Optimal tuning of weighted kNN- and diffusion-based methods for denoising single cell genomics data

Andreas Tjärnberg[1,2,3,*] **Omar Mahmood**[5,6] **Christopher A Jackson**[2,3]

Giuseppe-Antonio Saldi[2] **Kyunghyun Cho**[5,6] **Lionel A Christiaen**[1,3]

Richard A Bonneau[2,3,4,5,6,*]

[1]Center for Developmental Genetics, New York University, New York 10003 NY, USA

[2]Center For Genomics and Systems Biology, NYU, New York, NY 10008, USA

[3]Department of Biology, NYU, New York, NY 10008, USA

[4]Flatiron Institute, Center for Computational Biology, Simons Foundation, New York, NY 10010, USA

[5]Courant Institute of Mathematical Sciences, Computer Science Department, New York University, New York, NY 10003, USA

[6]Center For Data Science, NYU, New York, NY 10008, USA

[*]Correspondence: at145@nyu.edu and rb133@nyu.edu

Abstract

The analysis of single-cell genomics data presents several statistical challenges, and extensive efforts have been made to produce methods for the analysis of this data that impute missing values, address sampling issues and quantify and correct for noise. In spite of such efforts, no consensus on best practices has been established and all current approaches vary substantially based on the available data and empirical tests. The k-Nearest Neighbor Graph (kNN-G) is often used to infer the identities of, and relationships between, cells and is the basis of many widely used dimensionality-reduction and projection methods. The kNN-G has also been the basis for imputation methods using, *e.g.*, neighbor averaging and graph diffusion. However, due to the lack of an agreed-upon optimal objective function for choosing hyperparameters, these methods tend to oversmooth data, thereby resulting in a loss of information with regard to cell identity and the specific gene-to-gene patterns underlying regulatory mechanisms. In this paper, we investigate the tuning of kNN- and diffusion-based denoising methods with a novel non-stochastic method for optimally preserving biologically relevant informative variance in single-cell data. The framework, *Denoising Expression data with a Weighted Affinity Kernel and Self-Supervision* (DEWÄKSS), uses a self-supervised technique to tune its parameters. We demonstrate that denoising with optimal parameters selected by our objective function (i) is robust to preprocessing methods using data from established benchmarks, (ii) disentangles cellular identity and maintains robust clusters over dimension-reduction methods, (iii) maintains variance along several expression dimensions, unlike previous heuristic-based methods that tend to oversmooth data variance, and (iv) rarely involves diffusion but rather uses a fixed weighted kNN graph for denoising. Together, these findings provide a new understanding of kNN- and diffusion-based denoising methods and serve as a foundation for future research. Code and example data for DEWÄKSS is available at `https://gitlab.com/Xparx/dewakss/-/tree/Tjarnberg2020branch`.

1 Introduction

Single-cell RNA-seq (scRNA-seq) experimental methods measure the expression of individual cells from a heterogeneous sample. This allows identification of different cell subpopulations, and has been extensively used to map developmental trajectories. scRNA-seq experiments yield data with hundreds to hundreds of thousands of individual cell observations; however, the measured gene expression in each cell is noisy, due to undersampling caused by the extremely low quantities of RNA present in any individual cell[42]. Many computational applications have been developed that leverage the advantages of scRNA-seq experiments[10, 22, 31, 40]. Analysis has primarily focused on the interpretation of the cellular landscape; software suites incorporating customizable workflows have been developed to enable this analysis[1, 31, 41]. Denoising computational approaches to mitigating the sparsity of single-cell data (having few counts per cell) have corrected structural and sampling zeros[3], imputed missing values[9, 15, 24], or corrected measured expression values[11, 12, 39]. The modeling and motivational assumptions of these approaches vary and include cell-cell similarity, gene covariance, and temporal/trajectory stability.

In any individual cell, some genes will not be detected[21]; genes that have been biologically silenced or repressed and genes that have low stochastic expression may have zero expressed transcripts. Other genes have expressed transcripts in the cell but are measured as zero due to the sampling depth. For some scRNA-seq experimental techniques, there is evidence of zero inflation in measured expression levels[21], but newer droplet-based scRNA-seq methods do not appear to have more zero expression measurements than expected by chance[33]. Single-cell gene expression measurements are a function of transcript sampling depth, which varies widely from technique to technique, stochastic noise in transcript abundance within individual cells, and technical noise which affects genes of all expression levels. Some single-cell denoising methods consider measured zeros to be sampling 'dropouts', and only function to impute non-zero values in place of zero values; these bias corrections to low-expression genes, suppressing variance for these genes by over-correcting zeros and failing to denoise the data in a biologically relevant manner. Other methods are holistic, using the overall expression profile of all genes to guide denoising, including those which are highly expressed.

The results of these denoising methods for scRNA-seq data vary considerably based on the choice of hyperparameter values, and some methods require selection of an appropriate noise model. Empirical determination of hyperparameter values and noise models may not result in the most optimally denoised results. However, if we define an objective function that measures the performance of our algorithm, we can find an optimal model by choosing hyperparameters that maximize this objective. Self-supervision allows autoencoders to select model parameters to optimally denoise data by using an objective function based on the loss after reconstruction from a lower-dimensional manifold, and these methods have been applied to both genomics in general[12] and single-cell data specifically[9, 25]. However, the self-supervision of an autoencoder does not optimise for the choice of model hyperparameters, such as the loss function or the numbers of units and layers, and there is no principled way to tune these hyperparameters for a given dataset.

One of the most fundamental components of the single cell analysis framework is the k-Nearest Neighbor Graph (kNN-G), which connects each cell to the k cells near it based on the distance between their gene expression profiles. It is used to drive neighbor embedding methods that show the global structure of the data in a low-dimensionality projection[26], to detect communities or clusters of related cells[6, 36], and to establish trajectories through network connections that represent changes over time[13, 16, 17]. kNN algorithms are an attractive choice for denoising due to their simplicity, but it is still difficult to select optimal model hyperparameters, especially with regard to the use of prior knowledge to tune these algorithms[8]. Diffusion using a kNN-G is the process of averaging over an incrementally larger and larger number of neighbors derived through shared neighbor connections. Looking at a single cell, diffusing using one step means averaging over its neighbors, while a two-step diffusion means averaging over the cells' neighbors *and* their neighbors. One current method for denoising based on kNN-G is MAGIC[11], which diffuses on the kNN-G and maps back to gene expression of single genes, with final dimensionality limited by the initial input choice of the number of principal components. Another denoising approach smooths expression profiles by combining gene expression of kNN-G connected cells[38, 39]. These methods heuristically determine the number of neighbors used, which corresponds to the amount of smoothing applied; this is a drawback when compared to methods that use an objective function that minimizes a desired global objective function.

In this paper we propose using the noise2self self-supervision method[5] to select kNN-G denoising hyperparameters. This does not depend on an explicit noise model but on an invariant and independent function of the features of the data. We apply this underlying principle to optimally set parameters for denoising single-cell data in a framework called *Denoising Expression data with a Weighted Affinity Kernel and Self-Supervision* (DEWÄKSS), which incorporates a principled self-supervised objective function with weighted kNN-G averaging (Figure 1). We evaluate DEWÄKSS using previously established data and benchmark tests, and compare our self-supervised hyperparameter selection method to the state-of-the-art diffusion method MAGIC[10]. We find that DEWÄKSS performs at par with or better than other state-of-the-art methods, while providing a self-supervised and hence easily searchable hyper-parameter space, greatly simplifying the application of optimal denoising. We also find that diffusion, although conceptually attractive and previously described as beneficial, is, in fact, not optimal for any method in any setting on any dataset.

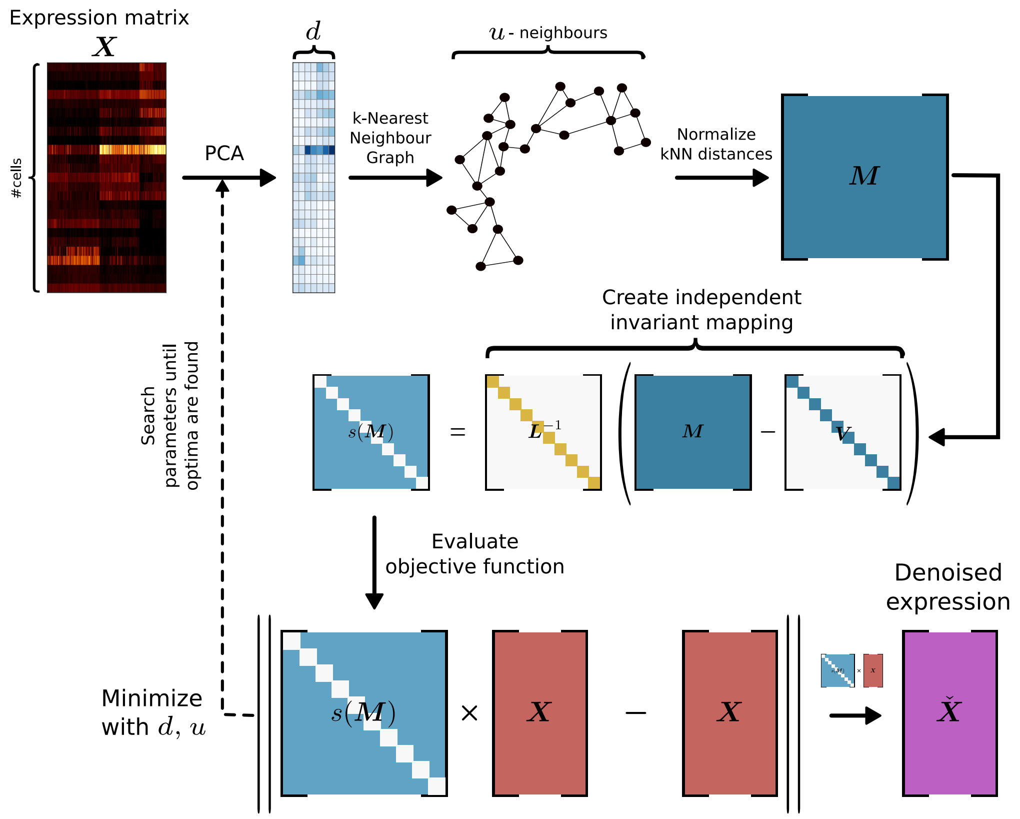

Expression matrix
X
#cells
PCA
d
k-Nearest Neighbour Graph
u - neighbours
Normalize kNN distances
M
Create independent invariant mapping
s(M)
=
L^{-1}
M
−
V
Search parameters until optima are found
Evaluate objective function
Denoised expression
Minimize with d, u
s(M)
×
X
−
X
X̌

Figure 1: The denoising process using DEWÄKSS. The procedure is as follows: (i) The PCA of the expression matrix $\boldsymbol{X}$ is computed. (ii) A cell-to-cell similarity structure is inferred from a k-Nearest Neighbor Graph (kNN-G). (iii) An invariant and independent mapping function is constructed. (iv) The objective function, with a defined optimum for denoising, minimizes the mean squared error (MSE) between the predicted weighted average of the neighbor cells' state and the observed cell state. DEWÄKSS tunes its objective function with two main input parameters, the number of PCs and the number k of neighbors.

2 Results

2.1 Benchmarking the DEWÄKSS algorithm

DEWÄKSS takes a normalized expression matrix and calculates a smoothed output expression matrix which has denoised gene expression for each cell. The DEWÄKSS expression matrix will have decreased stochastic sampling noise; expression values, including zeros that are likely the result of undersampling, will be weighted according to their sample-to-sample context. We will test the effectiveness of diffusion and kNN-based (diffusion with step size = 1) denoising methods that are tuned with DEWÄKSS objective function using three separate cases.

2.2 DEWÄKSS optimally groups cells and performs robustly, independent of data normalization method

Before any denoising can occur, single-cell data generally must be normalized. Recent work has established a benchmark for single-cell analysis methods which is broadly applicable to normalization and denoising techniques[34]. We have therefore compared the performance of DEWÄKSS to several previously-described denoising methods using two benchmark datasets that are generated with different methods. These artificially constructed RNA mixture datasets have a known ground truth; RNAmix_CEL-seq2 is derived from the CEL-Seq2 method[18] and RNAmix_Sort-seq is derived from the SORT-seq method[28]. Within each dataset, 'cells' that belong to the same group are samples that have the same proportions of mRNA from three different cell lines. Any differences between 'cells' in the same group can hence be attributed to technical noise.

Using a computational pipeline[32], we test the effect of normalization methods on denoising techniques, with output scoring defined by a known ground truth (Figure 2). Normalization methods are generally the same as previously tested[34], and include several bulk-RNA normalization methods (TMM, logCPM, DESeq2), several single-cell-specific methods (scone, Linnorm, scran), and a simple Freeman-Tukey transform (FTT). Overall, we find that DEWÄKSS yields expression profiles with high within-group correlation (averaged over all cells in the dataset) independent of the normalization method used, outperforming other denoising methods in the majority of cases (Figure 2A). This is not due to high correlation between cells in different groups (which could indicate oversmoothing), as cells of the same type are strongly correlated and cells of different types are weakly correlated when plotted as a heatmap (Figure S.1).

