Scalpel: Information-based Dimensionality Reduction for Rare Cell Type Discovery

Benjamin DeMeo[2,3] and Bonnie Berger[1,3,*]

[1]Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139, USA

[2]Department of Biomedical Informatics, Harvard University, Cambridge, MA 02138, USA

[3]Department of Mathematics, MIT, Cambridge, MA 02139, USA

Abstract

Single-cell RNA-sequencing (scRNA-seq) enables discovery of clinically and biologically interesting populations, but detecting rare cell types is a persistent challenge. Here we introduce Scalpel, a novel technique for extracting interpretable and maximally informative features from single-cell data, enabling population discovery, batch correction, and other downstream analyses at unprecedented resolution. On a collection of cytotoxic T-cells, Scalpel recovers subtle and biologically important populations, including gamma-delta T-cells and MAIT cells, which are invisible to standard pipelines. In multi-batched data, Scalpel effectively removes systemic batch effects, achieving robust and state-of-the-art performance. Unlike other methods, Scalpel is completely unsupervised, human-interpretable, and applicable to both continuous trajectories and clustered data, making it suitable in a wide range of analytic settings.

[*]Correspondence: bab@mit.edu

1 Introduction

Modern single-cell RNA-sequencing technologies measure tens of thousands of genes per cell, yielding high-dimensional datasets that are noisy and challenging to analyze [1]. As a result, nearly all single-cell analyses perform dimensionality reduction to identify a smaller set of features that capture the dataset's structure and diversity. Techniques like Principal Components Analysis (PCA) and Independent Components Analysis (ICA) are widespread, and more recently a range of more sophisticated approaches have emerged for reducing, visualizing, and clustering single-cell datasets [2, 3, 4, 5, 6, 7]. However, existing approaches for learning reduced feature representations heavily prioritize large cellular populations which contribute significantly to the dataset's overall variance, and lose information about smaller sub-populations that are biologically relevant [7].

Here, we present Scalpel, a novel technique for generating robust featurizations which recover rare cell populations and fine transcriptional substructures at unprecedented resolution. Scalpel leverages the mathematical notion of Shannon Information [8], whereby less probable events are considered more informative, to assign a significance score to each gene in each cell based on a statistical comparison of local and global expression. Rarer genes yield higher information scores where they are expressed, boosting the signal of small populations which may be identified by just a few key markers. These scores are used to generate a set of maximally-informative metagenes, which are used as features in downstream analyses.

Unlike other recent techniques, Scalpel is completely unsupervised, requiring no input information aside from the transcript counts. Furthermore, it makes no direct assumptions about the structure of the data, and is suitable for continuous trajectories as well as discrete clusterings. This contrasts with other recent approaches which require information about the number of clusters, or access to an existing atlas of known cell types [9, 10]. Scalpel's output is biologically interpretable, with a mathematically well-defined linear relationship to the input transcripts. This contrasts with recently-proposed nonlinear approaches [5, 6, 11], whose outputs are a complex function of the inputs. Such methods may be prone to undetectable methodological artifacts, and may be difficult to interpret biologically.

On a variety of datasets, both real and simulated, Scalpel recovers small populations of cells which are invisible to existing pipelines, including Mucosal-associated Invariant T-Cells and gamma-delta T-cells. At the same time, Scalpel preserves larger-scale distinctions between experimentally-determined cell types, and reproduces continuous trajectories with high fidelity. In addition to dimensionality reduction, we use Scalpel's scores to remove systemic batch effects in multi-batched data, achieving state-of-the-art performance with a simple feature reduction.

Scalpel leverages mathematical information theory to create featurizations which accurately reflect the true diversity of transcriptomic data. More broadly, we hope that Scalpel's information-theoretic paradigm forms a foundation for further innovations in feature extraction, both in single-cell analysis and elsewhere.

2 Results

2.1 Overview of Scalpel

Scalpel first converts the raw transcript counts X into a matrix S encoding the significance of each gene in each cell, which we call the *Score Matrix*. For each cell c_i and gene g_j, Scalpel identifies the k nearest neighbors of c_i, and performs a two-tailed binomial test to see whether g_j is expressed in more or fewer of these k neighbors than expected, given g_j's global probability of expression p_j (Figure 1a). The negative log p-value of this test can be interpreted as the Shannon information of g_j's local expression, and provides the magnitude of S_{ij}. The sign of S_{ij} is positive if g_j is over-expressed in c_i's neighborhood, and negative if it is under-expressed.

As shown in Figure 1b, this strategy gives genes high positive scores where they are markers, scores near zero where they represent noise, and low negative scores where they are conspicuously absent. Rare genes receive higher scores in regions where they are expressed, representing greater surprisal of encountering that gene. On the other hand, genes that are near-universally expressed receive strongly negative scores in regions that do not express them. Thus, small populations marked by the presence of a rare gene or the absence of a common one have a greater impact on the score matrix S.

Scalpel next identifies the linear combinations of genes, or metagenes, that encode the most information.

