts, normalized to sum to one. For the four thresholded workflows (DCSS, count, variance, and index of dispersion), the probability mass function for each cell is the subset vector of EC counts, normalized to one.

We evaluate the spectral clustering classification similarity using the adjusted Rand index (ARI) [34]. The ARI is a symmetric measure of similarity between two clustering assignments that counts the number of pairwise agreements between the two assignments, adjusted for chance. It takes values between −1 and 1. Perfect agreement between assignments has an ARI of 1, and random assignments have an expected ARI of 0. We compare the average ARI between the complete data and thresholded workflows, regarding the complete data workflow as the ground-truth. This is an imperfect measure, since the complete data has both noise and signal, but it is a reasonable measure for real biological data, since one does not have access to the ground-truth. Since spectral clustering requires a random initialization for $k$-means, the average is over $T = 10$ random initializations. Figs 5 and 6 show the average spectral clustering ARI for two spectral clusters for the workflow with the matrix $\mathbf{A}$ and the workflow with the column submatrix $\mathbf{C}$ for various $k$, $\epsilon$. The S1 File contains similar figures for nine clusters (Figs A and B in S1 File). The different cells were curated into 47 subtypes by [4], but we evaluate our method on coarser-grained classifications because we have higher confidence in the

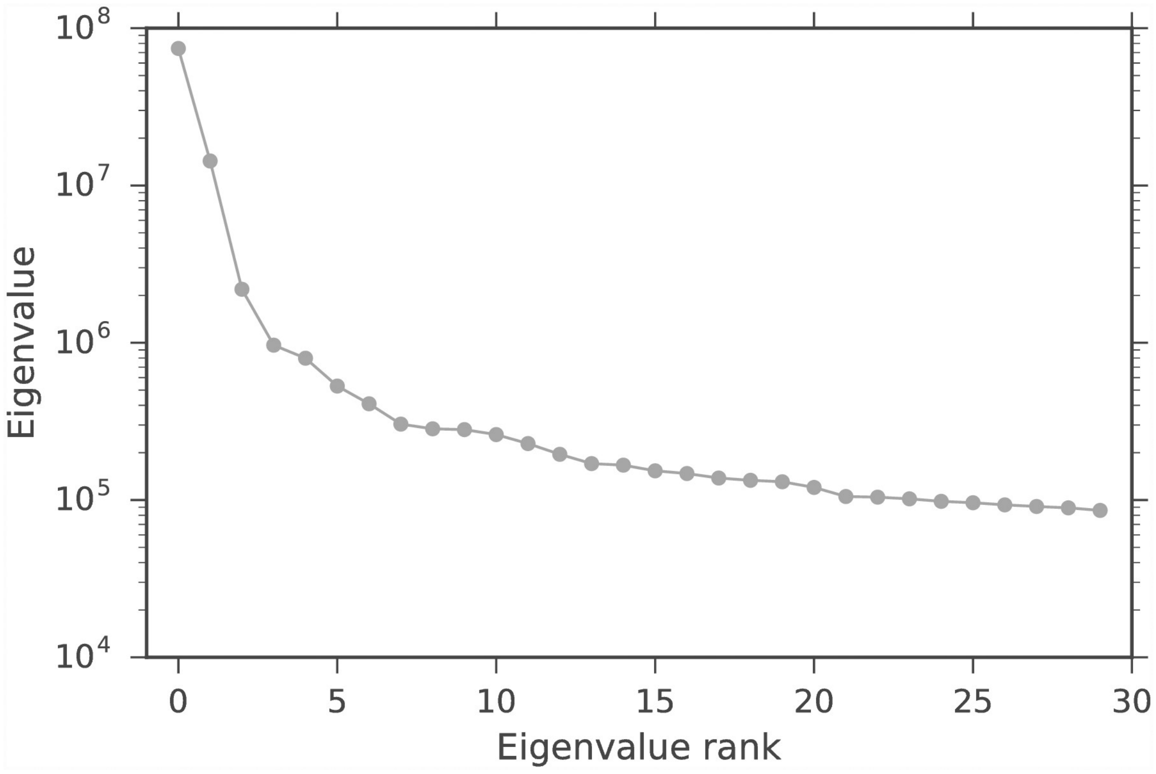

**Fig 7. Eigenvalues for $AA^T$, where A is the data matrix from the mouse bone marrow scRNA-Seq experiment [25] and the analysis workflow of [26].** "Elbows" are not as apparent as in Fig 1. The *wishbone* algorithm begins with a 15-dimensional PCA projection, so we choose the closest "elbow" at $k = 14$.

https://doi.org/10.1371/journal.pone.0210571.g007

spectral clustering ground-truth. Two spectral clusters identify neurons and non-neurons, while nine spectral clusters only loosely correspond to the nine major cell types. We also include the average ARI for the three simple column thresholding methods with the same number of columns as the DCSS method. We find that 0.3% of the total ECs give an average ARI of 0.93 compared to the complete data for two clusters for $k = 5$, $\epsilon = 0.1$ DCSS; only a small fraction of the gene expression profiles currently produced in scRNA-Seq experiments may be necessary to obtain the clusters reflecting cell types.

