University of Nevada, Reno

Novel Techniques for Single-cell RNA Sequencing Data Imputation and Clustering

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in Computer Science and Engineering

by

Bang Sy Tran

Dr. Tin Nguyen – Dissertation Advisor

August 2023

© by Bang Sy Tran 2023

All Rights Reserved

THE GRADUATE SCHOOL

We recommend that the dissertation
prepared under our supervision by

Bang Sy Tran

entitled

Novel Techniques for Single-cell RNA Sequencing Data Imputation and Clustering

be accepted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

Dr. Tin Nguyen
Advisor

Dr. Dongfang Zhao
Committee Member

Dr. Lei Yang
Committee Member

Dr. Hung (Jim) La
Committee Member

Dr. Pengbo Chu
Graduate School Representative

Markus Kemmelmeier, Ph.D., Dean
Graduate School

August, 2023

Abstract

Advances in single-cell technologies have shifted genomics research from the analysis of bulk tissues toward a comprehensive characterization of individual cells. These cutting-edge approaches enable the in-depth analysis of individual cells, unveiling the remarkable heterogeneity and complexity of cellular systems. By unraveling the unique signatures and functions of distinct cell types, single-cell technologies have not only deepened our understanding of fundamental biological processes but also unlocked new avenues for disease diagnostics and therapeutic interventions.

The applications of single-cell technologies extend beyond basic research, with significant implications for precision medicine, drug discovery, and regenerative medicine. By capturing the cellular heterogeneity within tumors, these methods have shed light on the mechanisms of tumor evolution, metastasis, and therapy resistance. Additionally, they have facilitated the identification of rare cell populations with specialized functions, such as stem cells and tissue-resident immune cells, which hold great promise for cell-based therapies.

However, one of the major challenges in analyzing scRNA-seq data is the prevalence of dropouts, which are instances where gene expression is not detected despite being present in the cell. Dropouts occur due to technical limitations and can introduce excessive noise into the data, obscuring the true biological signals. As a result, imputation methods are used to estimate missing values and reduce the impact of dropouts on downstream analyses. Furthermore, the high-dimensionality of scRNA-seq data presents additional challenges in effectively partitioning cell populations. Thus, robust computational approaches are required to overcome these challenges and extract meaningful biological insights from single-cell data.

There have been numerous imputation and clustering methods developed specifically to address the unique challenges associated with scRNA-seq data analysis. These methods aim to reduce the impact of dropouts and high dimensionality, allowing for accurate cell population partitioning and the discovery of meaningful biological insights. While these methods have unquestionably advanced the field of single-cell transcriptomics, they are not without limitations. Some methods may be computationally intensive, resulting in scalability issues with large datasets, whereas others may introduce biases or overfit the data, potentially affecting the accuracy of subsequent analyses. Furthermore, the performance of these methods can vary depending on the dataset's complexity and heterogeneity. As a result, ongoing research is required to improve existing methodologies and create new algorithms that address these limitations while retaining robustness and accuracy in scRNA-seq data analysis.

In this work, we propose three imputation approaches which incorporate with statistical and deep learning framework. We robustly reconstruct the gene expression matrix, effectively mitigating dropout effects and reducing noise. This results in the enhanced recovery of true biological signals from scRNA-seq data and leveraging transcriptomic profiles of single cells. In addition, we introduce a clustering method, which exploits the scRNA-seq data to identify cellular subpopulations. Our method employs a combination of dimensionality reduction and network fusion algorithms to generate a cell similarity graph. This approach accounts for both local and global structure within the data, enabling the discovery of rare and previously unidentified cell populations.

We plan to assess the imputation and clustering methods through rigorous benchmarking on simulated and more than 30 real scRNA-seq datasets against existing state-of-the-art techniques. We will show that the imputed data generated from our method can enhance the quality of downstream analyses. Also, we demonstrate that our clustering algorithm is efficient in accurately identifying the cells populations and capable of analyzing big datasets.

In conclusion, this thesis propose an alternative approaches to advance current state of scRNA-seq data analysis by developing innovative imputation and clustering methods that enable a more comprehensive and accurate characterization of cellular subpopulations. These advancements potentially have broad applicability in diverse research fields, including developmental biology, immunology, and oncology, where understanding cellular heterogeneity is crucial.

Dedication

To my loving father, Thắng, and my caring mother, Tình, who have always been my pillars of strength and sources of inspiration, I dedicate this thesis with my deepest gratitude. Your unwavering support and belief in me have been the driving forces behind my academic journey, and I am eternally grateful for the love, encouragement, and sacrifices you have made on my behalf.

To my dear wife, Quyên, thank you for your constant love, patience, and understanding as I pursued my dreams. Your companionship and support have been invaluable throughout this journey, and I am truly blessed to have you by my side.

To my precious son, Quang, you are my motivation to strive for excellence and to make the world a better place for you and future generations. I hope my accomplishments inspire you to chase your dreams and to never give up, no matter the challenges you may face.

To my supportive brother, Lợi, and my lovely niece, Ngân Khánh, thank you for always being there for me and for sharing in both the joys and struggles of this journey. Your unwavering faith in me has been a constant source of encouragement.

This thesis is a testament to the love, support, and guidance of my cherished family, and I dedicate it wholeheartedly to each one of you.

Acknowledgment

I would like to express my deepest gratitude to my advisor, Tin Nguyen, for his invaluable guidance, mentorship, and support throughout my doctoral journey. Your profound knowledge, wisdom, and unwavering belief in my capabilities have been instrumental in shaping my research and academic growth. I am truly honored to have had the opportunity to learn from and work with you.

I would also like to extend my sincere appreciation to my thesis committee members, Hung La, Dongfang Zhao, Lei Yang, and Pengbo Chu for their insightful feedback, constructive criticism, and encouragement during the course of my research. Your expertise and dedication to my development as a researcher have significantly contributed to the successful completion of this thesis.

My heartfelt thanks go to my colleagues and fellow graduate students in the Computer Science and Engineering at the Universty of Nevada, Reno for their camaraderie, intellectual discussions, and invaluable assistance throughout my doctoral studies. I am especially grateful to Hung Nguyen for their unwavering support, collaboration, and friendship.

Finally, I would like to dedicate this thesis to all the researchers who have come before me and those who will follow, in the pursuit of knowledge and the betterment of our world.

Table of Contents

Abstract **I**

1 Introduction **1**

2 Literature Review **5**

2.1 Single-cell RNA sequencing data imputation methods 6

2.1.1 Statistical-based scRNA-seq data imputation methods 7

2.1.2 Network-based scRNA-seq data imputation methods 9

2.1.3 Dimensionality reduction-based scRNA-seq data imputation methods 10

2.2 Single-cell RNA sequencing data clustering methods 12

3 Design of Computational Methods **14**

3.1 Sub-space regression-based imputation approach 16

3.1.1 Hypothesis testing and identification of dropout 17

3.1.2 Regression-based imputation 18

3.2 Hypergeometric testing and sub-space regression-based approach . . . 19

3.2.1 Hyper-geometric testing (Module 1) 21

3.2.2 Identifying gene subspaces (Module 2) 22

3.2.3 Subspace regression (Module 3) 25

3.3 Neural networks and regression-based imputation approach 26

3.3.1 Compressing data using autoencoders 27

3.3.2 Identifying dropouts and imputation 30
3.4 scRNA-seq data clustering using autoencoder and network fusion . . 31
3.4.1 Data compression using autoencoders (Module 1) 33
3.4.2 Network fusion and spectral clustering for cell segregation (Module 2) 36
3.4.3 Big data analysis (Module 3) 38
3.5 Validation 39
3.5.1 scRNA-seq data imputation validation 39
3.5.2 scRNA-seq clustering analysis validation 46

4 Method Validation and Results **53**
4.1 Results of imputation analysis using RIA 53
4.1.1 RIA improves the identification of sub-populations while preserving the biological landscape 54
4.1.2 RIA recovers temporal trajectories in embryonic developmental stages 58
4.2 Results of imputation analysis using scIDS 60
4.2.1 scIDS improves the identification of cells population. 60
4.2.2 scIDS preserves the biological landscape. 63
4.3 Results of imputation analysis using scISR 64
4.3.1 Cluster analysis of 25 scRNA-seq datasets 66
4.3.2 Preservation of the transcriptome landscape 72
4.3.3 Simulation studies 76
4.3.4 Robustness of scISR against non-uniform dropout probability 84
4.3.5 Simulation studies using Splatter package 91
4.3.6 Robustness of scISR against batch effect 92
4.4 Results of clustering analysis using scCAN 92