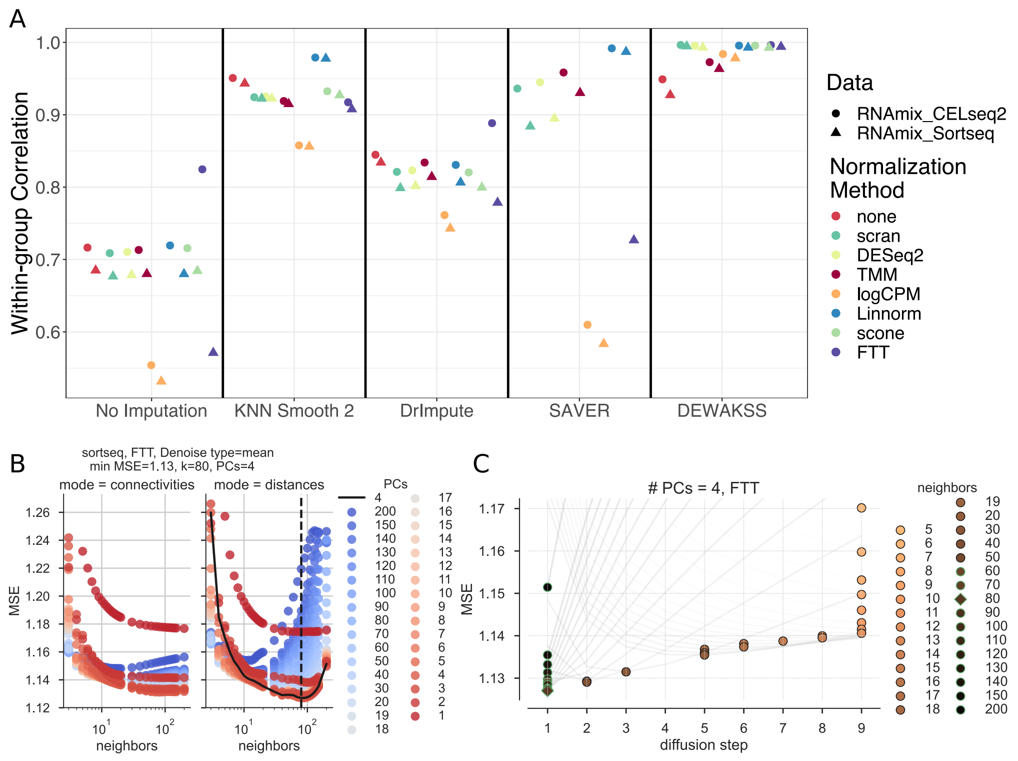


Figure 2: Benchmarking DEWÄKSS on a predefined benchmark test[34]. A) The average Pearson correlation coefficients between cells that are known to be in the same group, calculated as in Tian et al. [34] for the RNAmix_CEL-seq2 and RNAmix_Sort-seq benchmark datasets. DEWÄKSS yields highly correlated expression profiles for 'cells' in the same group, robustly across different normalization methods. B) Self-supervised hyperparameter grid search results (RNAmix_Sort-seq normalized by FTT). Neighbors are on the x-axis and PCs are colored. The optimal configuration neighbors are shown by the dotted black line and PCs are shown by the solid black line. C) Optimization behavior using optimal PCs=4 found in (B) for 5-200 neighbors. The lowest prediction error for each diffusion trajectory (line) is marked by a circle with a green outline if it corresponds to the number of iterations in the optimal configuration $i = 1$ or in black when the optimal number of iterations > 1. The optimal value is marked by a diamond. The number of diffusion steps decreases as the number of neighbors increases. The number of diffusion steps is truncated to 9 steps. The prediction error decreases as the number of neighbours increases from 5-80, and then increases.

DEWÄKSS has three essential input hyperparameters: the number of principal components (PCs) for initial data compression, the number of nearest neighbors to build the graph embedding with (k), and the

Normalization	Dataset	iteration	mode	neighbors	pcs	MSE
DESeq2	celseq2	1	distances	120	3	0.466
FTT	celseq2	1	distances	90	5	0.878
Linnorm	celseq2	1	distances	110	4	0.066
logCPM	celseq2	1	distances	100	6	4.567
none	celseq2	1	connectivities	14	120	4.428
scone	celseq2	1	distances	120	4	0.445
scran	celseq2	1	distances	130	3	0.484
TMM	celseq2	1	distances	50	6	0.378
DESeq2	sortseq	1	distances	100	3	0.513
FTT	sortseq	1	distances	80	4	1.127
Linnorm	sortseq	1	distances	100	4	0.083
logCPM	sortseq	1	distances	80	13	4.684
none	sortseq	1	distances	10	17	5.321
scone	sortseq	1	distances	100	4	0.484
scran	sortseq	1	distances	120	3	0.536
TMM	sortseq	1	distances	50	6	0.412

Table 1: Optimal hyperparameters selected by DEWÄKSS self-supervised objective function

connection mode for edge weights (either normalized distances or network density weighted connectivities). For this benchmark, the DEWÄKSS algorithm has grid searched through a model hyperparameter space, testing connection mode {distances, connectivities}, number of neighbors {1, 2, ..., 20, 30, 40, ..., 150, 200}, and PCs {1,2, ..., 20, 30, 40, ..., 150, 200}. Self-supervision selects optimal model hyperparameters by minimization of mean squared error (MSE); for the RNAmix_Sort-seq dataset that has been normalized by FTT, this is the normalized distance mode with 4 PCs and 80 neighbors (Figure 2B). The optimal selection of hyperparameters varies from dataset to dataset and by normalization method (Table 1). In most cases, the optimal MSE is found for the parameters given by normalized distances with between 50-130 neighbors and 3-13 PCs, but data which has not been normalized has very different optimal model hyperparameters. In general, using normalized distances as edge weights between neighbors outperforms using connectivities. In all cases, the optimal number of diffusion iterations is 1 (no diffusion) given a specific set of PCs and k, indicating that diffusion is not optimal on this data. The small number of PCs and large value for optimal neighbors suggests that this dataset is simplistic with weak local structure, which is a reasonable expectation given the artificial construction of the RNAmix datasets. The MSE is scaled differently depending on the data normalization and should not be compared between methods.

2.3 DEWÄKSS maintains cluster homogeneity and deconvolves cluster structure comparably to state-of-the-art methods.

To evaluate DEWÄKSS on higher-complexity data, we adapt the benchmark used by DeepImpute[2], using a dataset which contains 33 annotated celltypes in primary visual cortex cells from mice (GSE102827)[19]. We evaluate performance after preprocessing and denoising the count data by defining experimental clusters using the Leiden algorithm[36], which tunes the number of clusters based on merging and partitioning the kNN-G to find an optimal partitioning given the resolution parameter r. Differences between annotated cell types and experimental clusters are quantified by the Fowlkes-Mallows score. The silhouette score estimates the relative within-cluster density of the annotated cell-type clusters compared to the closest annotated neighbor-cluster distances – and is, therefore, independent of the cluster algorithm choice – and evaluates the closeness of known cell identities through different data transformations. The silhouette score is separately calculated on two dimension-reduction projections, (i) using 2 UMAP[26] components, and (ii) using PCA with the number of components used to compute the kNN-G used in the Leiden and UMAP algorithms.

We compare the results of DeepImpute, DEWÄKSS, and MAGIC to count data that has been preprocessed but has otherwise not been denoised (pp). The preprocessing is detailed in section 4.4, and, in short, consists of filtering and median normalizing the data followed by a Freeman-Tukey transformation. DeepImpute[2] takes as input the raw count data and needs to be preprocessed after, as above. MAGIC is run using the Seurat pipeline[31]. For this dataset, DEWÄKSS selects 100 PCs and 150 neighbors as optimal hyperparameters (Figure 3A)). After denoising we evaluate the performance metrics with a range of clustering and dimensionality reduction parameters to estimate the sensitivity of the performance metrics (clustering and cell dispersion during projection) to the choices of these parameters (Figure S.2). Overall performance (as determined by MSE) is poor when using few components (PCs) and a small number k of neighbors, which is similar to the default parameters in many processing pipelines (Figure S.2a). This underlines the importance of carefully considering the amount of variance to be used in the initial kNN-G construction.

Because the number of inferred clusters influences the Fowlkes-Mallows score, we also adjust, by doubling or halving, the resolution parameter r of the Leiden clustering algorithm to increase or decrease the number of clusters to be closer to the number of annotated clusters (33). r is increased from 1 to 2 for DeepImpute and pp and decreased from 1 to 0.5 for MAGIC. DEWÄKSS is not adjusted as the number of clusters falls close to the number of annotated clusters by default.

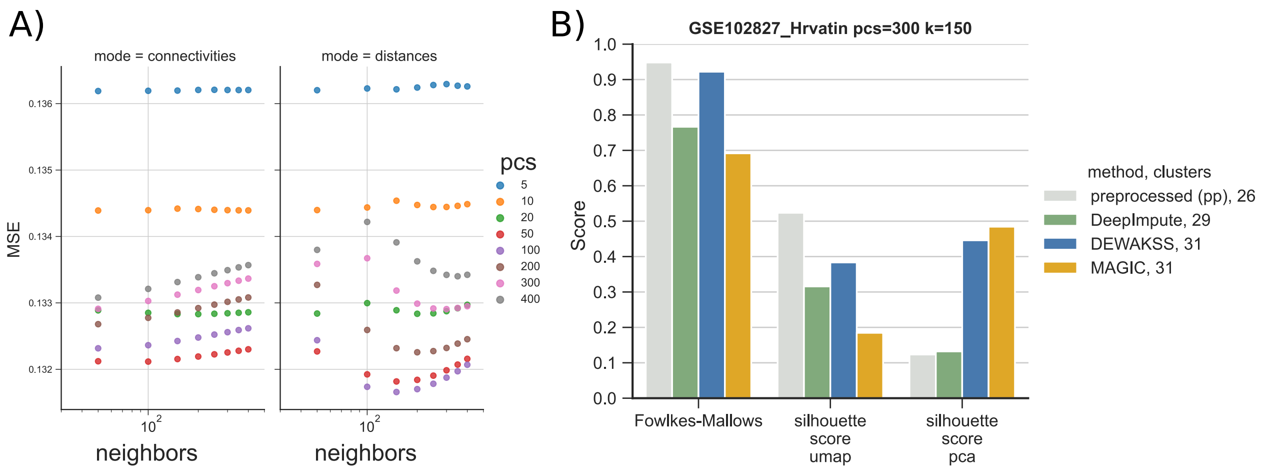


Figure 3: Denoising celltype-annotated data from Hrvatin et al. [19] using metrics from Arisdakessian et al. [2]. The dataset contains 33 annotated celltypes in 48267 cells. A) Optimal denoising of the expression data with DEWÄKSS requires 100 PCs and $k = 150$ neighbors. B) We benchmark four different denoising pipelines: (i) in-house preprocessed (section 4.4), (pp), (ii) DeepImpute, (iii) DEWÄKSS and (iv) MAGIC. After preprocessing & denoising, data is clustered with the Leiden algorithm[36] using 300 PCs and 150 neighbors (resolution is set to $r = 1$ for DEWÄKSS, $r = 2$ for preprocessed (pp) and DeepImpute, and $r = 0.5$ for MAGIC). Algorithm performance is measured with the Fowlkes-Mallows metric and silhouette score on two representations of the data, PCA and UMAP

2.4 Optimal kNN denoising does not involve diffusion

On all test datasets, we observed that the optimal configuration was found to have a single iteration (no diffusion) but variable (dataset specific) optimal number of PCs and neighbors (Figure 2C). This observation extended to all normalization methods if parameter spaces with sufficient numbers of neighbors were explored (Table 1). To determine if diffusion is improving denoising for real-world data, we applied DEWÄKSS to seven published single-cell datasets. We tested on mouse bone marrow (BM) data[29], on human cell line epithelial-to-mesenchymal transition (EMT) data[10], on *Saccharomyces cerevisiae* data from rich media (YPD) and on *Saccharomyces cerevisiae* data after treatment with rapamycin (RAPA)[20], on mouse visual cortex tissue (VisualCortex) data[19], on human embryonic forebrain tissue (hgForebrainGlut) data and on mouse dentate gyrus granule neuron (DentateGyrus) data[23]. The BM and EMT datasets are preprocessed following the vignette provided by the MAGIC package[10] (section 4.5). The YPD, RAPA and VisualCortex datasets are preprocessed using the procedure in section 4.4. The hgForebrainGlut and DentateGyrus datasets are preprocessed with the velocyto[23] and SCANPY[40] python packages using the provided vignettes (section 4.6).

We run DEWÄKSS on these datasets (searching for hyperparameters using ∼equidistant values in log space) to find the optimal configuration (Figure S.4). For the BM dataset we let the algorithm run 20 diffusion steps to map out the objective function. For all other datasets we use *run2best*, which finds the first minimum MSE during diffusion and then stops the search (figure S.5). All six real-world datasets result in optimal MSE when there is no diffusion (number of iterations $i = 1$) (Table 2).