## Mouse bone marrow single-cell gene expression

As a second application of the DCSS method, we focus on the genome-wide mRNA expression profiles of 4, 423 cells from mouse bone marrow myeloid progenitors [25], and the *wishbone* trajectory workflow of [26]. The contribution of [26] to scRNA-Seq is to use diffusion maps to identify components related to the development and maturation of cells, specifically myeloid and erythroid progenitors from hematopoietic stem and progenitor cells (HSPCs).

The [26] data matrix for the [25] dataset is **A** is 4, 423 cells × 14, 955 gene unique molecular identifier (UMI) counts. The [26] workflow is quite involved. In brief, the *wishbone* algorithm creates a nearest-neighbor Euclidean distance graph. This graph is used to estimate all of the shortest path distances between a set of randomly sampled cells and the rest of the cells, and the shortest path distances are used to make the trajectory and branch assignments. The *wishbone* algorithm acts on a set of diffusion components which are selected for immune cell differentiation through a gene-set enrichment analysis. The diffusion components are calculated

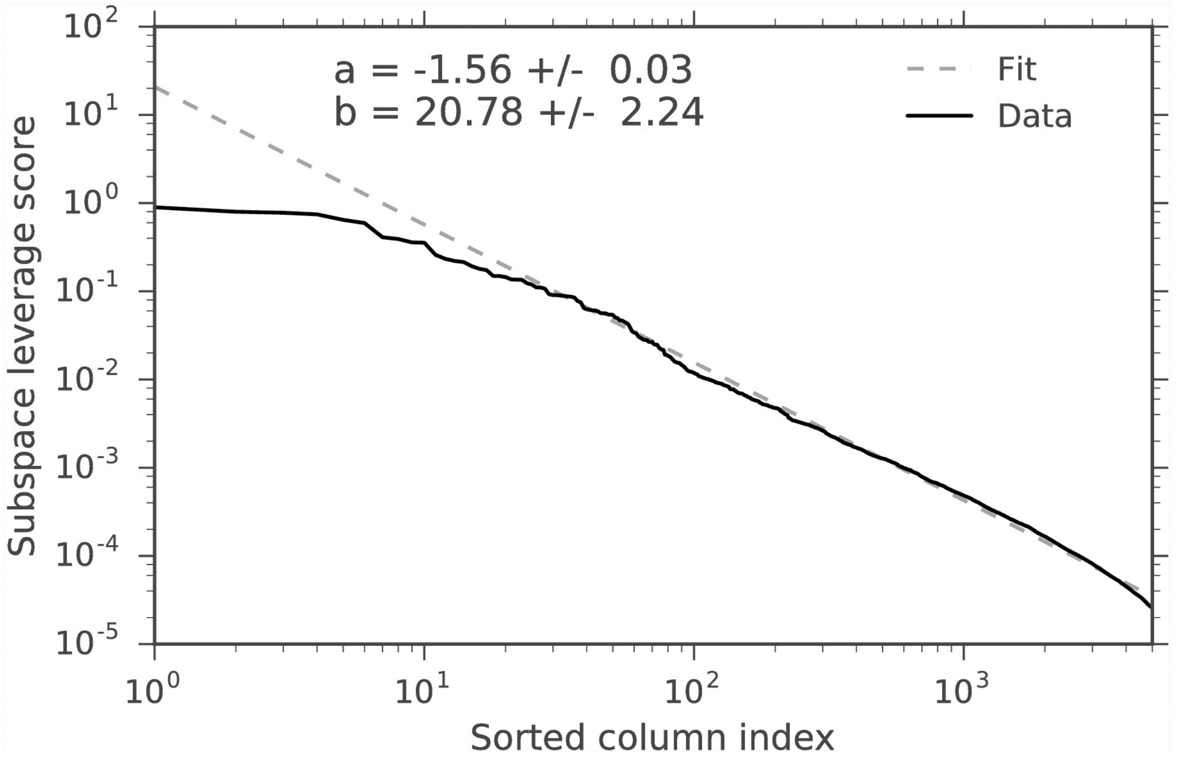

**Fig 8. Power-law decay of $k = 14$ subspace leverage scores with sorted index for each column in A, where A is the data matrix from the mouse bone marrow scRNA-Seq experiment [25] and the analysis workflow of [26].** The fit is toScore = b × (Index)$^{a}$. This shows that power-law decay is also a reasonable assumption for this dataset.

https://doi.org/10.1371/journal.pone.0210571.g008

from the diffusion map of the similarity matrix derived from the Gaussian kernel of the 10-nearest-neighbor Euclidean distance matrix from the 15-dimensional PCA projection of the normalized UMI gene counts [26].