4.4.1 Estimating the number of true cell types 94
4.4.2 Segregating cells of different types 96
4.4.3 Time and space complexity 103
4.4.4 Comparison of the clustering methods used in Modules 2 and 3 104
4.4.5 Effects of min-max scaling 108
4.4.6 Rare cell types detection 109
4.4.7 Scalability of scCAN 111

5 Conclusions and Future Research 114

References 118

Appendices 153

Appendix A Publication list 153

A.1 Journal articles 153
A.2 Conference proceedings 154

List of Tables

3.1 Description of the eight single-cell datasets used to assess the performance of RIA 42
3.2 Description of the eight single-cell datasets used to assess the performance of scIDS. 42
3.3 Description of the 25 single-cell datasets used to assess the performance of scISR against other methods. The first three columns describe the name, accession ID, and tissue, while the following seven columns show the sequencing protocol, cell isolation technique, quantification scheme, normalized unit, dropout rate, number of cell types, and number of cells. 45
3.4 Description of the 28 single-cell datasets used to assess the performance of scCAN. The first two columns describe the name and tissue while the next five columns show the number of cells, number of cell types, sequencing protocol, accession ID, and references. The first 27 datasets have true cell labels and can be used to assess the accuracy of the clustering methods. 47
3.5 Link to 28 single-cell datasets used to benchmark scCAN. 48

4.1 Comparisons of RIA performance against other methods using adjusted Rand index (ARI). 54
4.2 Comparisons of RIA performance against other methods using Jaccard Index 56

4.3 Comparisons of RIA performance against other methods using Purity Index 56

4.4 Comparisons of scIDS performance against other methods using adjusted Rand index (ARI). 61

4.5 Comparisons of scIDS performance against other methods using adjusted mutual information (AMI). 62

4.6 Comparisons of scIDS performance against other methods using V-measure. 62

4.7 Adjusted Rand Index (ARI) obtained from raw and imputed data. In each row, a cell is highlighted in green if the ARI value is higher than that of the raw data. scISR improves cluster analysis by having ARI values higher than those of the raw data in 21 out of 25 datasets. A one-sided Wilcoxon test also confirms that the ARI values of scISR are significantly higher than those of raw data ($p = 3.2 \times 10^{-5}$) and of all other methods ($p = 9.8 \times 10^{-6}$). 72

4.8 Estimation of the number of cell types of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by absolute log-modulus values. Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). The average absolute log-modulus value of scCAN is 0.59, which are smaller than the rest. 97

4.9 Performance of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by Adjusted Rand Index (ARI). Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). Cells highlighted in bold have the highest ARI values. The average ARI of scCAN is 0.81, which is much higher than the rest (SEURAT3 is the second best with an average ARI of 0.55). In addition, scCAN has the highest ARI values in all but three datasets (Camp, Montoro and Hrvatin). 99

4.10 Performance of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by Adjusted Mutual Information (AMI). Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). Cells highlighted in bold have the highest AMI values. The average AMI of scCAN is 0.77, which is much higher than the rest (SEURAT3 is the second best with an average AMI of 0.64). In addition, scCAN has the highest AMI values in all but four datasets (Camp, Montoro, Chen and Hrvatin). 100

4.11 Performance of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by V-measure. Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). Cells highlighted in bold have the highest V-measure values. The average V-measure of scCAN is 0.81, which is much higher than the rest (SEURAT3 is the second best with an average V-measure of 0.72). In addition, scCAN has the highest V-measure values in all but four datasets (Romanov, Montoro, Chen and Kanton). 102

4.12 Performance of the two clustering methods used in Module 2 (method 1) and Module 3 (method 2) on single-cell datasets measured by adjusted Rand index (ARI), adjusted mutual information (AMI), V-measure and running time (minutes). Cells with NA values indicate that the method was not able to analyze the dataset (out-of-memory). Cells highlighted in bold have the higher accuracy (ARI, AMI, and V-measure) or lower running time. 107

List of Figures

3.1 The overall pipeline of RIA. The algorithm consists of two modules. In the first module, we apply a hypothesis testing approach to determine which genes need to be imputed and which genes can be used as training. In the second module, we adopt the generalized linear model to impute the missing values from the imputable set. The algorithm outputs the imputed matrix that has the same number of rows and columns as of the input data. 18

3.2 Single-cell Imputation using Subspace Regression (scISR). (A) Input data visualized in cell/sample space. (B) Hypergeometric test to determine whether each zero value is induced by dropout. Based on the computed p-values for each entry, we separate the original data into two sets of data: training data and imputable data. (C) Training data in which none of the values is induced by dropout events. (D) Imputable data in which each gene has at least one entry that is likely to be induced by dropout events. (E) Gene subspaces determined by perturbation clustering. We perturb the training data to discover the natural structure of the genes. Based on the pair-wise similarity between genes, we separate genes into groups that share similar patterns. (F) Subspace regression. We assign each gene in the imputable data to the closest subspace and then perform a generalized linear regression on the subspace to estimate the zero-valued entries that are impacted by dropouts. (G) Output expression matrix obtained by concatenating the training data and imputed data. 20

3.3 The resilience of pair-wise connectivity. (A) The dataset consists of three classes of genes: the first class has expression values of $\mathcal{N}(0,1)$, the second has expression values of $\mathcal{N}(1,1)$, and the third class has expression values of $\mathcal{N}(-1,1)$. (B) The original connectivity matrix (upper panel) and perturbed connectivity matrix (lower panel) for $k = 2$. (C) The connectivity matrices for $k = 5$. (D) The connectivity matrices for $k = 3$. The perturbed connectivity matrices clearly reveal the true structure of the data. 24

3.4 The overall analysis pipeline of scIDS. The input is a matrix in which rows represent cells and columns represent genes. In the first module (A), we perform features selection, data embedding, and cells clustering using two autoencoders. In the second module (B), we apply the z-test hypothesis testing to determine which genes need to be imputed. From the obtained genes set, we segregate cells into different groups using cluster assignment obtained from module A. For each group of cells, we adopt the generalized linear model to estimate the missing values using embedding data in module A. and we perform. The algorithm outputs the imputed matrix that has the same number of rows and columns as of the input data. 28

3.5 The overall analysis pipeline of scCAN consists of three modules. In the first module (A), we perform data normalization, gene filtering, and latent variables generation using two autoencoders. In the second module (B), we adopt the network fusion-based clustering method to segregate cell types for small data. The third module (C) aims at clustering big data using a combination of the network fusion approach and K nearest neighbors (k-NN) algorithm. 32

4.1 Transcriptomics landscape of the Zeisel dataset. The scatter plot shows first two principle components calculated by t-SNE for raw and imputation data using RIA, scImpute, and MAGIC. RIA preserve the transcriptomics landscape of the data whereas scImpute and MAGIC introduces artificial signals and complete change the landscape. . . . 57

4.2 Transcriptomics landscape and temporal development stages. The scatter plots show the first two dimensions of the t-SNE results calculated from Biase, Yan, Goolam, and Deng datasets. Due to dropouts, it is difficult to recognize different temporal dynamics of cells. The raw data and imputed data using scImpute and DrImpute do not show clear patterns. On the contrary, RIA significantly elucidates the cell lineage identification such that it is clearly recognized in the 2-D scatter plots. 59
4.3 The similarity between the imputed and original landscapes. 63
4.4 The visualization of Baron dataset. 65
4.5 Running time of the six imputation methods on 25 real scRNA-seq datasets. scISR is the fastest and can impute the Darrah dataset in 50 minutes. 68
4.6 Adjusted Rand Index (ARI) obtained from raw and imputed data. The x-axis shows the names of the datasets while the y-axis shows ARI value of each method. scISR improves cluster analysis by having ARI values higher than those of the raw data in 21 out of 25 datasets. 70
4.7 Assessment results of each imputation method with respect to cell isolation techniques, quantification schemes, or normalized units. The analysis is performed with a log transformation of the data. Panel (A) shows the results using Adjusted Rand Index (ARI), while panels (B) and (C) show the results using Jaccard Index (JI) and Purity Index (PI). scISR consistently outperforms other methods in every grouping by having the highest ARI, JI, and PI values. 71