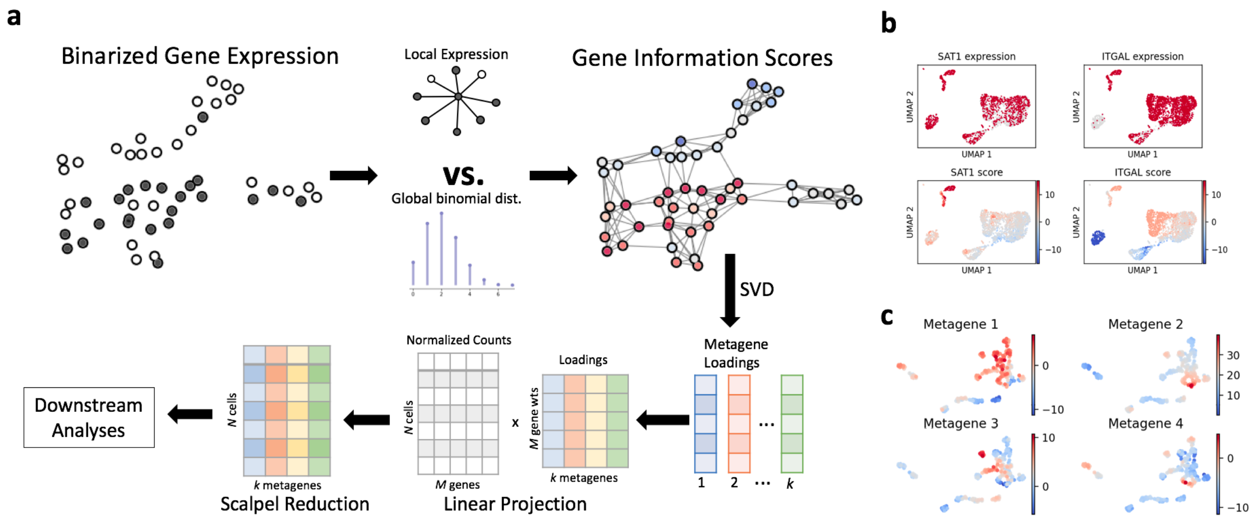


Figure 1: **a**: Overview of Scalpel. A binomial test for local over-expression yields information scores for each gene. SVD on the resulting matrix outputs the most informative metagenes, which are then used for downstream clustering and analysis. **b**: Example Gene Information Scores on SS3 PBMC data. *SAT1* is noisily expressed throughout the data, but achieves a high information score only where it is constitutively expressed. *ITGAL* achieves a strongly negative information score where it is conspicuously absent. **c**: Heatmaps of Scalpel metagenes on Scalpel-derived UMAP plots of the cytotoxic T-cell subset.

As shown in the Supplementary Methods, these are simply the right-eigenvectors of S with highest eigenvalue, i.e. its principal component vectors, which we denote $v_1, v_2, ..., v_m$. For a given target dimensionality D, the metagenes encoded by $v_1, v_2, ..., v_D$ act as the new features for the dimensionality-reduced X. In mathematical terms, we linearly project the original dataset to the span of $v_1, .., v_D$. This is like principal component analysis, but where the PCs are computed on S rather than X. Figure 1c shows an example of Scalpel metagene expression over a collection of cytotoxic T-cells analyzed below.

Although the process that generates $v_1, ..., v_D$ is nonlinear, requiring nearest-neighbor graphs and binomial score computation, Scalpel's output is a linear projection of its input onto their span. This places Scalpel firmly in the linear category, together with PCA and ICA; indeed, for fixed D, the coordinate systems defined by Scalpel and by these methods are related by rotation. Since linear projection does not stretch or shrink the data, this class of algorithms is particularly robust to distortion.

2.1.1 Iterative Bootstrapping

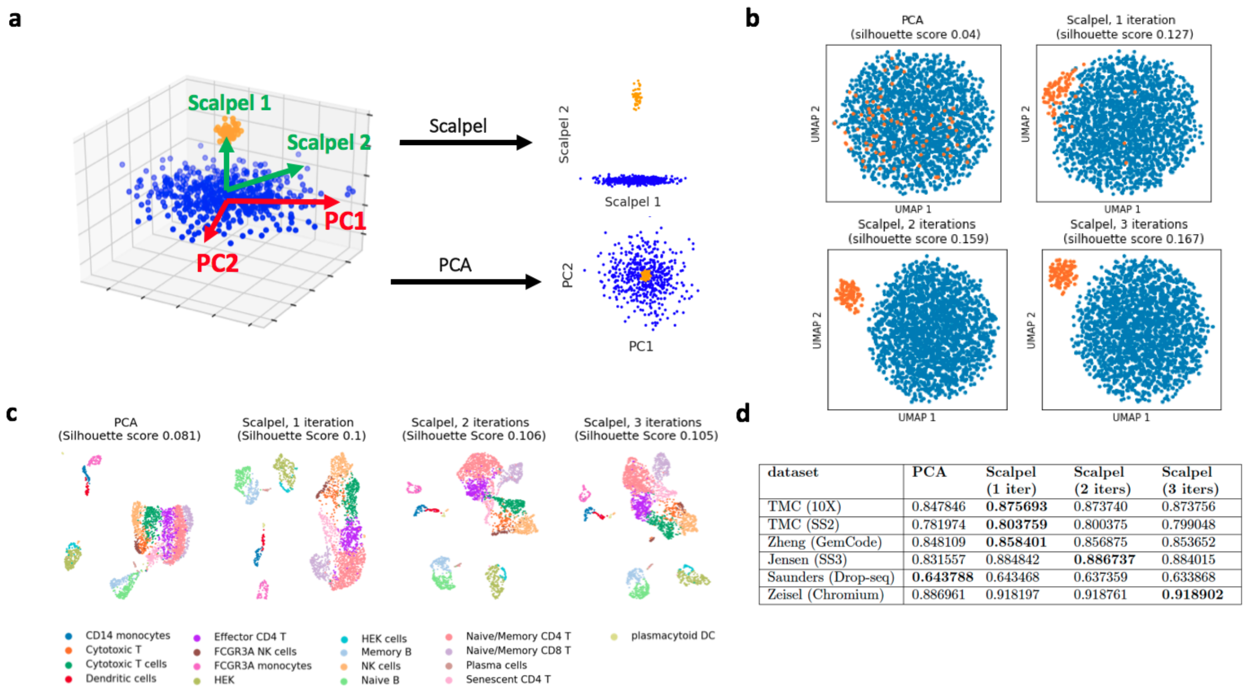
The above description leaves open the issue of choosing a metric for determining the k-nearest neighbors of a given cell c_i. By default, Scalpel uses Euclidean distance in PCA space, with user-configurable parameters. However, since Scalpel's reductions outperform PCA in many regards, we can instead use a Scalpel reduction to define nearest neighbors in a second application of Scalpel, generating another dimensionality reduction on the same dataset. This process can be repeated arbitrarily many times. This iteration further boosts the signal of rare cell populations, and often improves the discriminative power of the representation (Figure 2b).