Dataset	iteration	MSE	mode	neighbors	PCs
BM [29]	1	0.311	distances	100	50
EMT [10]	1	0.222	distances	100	100
VisualCortex [19]	1	0.132	distances	150	100
YPD [20]	1	0.217	distances	150	50
RAPA [20]	1	0.261	distances	175	20
hgForebrainGlut [23]	1	0.106	distances	100	20
DentateGyrus [23]	1	0.055	distances	100	100

Table 2: Optimal configurations found by hyperparameter search on DEWÄKSS on seven real-world single-cell datasets

2.5 DEWÄKSS preserves data variance for downstream analysis

The main goal of denoising scRNA-seq data is to reduce the influence of noise and to reveal biological variance. Biological variance is not easily separable from technical noise, and denoising methods risk oversmoothing, retaining only the strongest patterns (*e.g.* the first few principal components of the data) while discarding informative minor variation. It is therefore critical when tuning model parameters to have an objective function that takes into account the total variance of the data structure. We can evaluate the effect that denoising has on data variance by comparing the singular value structure of the denoised data for different methods, which represents the relative variance of all dimensions.

MAGIC[10] is currently among the most popular algorithms for denoising single-cell RNA-seq data. It uses a heuristic for determining optimal smoothing; as published, it used ΔR^2 between diffusion steps, but the most recent implementation has switched to Procrustes analysis of the differences between diffusion steps. Neither approach has an objective way to determine optimal smoothing. In the absence of crossvalidation or some other external method that prevents overfitting, we expect R^2 to decrease until all data is averaged, *i.e.*, to saturation, and a Procrustes analysis should behave similarly. MAGIC addresses this by hard-coding a stopping threshold which determines when the data is smoothed "enough"; because this threshold is not data-dependent, it can result in highly distorted outputs[2, 7, 20].

If this threshold is converted to a model hyperparameter, it is still necessary to tune with some external method as it has no lower bound for arbitrarily poor estimates.

We compare the effects of denoising using a heuristic as implemented in MAGIC[10], using DEWÄKSS in its optimal configuration and using DEWÄKSS in an oversmoothing (non-optimal) configuration for comparison. This comparison is performed on the previously-described mouse BM and human EMT data, preprocessed using the approach described in [10]. We run MAGIC with three sets of model parameters; the default parameters, default with early stopping (the diffusion parameter $t = 1$), and with the decay parameter $d = 30$. For DEWÄKSS, we scan a log-equidistant parameter range for the optimal configuration (Figures 4 B and S.5a) and find that the optimal configuration uses normalized distances with number of neighbors $k = 100$ and with number of principal components PCs$= 50$ for $i = 1$, giving MSE $= 0.3107$. Diffusion iterations i increment until a minimum MSE is found. For the BM data, the MSE generally decreases as the number of PCs increases. Beyond a certain point, however, continuing to increase the number of PCs (to 500) increases the MSE. The optimal number of PCs, 50, is small compared to the size of the data and suggests that some compression of the data is optimal before running the kNN algorithm. To oversmooth the data we extended the number of iterations to run DEWÄKSS to $i = 4$, beyond the optimal number of iterations (Figure 4).

To investigate how the variance structure of the data changes based on denoising we compute the singular values (section 4.7) and determine the number of components needed to explain 90% and 99% of the variance for each dataset after denoising (Figures 4A and S.3). We observe a striking difference between the oversmoothed data and the optimally denoised data. With optimal denoising, 90% of the variance is captured by 259 components. Conversely, only 2 components are needed to capture 90% of the post-processing variance when oversmoothing the data, showing that a substantial portion of the original information content of the data is lost in this regime. This is comparable to the results of using MAGIC with default parameters, where 90% of the variance in the post processed data can be captured with only 3 components. When using only one iterative step and default parameters, MAGIC captures this amount of variance using 25 components. In most cases the MAGIC algorithm generates shallow variance structures with a few components needed to express nearly all of the variance. The variance structure can differ greatly depending on the hyperparameters chosen for DEWÄKSS, and poor parameter selection results in shallow variance structures. However, the objective function automatically identifies an optimal configuration such that we expect to keep the relevant variance.

We can see the consequence of oversmoothing when plotting the expression of the erythroid marker Klf1, the myeloid marker Mpo, and the stem cell marker Ifitm1 (Figure 4D). Very few individual cells express both Klf1 and Mpo in the optimally-denoised data, but the oversmoothed data implies that there is a smooth continuous transition from high Klf1 expression, through co-expression of Klf1, Mpo, and Ifitm1 markers, to high Mpo expression. Although the difference in MSE is not large (Δ MSE < 0.0175) between these two denoised datasets, the resulting biological interpretation differs a great deal, and likely highlights a spurious relationship in the oversmoothed case.

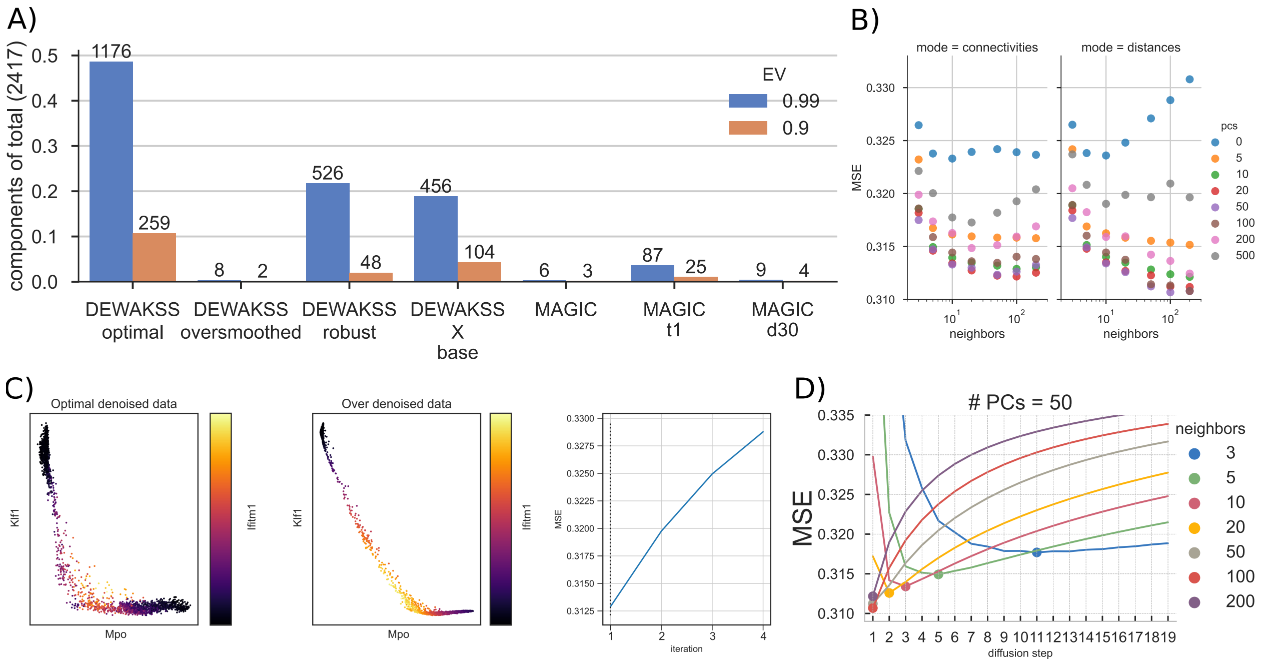


Figure 4: Mouse bone marrow (BM) data denoising. A) The numbers of principal components needed to explain 99% and 90% of the variance in the data for different hyperparameter values for DEWÄKSS and MAGIC. DEWÄKSS is run with **optimal** parameters ($k = 100$, $PCs = 50$, $i = i_{\min MSE}$), with **oversmoothed** parameters ($k = 100$, $PCs = 50$, $i = i_{\min MSE}$), with **robust** parameters ($k = 10$, $PCs = 13$ selected as in section S.1, $i = i_{\min MSE}$), and as **X base**, where normalized expression values are used instead of PCs with ($k = 100$, $i = i_{\min MSE}$). MAGIC is run with **defaults** ($d = 15$, $PCs = 100$, $k = 15$), with early stopping **t1** ($t = 1$), and with **d30** ($d = 30$). B) The lowest MSE over all iteration values as a function of each DEWÄKSS parameter configuration, using connectivity graphs in the left panel and distances in the right panel. C) Expression of erythroid marker gene Klf1, myeloid marker Mpo, and stem cell marker Ifitm1 in DEWÄKSS optimal and DEWÄKSS oversmoothed data. The MSE increases in each iteration. D) The objective function output as a function of diffusion steps for the optimal number of PCs = 50. The minimum MSE is found for 100 neighbors and 1 diffusion step, *i.e.*, using only the selected 100 neighbors.

We run a similar analysis on the EMT data (Figure 5) and find identical effects.

3 Discussion

In this paper we have introduced a novel objective function, based on noise2self[5], and applied it to self-supervised parameter tuning of weighted k-nearest neighbors (kNN) and diffusion-based denoising. The resulting algorithm, DEWÄKSS, is specifically designed to denoise single-cell expression data. The objective function has a global objective that can be minimized, removing the need to use often unreliable heuristics to select model parameters, which is a drawback to many current single-cell denoising methods. We demonstrate that this framework accurately denoises data by benchmarking against previously established methods, and find that it is robust to choice of normalisation method (Section 2.2).

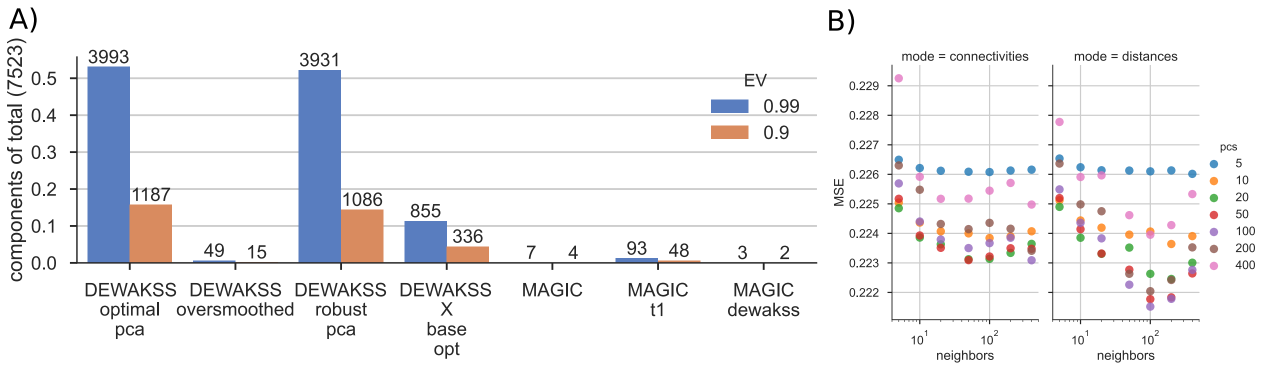


Figure 5: Epithelial-to-mesenchymal transition (EMT) data denoising. A) The numbers of components needed to explain 99% and 90% of the variance in the data for different methods and hyperparameter values. MAGIC is run with 3 settings: default, t1 = one iteration, and "dewakss", using the optimal configuration found by DEWÄKSS. DEWÄKSS is run with 4 different settings: (i) optimal, as found by iterating over a range of hyperparameters (panel B and Figure S.5b), (ii) oversmoothed, by running to $i = 4$ iterations, (iii) robust, i.e., using a different set of hyperparameters ($k = 100$, $PCs = 23$ selected as in section S.1, $i = i_{\min MSE}$) and (iv) X base ($k = 100$, $i = i_{\min MSE}$), using normalized expression values instead of principal components as input to the kNN-G algorithm. B) The lowest MSE over all iteration values as a function of each DEWÄKSS parameter configuration, using connectivity graphs in the left plot and distances in the right plot. The lowest MSE configuration is found using distances with 100 PCs and $k = 100$ neighbors.

Due to the difficulty in establishing a ground truth for most real-world single-cell data, denoising algorithms are frequently tested on synthetic or artificial data. Maintaining biological variance is a crucial aspect of denoising; common downstream applications such as cell type identification, differential gene expression, marker identification, and regulatory network inference rely on biological variance to function. We therefore believe that it is necessary to extensively test on experimental data (a notable strength of van Dijk et al. [10] is testing on real-world data). On larger datasets with higher complexity, DEWÄKSS performs well in terms of deconvolving cell types. We find that in general, the amount of variance included when clustering the data has a large impact on the performance of all methods tested, and that DEWÄKSS outperforms other denoising algorithms in this area. While it is still an open question how much variance should be used to project and cluster single-cell data, it is clear that it is an essential component of accurate interpretation.

To investigate the properties of our method we run the algorithm on seven different published single-cell gene expression datasets. In all cases, the optimal denoising configuration (as determined by the objective function) uses the closest neighborhood, and is not improved by diffusion on the kNN graph. Diffusion causes a decrease in denoising performance, compressing almost all of the variance into a handful of dimensions. This may have some advantages for visualizing high-dimensional gene expression data, but most non-visualization analyses are impaired by the loss of variance. We also find that the number of neighbors k and the number of principal components to use tend to be large compared to the default parameters of other methods and conventions used in computational pipelines. In general, there is an advantage to the inclusion of more principal components than called for by common rules of thumb, like using the knee in an explained variance vs number of components plot. However, including an arbitrary number of principal components is not ideal, as excess principal components do decrease performance. Comparing the use of a distance matrix versus the use of a connectivity matrix as a representation of the kNN-G shows that a distance matrix yields better results. The degree of similarity between the expression profile of one cell to that of another cell is relevant for denoising, not just whether cells are more or less similar than other cells' expression profiles in the experiment.