We choose $k = 14$ for the DCSS workflow by a combination of prior selection by the *wishbone* algorithm and the elbow method (Fig 7). The *wishbone* algorithm begins with a 15-dimensional PCA projection, and $k = 14$ is the closest (albeit slight) elbow to $k = 15$. We select an error tolerance of $\epsilon = 0.05$. The DCSS algorithm ran in less than a minute. The rank-14 subspace leverage scores and the power-law fit for the top-scored 5, 000 genes are shown in Fig 8. The column submatrix **C** has 4, 693 genes, or approximately 31.4% of the total genes. These genes contain 90.4% of the UMI counts.

We compare DCSS thresholding with $k = 14$, $\epsilon = 0.05$ to the three simple column thresholding methods with the same number of columns in Figs 9 and 10. The distribution of columns on the count-variance plots are qualitatively different between the mouse brain data (Fig 9) and the mouse bone marrow data (Fig 3). This difference is expected due to the differences between ECs and gene UMI counts. Although the index of dispersion method is more differentiated from the other methods on the mouse bone marrow dataset, the behavior of the DCSS method in relation to the simple column thresholding methods is similar between the datasets.

We calculate the average *wishbone* classification ARI between the two workflows, again regarding the complete data workflow as the ground-truth. Since the *wishbone* algorithm

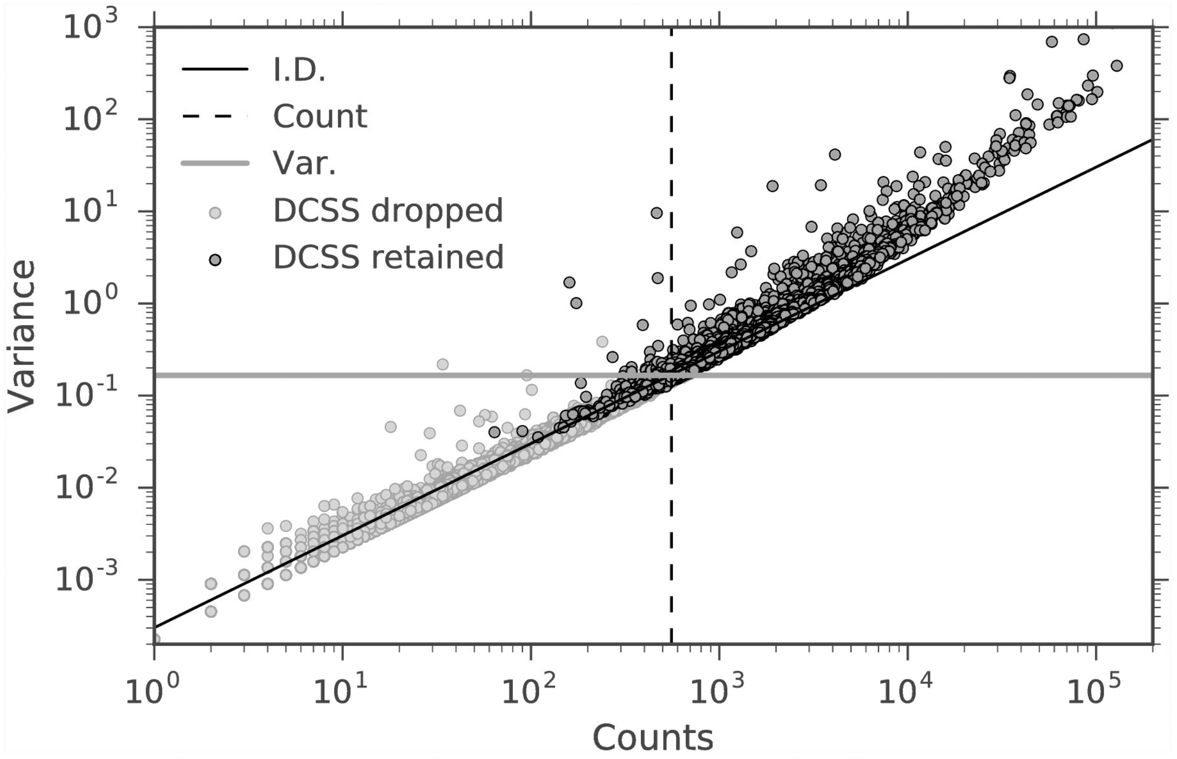

**Fig 9. Count-variance plot for each column of A, where A is the data matrix from the mouse bone marrow scRNA-Seq experiment [25] and the analysis workflow of [26].** The color for each column represents whether the column is selected or not by $k = 14$, $\epsilon = 0.05$ DCSS. The plot also shows the thresholds for count, variance, and index of dispersion with same number of selected columns as DCSS. The columns selected by DCSS are highly variable and have large counts, similar to Fig 3.