Regardless of the number of iterations Scalpel is run for, the final output is still just a linear projection of the input dataset X – iteration simply refines the subspace onto which Scalpel projects. Thus, Scalpel retains the robustness inherent to linear methods, as discussed above. In practice, we observe improved performance with more iterations (Figure 2).

2.2 Scalpel Distinguishes Synthetic Clusters

To test Scalpel's power to recover rare cell populations in the presence of noise, we generated a synthetic dataset with 2000 cells and 105 genes. Five genes are expressed only in a subpopulation of 80 cells (4% of the total), and the remaining 100 genes are expressed uniformly at random. We reduced this dataset using a standard PCA pipeline, and using Scalpel with 1-3 iterations (Methods). For each reduction, we assessed separation visually using UMAP coordinates, and quantitatively by assessing the silhouette score [12].

The results are shown in Figure 2. In all metrics, Scalpel drastically outperforms PCA. PCA with 50 components fails to separate the two populations, likely because the noisy genes account for more of the global variance and receive high loadings. On the other hand, a set of 50 features produced with Scalpel produces a very good separation after one iteration, and a near-perfect separation after two, with substantial improvement in the silhouette score. This improvement is due to Scalpel correctly identifying the marker genes as having high information, and thus giving them higher impact on the final reduction.



dataset	PCA	Scalpel (1 iter)	Scalpel (2 iters)	Scalpel (3 iters)
TMC (10X)	0.847846	**0.875693**	0.873740	0.873756
TMC (SS2)	0.781974	**0.803759**	0.800375	0.799048
Zheng (GemCode)	0.848109	**0.858401**	0.856875	0.853652
Jensen (SS3)	0.831557	0.884842	**0.886737**	0.884015
Saunders (Drop-seq)	**0.643788**	0.643468	0.637359	0.633868
Zeisel (Chromium)	0.886961	0.918197	0.918761	**0.918902**

Figure 2: **a**: Schematic cartoon of synthetic test data, explaining the difference in performance. A small population of 80 cells is distinguished from a large and dispersed population by the expression of a 5 marker genes. Principal components align with the higher-variance cluster, whereas Scalpel detects the smaller cluster due to higher information. **b**: UMAP plots generated using PCA and using Scalpel with 1-3 iterations. Iterating Scalpel improves recovery of a rare population of 80 cells (4%) expressing 5 marker genes. **c**: UMAP plots of the smart-seq 3 dataset after transformation with PCA, or Scalpel for 1-3 iterations. **d**: Fidelity of near-neighbor graphs under different pre-processing steps. Graphs were computed by connecting each cell to its 15 nearest neighbors in the reduction computed by each method.

2.3 Scalpel better recovers known cell types

We next tested whether Scalpel's reductions better separate known populations of cells in real, large-scale single-cell datasets. We selected 6 scRNA-seq datasets spanning a wide range of technologies [13, 14, 15, 16, 17], with experimentally-determined or expert-curated cell type annotations (see Methods for details). Each dataset was first subset to at most 5000 cells using Hopper [18]. As above, we computed reductions using PCA or Scalpel with 50 components. For all samples, we ran Scalpel with a neighborhood size of 15 cells. For each reduction, we assessed the *fidelity*, defined as the percentage of edges in the 15-nearest neighbor graph which connect cells of the same type (Methods).

In all but one of the tested datasets, Scalpel reductions increased the fidelity after a single iteration, with additional iterations sometimes increasing it further (Figure 2d). This effect is visible in UMAP plots produced downstream: Scalpel-derived embeddings show better separation between cell types, and generally improve with more iteration (Figure 2c; Supplementary figure 1). Furthermore, cell types that appear as globular clusters in PCA space often acquire additional structure in the Scalpel-derived visualization, especially immune cells (Figure 2c; Supplementary Figure 1). We examine this structure more closely in the next section.

2.4 Scalpel uncovers rare immune subpopulations

In Scalpel reductions of many tested datasets, we observed fine-grained sub-populations of cells which are not apparent from PCA-derived reductions. Immune populations were particularly rich in this regard. We therefore sought to characterize these clusters and assess their biological significance.

We applied Scalpel and PCA to the cytotoxic T-cell subpopulation of the SS3 dataset, comprising 307 cells. Due to the smaller number of cells, we used only 20 dimensions for each method. For Scalpel, we ran three iterations with a neighborhood size of 15. As above, we computed a Euclidean 15-nearest neighbor graph in each reduction using scanpy [19], followed by Louvain clustering [20], UMAP visualization [3], and identification of differentially-expressed genes (Methods).