Overall, the DEWÄKSS framework presented here has substantial advantages over heuristic parameter selection. Heuristic-based denoising methods set hyperparameters without a clear basis for effectiveness, often with opaque reasoning for choices. At best, this is likely to result in sub-optimal denoising performance; at worst, it may result in data that is dominated by oversmoothing effects, and which yields incorrect biological interpretations. Our objective function-based method provides a rigorous way of choosing an effective configuration. The difficulties of evaluating how to denoise single-cell data should not be underestimated. It is vital that the effectiveness of single-cell processing methods be quantifiable, so that the methods can be tuned for performance. We have chosen to use Euclidean distances for all analysis, but DEWÄKSS can accept any graph derived with any distance metric to create the kNN matrix. By constructing a denoising function that uses a k-nearest neighbors graph and is consistent with the conditions laid out in noise2self, we have derived an easily-evaluated method that can denoise single-cell data in a self-supervised manner. The DEWÄKSS objective may also have applications to other graph-based algorithms.

4 Methods

We begin by presenting a review of the mathematical constraints on our denoising function. We then present the core DEWÄKSS method and objective function. We end with descriptions of our preparatory and preprocessing methods.

4.1 Fundamental principle of noise2self

Batson and Royer [5] present the approach noise2self and applied it for UMI counts[4], in which they partition observed features (in this case raw transcript counts) $x_{i\in\mathbb{J}}$, $\mathbb{J} = \{1, \dots, 2m\}$, into two groups $\{X_J, X_{J^c}\} = \{\{x_1, \dots, x_m\}, \{x_{m+1}, \dots, x_{2m}\}\}$ where the superscript c represents the complement set. The task is then to find an invariant function $g(x)_J : \mathbb{R}^{2m} \to \mathbb{R}^{2m}$ that operates only on X_{J^c} and yields an output $\hat{x}$ whose entries at indices J are predictors of x_J. This function is independent of x_J; some of the features of each datapoint are predicted using another independent set of features of the same datapoint. This practical implementation of the original noise2self does not employ a kNN graph.

4.2 Denoising expression data with a weighted affinity kernel and self-supervision

In DEWÄKSS, as in other state-of-the-art methods, we start by computing a lower, d-dimensional representation of the data using PCA. We then compute a connectivity or distance-weighted kNN-G with u neighbors. Our approach is similar to that of MAGIC[10] but differs in two key ways: (i) we use a self-supervised objective function for hyperparameter selection, and (ii) we denoise on the expression values directly to avoid a loss of information/variance that results from overreduction of dimensionality (reducing the data to a latent representation with low rank or low dimensionality such that key biological variation is lost). To calculate the kNN-G, we use the algorithm UMAP[26] and its implementation[27] and create a right-stochastic matrix $\boldsymbol{M}_{d,u}$ from the connectivity/distance matrix. In practice, any graph can be provided as input to DEWÄKSS for denoising. We use the UMAP neighbor algorithm due to its versatility and speed, but alternative methods could be used here.

Denoising using a normalized kNN-G $\boldsymbol{M}$ can be carried out through simple matrix multiplication

$$\begin{aligned} \check{\boldsymbol{X}} &= \boldsymbol{M}\boldsymbol{X} \\ \check{\boldsymbol{X}}_2 &= \boldsymbol{M}\check{\boldsymbol{X}} \end{aligned} \tag{1}$$

and so on, where $\boldsymbol{X}$ is the expression matrix for z cells, with each column $\boldsymbol{x}_{*j}$ containing the expression values of a single gene j for each cell $k \in K = \{1, \dots, z\}$, $\check{\boldsymbol{X}}$ is $\boldsymbol{X}$ after one step of denoising and $\check{\boldsymbol{X}}_2$ is $\boldsymbol{X}$ after two diffusion steps of denoising. For a given gene j in a single cell k, this equation calculates the weighted average influence of the expression value of the same gene in each neighboring cell. The expression value $\check{x}_{kj}$ is hence set to this weighted sum:

$$\check{x}_{kj} = \sum_{\hat{k}=1}^{z} \mu_{\hat{k}j} x_{\hat{k}j} \tag{2}$$

where $\mu_{\hat{k}j}$ is the $\hat{k}j$-th element of $\boldsymbol{M}$ and $\sum_{\hat{k}=1}^{z} \mu_{\hat{k}j} = 1$.

In general, a Markov process can be forward-iterated as

$$\boldsymbol{M}^2 = \boldsymbol{M}\boldsymbol{M} \tag{3}$$

for a two-step iteration, generalized to $\boldsymbol{M}^n$ for an n-step forward process. Denoising is then carried out as follows:

$$\check{\boldsymbol{X}}_n = \boldsymbol{M}^n \boldsymbol{X} \tag{4}$$

In DEWÄKSS we implement a self-supervised procedure by noting that the operation in equation 4 is the application of an invariant function $g(x)_J$ on each entry of $\boldsymbol{X}$ if the diagonal elements of $\boldsymbol{M}$ at each step n

are set to 0 (to enforce the independence noted in section 4.1). Here $J = \{j\}$, so the expression value of a gene j in cell k is calculated using the expression values of j in all cells except cell k. Equation 2 reduces to:

$$\check{x}_{kj} = \sum_{\hat{k} \neq k}^{z} \mu_{\hat{k}j} x_{\hat{k}j}. \tag{5}$$

That is, for each gene there is an invariant function over the expression values of the gene across all cells.

The Markov property guarantees the independence of the neighborhood graph from past steps. At each step, we set diag $\boldsymbol{M} = 0$ and renormalize to a right stochastic matrix. Let s be a function that removes the diagonal elements of a matrix and then normalizes the resulting matrix to a right stochastic matrix. Let $\mu_{\hat{k}j}^{(d,u,n)}$ be the $\hat{k}j$-th element of $\boldsymbol{M}_{d,u}^{\bar{n}}$. Then

$$s(\boldsymbol{M}_{d,u}^{\bar{n}}) = \boldsymbol{L}_{d,u,n}^{-1}(\boldsymbol{M}_{d,u}^{\bar{n}} - \boldsymbol{V}_{d,u,n}) \tag{6}$$

for each n with

$$\boldsymbol{V}_{d,u,n} = \begin{bmatrix} \mu_{1,1}^{(d,u,n)} & & \\ & \ddots & \\ & & \mu_{z,z}^{(d,u,n)} \end{bmatrix} \tag{7}$$

and

$$\boldsymbol{L}_{d,u,n} = \begin{bmatrix} \sum_{j\neq 1} \mu_{1,j}^{(d,u,n)} & & \\ & \ddots & \\ & & \sum_{j \neq z} \mu_{z,j}^{(d,u,n)} \end{bmatrix} \tag{8}$$

with the following notation:

$$\begin{aligned} \boldsymbol{M}_{d,u}^{\bar{2}} &= s(\boldsymbol{M}_{d,u})s(\boldsymbol{M}_{d,u}) \\ \boldsymbol{M}_{d,u}^{\bar{3}} &= s(\boldsymbol{M}_{d,u}^{\bar{2}})s(\boldsymbol{M}_{d,u}). \end{aligned} \tag{9}$$

Equation 4 can be rewritten and incorporated into the mean square error minimization objective as follows:

$$d^*, u^*, n^* = \underset{d,u,n}{\arg\min} \left\| s(\boldsymbol{M}_{d,u}^{\bar{n}})\boldsymbol{X} - \boldsymbol{X} \right\| \tag{10}$$

$$\boldsymbol{X}^* = s(\boldsymbol{M}_{d^*,u^*}^{\bar{n}^*})\boldsymbol{X} \tag{11}$$

We use this equation to find n that denoises $\boldsymbol{X}^*$ enough to best capture its underlying structure while attenuating variation due to noise. To make sure we keep the invariant sets independent we consider each step an independent Markov process and apply the function $s(.)$ at each step.

4.3 Converting kNN to a right stochastic transition matrix

To allow for the use of UMAP distance metrics, DEWÄKSS uses the transformation on distances used in [10],

$$\boldsymbol{M} = e^{-(\boldsymbol{D}/\bar{\boldsymbol{d}}).^{\alpha}} \tag{12}$$

where $\boldsymbol{D}$ is the matrix of distances between the gene expression profiles of different cells and $\bar{\boldsymbol{d}}$ is the mean of the nonzero elements of $\boldsymbol{D}$. A decay rate α is also used, and the . in the equation indicates element-wise multiplication. This decay rate is applied on a connectivity matrix as $\boldsymbol{C}^{\cdot\alpha}$, where $\boldsymbol{C}$ has elements $c \in [0, 1]$. It should be noted that in [10], the decay rate is also applied during the construction of the kNN-G to estimate distance thresholds and there may not be a 1-to-1 correspondence in the algorithms. To stabilize the denoising procedure, the final step before normalizing to a right stochastic transition matrix is to symmetrize $\boldsymbol{M}$ so that

$$\boldsymbol{M} = \frac{\boldsymbol{M} + \boldsymbol{M}'}{2}$$

4.4 Preprocessing of scRNA-Seq expression data

We preprocess single cell RNA-seq datasets before applying denoising. Unless otherwise stated, all datasets are preprocessed using the same steps if no guidelines were provided from the dataset or publication source documents.

Preprocessing is carried out using the following steps: (i) Filtering cells by using only those that have greater than a certain number of expressed genes and greater than a certain total cell UMI count. This is done mainly by visual inspection, clipping $1-2\%$ of the data – in practice, removing the top and bottom 0.5 percentile of data points; (ii) Removing genes not expressed in more than n cells, with $n \leq 10$; (iii) Normalizing the counts of each cell so that each cell has the same count value as the median cell count over the cell population; and (iv) applying the Freeman-Tukey transform (FTT)[14] with an adjustment term -1 to preserve sparsity,

$$\tilde{\text{FTT}}(x) = \sqrt{x} + \sqrt{x+1} - 1 \tag{13}$$

The FTT stabilizes the variance of Poisson distributed data. Wagner et al. [39] showed that the FTT is a good choice for single-cell data compared to the log-TPM and log-FPKM transforms as it does not underestimate the relative variance of highly expressed genes and thus balances the influence of lowly expressed gene variation. In other words, the relative variance of highly expressed genes versus more lowly expressed genes should be preserved after transformation. This is an essential property for inferring a relevant kNN-G.

We process all data with the help of the SCANPY framework[40]. DEWÄKSS can accept the SCANPY AnnData object, a regular numpy array or a scipy sparse array as input.

4.5 Preprocessing for comparison with MAGIC

The BM dataset is preprocessed using the same approach as used by [10], as detailed here: https://nbviewer.jupyter.org/github/KrishnaswamyLab/MAGIC/blob/master/python/tutorial_notebooks/bonemarrow_tutorial.ipynb) The EMT dataset is preprocessed as detailed here: https://nbviewer.jupyter.org/github/KrishnaswamyLab/magic/blob/master/python/tutorial_notebooks/emt_tutorial.ipynb).

4.6 Preprocessing hgForebrainGlut and DentateGyrus data

The hgForebrainGlut and DentateGyrus datasets are preprocessed by replicating the process provided by velocyto[23] here https://github.com/velocyto-team/velocyto-notebooks/blob/master/python/hgForebrainGlutamatergic.ipynb and here https://github.com/velocyto-team/velocyto-notebooks/blob/master/python/DentateGyrus.ipynb. The package SCANPY is used to carry out the computations[40].

4.7 Preservation of variance and PCA computation

To estimate the variance structure of our expression matrix before and after denoising, we take the standard normalization of each variable j in the data so that each observation k in column j is

$$\check{x}_{k,j} = \frac{x_{k,j} - \mathbb{E}_k\ x_{k,j}}{\sigma(x_{,j})} \tag{14}$$

with $\mathbb{E}_k$ denoting expected value with respect to k and σ indicating standard deviation.

Computing the singular value decomposition, we get

$$\check{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{15}$$

The singular values are invariant to the transpose, meaning that they explain the variance in the data independent of whether the data is projected onto cells or genes, and nonzero singular values are bounded by $\min\{m, n\}$. To estimate the rank of $\check{\boldsymbol{X}}$ and the nonzero singular values we use the cutoff from numpy[37]:

$$\boldsymbol{S} \leq \max(\boldsymbol{S}) \times \max\{m, n\} \times \epsilon \tag{16}$$

with ϵ being the machine precision of a numpy float 32 type.

The relative variance is then calculated as

$$\eta_i^2 = \frac{s_i^2}{\sum_i s_i^2} \tag{17}$$

The relative condition number of each singular value can be calculated as

$$|\kappa_i| = \frac{s_i}{\underline{s}} \tag{18}$$

with $\underline{s}$ representing the minimum nonzero singular value defined by equation 16.