https://doi.org/10.1371/journal.pone.0210571.g009

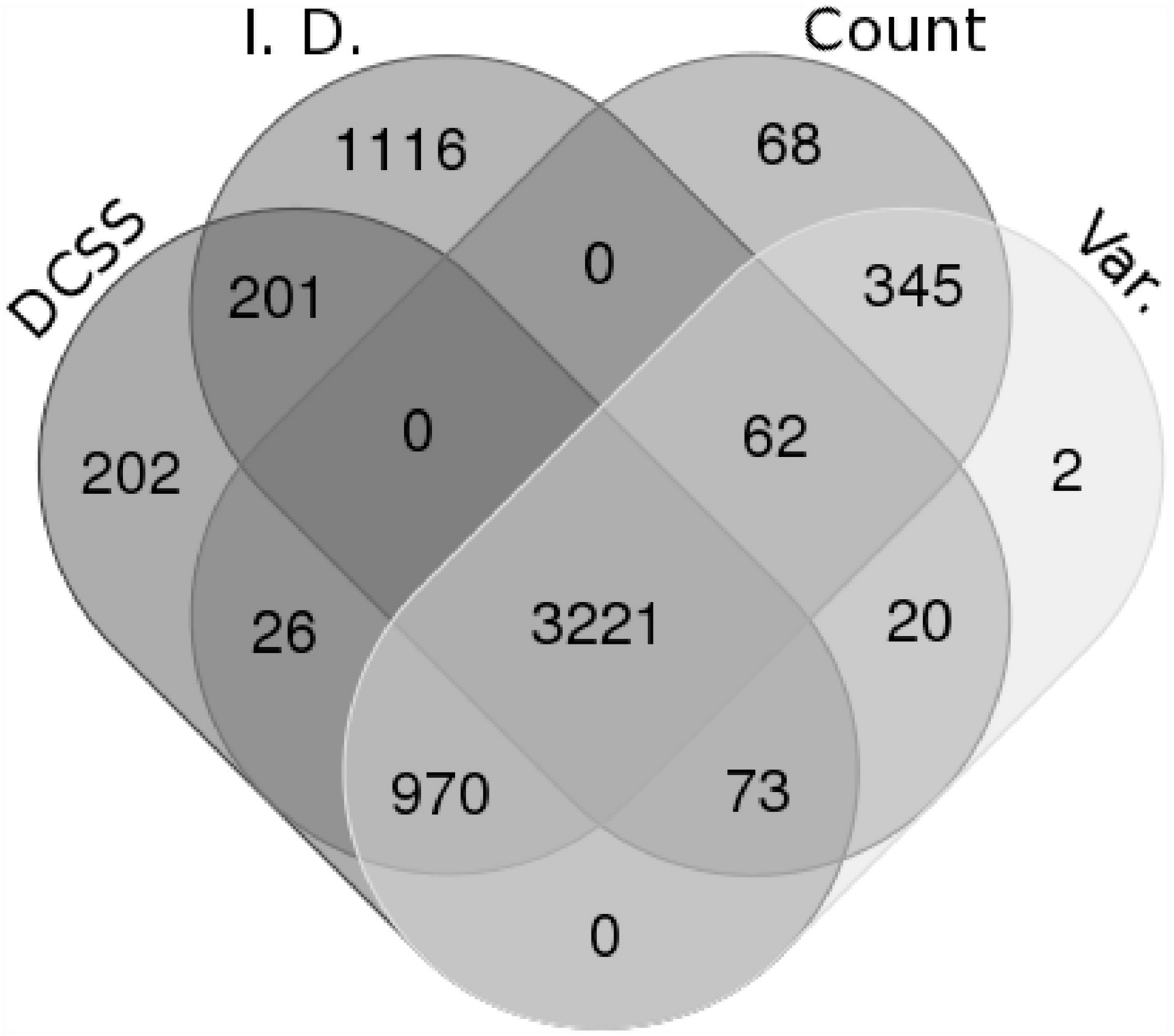

**Fig 10. Venn diagram comparing the overlap between selected columns between $k = 14$, $\epsilon = 0.05$ DCSS, count, variance, and index of dispersion thresholding on the data matrix from the mouse bone marrow scRNA-Seq experiment [25] and the analysis workflow of [26].** Figure tool credit: [33].