As shown in Figure 3a, Scalpel's embedding was far more granular, with 13 well-isolated clusters, com-

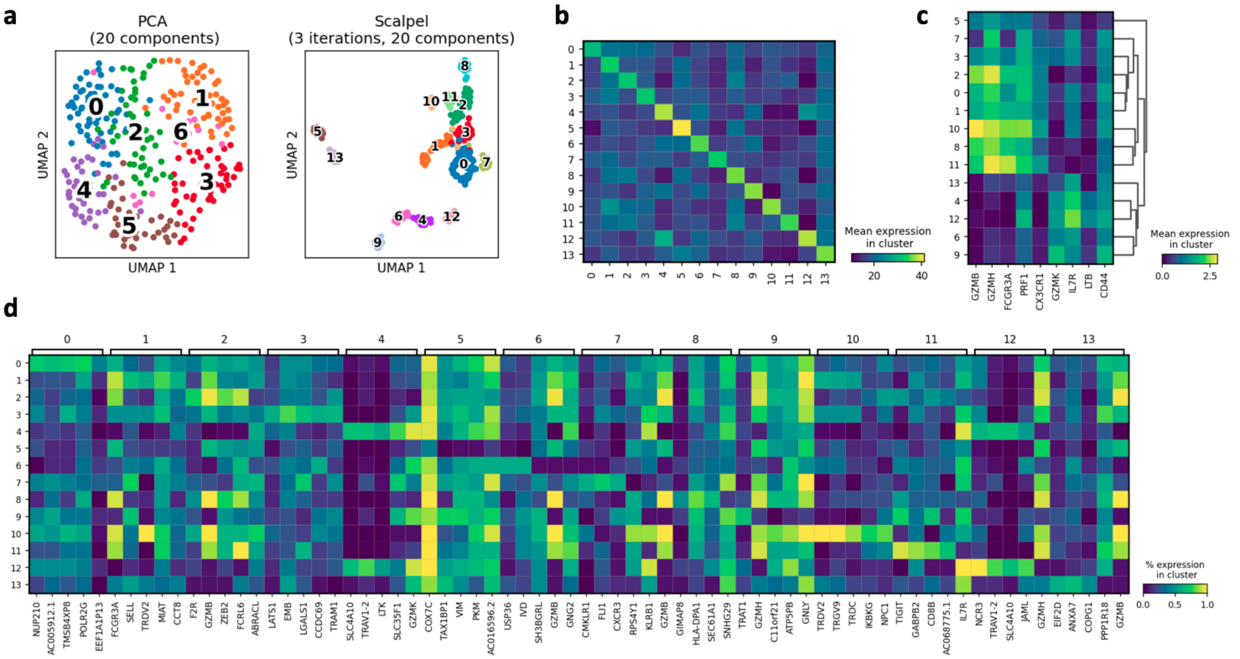


Figure 3: Scalpel recovers small cytotoxic T-cell subpopulations. **a**: Comparison of UMAP embeddings computed using PCA reductions (left), or Scalpel reductions (right). Scalpel's clustering is far more granular. **b**: Matrix plot of marker metagene expression in each cluster. Each marker metagene is the sum of 50 binarized DE genes of the corresponding cluster, with positive sign if the gene is up-regulated and negative sign if the gene is down-regulated. **c**: Matrix plot showing separation of clusters by cytotoxicity markers. **d**: Full matrix plot showing the top 5 DE genes of each cluster, colored by frequency of expression. Within clusters, genes are ordered from most over-expressed to most under-expressed.

pared to 5 somewhat overlapping clusters in the PCA representation. As discussed above, Scalpel's linearity precludes extreme distortion of the data; nevertheless, we performed several checks to ensure that Scalpel's clusters represent biologically relevant variation. We first verified that the clusters separated cells expressing Granzyme B, a key cytotoxic marker [21], from those that do not (Figure 3c). We then performed differential gene expression analysis to identify marker and/or dropout genes in each cluster, formed marker metagenes in each group incorporating the top 50 DE genes with equal weights (Methods), and verified that they strongly distinguished the 13 clusters (Figure 3b). This shows that each cluster has a robust biological signal, motivating further analyses.

Figure 3 shows a differential expression matrix for the top five DE genes identifying each cluster. Granzyme B and Granzyme K show largely disjoint expression, and coarsely partition the clusters. The Granzyme B-expressing clusters are elevated in several other markers of cytotoxicity, such as Perforin (*PRF1*), CD16 (*FCGR3A*), and Granzyme H (*GZMH*), suggesting that these cells are actively lysing other cells. Within the Granzyme B-expressing compartment, clusters are distinguished by combinatorial expression of various receptors, transcription factors, and metabolic proteins. For example, Cluster 0 marks a region of active transcription and translation, evidenced by increased expression of elongation factors (e.g. *EEF1A1P13*) and nucleoporins (*NUP210*), and by elevated total transcript counts (Supplementary Figure 2). Cluster 10 contains gamma-delta T-cells, evidenced by constitutive expression of both the delta and gamma chains of T-cell receptors (*TRDV2* and *TRGV9*, respectively) [22].

The Granzyme K-expressing populations are also heterogenous. Clusters 4 and 12 show highly elevated expression of *SLC4A10* and *TRAV1-2*, and lack *CX3CR1* and *CD16*, suggesting Mucosal Associated Invariant T (MAIT) populations [23]. Cluster 12 is further distinguished by elevated expression of several genes, including *NCR3*, *LPXN*, and *KLRG1*. Clusters 5 and 13 are marked by decreased expression of many genes, possibly indicating poor cell quality or hybridization errors; indeed, both show significantly fewer total transcripts, and a higher percentage of mitochondrial and ribosomal genes (Supplementary Figure 2). Cluster 6 likewise shows few transcript counts, but is slightly enriched in the ubiquitin-specific peptidase *USP36*; *ABCA7*, which is associated with phagocytosis [24]; and *LINC02273*, which has been implicated in