4.8 The distance correlation between raw data and imputed data using the first two components obtained from t-SNE and UMAP. Higher correlation values indicate more similarity between the imputed and original landscapes. Different colors represent different imputation methods. scISR has the highest mean correlation with the smallest variance. A one-sided Wilcoxon test indicates that the correlation values obtained from scISR are significantly higher than the rest ($p = 3 \times 10^{-9}$ and 2.8×10^{-7} for t-SNE and UMAP, respectively). 75

4.9 Assessment of MAGIC, scImpute, SAVER, and scISR using simulation (100 cells and 300 genes). (A) – (H) The visualization of the *complete data*, *masked data* and *imputed data* recovered by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR. In each subfigure, the left panel shows the transcriptome landscape using t-SNE while the right panel shows the gene-cell heatmap. (I) Mean Absolute Error (MAE) and correlation coefficients obtained by comparing masked/imputed data with the complete data. We calculate the MAE and correlation values for each gene and then plot the distributions of each metric using boxplot. The transcriptome landscapes and heatmaps show that scISR comes closest to recovering the complete data. scISR also has significantly smaller MAE values as well as significantly higher correlation coefficients than other methods with p-values 1.6×10^{-64} and 9.2×10^{-63}, respectively (Wilcoxon test). 78

4.10 Assessment of MAGIC, scImpute, SAVER, and scISR using simulation of 1,000 cells. (A) – (H) The visualization of the *complete data*, *masked data* and *imputed data* recovered by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR. In each subfigure, the left panel shows the transcriptome landscape using t-SNE while the right panel shows the gene-cell heatmap. (I) Mean Absolute Error (MAE) and correlation coefficients obtained by comparing masked/imputed data with the complete data. We calculate the MAE and correlation values for each gene and then plot the distributions of each metric using boxplot. The transcriptome landscapes and heatmaps show that scISR comes closest to recovering the complete data. scISR also has significantly smaller MAE values as well as significantly higher correlation coefficients than other methods with p-values $< 10^{-100}$ and $< 10^{-100}$, respectively (using Wilcoxon test). 79

4.11 Assessment of MAGIC, scImpute, SAVER, and scISR using simulation of 10,000 cells. (A) – (H) The visualization of the *complete data*, *masked data* and *imputed data* recovered by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR. In each subfigure, the left panel shows the transcriptome landscape using t-SNE while the right panel shows the gene-cell heatmap. (I) Mean Absolute Error (MAE) and correlation coefficients obtained by comparing masked/imputed data with the complete data. We calculate the MAE and correlation values for each gene and then plot the distributions of each metric using boxplot. The transcriptome landscapes and heatmaps show that scISR comes closest to recovering the complete data. scISR also has significantly smaller MAE values as well as significantly higher correlation coefficients than other methods with p-values $< 10^{-100}$ and $< 10^{-100}$, respectively (using Wilcoxon test). 80

4.12 Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using simulation studies. Mean Absolute Error (MAE) and correlation coefficients were obtained by comparing imputed data with the complete data. In each analysis, scISR has smaller MAE values and higher correlation coefficients than other methods. 82

4.13 Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using simulated datasets with different dropout distributions and sample sizes. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. In each panel, the left side shows the results for uniform distributions while the right side shows the results for normal distributions. For small datasets (e.g., datasets with 1,000 cells) with high dropout rates, scISR is less accurate when the dropout probability is normally distributed. When the sample size increases, scISR becomes more accurate. For datasets with 7,000 cells or more, scISR performs well for both uniform and normal distributions alike across all dropout rates. For most of the dataset sizes and dropout rates, scISR have a much better median MAE and correlation compared to other methods. 87

4.14 Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using simulated datasets with different dropout distributions and sample sizes. In each dataset, cells of the same type have high correlation and cells of different types have low correlation. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. In each panel, the left side shows the results for uniform distributions while the right side shows the results for normal distributions. For small datasets (e.g., datasets with 1,000 cells) with high dropout rates, scISR is less accurate when the dropout probability is normally distributed. When the sample size increases, scISR becomes more accurate. For datasets with 7,000 cells or more, scISR performs well for both uniform and normal distributions alike across all dropout rates. For most of the dataset sizes and dropout rates, scISR have a much better median MAE and correlation compared to other methods. 89

4.15 The accuracy of scISR hypothesis testing using F-score. The F-score measures how well the algorithm distinguish between true zero values and dropouts. The left panel shows the F-scores for datasets with uniform distribution while the right panel shows the F-scores for datasets with normal distribution. For datasets with 7,000 cells or more, the median F-scores are close to 1 for both uniform and normal distributions alike across all dropout rates. In other words, scISR accurately identifies the zero values that need to be imputed. 90

4.16 Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using datasets simulated by Splatter. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. scISR and scScope are the only methods that can perform imputation on the biggest dataset, while MAGIC, SAVER, scImpute, and scGNN stop working with datasets bigger than 100,000, 10,000, 10,000, and 50,000 cells, respectively. scISR is the only method that can improve the dropout data in all scenarios. 93

4.17 Running time of the six imputation methods on simulated datasets. These datasets have 15,000 cells and varying number of cells (5,000 to 200,000). scISR and scScope are the only methods that can analyze all datasets. The two methods can finish the analysis of 200,000 cells in 200 and 100 minutes, respectively. 94

4.18 Impact of batch effects on scISR. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. In each panel, the left 10 boxes show the results for data without batch effects while the right 10 boxes show the results for data with batch effects. Overall, batch effects do not have a significant impact on the performance of scISR. 94

4.19 Absolute log-modulus values obtained from CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN for 27 real scRNA-seq datasets. This metric measures the difference between the number of clusters and the number of true cell types. The average log modulus of scCAN is 0.59 while those of Monocle3, SCANPY, SHARP, SEURAT3, and CIDR are 1.35, 1, 0.72, 0.64, and 0.63, respectively. scCAN significantly outperforms other methods by having the smallest absolute log-modulus values (Wilcoxon p-value of $p = 8.6 \times 10^{-4}$). Note that the dataset Brain 1.3M was excluded from this analysis because it does not have true cell type information. 95

4.20 Accuracy assessment of the six clustering methods using adjusted Rand index (ARI), adjusted mutual information (AMI), and V-measure. scCAN consistently and substantially outperforms other methods in every assessment by having the highest ARI, AMI, and V-measure values across 27 real scRNA-seq datasets. 98

4.21 Assessment of CIDR, SEURAT3, Monocle3, SHARP, SCANPY and, scCAN against dropouts. Simulations were obtained by varying the number of zeros in each of 27 real biological datasets from 50% to about 90%, respectively. Each box plot shows the ARI values obtained from each method for a specific dropout portion. Wilcoxon test shows that the ARI values obtained from scCAN are significantly higher than CIDR, SEURAT3, Monocle3, SHARP, SCANPY ($p < 2.2 \times 10^{-16}$). . 103

4.22 Running time of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN for the analysis of 28 real scRNA-seq datasets. The horizontal axis shows the number of cells while the vertical axis shows the running time in the log scale (base 60) of minutes. scCAN and SCANPY are the only two methods that can analyze datasets with more than 200,000 cells. 104

4.23 Impact of min-max scaling on scCAN. The analysis without scaling has higher variability and lower ARI values. 109

4.24 Rare cell type detection using the Zilionis dataset as example. The dataset has a total of 34,558 cells, in which there are 108 tRBC cells (rare cell type with 0.3% prevalence). (A) Transciptome landscape and true cell types. (B) Clustering results using scCAN with default sample size ($samp.size = 5,000$), in which tRBC are mistakenly grouped with tPlasma cells. (C) Clustering results with sample size of 10,000 ($samp.size = 10,000$). In this case, scCAN properly separates tRBC cells in cluster 2 with an F1 score of 0.9. (D) Clustering results using two-stage strategy and default sample size ($samp.size = 5,000$). scCAN properly separates tRBC cells in cluster 2 with a perfect F1 score of 1. 112

4.25 Clustering results of the Brain 1.3M dataset using scCAN and SCANPY. The left panel shows cell annotation of 20 clusters discovered by SCANPY. The right panel shows the cell partitions of 19 clusters identified from scCAN. 113

4.26 Visualizing of the Cao dataset using t-SNE. (A) Transcriptome landscape with true cell type information. (B) Transcriptome landscape of the clusters identified by SCANPY. (C) Transciptome landscape of clusters identified by scCAN. scCAN outperforms SCANPY by having a higher ARI value. 113

Chapter 1

Introduction

Single-cell RNA sequencing (scRNA-seq) was first known in 2009 when Tang et al. [1] monitored how individual cells respond to signals and other environmental cues at critical stages of cell-fate. However, scRNA-seq had not gain major attention until 2014 when sequencing cost became more affordable. Since then, a number of scRNA-seq protocols have been developed to isolate single cells and to prepare cDNA libraries using next generation sequencing (NGS) platforms [2, 3]. These advancements in single-cell sequencing hold enormous opportunities for both basic biology and clinical applications. For example, scRNA-seq disclosed diverse characteristic of cells within a seemingly analogous cell population or tissue, and revealed insights into cell identity, cell fate, and cellular functions [4]. Single-cell data was also used to detect highly variable genes (HVGs) that contribute for heterogeneity across cells in a cell population, to discover the relationship between genes and cellular phenotypes, or to identify new rare cell types via dimensionality reduction and clustering.