https://doi.org/10.1371/journal.pone.0210571.g010

utilizes random sampling, the average is over $T = 10$ *wishbone* branch assignments. The original *wishbone* analysis included only diffusion components 1 and 2. We additionally include diffusion component 4, since it is enriched for T cell differentiation according to the GSEA (see Table 2). For the mouse bone marrow dataset, *wishbone* assigns cells to three branches. [26] used the behavior of four markers (CD34, Gata1, Gata2, and Mpo) to verify that the three branches correspond to HSPCs, myeloid progenitors, and erythroid progenitors, and the behavior does not change with the inclusion of component 4. Figs 11 and 12 show the average branch assignment classification ARI for the workflow with the matrix **A** and the workflow with the column submatrix **C** for various $k$, $\epsilon$, and also the three simple column thresholding methods with the same number of columns as the DCSS method for each $k$, $\epsilon$ point. Not all the thresholding methods successfully complete the *wishbone* workflow at large $\epsilon$, due to the sensitivity of the diffusion component GSEA enrichment analysis, which we perform with keyword

**Table 2. *wishbone* diffusion component 4, calculated from the complete data workflow, is enriched for T cell differentiation, according to the *wishbone* GSEA.**

| Gene ontology (GO) term | FDR q-value |
|---|---|
| Component 4, positive enrichment scores | |
| ribosome biogenesis (GO:0042254) | 0.0 |
| ribonucleoprotein complex biogenesis (GO:0022613) | 0.0 |
| rRNA processing (GO:0006364) | 0.0 |
| ncRNA metabolic process (GO:0034660) | 0.0 |
| rRNA metabolic process (GO:0016072) | 0.0 |
| Component 4, negative enrichment scores | |
| T cell differentiation (GO:0030217) | 4.73E-03 |
| second-messenger-mediated signaling (GO:0019932) | 5.52E-03 |
| regulation of actin cytoskeleton organization (GO:0032956) | 5.76E-03 |
| taxis (GO:0042330) | 4.32E-03 |
| regulation of actin filament-based process (GO:0032970) | 5.04E-03 |

https://doi.org/10.1371/journal.pone.0210571.t002

string matching. We find that for the $k = 14$, $\epsilon = 0.05$ DCSS, 31.4% of the total genes give an average ARI of 0.91 for three branch assignments compared to the complete data; this supports our conclusion that only a small fraction of the gene expression profile from scRNA-Seq experiments may be necessary to obtain meaningful cell-type classifications.

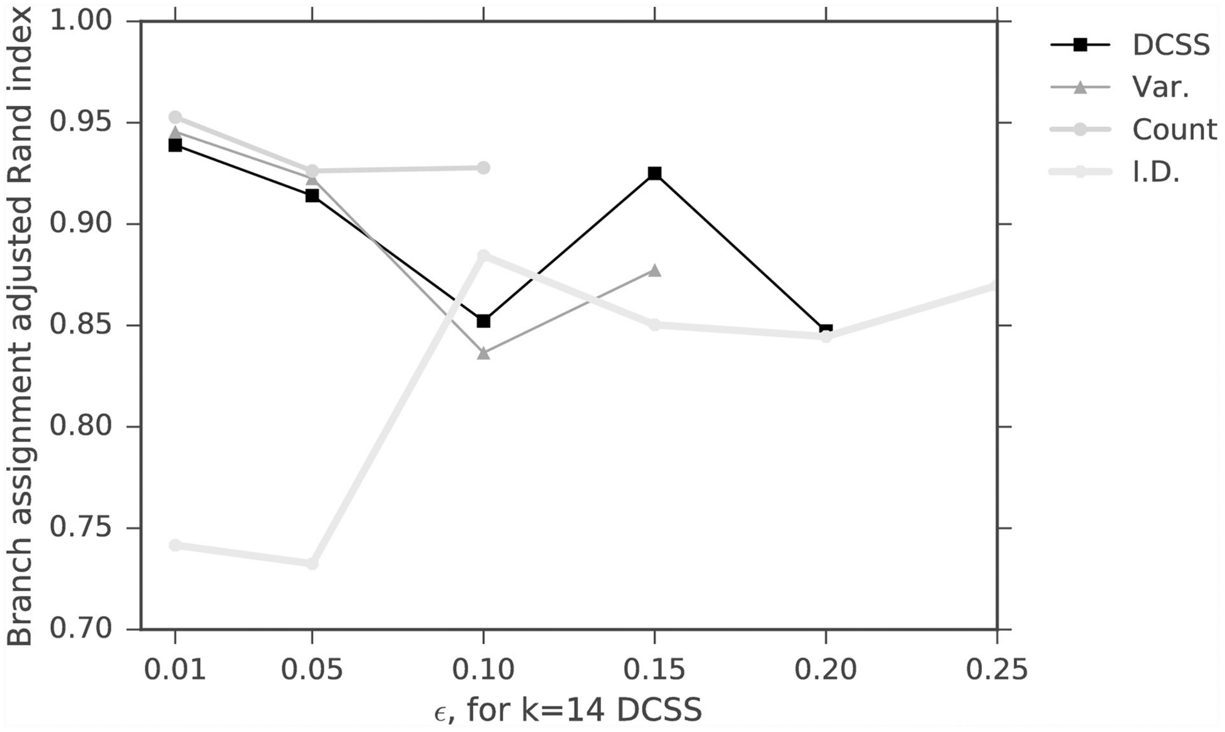

**Fig 11. Average branch assignment ARI, varying the error tolerance $\epsilon$ with $k = 14$ for DCSS for the [25] and [26] dataset.** We also threshold according to count, variance, and index of dispersion. The trend between error tolerance and average ARI is not as clear as in Fig 5.