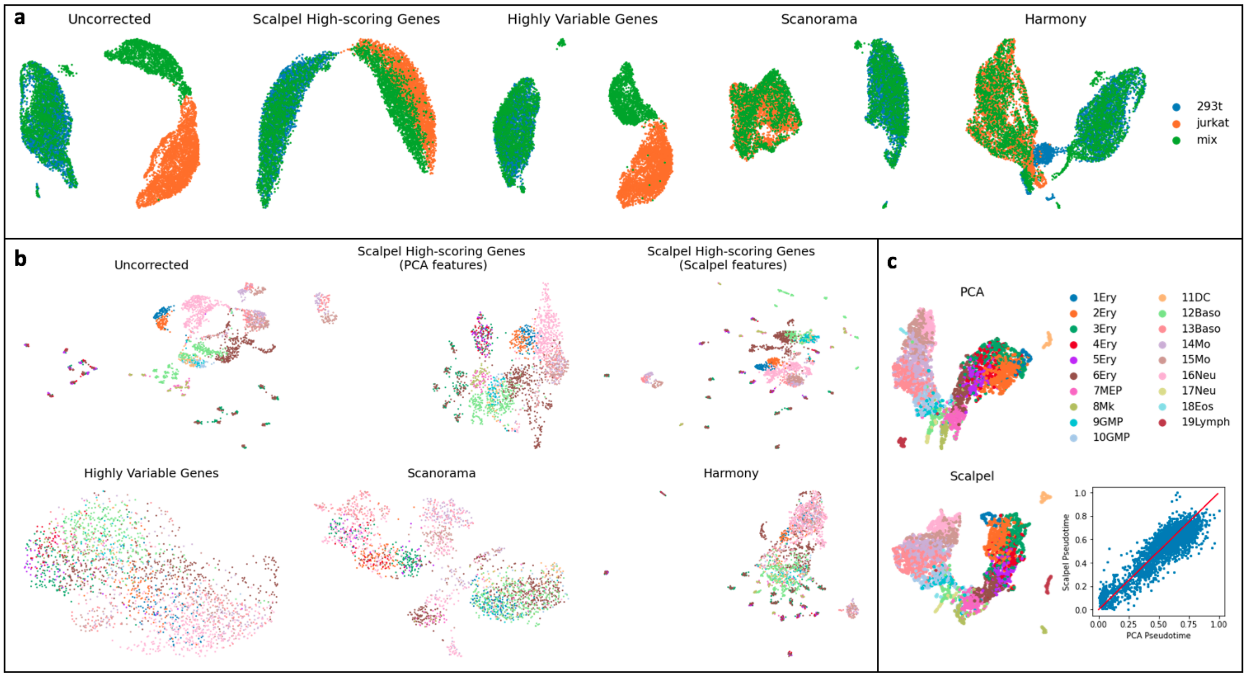


Figure 4: **a**: UMAP plots of 293T and Jurkat datasets from the Scanorama paper [26], integrated through various approaches. One batch (green) contains 293T cells; one (orange) contains Jurkat cells, and one (Blue) contains an equal mixture of both. **b**: UMAP plots of cytotoxic T-lymphocyte data from Patil et. al. [27] after integration with various approaches. Each UMAP is colored by donor. **c**: PCA and Scalpel-derived UMAPs of developing myeloid progenitors from Paul et. al. [28]. Scalpel accurately captures the continuous nature of the data. Using PAGA [29], diffusion pseudotime computed on the Scalpel representation closely matches diffusion pseudotime computed on the PCA representation (bottom right).

breast cancer metastasis [25].

Thus, Scalpel enables the detection of small cellular populations of biological, clinical, and methodological importance, which are not recovered through standard pipelines.

2.5 Scalpel Corrects Systemic Batch Effects

Integrating data from different technical and biological replicates is a central problem in scRNA-seq, and the goal of many existing methods [26, 30, 5, 31, 11]. Batch effects arise from a variety of technical factors, such as capture efficiency, sample quality, and library size. We hypothesize that many batch effects occur uniformly across each sample, producing a noisy signal independent of cell type. Since Scalpel's score matrices eliminate such signals, we hypothesized that Scalpel should be able to remove these systemic batch effects simply by filtering genes based on their score.

We implemented Scalpel-based batch correction by keeping only those genes which are highly informative in at least one batch. We applied Scalpel for three iterations to produce score matrices for each batch, ranked genes by the highest score attained across all cells in all batches, and kept only the top genes. This protocol is implemented as a single function with tunable parameters within the Scalpel API.

As with other Scalpel functions, this approach is conceptually very simple (it simply filters genes), and has limited capacity to distort the data. Other methods vary in the degree of batch correction, and may over-correct, leading to variable performance (Figure 4).

We ran Scalpel's integration protocol on two datasets, keeping the top 1000 top-scoring genes after three iterations of Scalpel on each batch. On 293T/Jurkat cell mixtures from the Scanorama paper [26], Scalpel successfully merged the two Jurkat cell populations across batches, performing similarly to Scanorama [26] and Harmony [30] (Figure 4a). Simply taking the top 1000 most variable genes failed to achieve this, showing that variable genes are not necessarily informative.

On another dataset consisting of cytotoxic T-lymphocytes from 8 donors [27], Scalpel again merges cells from different batches, whilst preserving major clusters (Figure 4b). Here, Scanorama and Harmony produce drastically different embeddings: Scanorama merges several small clusters on the periphery, whereas Harmony keeps them separate. Scalpel's embedding merges some of these clusters while keeping others separate in the PCA-derived UMAP; however, the Scalpel-derived UMAP of the integrated data separates all of the small clusters again. Notably, taking highly variable genes completely loses the finer structural details, reflecting the discriminative power of rare, low-variance genes.