4.8 Processing Tian et al. [34] benchmark data

In order to evaluate DEWÄKSS and existing methods in accordance with the benchmark analysis of Tian et al. [34], we use the R code provided in Tian et al. [35] to apply all normalization methods that we can successfully run on the RNAmix_CEL-seq2 and RNAmix_Sort-seq datasets. We also apply our FTT-based preprocessing method on the data in Python and combine the result with the other normalization results into a single data structure.

We then use the same codebase to run the denoising (imputation) methods in [34] on the output of each of the normalization methods on each dataset. We transfer each of these outputs to Python, perform a hyperparameter search using DEWÄKSS on it and record the best parameter configurations along with the corresponding mean squared error (MSE) in Table 1. We apply DEWÄKSS on each normalized input using the optimal configuration for that input and transfer the results back to R, combining them with the other denoising results into a single data structure. Note that some normalization-imputation method combinations are not represented in our figures as we could not successfully run these using the provided pipeline. We use the postprocessing and plotting scripts in [35] to generate plots for our analysis.

Acknowledgement

LC and RB is funded by the NIH R01HD096770. RB is funded by the NSF IOS-1546218, NSF CBET-1728858 and NIH R01 CA240283.

References

[1] Amezquita RA, Carey VJ, Carpp LN, Geistlinger L, Lun ATL, Marini F, et al. Orchestrating Single-Cell Analysis with Bioconductor. bioRxiv 2019;.

[2] Arisdakessian C, Poirion O, Yunits B, Zhu X, Garmire LX. DeepImpute: an accurate, fast, and scalable deep neural network method to impute single-cell RNA-seq data. Genome Biology 2019;20(1):211.

[3] Bacher R, Kendziorski C. Design and computational analysis of single-cell RNA-sequencing experiments. Genome Biology 2016 Apr;17(1):63.

[4] Batson J, Royer L, Webber J. Molecular Cross-Validation for Single-Cell RNA-seq. bioRxiv 2019;.

[5] Batson JD, Royer L. Noise2Self: Blind Denoising by Self-Supervision. CoRR 2019;abs/1901.11365.

[6] Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008 oct;2008(10):P10008.

[7] Chen M, Zhou X. VIPER: variability-preserving imputation for accurate gene expression recovery in single-cell RNA sequencing studies. Genome Biology 2018;19(1):196.

[8] Cooley SM, Hamilton T, Deeds EJ, Ray JCJ. A novel metric reveals previously unrecognized distortion in dimensionality reduction of scRNA-Seq data. bioRxiv 2019;.

[9] Deng Y, Bao F, Dai Q, Wu LF, Altschuler SJ. Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning. Nature Methods 2019;16(4):311–314.

[10] van Dijk D, Sharma R, Nainys J, Yim K, Kathail P, Carr AJ, et al. Recovering Gene Interactions from Single-Cell Data Using Data Diffusion. Cell 2018;174(3):716 – 729.e27.

[11] van Dijk D, Sharma R, Nainys J, Yim K, Kathail P, Carr AJ, et al. Recovering Gene Interactions from Single-Cell Data Using Data Diffusion. Cell 2018;174(3):716 – 729.e27.

[12] Eraslan G, Simon LM, Mircea M, Mueller NS, Theis FJ. Single-cell RNA-seq denoising using a deep count autoencoder. Nature Communications 2019;10(1):390.

[13] Farrell JA, Wang Y, Riesenfeld SJ, Shekhar K, Regev A, Schier AF. Single-cell reconstruction of developmental trajectories during zebrafish embryogenesis. Science 2018;360(6392).

[14] Freeman MF, Tukey JW. Transformations Related to the Angular and the Square Root. Ann Math Statist 1950 12;21(4):607–611.

[15] Gong W, Kwak IY, Pota P, Koyano-Nakagawa N, Garry DJ. DrImpute: imputing dropout events in single cell RNA sequencing data. BMC Bioinformatics 2018 Jun;19(1):220.

[16] Haghverdi L, Buettner F, Theis FJ. Diffusion maps for high-dimensional single-cell analysis of differentiation data. Bioinformatics 2015;31(18):2989–2998.

[17] Haghverdi L, Büttner M, Wolf FA, Buettner F, Theis FJ. Diffusion pseudotime robustly reconstructs lineage branching. Nature Methods 2016 Aug;13:845 EP –.

[18] Hashimshony T, Senderovich N, Avital G, Klochendler A, de Leeuw Y, Anavy L, et al. CEL-Seq2: sensitive highly-multiplexed single-cell RNA-Seq. Genome Biology 2016 Apr;17:77.

[19] Hrvatin S, Hochbaum DR, Nagy MA, Cicconet M, Robertson K, Cheadle L, et al. Single-cell analysis of experience-dependent transcriptomic states in the mouse visual cortex. Nature neuroscience 2018 January;21(1):120–129.

[20] Jackson CA, Castro DM, Saldi GA, Bonneau R, Gresham D. Gene regulatory network reconstruction using single-cell RNA sequencing of barcoded genotypes in diverse environments. eLife 2020 jan;9:e51254.

[21] Kharchenko PV, Silberstein L, Scadden DT. Bayesian approach to single-cell differential expression analysis. Nature Methods 2014 May;11:740 EP –.

[22] Kiselev VY, Andrews TS, Hemberg M. Challenges in unsupervised clustering of single-cell RNA-seq data. Nature Reviews Genetics 2019;.

[23] La Manno G, Soldatov R, Zeisel A, Braun E, Hochgerner H, Petukhov V, et al. RNA velocity of single cells. Nature 2018;560(7719):494–498.

[24] Li WV, Li JJ. An accurate and robust imputation method scImpute for single-cell RNA-seq data. Nature Communications 2018;9(1):997.

[25] Lopez R, Regier J, B Cole M, Jordan M, Yosef N. Deep generative modeling for single-cell transcriptomics. Nature Methods 2018 11;15.

[26] McInnes L, Healy J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. ArXiv e-prints 2018 feb;.

[27] McInnes L, Healy J, Saul N, Grossberger L. UMAP: Uniform Manifold Approximation and Projection. The Journal of Open Source Software 2018;3(29):861.

[28] Muraro M, Dharmadhikari G, GrÃijn D, Groen N, Dielen T, Jansen E, et al. A Single-Cell Transcriptome Atlas of the Human Pancreas. Cell Systems 2016 Oct;3(4):385–394.e3.

[29] 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 Dec;163(7):1663–1677.

[30] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 2011;12:2825–2830.

[31] Stuart T, Butler A, Hoffman P, Hafemeister C, Papalexi E, Mauck WM, et al. Comprehensive integration of single cell data. bioRxiv 2018;.

[32] Su S, Freytag S, Tian L, Dong X, Ritchie M. CellBench: Construct Benchmarks for Single Cell Analysis Methods; 2019, r package version 1.1.0.

[33] Svensson V. Droplet scRNA-seq is not zero-inflated. bioRxiv 2019;.

[34] Tian L, Dong X, Freytag S, Lê Cao KA, Su S, JalalAbadi A, et al. Benchmarking single cell RNA-sequencing analysis pipelines using mixture control experiments. Nature Methods 2019;16(6):479–487.

[35] Tian L, Dong X, Freytag S, Lê Cao KA, Su S, JalalAbadi A, et al., Single cell mixology: single cell RNA-seq benchmarking; 2019.

[36] Traag V, Waltman L, van Eck NJ, From Louvain to Leiden: guaranteeing well-connected communities; 2018.

[37] van der Walt S, Colbert SC, Varoquaux G. The NumPy Array: A Structure for Efficient Numerical Computation. Computing in Science Engineering 2011 March;13(2):22–30.

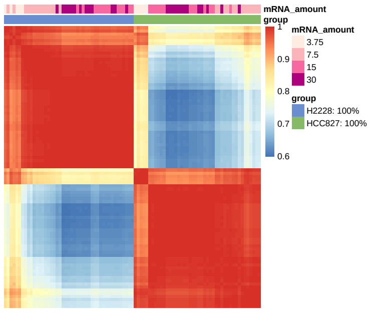
[38] Wagner F, Barkley D, Yanai I. ENHANCE: Accurate denoising of single-cell RNA-Seq data. bioRxiv 2019;.

[39] Wagner F, Yan Y, Yanai I. K-nearest neighbor smoothing for high-throughput single-cell RNA-Seq data. bioRxiv 2017;.

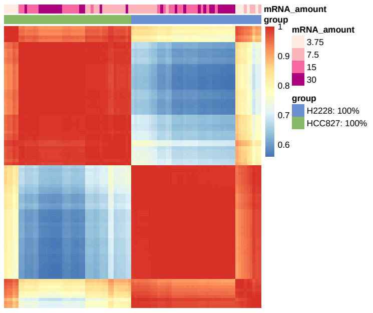
[40] Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. Genome Biology 2018 Feb;19(1):15.

[41] Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. Genome Biology 2018 Feb;19(1):15.

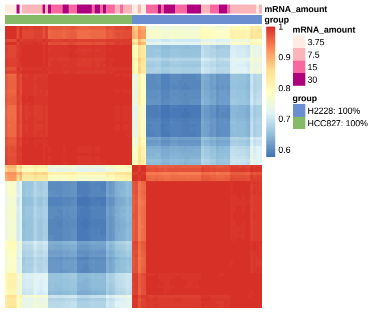
[42] Ziegenhain C, Vieth B, Parekh S, Reinius B, Guillaumet-Adkins A, Smets M, et al. Comparative Analysis of Single-Cell RNA Sequencing Methods. Molecular Cell 2017;65(4):631 – 643.e4.