https://doi.org/10.1371/journal.pone.0210571.g011

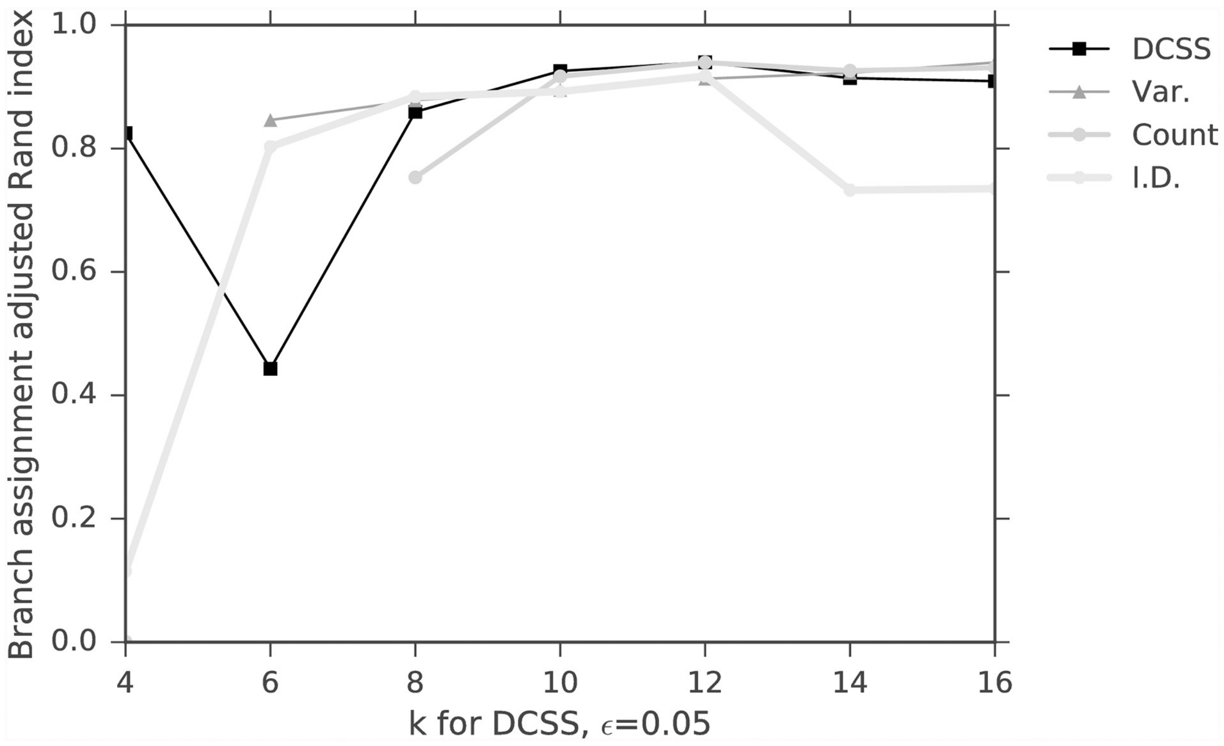

**Fig 12. Average branch assignment ARI, varying the dimension *k* with fixed error tolerance $\epsilon = 0.05$ for DCSS for the [25] and [26] dataset.** We also threshold according to count, variance, and index of dispersion. The trend between dimension and average ARI is not as clear as in Fig 6.

https://doi.org/10.1371/journal.pone.0210571.g012

## Discussion

scRNA-Seq experiments allow researchers to probe the cell-specific heterogeneity in gene expression. Quality control and technical variability are significant challenges for scRNA-Seq experiments, and additionally the whole-genome expression profile is high-dimensional. In this note, we explore three simple column thresholding schemes– count, variance, and index of dispersion– and propose a novel application of a thresholding scheme—DCSS– to select informative features and control quality and technical variability. We prove a novel bound on the "closeness" of the DCSS data submatrix to the complete data matrix (Eq 9). This bound enlarges upon the existing set of error guarantees for DCSS [22] and provides a theoretical advantage over the three simple column thresholding schemes which have no similar guarantees. Other advantages of DCSS over simple column thresholding include sensitivity to collinearity of features and covariance of cells. Since scRNA-Seq experiments are frequently used to cluster and classify cells, we choose to evaluate these thresholding schemes for clustering and classification compared to the complete data and using the ARI.