The variability in the outputs of integration methods are the result of variable assumptions and stylistic decisions. Scalpel shows that state-of-the-art performance can be achieved via basic feature selection, which is inherently simpler, more conservative, and easier to interpret than existing nonlinear approaches.

2.6 Scalpel Preserves Continuous Trajectories

Scalpel excels at identifying small clusters when they are present; however, unlike other approaches, it does not assume that the data admits a natural partition into clusters. In particular, Scalpel embeds continuous trajectories smoothly, and preserves pseudotime inference. To demonstrate this, we ran Scalpel on a dataset of developing myeloid progenitors from Paul et. al [28]. Scalpel's UMAP embedding is nearly identical to that of PCA, retaining the continuity of the trajectory (Figure 4c). We then inferred pseudotime using PAGA [29] on each of the two embeddings, and obtained similar pseudotime values regardless of the embedding used ($R^2 = 0.82$; Figure 4c).

3 Discussion

Scalpel provides a mathematically principled approach to extracting signal from noise in high-dimensional datasets that is powerful, interpretable, and broadly applicable. Its power derives from a novel, locality-sensitive, and statistically sound assessment of the significance of each transcript count, amplifying true transcriptional differences whilst reducing transcriptional noise. Rare populations and small-scale structures receive higher scores, enabling their discovery in downstream analyses.

Scalpel's information scores are similar in principle to Inverse Document Frequency (IDF), a normalization approach widely used in text processing and in some single-cell applications, whereby each feature (gene) is weighted by the logarithm of its inverse frequency. GiniClust [7], a more tailored approach, selects genes with high Gini index, a measure of statistical dispersion which detects unequal expression across the data. However, these approaches are only gene-specific, assigning genes the same importance wherever they are expressed. By incorporating counts from local neighborhoods of each cell, Scalpel allows genes to have variable scores across the dataset, achieving high-magnitude scores where they are discriminative and near-zero scores where they are noise (Figure 1b). This reflects true biology, where genes may be expressed sporadically across the entire dataset, but mark informative distinctions only within a small subpopulation.

Scalpel is linear, and therefore related to other linear dimensionality reduction methods (e.g. PCA) by change of basis. Intuitively, Scalpel simply changes the "perspective" from which the data is viewed. It is remarkable, then, that Scalpel's reductions look so different in downstream analyses from those of PCA (see e.g. Figure 3a). This is possible because high-dimensional space offers a far wider variety of perspectives than the three-dimensional space we often think in, giving linear methods more richness than they are usually credited for.

RNA sequencing arose from the need to understand biological tissues at cellular resolution, and as the field evolves, it becomes ever more important to recover clinically relevant cellular populations. Scalpel offers a robust, scalable, and interpretable framework for doing so. Scalpel is implemented in Python, and offers integration with scanpy, a popular scRNA processing framework. It is freely available at `github.com/bendemeo/scalpel`.

4 Methods

4.1 Synthetic Data Experiments

The synthetic dataset analyzed in Section 2.2 was generated using numpy [32] and scanpy [19]. Expression levels are binarized, spanning 105 genes and 2000 cells. 100 of the genes are expressed uniformly at random throughout the entire population, with each cell having a 50% chance of expressing each gene. Five "marker genes" are expressed only in the same subgroup of 80 cells. The marker genes define two populations, indicated by color in Figure 2b.

For PCA representations, we compute 50 PCs using scanpy's `pca` function, which in turn uses the implementation of `scikit-learn` [33], with default parameters. We ran Scalpel with a neighborhood size of 5 for 1-3 iterations, starting with the PCA embedding and keeping 50 metagenes after each iteration. In each representation, we computed the 15-nearest-neighbor graph using cosine distance, and used the resulting graphs to generate the UMAP plots shown in Figure 2b. Silhouette scores were computed using the `silhouette_score` method from `scikit-learn` [33, 12].

4.2 Dataset Fidelity Tests

We downloaded high-throughput single-cell datasets using accessions provided by the original study authors, as follows:

dataset	Accession	DOI
TMC datasets [13]	GSE132042	10.1101/661728
Zheng [14]	SRP073767	10.1038/ncomms14049
Jensen SS3 [15]	E-MTAB-8735	10.1038/s41587-020-0497-0
Saunders [16]	GSE116470	10.1016/j.cell.2018.07.028
Zeisel [17]	SRP135960	10.1016/j.cell.2018.06.021

Table 1: Accessions and DOIs for the datasets summarized in Figure 2d.

True cell type labels were determined through a variety of experiments by the original authors, and obtained from the accessions above. Below, we briefly summarize the techniques used to determine cell type labels, and refer the reader to the original studies for more details:

- **TMC datasets**: Cells were processed with droplet-based scRNA-seq and fluorescence-activated cell sorting (FACS). Initial clusters were determined by Louvain and Leiden-based clustering, mapped to an existing mouse cell atlas, and reviewed by experts in each of the tissues profiled. More details are available at the authors' github: `https://github.com/czbiohub/tabula-muris-senis`

- **Zheng PBMCs**: Raw counts were first processed *in silico* using K-means clustering, and the resulting clusters were refined through marker gene analysis. The authors then generated reference transcriptome profiles through scRNA-seq of bead-enriched subpopulations, identifying purified populations to which the original transcripts were matched, further enhancing the clustering.

- **Jensen SS3**: Clusters represent several different samples from the Human Cell Atlas benchmarking dataset, with cells from several tissues combined and cryopreserved before sequencing. Cell types were deduced from the tissue of origin, together with manual marker gene analysis for more complex tissues (e.g. PBMCs).