However, scRNA-seq data come with additional challenges [5, 6]. One challenge is that sequencing mRNA within individual cells requires artificial amplification of DNA materials millions of times, leading to disproportionate distortions of relative transcript abundance and gene expression. Another outstanding challenge is the “dropout” phenomenon where a gene was highly expressed in one cell but did not express at all in another cell [7]. These dropout events usually occur due to the limitation of sequencing technologies when only a low amount of starting mRNA in individual cells can be captured, leading to low sequencing depth and failed amplification [8–10].

Imputation has emerged as an effective strategy to mitigate the impact of dropouts in scRNA-seq data by estimating and filling in the missing gene expression values. By leveraging the intrinsic structure and patterns present in the data, imputation algorithms can enhance the quality and reliability of the gene expression profiles, thus facilitating downstream analyses such as clustering and differential expression analysis. The application of imputation methods [11–20] in scRNA-seq data processing not only improves the accuracy of biological interpretation but also enables a more comprehensive understanding of cellular heterogeneity and function.

Besides imputation, unsupervised clustering is an important application for analyzing scRNA-seq data and identifying cell types or sub-populations [16, 21–24]. It allows us to identify putative cell types and can provide insight into cellular function. This opportunity has spurred the creation of several atlas projects, most notably the Human Cell Atlas [25]. These atlas projects aim to build comprehensive references for all cell types present in an organism or tissue at various stages of development. In addition to providing a deeper understanding of the basic biology, atlases will also be useful as references for disease studies. For a cell atlas to be of practical use, reliable methods for unsupervised clustering of the cells will be one of the key computational challenges.

Although considerable progress has been made in terms of imputing and clustering algorithms over the past few years, a number of questions remain challenged.

1. The high dropout rate in scRNA-seq data is a significant challenge for imputation methods, as it may lead to false negatives and impact the downstream

analysis. Accurate imputation is essential to minimize the impact of these false negatives. The high data sparsity makes it challenging for imputation methods to differentiate between true zero values and dropout events. Also, this sparsity makes it difficult for clustering methods to identify meaningful patterns and differentiate between true biological variability and technical noise.

2. scRNA-seq data is high-dimensional, with thousands of genes measured for each cell. High dimensionality can lead to the "curse of dimensionality" in imputation and clustering, where the increased complexity hinders the ability to recover the true genes expressions and identification of meaningful cell patterns.

3. scRNA-seq data contains cells from different types, states, and conditions, which increases the complexity of the imputation task. Imputation methods need to account for this heterogeneity without introducing biases. Also, cellular heterogeneity in scRNA-seq data makes it challenging to identify the correct number of clusters and accurately separate different cell types or states.

4. Some imputation methods can over-smooth the data or overestimate gene expression levels, which might lead to the loss of biologically relevant information and distort downstream analyses.

5. Clustering results can be sensitive to the choice of algorithm, distance metric, and parameter settings. Ensuring the stability and reproducibility of clustering results is crucial for reliable interpretation.

Despite aforementioned challenges, scRNA-seq data imputation and clustering remain as important applications for downstream analyses of scRNA-seq data. Imputation methods can help to address missing values in the data, while clustering allows for the identification of distinct cell types and subtypes. These applications can provide important insights into the cellular and molecular mechanisms underlying biological processes, and have the potential to drive new discoveries in fields such as developmental biology, cancer research, and immunology.

In this proposal, we will first attempt to eliminate dropouts phenomenon in scRNA-seq data. In the first and second framework, we introduce RIA and scIR, regression-based approaches, that are able to reliably recover the missing values in single-cell data and thus can effectively improve the performance of downstream analyses. In the third framework, we propose scIDS, another novel imputation method that is a combination of deep autoencoder neural networks and subspace regression to reliably recover the missing values in scRNA-seq data.

The second part of the proposal aims to address the challenge of clustering scRNA-seq data in a scalable and efficient manner. To achieve this, we propose to develop a novel pipeline that utilizes multi-stage autoencoders and spectral clustering algorithm. The proposed pipeline will involve encoding the high-dimensional scRNA-seq data into low-dimensional representations using multi-stage autoencoders. These low-dimensional representations will then be clustered using the spectral clustering algorithm, which has been shown to be highly effective for clustering high-dimensional data. The proposed pipeline is expected to provide a scalable solution for clustering large scRNA-seq datasets, and to enable the identification of novel cell types and subtypes in a wide range of biological systems. Overall, the second part of the proposal represents an important step towards improving our understanding of cellular diversity and function in health and disease.

The rest of this proposal is outlined as follows: in Chapter 2, we provide an overview of related studies and state-of-the-art scRNA-seq imputation and clustering methods. In Chapter 3, we will present our proposed approaches, analyzed datasets and our plan for validation. Finally, we conclude this proposal by summarizing the contributions of our work.

Chapter 2

Literature Review

With the rapid advancement of single-cell RNA sequencing (scRNA-seq) technology, there have been significant developments in imputation and clustering methods to analyze scRNA-seq data. The initial release of these methods [11, 16] has provided a valuable foundation for the field of scRNA-seq analysis. However, despite these advancements, there are still limitations in the current imputation and clustering methods. For instance, some methods may not perform well on low-quality datasets, and others may have limitations in scalability or may require complex parameter tuning. Therefore, further studies are needed to improve these methods and to develop new ones that can overcome these limitations. In this chapter, we aim to provide an overview of the current status of imputation and clustering methods for scRNA-seq data, including their strengths, limitations, and future directions for improvement.

2.1 Single-cell RNA sequencing data imputation methods

scRNA-seq sequencing technologies offer power tools to measure gene expression in individual cell [26–29]. In contrast to bulk RNA-sequencing (RNA-seq), a distinctive feature of data measured using single-cell RNA-sequencing (scRNA-seq) is the increased sparsity, or fraction of observed "dropouts," where a number of zeros refers to no reads mapping to a given gene in a cell [7, 30–32]. These observed zeros can be due to biological fluctuations in the measured trait or technical limitations related to challenges in quantifying small numbers of molecules. To address the increased sparsity observed in scRNA-seq data, recent work has led to the development of "imputation" methods, in a similar spirit to imputing gene expression data that are missing or not observed.

Single-cell RNA sequencing (scRNA-seq) data imputation methods can be broadly categorized into three main groups: statistical-based, network-based, and dimensionality reduction-based methods. The first group consists of imputation methods that represent sparsity directly using probabilistic models. These methods might or might not differentiate between biological and technical zeros, but if they do, they typically only impute gene expression values for technical zeros. A second method modifies (typically) all values (zero and non-zero) by smoothing or diffusing gene expression values in cells with similar expression profiles identified, for instance, by graph neighbors. The third approach identifies a latent space representation of the cells using either low-rank matrix-based methods (capturing linear relationships) or deep-learning methods (capturing non-linear relationships), and then reconstructs the observed expression matrix from the no longer sparse low-rank or estimated latent spaces. For deep-learning techniques, such as variational autoencoders, both the estimated latent spaces and the "imputed" data (i.e., the reconstructed expression matrix) can be used for downstream analyses, whereas ordinarily only the imputed data is provided for downstream analyses.

2.1.1 Statistical-based scRNA-seq data imputation methods

Methods in the first category include SAVER [17], SAVER-X [33], scImpute [18], BISCUIT [19], bayNorm [34], scRecover [35], and VIPER [36] that model dropouts in scRNA dataset as a mixture of different distributions. Those methods share comment steps to perform imputation: (i) dropouts modeling, (ii) parameter estimation, (iii) dodel fitting, and (iv) dropout imputation using fitted model.