We present two case studies, the first on mouse cortex and hippocampus scRNA-Seq [4, 24], and the second on mouse bone marrow scRNA-Seq [25, 26]. For the mouse brain, the data matrix is cells × ECs, and only an incredibly small fraction of the ECs are necessary to obtain neuron and non-neuron cell clusters. For the mouse bone marrow, the data matrix is cells × genes, and only a small fraction of the genes are necessary to obtain HSPC, myeloid progenitor, and erythroid progenitor branch assignments. For both case studies, DCSS performs similarly to the simple column thresholding schemes with the same number of columns, in that it reduces the low abundance genes, maintains the most variable and over-dispersed genes, and provides the additional benefit of theoretical guarantees. This supports our recommendation to use DCSS to control quality and technical variability. In both case studies, there is high similarity between the clustering computed with the complete expression profile and the reduced expression profile, suggesting that the clustering algorithms rely on a small subset of informative features.

## Supporting information

**S1 File. The supplementary material document contains supplementary figures, the mathematical definitions, a brief linear algebra review, and the proofs.**
(PDF)

## Acknowledgments

SRM would like to acknowledge Ilan Shomorony, Elaine Angelino, and Robert Tunney for useful comments.

## Author Contributions

**Conceptualization:** Shannon R. McCurdy, Vasilis Ntranos.

**Formal analysis:** Shannon R. McCurdy.

**Methodology:** Shannon R. McCurdy, Vasilis Ntranos.

**Software:** Shannon R. McCurdy.

**Supervision:** Lior Pachter.

**Validation:** Shannon R. McCurdy, Vasilis Ntranos.

**Visualization:** Shannon R. McCurdy.

**Writing – original draft:** Shannon R. McCurdy.

**Writing – review & editing:** Shannon R. McCurdy, Vasilis Ntranos.

## References

1. Tang F, Barbacioru C, Wang Y, Nordman E, Lee C, Xu N, et al. mRNA-Seq whole-transcriptome analysis of a single cell. Nature Methods. 2009; 6(5):377–382. https://doi.org/10.1038/nmeth.1315 PMID: 19349980
2. Stegle O, Teichmann SA, Marioni JC. Computational and analytical challenges in single-cell transcriptomics. Nature Reviews Genetics. 2015; 16(3):133–145. https://doi.org/10.1038/nrg3833 PMID: 25628217
3. Wagner A, Regev A, Yosef N. Revealing the vectors of cellular identity with single-cell genomics. Nature Biotechnology. 2016; 34(11):1145–1160. https://doi.org/10.1038/nbt.3711 PMID: 27824854
4. Zeisel A, Muñoz-Manchado AB, Codeluppi S, Lönnerberg P, Manno GL, Juréus A, et al. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. Science. 2015; 347(6226):1138–1142. https://doi.org/10.1126/science.aaa1934 PMID: 25700174
5. Satija R, Farrell JA, Gennert D, Schier AF, Regev A. Spatial reconstruction of single-cell gene expression data. Nature Biotechnology. 2015; 33(5):495–502. https://doi.org/10.1038/nbt.3192 PMID: 25867923
6. Trapnell C, Cacchiarelli D, Grimsby J, Pokharel P, Li S, Morse M, et al. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nature Biotechnology. 2014; 32(4):381–386. https://doi.org/10.1038/nbt.2859 PMID: 24658644
7. Finak G, McDavid A, Yajima M, Deng J, Gersuk V, Shalek AK, et al. MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA

sequencing data. Genome Biology. 2015; 16:278. https://doi.org/10.1186/s13059-015-0844-5 PMID: 26653891

8. Zheng GXY, Terry JM, Belgrader P, Ryvkin P, Bent ZW, Wilson R, et al. Massively parallel digital transcriptional profiling of single cells. Nature Communications. 2017; 8:14049. https://doi.org/10.1038/ncomms14049 PMID: 28091601

9. McCarthy DJ, Campbell KR, Lun ATL, Wills QF. Scater: pre-processing, quality control, normalization and visualization of single-cell RNA-seq data in R. Bioinformatics. 2017; 33(8):1179–1186. https://doi.org/10.1093/bioinformatics/btw777 PMID: 28088763

10. Lun ATL, McCarthy DJ, Marioni JC. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. F1000Research. 2016; 5:2122. https://doi.org/10.12688/f1000research.9501.2 PMID: 27909575

11. Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. Genome Biology. 2018; 19:15. https://doi.org/10.1186/s13059-017-1382-0 PMID: 29409532

12. Shalek AK, Satija R, Shuga J, Trombetta JJ, Gennert D, Lu D, et al. Single-cell RNA-seq reveals dynamic paracrine control of cellular variation. Nature. 2014; 510(7505):363–369. https://doi.org/10.1038/nature13437 PMID: 24919153

13. Brennecke P, Anders S, Kim JK, Kołodziejczyk AA, Zhang X, Proserpio V, et al. Accounting for technical noise in single-cell RNA-seq experiments. Nature Methods. 2013; 10(11):1093–1095. https://doi.org/10.1038/nmeth.2645 PMID: 24056876