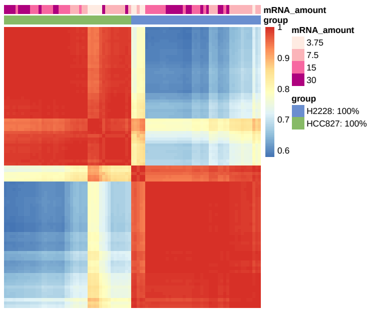
(a) Linnorm



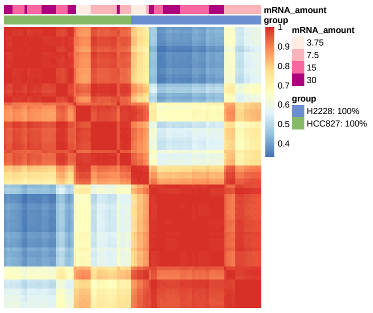
(b) scone



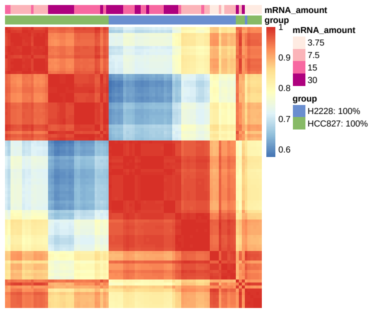
(c) scran



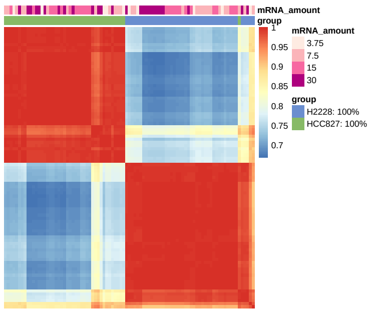
(d) DESeq2



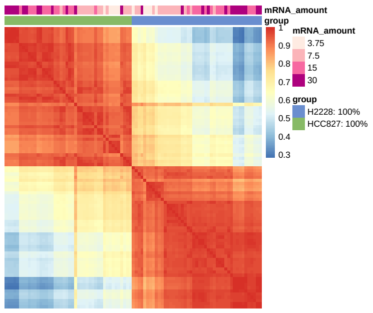
(e) logCPM



(f) TMM



(g) FTT



(h) none

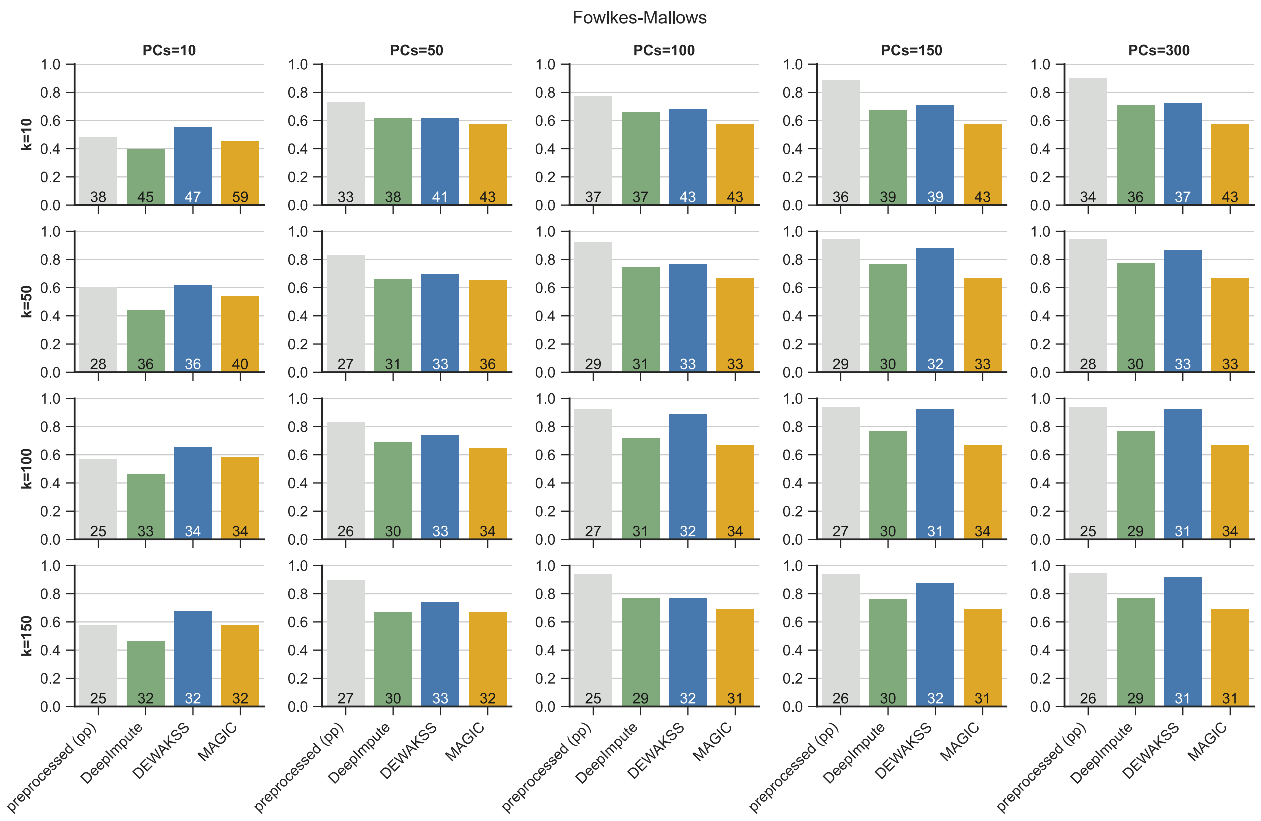
Figure S.1: Heatmaps of normalized and denoised data on the RNAmix_CEL-seq2 dataset [34]. We use the R code provided in Tian et al. [35] to apply all normalization methods that we can successfully run on the RNAmix_CEL-seq2 dataset. We also apply our FTT-based preprocessing method on the data. We use the postprocessing and plotting scripts in [35] to generate plots for our analysis.

S.1 Implementation of Molecular cross-validation [4]

The most straightforward method of denoising data is to use a principal component analysis (PCA); a subset of principal components (PCs) which explain a desired amount of variance are identified, and the expression data is projected onto this subset. This removes the effect of low-variance, presumably noise-driven components. The choice of how many PCs to retain greatly influences the structure of the data and the amount of remaining biological variation. DEWÄKSS utilizes PCA only to constructing the kNN-G; after the graph has been constructed denoising proceeds using the full expression matrix.

We explored selecting a reasonable number of PCs using the method Molecular cross-validation (MCV), presented by Batson et al. [4] (for use in projection algorithms). MCV takes in count data and partitions the individual counts randomly into two partitions, $(\boldsymbol{X}_J, \boldsymbol{X}_{J^c}) \in \mathbb{R}^{N,M}$. A normalization pipeline then normalizes each data partition individually to yield $(\bar{\boldsymbol{X}}_J, \bar{\boldsymbol{X}}_{J^c})$ and carries out PCA on $\bar{\boldsymbol{X}}_J$.

For different numbers of PCs, we project and then reconstruct $\bar{\boldsymbol{X}}_J$ and then calculate the Mean Squared Error (MSE) between the reconstruction and $\bar{\boldsymbol{X}}_{J^c}$. This is done as follows: let $\bar{\boldsymbol{X}}_J$ be the normalized expression matrix and $\boldsymbol{V}$ be the orthogonal matrix with columns as principal vectors. Then the mapping



(a) Clustering performance metric computed against cell type clusters. Clusters were inferred with the Leiden algorithm[36]. The number at the bottom of each bar represents the inferred number of clusters. The Leiden algorithm's resolution parameter r was set to 0.5 for MAGIC and 2 for pp and DeepImpute to reduce or increase number of clusters, respectively, to comparative numbers given the previously annotated clusters.

$\boldsymbol{Z} = \bar{\boldsymbol{X}}_{J^c}\boldsymbol{V}$ gives the projected features $\boldsymbol{Z}$ that we can map back to $\bar{\boldsymbol{X}}_J = \boldsymbol{Z}\boldsymbol{V}^T$. Using all columns of $\boldsymbol{V}$ will map $\bar{\boldsymbol{X}}_J \to \bar{\boldsymbol{X}}_J$; however, using a subset $\boldsymbol{V}_K$ of the PCs we get

$$\check{\boldsymbol{X}} = \bar{\boldsymbol{X}}_J \boldsymbol{V}_{1:k}\boldsymbol{V}_{1:k}^T \tag{S19}$$

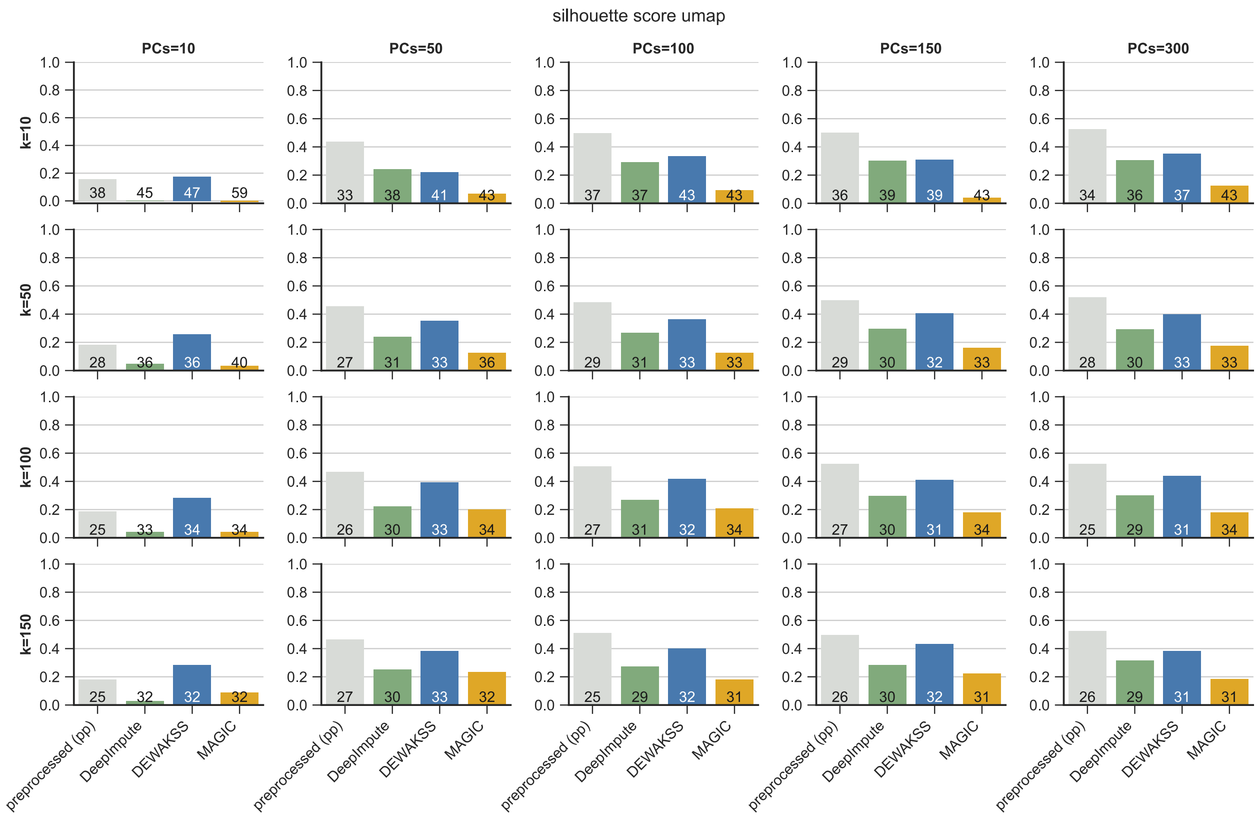
which we can use to calculate the MSE:

$$\text{MSE} = \left\| \check{\boldsymbol{X}} - \bar{\boldsymbol{X}}_{J^c} \right\|. \tag{S20}$$

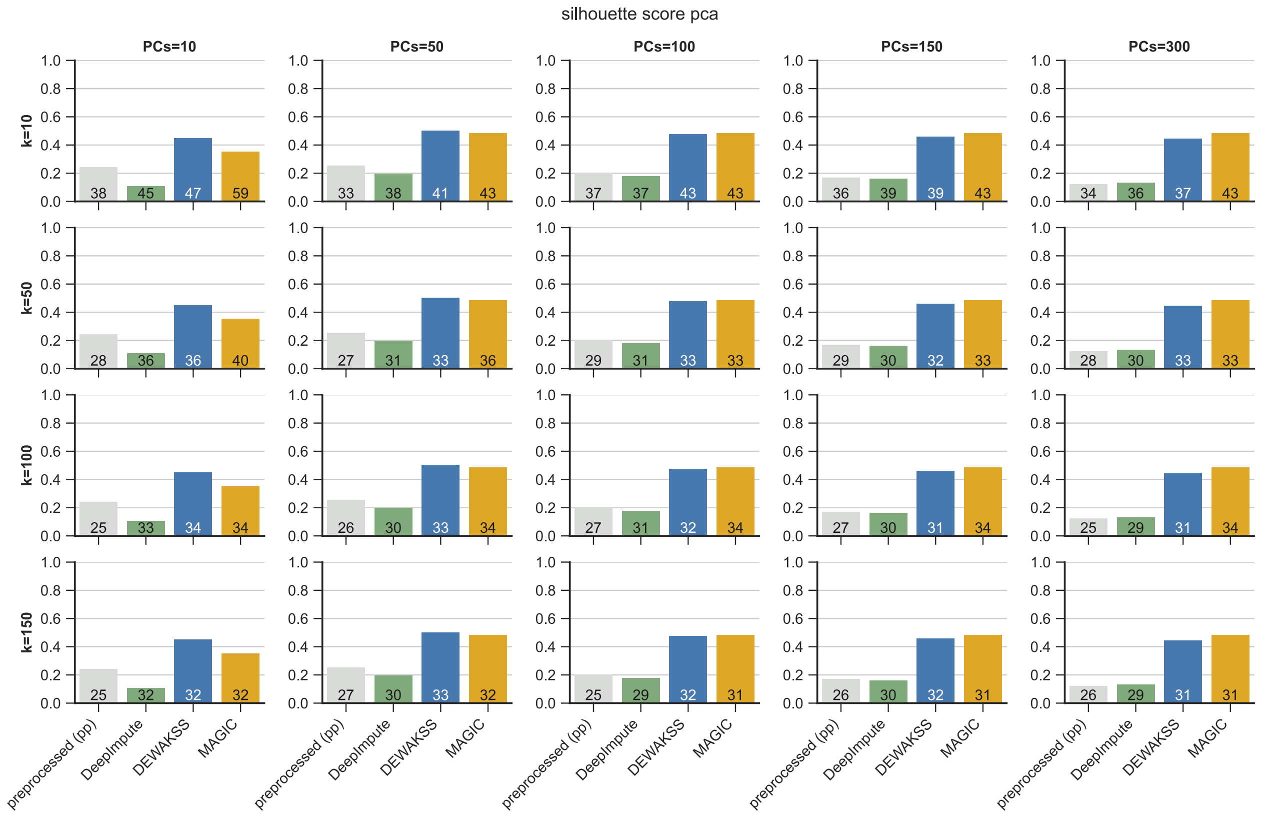
We can calculate the MSE for each number of PCs $k \in [1, K]$ and find k that corresponds to the minimum MSE:

$$k^* = \arg\min_k \left\| \boldsymbol{X}_J \boldsymbol{V}_{1:k}\boldsymbol{V}_{1:k}^T - \boldsymbol{X}_{J^c} \right\| \tag{S21}$$

We applied this algorithm using *TruncatedSVD* [30] to perform PCA and reconstruct the input matrix using the operation in equation 21.



(b) Silhouette score computed on 2 UMAP components using the 33 predefined cell type clusters. The input parameters to the umap algorithm is annotated with x and y labels of each panels row and column.



(c) Silhouette score computed on a varying numbers of PC components using the 33 predefined cell clusters. The number of PCs used corresponds to the number on the upper edge of the graph.

Figure S.2: Denoising celltype annotated data from Hrvatin et al. [19]. The dataset contains 33 annotated celltypes in 48267 cells.