SAVER models read counts as a mixture of Poisson-Gamma and then uses a Bayesian approach to estimate true expression values of genes by borrowing information across genes. Similar to SAVER, SAVER-X also offers the option to pre-train hyper-parameters estimated from preexisting datasets using transfer learning. Despite initial success on some small datasets [37–40], both assumes that gene-gene relationships are linear. This may not be true for all genes, leading to inaccurate imputation for genes with non-linear relationships. Those methods can be computationally expensive, especially for large datasets with many cells and genes. In addition, SAVER-X is susceptible for overfitting when pre-trained model is too specialized for the source dataset. The model may not generalize well to the target dataset.

scImpute models the gene expression as a mixture of two different distributions: the Gaussian distribution represents the actual gene expression while the Gamma distribution accounts for the dropout events. scImpute estimates the parameters of the mixture model using the Expectation-Maximization (EM) algorithm [41]. Genes with a high dropout rate are consider imputable while genes with low dropout rate do not need imputation. The method then uses a non-negative least square to impute genes with high dropout rates. The EM-based strategy involves estimation of many parameters for all genes across the whole genome. This makes the methods very slow and vulnerable to overfitting. scImpute attempts to alter the expression of all genes, including those that are not affected by dropout events.

Similar to SAVER and scImpute, BISCUIT [19] uses the Dirichlet process mixture model [20] to repeatedly perform the processing steps such as normalization, sc-RNA data imputation, and cells clustering by simultaneously inferring clustering parameters, estimating technical variations (e.g. library size), and learning co-expression structures of each cluster. Another method, BayNorm models gene expression levels using a gamma-Poisson hierarchical model, which captures both technical noise and biological variability. BayNorm estimates cell-specific scaling factors that account for differences in sequencing depth and efficiency among cells. These scaling factors help to normalize the data and reduce technical biases. The method accounts for dropout events by estimating the probability of observing a zero count for each gene in each cell. This allows for the imputation of missing gene expression values and the recovery of dropout events. BayNorm employs a Bayesian approach to estimate model parameters, using Markov chain Monte Carlo (MCMC) [42] sampling to obtain posterior distributions for each parameter. While BayNorm provides an effective method for gene expression recovery, ensuring convergence of the MCMC chains can be challenging, and improper convergence may lead to biased parameter estimates.

scRecover [35] is based on a zero-inflated negative binomial distribution model that attempts to adapt to high drop-out rates, whereas VIPER [36] is based on a sparse non-negative regression model. The models were estimated based on log-transformed normalized gene expression data, it may not perform as well when dealing with count data that has low sequencing depth. In such cases, VIPER's assumptions and modeling techniques might not adequately capture the characteristics of the data, leading to less accurate imputations and potentially impacting downstream analyses.

In summary, model-based imputation methods for single-cell data have emerged as effective tools for dealing with missing gene expression values in scRNA-seq datasets. However, they have some main drawbacks that must be considered. These methods rely on assumptions about the underlying data distribution, gene-gene relationships, and gene expression patterns, which may not always be correct, resulting in inaccurate imputations. Furthermore, the computational complexity associated with model-based methods can be difficult, especially for large scRNA-seq datasets with many cells and genes, which may necessitate powerful hardware or efficient algorithms. Furthermore, when dealing with sparse data, as well as lowly expressed or rare genes, where there may not be enough information to accurately model their expression patterns, their performance can be suboptimal.

2.1.2 Network-based scRNA-seq data imputation methods

Network-based methods rely on the assumption that cells with similar gene expression profiles are likely to have similar missing values, and thus impute the missing data by leveraging relationships between cells in the high-dimensional space. Methods in this category includes MAGIC [11], DrImpute [13], knn-smoothing [43], netSmooth [44], and 2S3 [45].

MAGIC [11] was one of the first imputation method that is able to impute single-cell data on a genomic scale. MAGIC imputes zero expression value by using heat diffusion [12] concept. It first constructs the affinity matrix between cells using Gaussian kernel and then constructs a Markov transition matrix by normalizing the sc-RNA similarity matrix. Next, the weights of the other cells are estimated by the transition matrix.

DrImpute which is based on the cluster ensemble strategy [14] using consensus clustering [15, 16] as the basic clustering algorithm. It performs clustering for a predefined number of times and imputes the data by averaging value of similar cells. If the number of clusters is not provided by users, DrImpute will use some default values that might not be optimal for the data. DrImpute relies on many parameters to fine-tune their model, which often leads to overfitting. This makes their results unreliable, i.e., the imputation is sensitive to a slight change in the input data or in parameter settings. knn-smoothing leverages the k-Nearest Neighbors (KNN) algorithm to estimate and fill in the missing values by considering the similarities between data points. Similarly, 2S3 and netSmooth incorporate the biological network, effectively smoothing the data by incorporating information from neighboring nodes.

2.1.3 Dimensionality reduction-based scRNA-seq data imputation methods

Dimensionality reduction-based scRNA-seq data imputation is a powerful technique for handling the inherent noise and sparsity present in single-cell RNA sequencing data. This approach first identifies a latent space representation of the cells, either using low-rank matrix-based methods (capturing linear relationships) or deep-learning methods (capturing non-linear relationships), and then reconstructs the observed expression matrix from the low-rank or estimated latent spaces, which are no longer sparse. In this section we will systematically review available methods that employ dimensionality reduction-based approach in their pipeline

Deep learning-based scRNA-seq data imputation methods

Deep learning algorithms, such as autoencoders and convolutional neural networks, have demonstrated remarkable capabilities in capturing complex patterns and structures within the data. By leveraging these architectures, deep learning-based imputation methods can effectively model the gene expression dependencies and provide more accurate imputations for scRNA-seq data. This section provides a brief survey of imputation methods that employ deep learning algorithms in the analysis pipeline.

In this work, we provide an review of six imputation methods, namely, AutoImpute [46], DCA [47], DeepImpute [48], SAUCIE [49], scScope [50], and scVI [51]. In these methods, a latent space is constructed using deep learning models to represent cells by low-dimensional latent variables which are used to reconstruct gene expression. The latent space representation can be used for downstream analyses, such as clustering the cells or inferring pseudotime trajectories on the cells, but not for differential gene expression analysis. DCA is a deep count autoencoder network that uses a negative binomial noise model with or without zero-inflation (depending on the dispersion learned form data) and captures nonlinear gene-gene dependencies. scVI is based on a hierarchical Bayesian model and applies deep neural networks to specify the conditional distributions of variables where the latent variables are mapped to the zero-inflated negative binomial distribution. AutoImpute applies overcomplete autoencoders and tends to be more conservative by considering the expression values as truly zeros if the genes are silenced across bulk samples. DeepImpute constructs multiple sub-neural networks to impute sets of target genes using genes highly correlated with the target genes. SAUCIE is a regularized autoencoder that uses the reconstructed signal from autoencoder to denoise and impute the data. ScScope iteratively performs imputation using a recurrent network layer.

Low-rank matrix representation-based scRNA-seq data imputation methods

Low-rank matrix representation-based scRNA-seq data imputation method, which leverages the intrinsic low-dimensional structure of gene expression data to accurately estimate and fill in missing values. This approach assumes that the underlying true gene expression matrix can be approximated by a low-rank matrix, thus exploiting the correlations among genes and cells to effectively recover the missing information. These include 3 methods, namely, ALRA [52], mcImpute [53], and PBLR [54]. In these low-rank matrix-based methods, cell profiles are mapped to a low-dimensional linear space for imputation. ALRA uses SVD decomposition followed by a thresholding scheme. mcImpute uses nuclear norm minimization, a matrix completion algorithm. PBLR first groups cells into subpopulations and then runs a bounded low-rank matrix recovery method within each cell sub-population.

2.2 Single-cell RNA sequencing data clustering methods

The large number of cells (up to millions) and the high-dimensionality of the data (tens of thousands of genes or features) present substantial challenges to computational methods. This section provides a survey of the current clustering method.

One prominent strategy is to reduce the dimensionality of the data before performing cluster analysis. Methods in this category include SC3 [16], CIDR [21], pcaReduce [22], SEURAT2 [23], SIMLR [24], and SHARP [55]. These methods typically apply dimension reduction techniques such as PCA [56], t-SNE [57] and UMAP [58] to obtain a lower-dimensional representation of the data. Deep-learning-based approaches, including scDeepCluster [59], scAIDE [60], SCA [61], AAE-SC [62], and scGMAI [63], often use autoencoders to select important features and to project the data onto a low-dimensional latent space. Next, these clustering methods partition the cells using established clustering algorithms (e.g., k-means, spectral clustering, etc.). Since these dimension reduction techniques are sensitive to sequencing platforms [64] and dropouts [10], the quality of clustering results also vary accordingly.