- **Saunders**: This dataset contains cells from 9 different regions of the mouse brain, which were separately analyzed using a two-stage procedure. The cells were first clustered *in silico* via ICA and modularity-based clustering, then refined using Independent Components (ICs) selected manually by experts to be biologically meaningful.

- **Zeisel**: The authors produced an initial clustering of cytograph, then verified concordance with six previously published and experimentally validated scRNA-seq datasets. The authors then computed trinarization and enrichment scores, identifying marker gene sets identifying each cluster with high probability.

Before applying Scalpel or PCA, we subset each dataset to 5000 cells using Hopper [18], and natural log-transformed the raw counts with a pseudocount of 1 – that is, for each count c,

$$\text{log1p}(c) = \ln(c+1)$$

50 Principal components were computed using the `pca` function of scanpy [19, 33]. We ran Scalpel on each dataset with a neighborhood size of 15, using the PCA representation to generate the initial neighborhood graph, and keeping the top 50 metagenes in the final representation. For both representations, near-neighbor graphs were generated by connecting each cell to its 15 nearest neighbors. We used the cosine distance to assess nearness, which resulted in superior fidelity over Euclidean distance for all datasets regardless of representation.

As described above, the fidelity of a representation R is the fraction of edges in the near-neighbor graph G_R which connect cells of the same type, where G_R is computed using cosine distance in the representation R. That is:

$$\text{fidelity}(R) = \frac{|\{(i,j) \in E(G_R) : \text{label}(c_i) = \text{label}(c_j)|\}}{|E(G_R)|}.$$

Note that fidelity falls between 0 and 1, with 1 indicating that cells are always closer to cells of the same label.

4.3 Cytotoxic T-cell Population Discovery

We extracted all cytotoxic T-cells from the Jensen SS3 dataset using the authors' cell type annotations, obtaining 307 cells in total. For PCA, we used scanpy's `pca` function with 20 components. For Scalpel, we ran three iterations with 20 components each, starting with the PCA representation. 15-nearest neighbors graphs were computed in both representations using Euclidean distance, and the results were used to generate the UMAP plots in Figure 3.

Differentially-expressed genes were computed via one-vs-rest logistic regression on the binarized gene counts, using scanpy's `rank_genes_groups` function. Underexpressed genes were retained by setting `rankby_abs=False`. Matrix plots were then generated using scanpy's `rank_genes_groups_matrixplot` function on the binarized counts. To create marker metagenes for each cluster, we added together the overexpressed genes and subtracted the underexpressed genes, among the top 50 significantly DE genes.

4.4 Integration

293T and Jurkat cell mixtures were downloaded from `http://scanorama.csail.mit.edu/`, provided by the authors of Scanorama [26], who originally obtained them from Zheng et. al. [14]. CD4+ T-lymphocytes were downloaded from the European Bioinformatics Institute [34] under accession E-GEOD-106540, accessed via scanpy's `ebi_expression_atlas` function.

293T/Jurkat data were log-transformed, and T-lymphocyte data binarized, as these transformations produced the clearest visual separation between known clusters regardless or representation. For both methods, we applied Scanorama with default parameters, and Harmony on the PCA representation with 50 PCs. To generate Scalpel information scores, we used 3 iterations with 20 components each. The same parameters were used to generate a Scalpel embedding for the representation pictured in the top right of Figure 4b. Highly variable genes were identified for comparison using Scanpy's `highly_variable_genes` function.

4.5 Trajectory Analysis

Our analysis followed the vignette at `https://scanpy-tutorials.readthedocs.io/en/latest/paga-paul15.html`. 20 components were computed in each representation (PCA and Scalpel). 4-nearest-neighbor graphs were used to generate the UMAPs in Figure 4c, and to compute Leiden clusters in each representation [35]. We then computed diffusion maps with scanpy's `diffmaps` function, re-computed clusters in diffusion space based on a 10-nearest-neighbor graph, and applied PAGA to the result. Following the vignette, we identified pseudotime roots for each PAGA graph by marker gene expression, and applied scanpy's `dpt` function, an implementation of Diffusion pseudotime [36], to compute pseudotimes for each cell in each representation.

4.6 Scalpel Implementation Details

Scalpel is written in Python. Binomial scores were computed using the `binom_test` function from scipy [37]. Singular value decompositions were computed using the `pca` function from the Python package `fbpca`.

Since computing a binomial test for each gene and cell is time-consuming, we amortized this computation by pre-computing binomial scores for each global gene frequency and neighborhood occurrence rate, for a fixed neighborhood size. This requires around kM computations for a neighborhood size of k, rather than MN computations naively, with identical performance.

As a further speedup and memory reduction, we optionally allow dividing of the gene frequencies into a fixed number B of bins, requiring only kB binomial score computations. Bin divisions can optionally be chosen so that the resulting frequency approximation is accurate to within a constant factor, with bins spaced according to a power law. In practice, this means that the binning is denser at lower probabilities, allowing more accurate binomial scores for rare genes. This improves slightly over equally-spaced bins, and is Scalpel's default behavior. All of the above experiments were performed with $B = 500$ evenly-spaced bins.

5 Acknowledgements

The authors are grateful to Ashwin Narayan, Brian Hie, Sarah Nyquist, Rohit Singh, Ellen Zhong, and other members of the Berger lab for their continual support and helpful feedback. We also thank Josh Peters for helpful conversations on Scalpel's applications. This work was supported by the National Institutes of Health [R01GM081871 and R01GM108348 to B.B.].

References

[1] Saliba, A.-E., Westermann, A. J., Gorski, S. A. & Vogel, J. Single-cell rna-seq: advances and future challenges. *Nucleic acids research* **42**, 8845–8860 (2014).