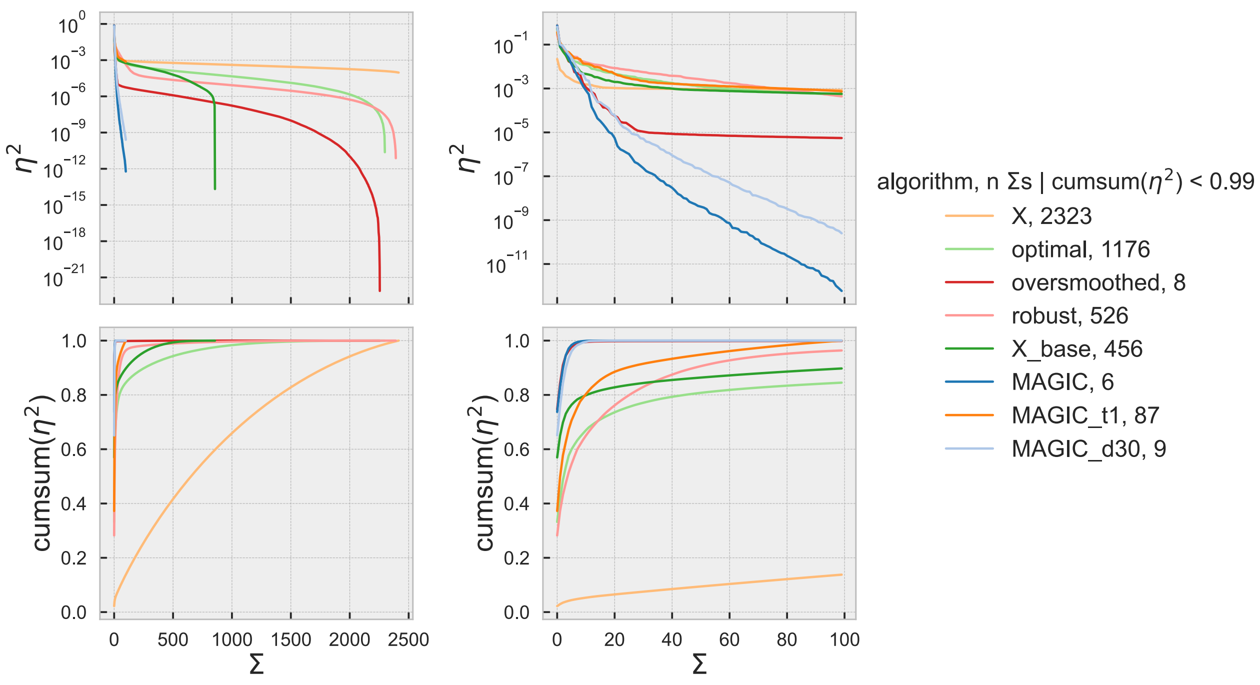
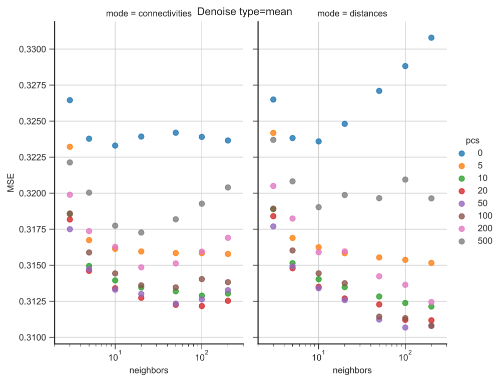
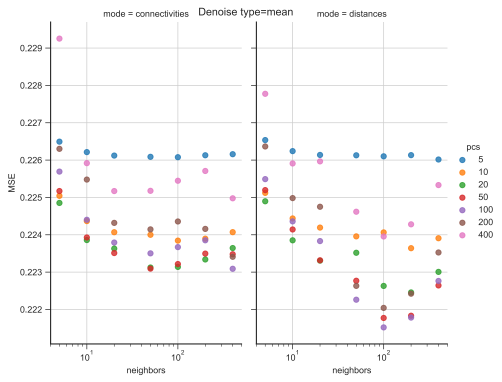
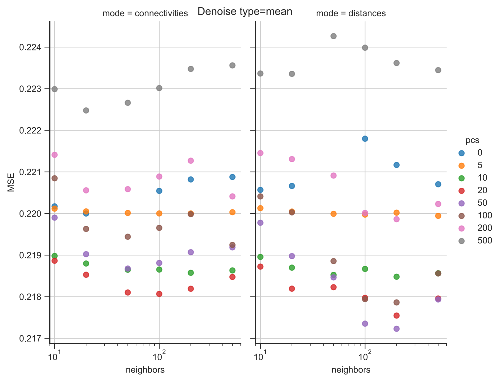
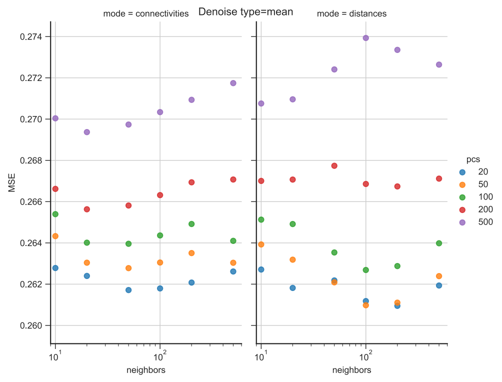


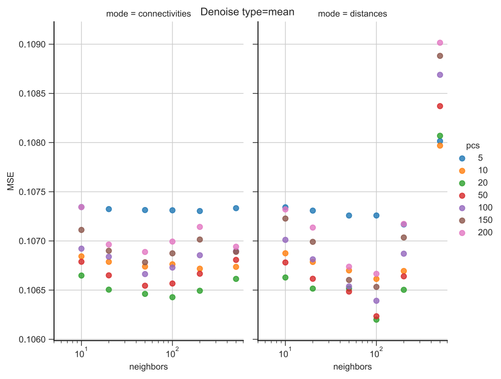
Figure S.3: BM data [29]. Explained variance η^2 for each component Σ (top row) and cumulative sum of η^2 for each component Σ (bottom row). Each colored line indicates the data denoised with a specific set of parameters for the algorithms DEWÄKSS and MAGIC. X indicates the data without denoising. MAGIC truncates the number of possible components to the number of PCs used in the algorithm, which here equals 100. The right column only shows the first 100 components for the respective η^2 and $cumsum(\eta^2)$. MAGICd1 is removed from the top right-hand figure because it compresses the other lines. The explained variance is computed through the singular value decomposition and singular values lower than the numerical precision threshold are considered equal to 0 and removed. This threshold is determined by the criterion $\sigma_i \leq \sigma_1 \times \max(i,j) \times \epsilon$, where i, j are the data dimensions and ϵ is the machine precision (numpy matrix_rank).


(a) Paul et al. [29] BM dataset.


(b) van Dijk et al. [10] EMT dataset.


(c) Jackson et al. [20] YPD growth condition


(d) Jackson et al. [20] YPDRapa growth condition


(e) La Manno et al. [23] hgForebrainGlut

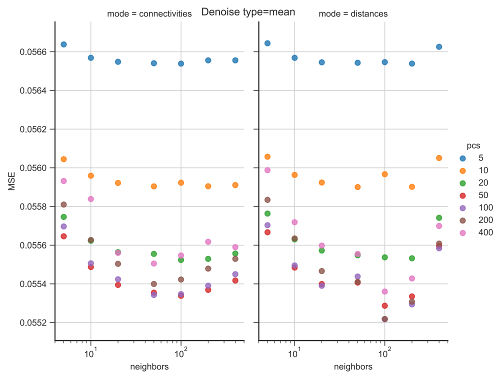
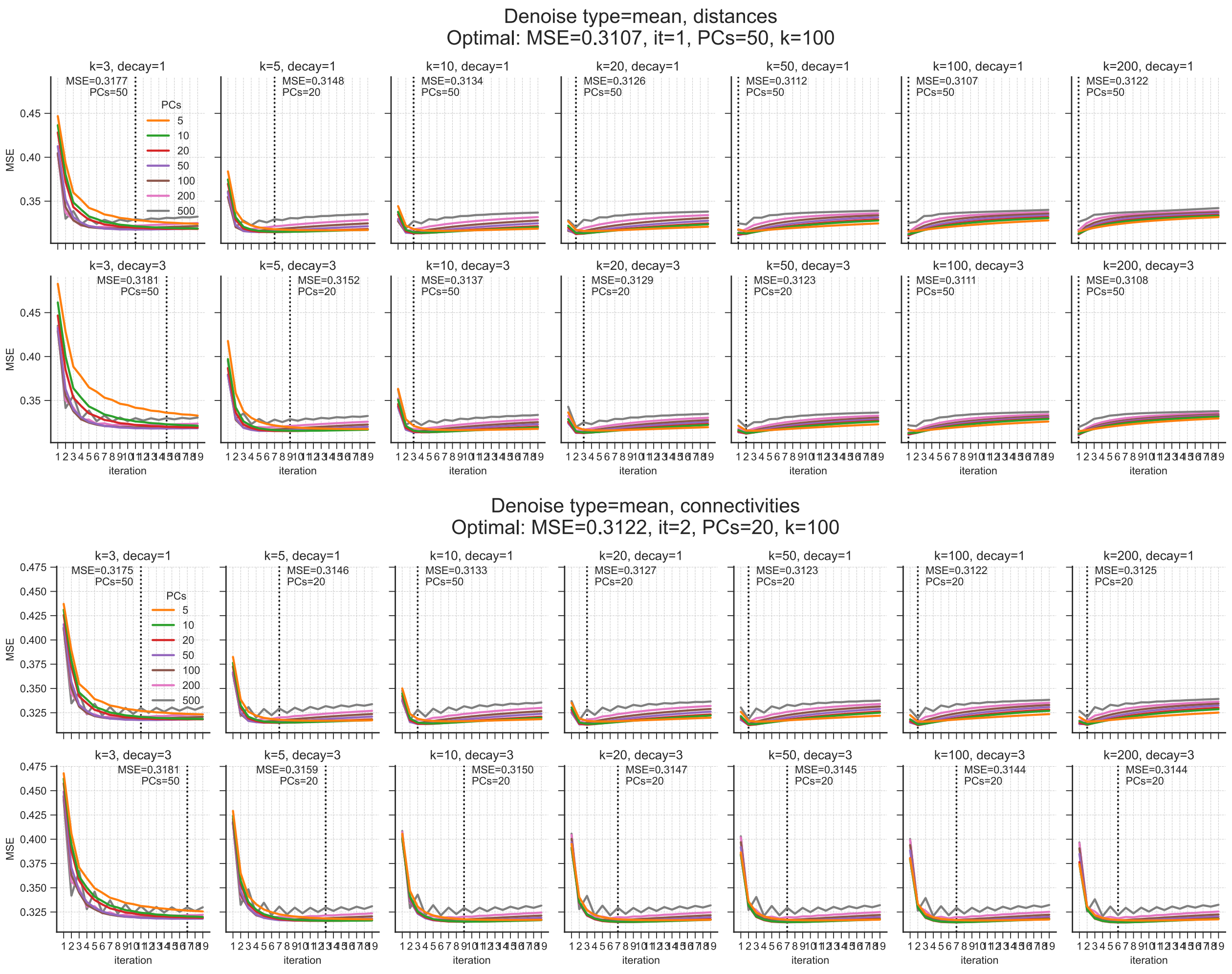
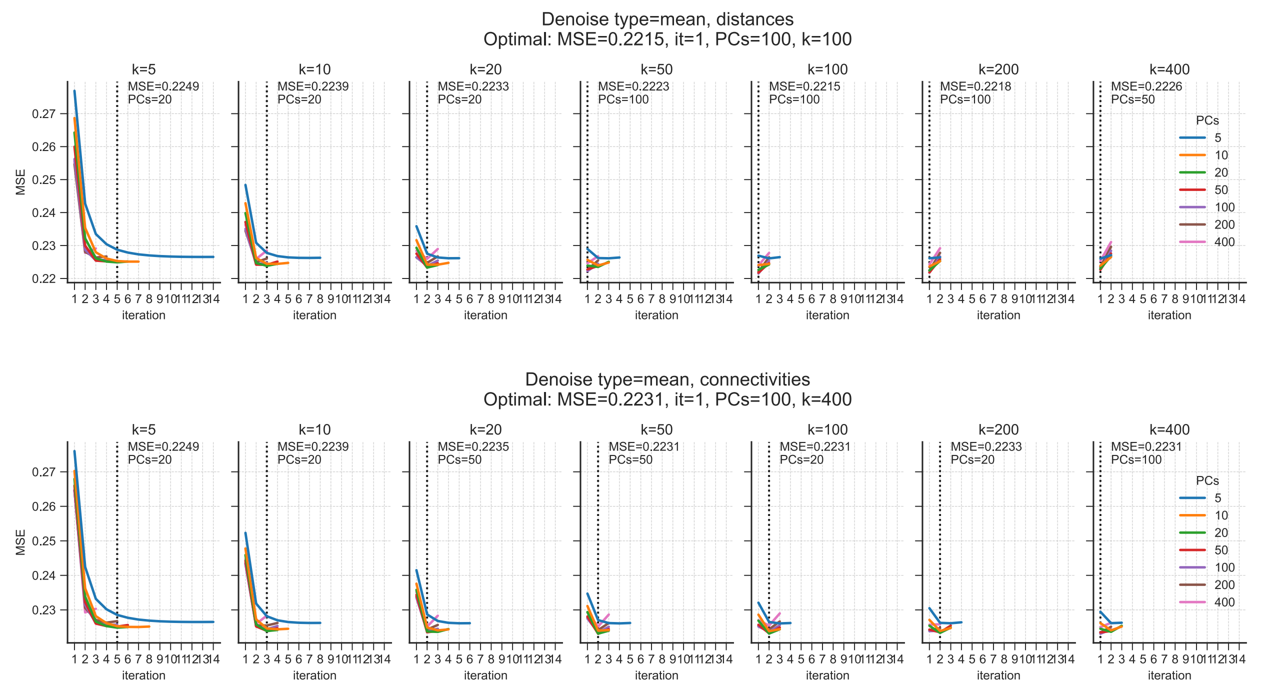