14. Andrews TS, Hemberg M. Dropout-based feature selection for scRNASeq. bioRxiv. 2018; p. 065094.

15. Kharchenko PV, Silberstein L, Scadden DT. Bayesian approach to single-cell differential expression analysis. Nature Methods. 2014; 11(7):740–742. https://doi.org/10.1038/nmeth.2967 PMID: 24836921

16. Pierson E, Yau C. ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. Genome Biology. 2015; 16:241. https://doi.org/10.1186/s13059-015-0805-z PMID: 26527291

17. Lin P, Troup M, Ho JWK. CIDR: Ultrafast and accurate clustering through imputation for single-cell RNA-seq data. Genome Biology. 2017; 18:59. https://doi.org/10.1186/s13059-017-1188-0 PMID: 28351406

18. Tropp JA. Column Subset Selection, Matrix Factorization, and Eigenvalue Optimization. In: Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA'09. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 2009. p. 978–986. Available from: http://dl.acm.org/citation.cfm?id=1496770.1496876.

19. Velleman PF, Welsch RE. Efficient Computing of Regression Diagnostics. The American Statistician. 1981; 35(4):234–242. https://doi.org/10.1080/00031305.1981.10479362

20. Chatterjee S, Hadi AS. Influential Observations, High Leverage Points, and Outliers in Linear Regression. Statistical Science. 1986; 1(3):379–393. https://doi.org/10.1214/ss/1177013622

21. Drineas P, Mahoney MW, Muthukrishnan S. Relative-Error $CUR$ Matrix Decompositions. SIAM J Matrix Anal Appl. 2008; 30(2):844–881. https://doi.org/10.1137/07070471X

22. Papailiopoulos D, Kyrillidis A, Boutsidis C. Provable Deterministic Leverage Score Sampling. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD'14. New York, NY, USA: ACM; 2014. p. 997–1006. Available from: http://doi.acm.org/10.1145/2623330.2623698.

23. Maaten Lvd, Hinton G. Visualizing Data using t-SNE. Journal of Machine Learning Research. 2008; 9 (Nov):2579–2605.

24. Ntranos V, Kamath GM, Zhang JM, Pachter L, Tse DN. Fast and accurate single-cell RNA-seq analysis by clustering of transcript-compatibility counts. Genome Biology. 2016; 17(1):1–14. https://doi.org/10.1186/s13059-016-0970-8

25. Paul F, Arkin Y, Giladi A, Jaitin DA, Kenigsberg E, Keren-Shaul H, et al. Transcriptional Heterogeneity and Lineage Commitment in Myeloid Progenitors. Cell. 2015; 163(7):1663–1677. https://doi.org/10.1016/j.cell.2015.11.013 PMID: 26627738

26. Setty M, Tadmor MD, Reich-Zeliger S, Angel O, Salame TM, Kathail P, et al. Wishbone identifies bifurcating developmental trajectories from single-cell data. Nature Biotechnology. 2016; 34(6):637–645. https://doi.org/10.1038/nbt.3569 PMID: 27136076

27. Wang B, Zhu J, Pierson E, Ramazzotti D, Batzoglou S. Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning. Nature Methods. 2017; 14(4):414–416. https://doi.org/10.1038/nmeth.4207 PMID: 28263960

28. Kwon H, Fan J, Kharchenko P. Comparison of Principal Component Analysis and t-Stochastic Neighbor Embedding with Distance Metric Modifications for Single-cell RNA-sequencing Data Analysis. bioRxiv. 2017; p. 102780.

29. Rao CR. Linear statistical inference and its applications. 2nd ed. Wiley series in probability and mathematical statistics. New York: Wiley; 1973. Available from: http://dx.doi.org/10.1002/9780470316436.

30. Jolliffe IT. Principal component analysis. 2nd ed. Springer series in statistics. New York: Springer; 2002. Available from: http://dx.doi.org/10.1007/b98835.

31. Nicolae M, Mangul S, Măndoiu II, Zelikovsky A. Estimation of alternative splicing isoform frequencies from RNA-Seq data. Algorithms for molecular biology: AMB. 2011; 6(1):9. https://doi.org/10.1186/1748-7188-6-9 PMID: 21504602

32. The Gene Ontology Consortium. Gene Ontology Consortium: going forward. Nucleic Acids Research. 2015; 43(D1):D1049–D1056. https://doi.org/10.1093/nar/gku1179 PMID: 25428369

33. VIB / UGent Bioinformatics and Evolutionary Genomics. Calculate and draw custom Venn diagrams. Available from: http://bioinformatics.psb.ugent.be/webtools/Venn/.

34. Hubert L, Arabie P. Comparing partitions. Journal of Classification. 1985; 2(1):193–218. https://doi.org/10.1007/BF01908075