Another strategy is to iteratively search for hierarchical structures over both cells and genes. Methods using this strategy include BackSPIN [40], SAIC [65], and Panoview [66]). These methods attempt to iteratively divide cells and genes into sub-groups to maximize cell similarity within each cluster. These methods, however, require excessive computational power (due to the iteration), and overestimate the number of cell types.

Many single-cell methods also utilize community detection algorithms such as Louvain [67] and Leiden [68]. SEURAT3 [69], SCANPY [70], and Monocle3 [71] embed community detection algorithms in their analysis pipeline. These methods first convert scRNA-seq data into networks in which cells are nodes and the edges represent similarity among them. Next, they partition the network using community detection algorithms that are known to be fast. The quality of clustering results strongly depends on the construction of the similarity network. Further, although community detection algorithms can produce reasonable results, they often overestimate the number of cell communities (cell types).

Lastly, cluster ensemble is another strategy that aims to aggregate results from multiple clustering models. Methods of this class include SAFE [72], SAME [73], and Sc-GPE [74]; these methods selectively combine the resulted obtained from multiple clustering algorithms, including SC3, CIDR, SEURAT, CIDR, SIMLR, SNN-cliq [75], SSNN-Louvain [76], and MPGS-Louvain [77]. One of the main drawbacks of clustering ensemble methods is that they do not scale well for large datasets. Moreover, evaluating the quality of clustering results obtained from each individual method is a difficult task because there is no universally agreed standard on what constitutes good quality clusters in the first place [78].

Chapter 3

Design of Computational Methods

This chapter is based on the following publications:

1. ***Bang Tran**, Duc Tran, Hung Nguyen, Nam Sy Vo, and Tin Nguyen. RIA: a novel regression-based imputation approach for single-cell RNA sequencing. In Proceedings of the 11th International Conference on Knowledge and Systems Engineering (KSE), 2019.*

2. *Duc Tran, **Bang Tran**, Hung Nguyen, and Tin Nguyen. A novel method for single-cell data imputation using subspace regression. Scientific Reports, 2022. DOI: 10.1038/s41598-022-06500-4.*

3. ***Bang Tran**, Quyen Nguyen, Sangam Shrestha, and Tin Nguyen. scIDS: Single-cell Imputation by combining Deep autoencoder neural networks and Subspace regression. In Proceedings of the 13th International Conference on Knowledge and Systems Engineering (KSE), 2021.*

4. ***Bang Tran**, Duc Tran, Hung Nguyen, Seungil Ro, and Tin Nguyen. scCAN: single-cell clustering using autoencoder and network fusion. Scientific Reports, 2022. DOI: 10.1038/s41598-022-06500-4.*

In the rapidly evolving field of scRNA-seq data analysis, the development of efficient and accurate methods for handling missing or dropout values is of utmost importance. Among the many approaches, we propose on two highly effective methods, RIA, scISR and scIDS. RIA employs a regression-based imputation approach that leverages the inherent relationships between gene expression levels to estimate and replace missing values in scRNAseq data. This technique offers a robust and straightforward solution to address the dropout problem and provides an improved basis for subsequent data analysis tasks. scISR improves the dropouts identification using hypergeometric testing and impute the missing values by sub-space regression approach. On the other hand, scIDS combines the power of neural networks with regression-based imputation, integrating the advantages of both techniques to achieve a more comprehensive imputation. In scIDS, a neural network learns the complex patterns and dependencies within the data, while regression-based imputation serves as a supplementary mechanism to fill in missing values. This combination results in a more accurate representation of the underlying biological processes, ultimately leading to a better understanding of cellular heterogeneity and function.

Alongside these imputation methods, the ability to cluster and analyze large-scale scRNAseq data has become increasingly crucial. In response to this need, we introduce scCAN (Single-cell Clustering using Autoencoders and Nearest-Neighbor), a novel scRNAseq method that efficiently clusters big data by combining the strengths of autoencoder-based dimensionality reduction and spectral clustering techniques. The autoencoder, a powerful deep learning model, is employed to compress high-dimensional scRNAseq data into a lower-dimensional space, effectively preserving the critical features and inherent structure of the data. This compact representation not only reduces computational complexity but also mitigates the curse of dimensionality, enhancing the performance of clustering algorithms. Subsequently, spectral clustering is applied to the reduced-dimension data to reveal the underlying structure and identify biologically relevant cell subpopulations. By harnessing the synergistic effects of autoencoders and spectral clustering, scCAN offers a highly effective solution for analyzing large-scale scRNAseq data, facilitating the discovery of novel cell types, differentiation pathways, and gene regulatory networks in complex biological systems.

Also in this chapter, we will present our validation plan for the two imputation methods and one clustering method. We will use more than 30 publicly available datasets with more than 1 millions of cells collected from Expression Omnibus (GEO) [79], European Bioinformatics Institute (`https://www.ebi.ac.uk/gxa/sc/experiments/`), Broad Institute Single Cell Portal (`https://singlecell.broadinstitute.org/single_cell`), and 10X Genomics website (`https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons`). In each of the proposed method, we will benchmark the performance against state-of-the-art methods.

3.1 Sub-space regression-based imputation approach

Here we propose a new approach, RIA, that can reliably impute missing values from single-cell data. Our method consists of two modules. The first module performs a hypothesis testing to identify the values that are likely to be impacted by the dropout events. The second module estimates the missing value using a robust regression approach. All of the parameters are learned from the data themselves. The approach is tested using five benchmarking datasets with a total of 3,535 cells. We demonstrate that RIA outperforms existing imputation methods in improving the identification of cell population and temporal trajectories.

Figure 3.1 shows the overall analysis pipeline of RIA. The input of RIA is a matrix in which rows represent genes/components and columns represent cells/samples. RIA first performs a hypothesis testing to determine genes that have accurate values with high confidence. These genes are will be used as the training set. The rest of the genes (genes that need to be imputed) will be the imputable set. The method then uses a generalized linear model to learn from the training set and to impute the missing data in the imputable set. Finally, RIA concatenates the two sets of genes and outputs a matrix that has the same number of rows and columns as of the input matrix.

3.1.1 Hypothesis testing and identification of dropout

In order to impute the missing data without introducing false signals to the original data, it is important to determine which genes are impacted by dropouts and which genes do not need imputation. Therefore, we have developed a hypothesis testing approach to determine the set of genes that are likely to be impacted by dropouts.

Our approach is based on the observation that for genes that are not impacted by dropouts, the log-transformed expression values are normally distributed [18, 80]. Therefore, we use z-test to determine whether a zero value is observed by chance or by the impact of dropout events. For each gene g, we use the non-zero expression values to determine the parameters μ and σ of the Gaussian distribution. Next, we use z-test to estimate how likely a zero value occurs, given that the expression values follow the estimated Gaussian distribution. If the chance of observing a zero value is less than the significance threshold (0.05), we conclude that gene g is likely to be affected by dropout. By repeating this process for all genes, we can divide our data into two sets of genes: a set G that include genes affected by dropout, and a set M that have high confidence of not being affected by dropout.

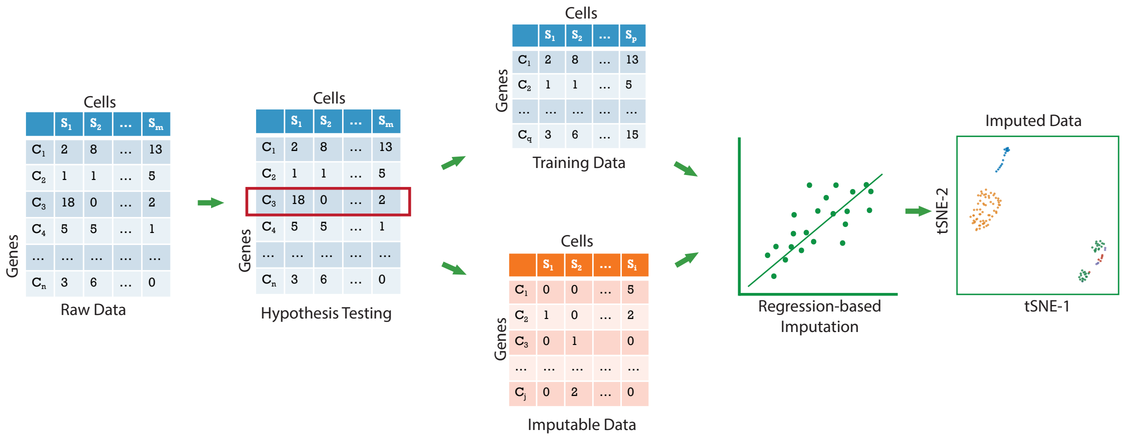

Figure 3.1: The overall pipeline of RIA. The algorithm consists of two modules. In the first module, we apply a hypothesis testing approach to determine which genes need to be imputed and which genes can be used as training. In the second module, we adopt the generalized linear model to impute the missing values from the imputable set. The algorithm outputs the imputed matrix that has the same number of rows and columns as of the input data.