[2] Narayan, A., Berger, B. & Cho, H. Density-preserving data visualization unveils dynamic patterns of single-cell transcriptomic variability. *bioRxiv* (2020).

[3] McInnes, L., Healy, J. & Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[4] Maaten, L. v. d. & Hinton, G. Visualizing data using t-sne. *Journal of machine learning research* **9**, 2579–2605 (2008).

[5] Lopez, R., Regier, J., Cole, M. B., Jordan, M. I. & Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature methods* **15**, 1053–1058 (2018).

[6] Tian, T., Wan, J., Song, Q. & Wei, Z. Clustering single-cell rna-seq data with a model-based deep learning approach. *Nature Machine Intelligence* **1**, 191–198 (2019).

[7] Jiang, L., Chen, H., Pinello, L. & Yuan, G.-C. Giniclust: detecting rare cell types from single-cell gene expression data with gini index. *Genome biology* **17**, 144 (2016).

[8] Shannon, C. E. A mathematical theory of communication. *The Bell system technical journal* **27**, 379–423 (1948).

[9] 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* **14**, 414–416 (2017).

[10] Han, X. *et al.* Mapping the mouse cell atlas by microwell-seq. *Cell* **172**, 1091–1107 (2018).

[11] van Dijk, D. *et al.* Magic: A diffusion-based imputation method reveals gene-gene interactions in single-cell rna-sequencing data. *BioRxiv* 111591 (2017).

[12] Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* **20**, 53–65 (1987).

[13] Schaum, N. *et al.* Single-cell transcriptomics of 20 mouse organs creates a tabula muris: The tabula muris consortium. *Nature* **562**, 367 (2018).

[14] Zheng, G. X. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nature communications* **8**, 1–12 (2017).

[15] Hagemann-Jensen, M. *et al.* Single-cell rna counting at allele and isoform resolution using smart-seq3. *Nature Biotechnology* **38**, 708–714 (2020).

[16] Saunders, A. *et al.* Molecular diversity and specializations among the cells of the adult mouse brain. *Cell* **174**, 1015–1030 (2018).

[17] Zeisel, A. *et al.* Molecular architecture of the mouse nervous system. biorxiv. *Preprint]* **10** (2018).

[18] DeMeo, B. & Berger, B. Hopper: a mathematically optimal algorithm for sketching biological data. *Bioinformatics* **36**, i236–i241 (2020).

[19] Wolf, F. A., Angerer, P. & Theis, F. J. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology* **19**, 15 (2018).

[20] BLONDEL, V. *et al.* Fast unfolding of communities in large networks. 25 jul. 2008 (2018).

[21] Afonina, I. S., Cullen, S. P. & Martin, S. J. Cytotoxic and non-cytotoxic roles of the ctl/nk protease granzyme b. *Immunological Reviews* **235**, 105–116 (2010). URL `https://onlinelibrary.wiley.`

com/doi/abs/10.1111/j.0105-2896.2010.00908.x. https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.0105-2896.2010.00908.x.

[22] Holtmeier, W. & Kabelitz, D. $\gamma\delta$ t cells link innate and adaptive immune responses. In *Mechanisms of epithelial defense*, vol. 86, 151–183 (Karger Publishers, 2005).

[23] Treiner, E. *et al.* Selection of evolutionarily conserved mucosal-associated invariant t cells by mr1. *Nature* **422**, 164–169 (2003).

[24] Iwamoto, N., Abe-Dohmae, S., Sato, R. & Yokoyama, S. Abca7 expression is regulated by cellular cholesterol through the srebp2 pathway and associated with phagocytosis. *Journal of lipid research* **47**, 1915–1927 (2006).

[25] Xiu, B. *et al.* Linc02273 drives breast cancer metastasis by epigenetically increasing agr2 transcription. *Molecular cancer* **18**, 1–20 (2019).

[26] Hie, B., Bryson, B. & Berger, B. Efficient integration of heterogeneous single-cell transcriptomes using scanorama. *Nature biotechnology* **37**, 685–691 (2019).

[27] Patil, V. S. *et al.* Precursors of human cd4+ cytotoxic t lymphocytes identified by single-cell transcriptome analysis. *Science immunology* **3** (2018).

[28] Paul, F. *et al.* Transcriptional heterogeneity and lineage commitment in myeloid progenitors. *Cell* **163**, 1663–1677 (2015).

[29] Wolf, F. A. *et al.* Paga: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome biology* **20**, 1–9 (2019).

[30] Korsunsky, I. *et al.* Fast, sensitive and accurate integration of single-cell data with harmony. *Nature methods* 1–8 (2019).

[31] Polański, K. *et al.* Bbknn: fast batch alignment of single cell transcriptomes. *Bioinformatics* **36**, 964–965 (2020).

[32] Harris, C. R. *et al.* Array programming with numpy. *Nature* **585**, 357–362 (2020).

[33] Pedregosa, F. *et al.* Scikit-learn: Machine learning in python. *the Journal of machine Learning research* **12**, 2825–2830 (2011).

[34] Madeira, F. *et al.* The embl-ebi search and sequence analysis tools apis in 2019. *Nucleic acids research* **47**, W636–W641 (2019).

[35] Traag, V. A., Waltman, L. & van Eck, N. J. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports* **9**, 1–12 (2019).

[36] Haghverdi, L., Büttner, M., Wolf, F. A., Buettner, F. & Theis, F. J. Diffusion pseudotime robustly reconstructs lineage branching. *Nature methods* **13**, 845 (2016).

[37] Virtanen, P. *et al.* SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **17**, 261–272 (2020).