(f) La Manno et al. [23] DentateGyrus

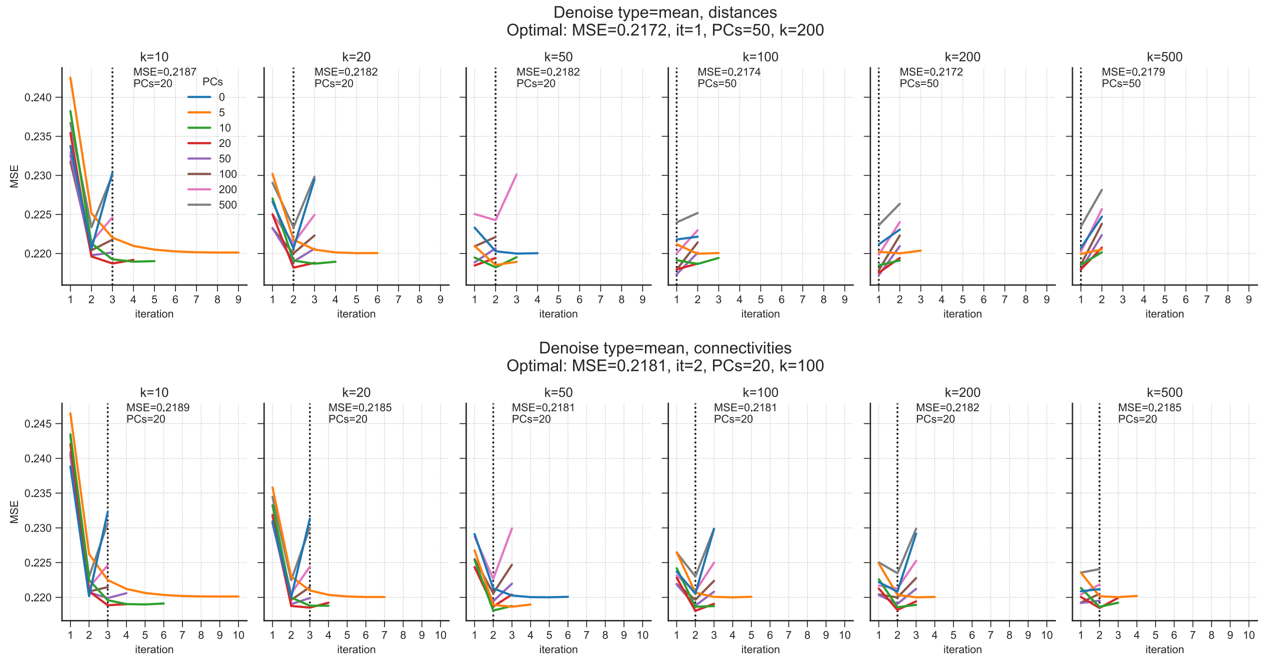
Figure S.4: Optimal hyperparameter search for single-cell datasets. The optimal number of PCs and neighbors is independent of the number of diffusion steps.



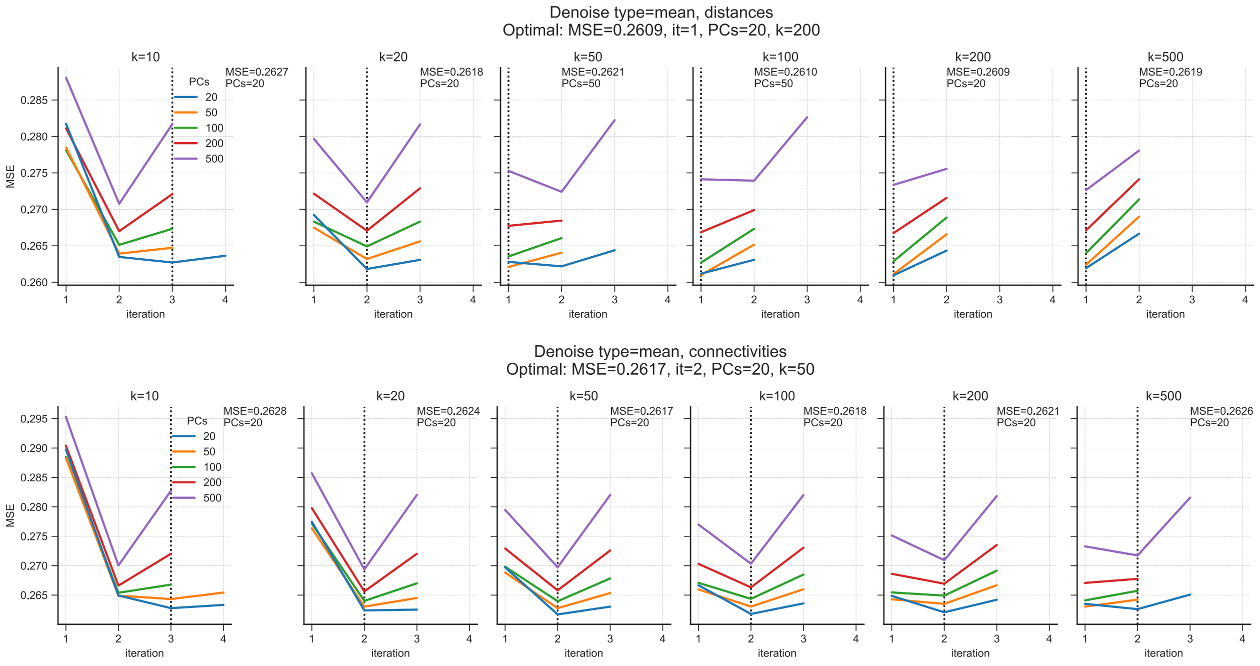
(a) BM data. DEWÄKSS objective function values, mean square error, MSE during optimization of a range of hyperparameters, number of principal components (PCs) $= \{5, 10, 20, 50, 100, 200, 500\}$, number of initial neighbors (k) $= \{3, 5, 10, 20, 50, 100, 200\}$ and decay $= \{1, 3\}$. Optimal settings are found to be 50 PCs, decay $= 1$, and $k = 100$ with MSE $= 0.3107$



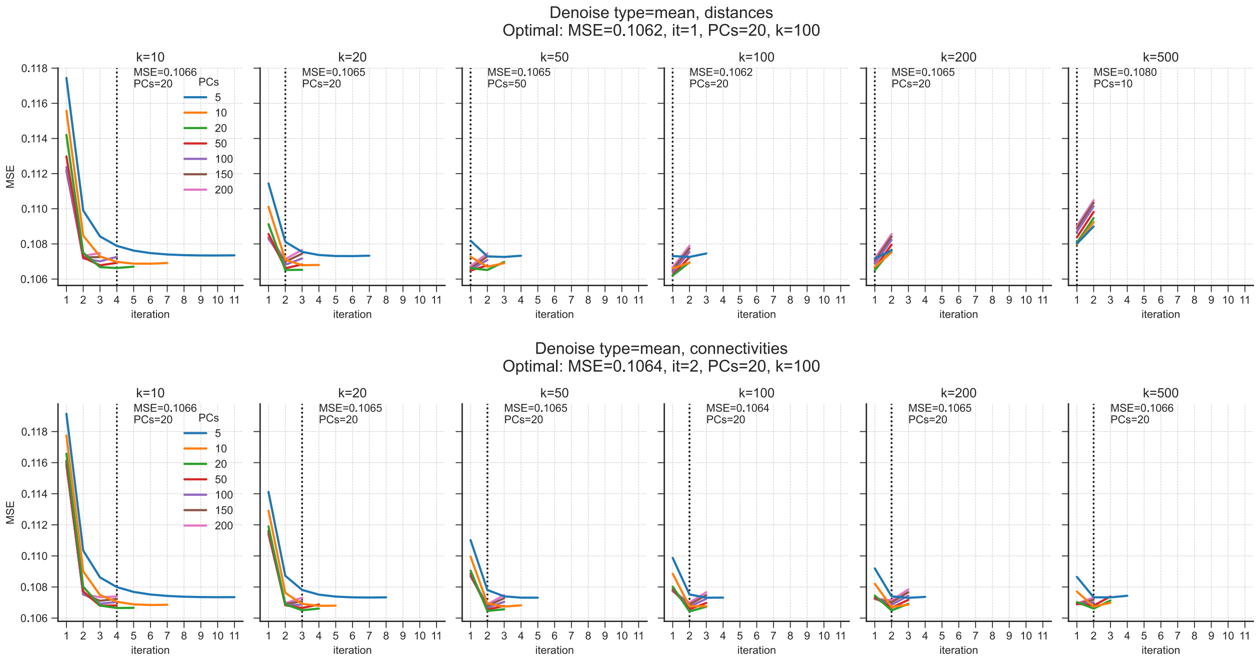
(b) EMT data. DEWÄKSS objective function values, mean square error, MSE during optimization of a range of hyperparameters, number of principal components (PCs) = 5, 10, 20, 50, 100, 200, 400}, number of initial neighbors (k) = {5, 10, 20, 50, 100, 200, 400}. The top row uses distances on the kNN-G and the bottom row uses connectivites of the kNN-G. Optimal settings are found to be 100PCs, $k = 100$ using distances with MSE $= 0.2215$. The algorithm's option to stop after the minimum is found is used here to stop the algorithm once the optimum (min MSE) is found.



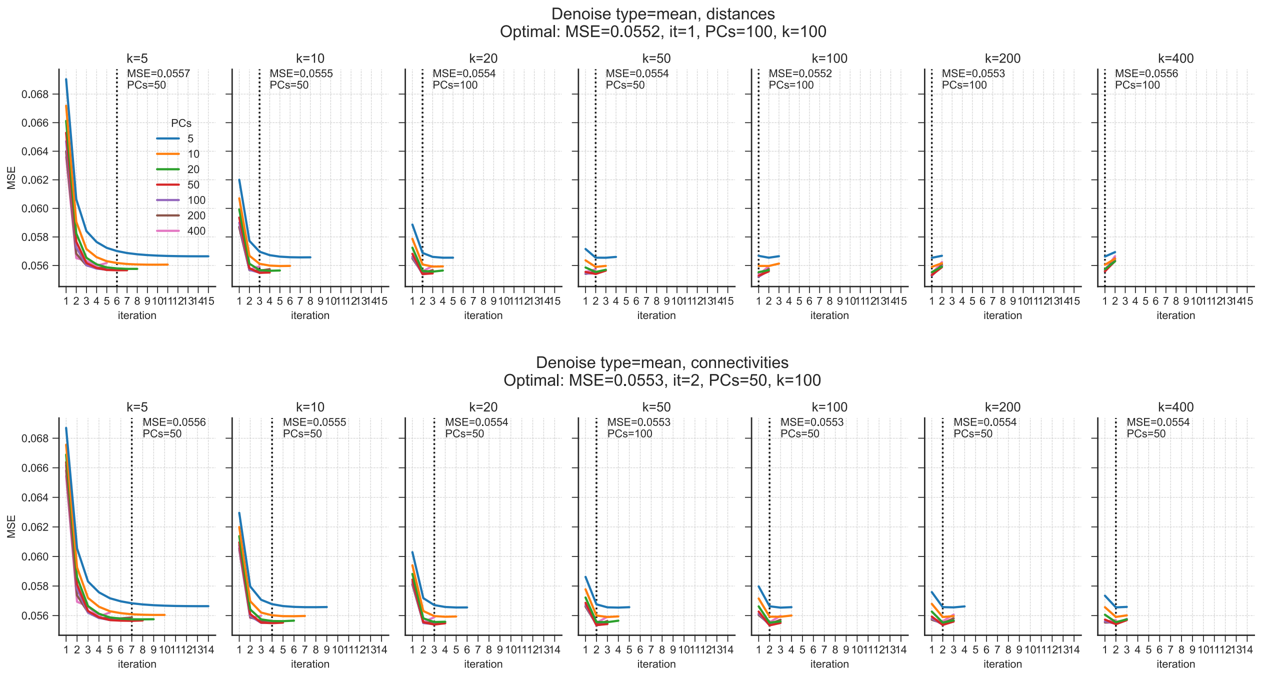
(c) Jackson et al. [20] YPD growth condition



(d) Jackson et al. [20] YPDRapa growth condition



(e) La Manno et al. [23] hgForebrainGlut



(f) La Manno et al. [23] DentateGyrus

Figure S.5: Optimal hyperparameter search for single-cell datasets. Each figure shows the algorithm, using distances (top row) and connectivities (bottom row). Each panel is scaled to the maximum number of observed iterations (x-axis) for any configuration run, with the objective function value MSE on the y-axis. The colored lines indicate the number of PCs used as input in the kNN-G distance computation. Each column corresponds to the initial number of neighbors k used for constructing the kNN-G.