3.1.2 Regression-based imputation

Based on the hypothesis testing described above, we divide the data into two groups of genes: i) a group G in which all of the genes are likely to be affected by dropouts (imputable set), and ii) a group of genes M that have accurate gene expression that do not need imputation (training set). The linear regression process consists of two steps. The first step is to select genes from the training set that are highly correlated with the gene we need to impute. In the second step, we train the linear model using these highly-correlated genes and then estimate the missing values.

For a gene $g \in G$ (imputable set), let us denote y as the non-zero part of g. In the first step we calculate the Pearson correlation coefficient of y with the corresponding values of every gene in M (training set). We then select 10 genes from M with the highest correlation coefficients. Denoting $\{m_{i_1}, \ldots, m_{i_{10}}\}$ as the selected genes in M, we have $\{x_{i_1}, \ldots, x_{i_{10}}\}$ as the vectors obtained from $\{m_{i_1}, \ldots, m_{i_{10}}\}$ that are highly correlated with y. Note that each vector x_{i_j} is a part of m_{i_j}. We train the generalized linear model in which $\{x_{i_1}, \ldots, x_{i_{10}}\}$ are the predictor variables and y is the outcome variable. In our implementation, we adopt the *lm* function that is available in the *stats* package. Next, we use the trained linear model to estimate the missing values in g, using $\{m_{i_1} \backslash x_{i_1}, \ldots, m_{i_{10}} \backslash x_{i_{10}}\}$ as the predictors, where $m_{i_j} \backslash x_{i_j}$ is that part of m_{i_j} that do not belong to x_{i_j}.

3.2 Hypergeometric testing and sub-space regression-based approach

Here we propose a new approach, scISR, that can reliably impute missing values from single-cell data. Our method consists of three modules. The first module performs hypothesis testing to identify the values that are likely to be impacted by the dropout events. By not altering the true zero values, we can avoid false imputations. The second module utilizes a data perturbation technique [81] to automatically group genes with similar patterns into smaller groups. The third module imputes missing values affected by dropout events (identified in the first module) by learning the gene patterns in each gene group (identified in the second module).

The schematic pipeline of scISR is shown in Figure 3.2. The input is an expression matrix, in which rows represent genes/transcripts and columns represent cells/samples (Figure 3.2A). The method consists of three modules. In the first module, we focus on identifying entries that are likely to be induced by dropouts (Figure 3.2B). For this purpose, we perform a hypergeometric test on each zero-valued entry using the expression values in the corresponding gene-cell pair. An entry is imputable only if the p-value obtained from the test is significant. We then divide the data into two sets of data: (i) training data in which all values are trustworthy, i.e., no entry needs to be imputed (Figure 3.2C), and (ii) imputable data in which each gene has at least one entry that needs to be imputed (Figure 3.2D). In the second module, we aim at identifying similar gene groups (gene subspaces) in the training data that share similar expression patterns (Figure 3.2E). For this purpose, we utilize the perturbation clustering we recently developed [81, 82]. Finally, in the third module, we estimate the missing values in the imputable data using the identified gene subspaces (Figure 3.2F). The method then merges the two matrices (training data and imputed data) and outputs a single matrix (Figure 3.2G).

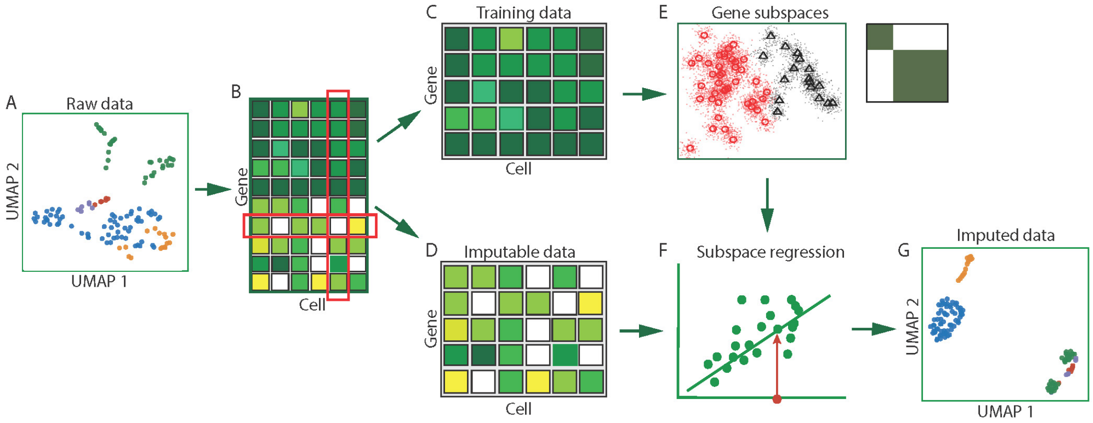

Figure 3.2: Single-cell Imputation using Subspace Regression (scISR). (A) Input data visualized in cell/sample space. (B) Hypergeometric test to determine whether each zero value is induced by dropout. Based on the computed p-values for each entry, we separate the original data into two sets of data: training data and imputable data. (C) Training data in which none of the values is induced by dropout events. (D) Imputable data in which each gene has at least one entry that is likely to be induced by dropout events. (E) Gene subspaces determined by perturbation clustering. We perturb the training data to discover the natural structure of the genes. Based on the pair-wise similarity between genes, we separate genes into groups that share similar patterns. (F) Subspace regression. We assign each gene in the imputable data to the closest subspace and then perform a generalized linear regression on the subspace to estimate the zero-valued entries that are impacted by dropouts. (G) Output expression matrix obtained by concatenating the training data and imputed data.

3.2.1 Hyper-geometric testing (Module 1)

This section describes the first module in scISR which aims at determining whether each zero value observed is the result of dropouts. Our hypothesis is that dropout events happen randomly for a gene affected by this phenomenon. By treating each cell as an instance of the population, we also assume that the ratio of zero values (dropout probability) reported for each cell differ from each other. Using dropout probabilities from both genes and cells, we can calculate how likely each zero values is affected by dropout. If zero values caused by dropout are over-represented in a gene, we conclude that this gene is affected by dropout events.

Given a zero-valued entry, let us denote p_1 and p_2 as the probability of observing a zero value in the corresponding gene and cell, respectively. It follows that the chances of having zero values in a gene and in a cell follow binomial distributions denoted by $X \sim$ *Bin(n,* p_1*)* and $Y \sim$ *Bin(m,* p_2*)*, respectively. n is the number of measured values for a gene, and m is the number of measured values for a cell. Under the null, we have $p = p_1 = p_2$. If X and Y are independent, we have $X + Y \sim$ *Bin(n+m, p)*. Therefore, the conditional distribution of X, $P(X = x|X + Y = r)$, is a hyper-geometric where x is the number of observed zero values in the gene and r is the total number of observed zero values in the selected pair of gene and cell. The probability mass function of the hyper-geometric distribution can be written as follows:

$$P(X = x-1|X+Y = r-1) = \frac{\binom{n-1}{x-1}\binom{m}{r-x}}{\binom{n+m-1}{r-1}} \tag{3.1}$$

Note that X and Y have an overlapping entry for each gene and cell pair. Therefore, we remove the overlapping entry from the hypergeometic formula by using: i) $n+m-1$ (instead of $n+m$) as the total number of of observed values in the selected pair of gene and cell, ii) $n-1$ (instead of n) as the number of measured values for the gene, and iii) $x-1$ (instead of x) as the number of zero values observed in the gene.

Applying Equation (3.1), we calculate the p-value for every zero-valued. We perform two different kinds of tests: an under-representation and over-representation analysis with a significance threshold set to 0.01 for both analyses. An entry with a significant p-value in the over-representation analysis is considered untrustworthy and should be imputed (imputable). An entry with a significant p-value in the under-representation analysis is considered trustworthy. An entry that is neither trustworthy nor untrustworthy should be left alone. These values will not be imputed, nor be used to impute other values. A gene is trustworthy if all of its entries are trustworthy. A gene is imputable when at least one of its values is imputable. Based on this hypothesis testing procedure, we obtain a set of genes that can be used for training (training data), and a set of genes that needed to be imputed (imputable data). See Supplementary Section 4.2, Figures S19, S21, and S24 for discussion about the robustness of scISR.

3.2.2 Identifying gene subspaces (Module 2)

It is crucial that the missing values of a gene are inferred using related genes that share similar expression patterns. Therefore, this module aims at identifying gene groups of the training data, i.e., gene subspaces that share similar patterns. For this purpose, we utilize the perturbation clustering [81, 82] that we recently developed. The method is based on the observation that small changes in quantitative assays will be inherently presented even when there is no significant difference between genes. If distinct gene groups do exist, they must be stable with respect to small degrees of data perturbation. This is indeed the case, as we have demonstrated in our previous work that the pair-wise connectivity between data points of the same group is preserved when the data are perturbed.

We will describe this approach using an illustrative example shown in Figure 3.3. In this simulated dataset, we have three distinct classes of genes in which the expressions of genes in each class are generated using a standard normal distribution. This distribution for the first class is $\mathcal{N}(0,1)$, for the second class is $\mathcal{N}(1,1)$ to simulate up-regulated genes, and for the third class is $\mathcal{N}(-1,1)$ to simulate down-regulated genes.

Assuming that we do not know the number of classes in this dataset, we set $k = 2$ (number of clusters) and then partition the genes. The upper panel in Figure 3.3B shows the connectivity between genes after clustering: green when they belong to the same cluster, and white otherwise. Note that two of the three true classes are wrongfully grouped together due to the wrong number of clusters. Now we repeatedly perturb the molecular measurements (by adding Gaussian noise) and partition the genes again (still with $k = 2$). The lower panel in Figure 3.3B shows the average connectivity between genes when the data is perturbed. The perturbed connectivity matrix suggests that the larger cluster is not stable. Similarly, the discordant connectivity in Figure 3.3C states that the partitioning using $k = 5$ is not correct either. The perturbed connectivity matrices (Figure 3.3B, C) suggest that there are three distinct classes of genes. Finally, when we set $k = 3$, the perturbed and original connectivity matrices are identical (Figure 3.3D).

The perturbed connectivity matrices suggest that there are three distinct classes of genes. This demonstrates that for truly distinct gene groups the true connectivity between genes within each class is recovered when the data is perturbed, no matter how we set the value of k. This resilience of pair-wise connectivity occurs consistently regardless of the clustering algorithm being used (e.g., k-means, hierarchical clustering, or partitioning around medoids), or the distribution of the data. When there are no truly distinct subgroups, the connectivity is randomly distributed. When the

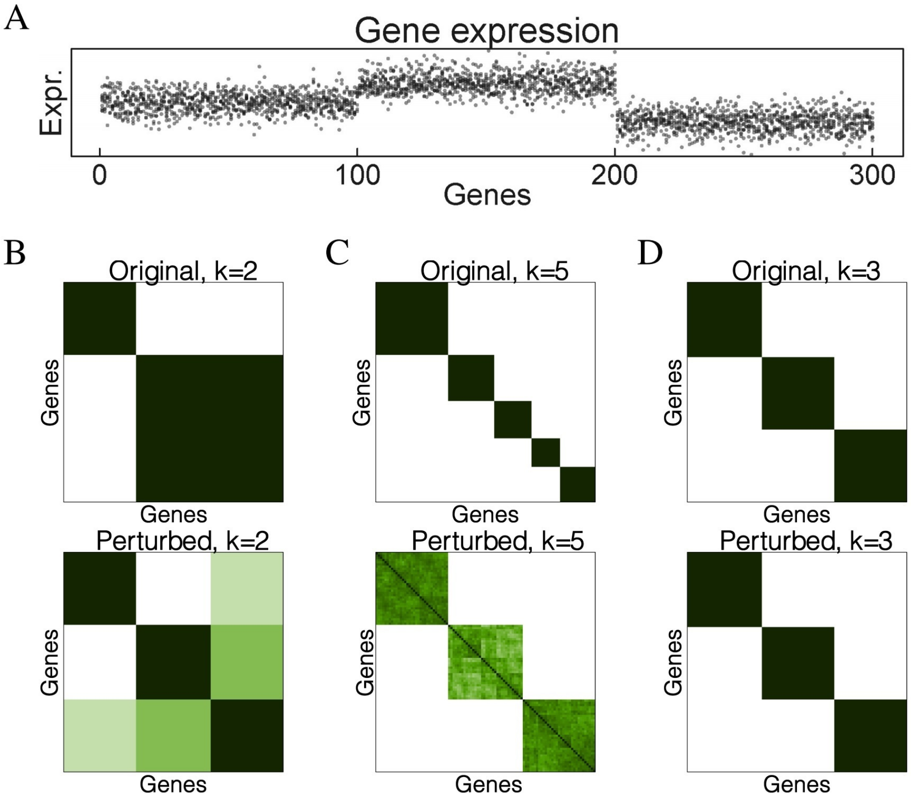

Figure 3.3: The resilience of pair-wise connectivity. (A) The dataset consists of three classes of genes: the first class has expression values of $\mathcal{N}(0,1)$, the second has expression values of $\mathcal{N}(1,1)$, and the third class has expression values of $\mathcal{N}(-1,1)$. (B) The original connectivity matrix (upper panel) and perturbed connectivity matrix (lower panel) for $k = 2$. (C) The connectivity matrices for $k = 5$. (D) The connectivity matrices for $k = 3$. The perturbed connectivity matrices clearly reveal the true structure of the data.

number of true classes changes, the perturbed connectivity always reflects the true structure of the data.

To identify the optimal partitioning, we calculate the absolute difference between the original and the perturbed connectivity matrices and compute the empirical cumulative distribution functions of the entries of the difference matrix (CDF-DM). In the ideal case of perfectly stable clusters, the original and perturbed connectivity matrices are identical, yielding a difference matrix of 0s, a CDF-DM that jumps from 0 to 1 at the origin, and an area under the curve (AUC) of 1 [81, 82]. We choose the partitioning with the highest AUC and then partition the genes into subgroups that are strongly connected in those perturbation scenarios. We note that the idea of determining subspaces can be realized for both genes and cells simultaneously. We do not focus on such simultaneous clustering in this manuscript, but it is of great interest.

3.2.3 Subspace regression (Module 3)

In the first module, we divide the genes into two sets: i) a set I in which all of the genes are likely to be affected by dropouts (imputable set), and ii) a set T that have accurate gene expression that does not need to impute (training set). In the second module, we segregate T into smaller groups of genes (gene subspaces) that share similar expression patterns. In this third module, we will impute dropout values in group I using a generalized linear regression model on gene subspaces.

Given a gene in the imputable set $g \in I$, we calculate the Euclidean distance between the gene to the centroid of each gene subspaces. Based on the calculated distances, we assign the gene to the closest subspace (with the smallest Euclidean distance). In order to impute dropout values in g, we train a generalized linear model using only highly-correlated genes within the assigned subspace in T. The linear regression process consists of two steps. The first step is to select genes from the training set that are highly correlated with the gene we need to impute. In the second step, we train the linear model using these highly correlated genes and then estimate the missing values [83].

Denoting $y \subset g$ as the non-zero part of g, S as the gene subspace in T that g was assigned to, $\{s_i \in S\}$ are expression vectors of genes in S; and $\{x_i \subset t_i\}$ are the parts of $\{t_i\}$ that correspond with y. We calculate the Pearson correlation between y and x_i and then select the 10 genes $\{t_1, \ldots, t_{10}\}$ in T with the highest correlation coefficients (see Supplementary Figure S5 for the discussion with regard to this parameter). We train a linear model in which $\{x_1, \ldots, x_{10}\}$ are the predictor variables and y is the outcome variable. In our implementation, we adopt the *lm* function that is available in the *stats* R package. Next, we use the trained linear model to estimate the missing values in $g \setminus y$, using $\{t_1 \backslash x_1, \ldots, t_{10} \backslash x_{10}\}$ as the predictors, where $t_i \backslash x_i$ is the part of t_i that does not belong to x_i. To avoid adding excessive weight to genes with high expression values, we always rescale the data to an acceptable range (default is [0,100]) using log transformation (base 2).

3.3 Neural networks and regression-based imputation approach

Here we propose a new approach, scIDS, that can reliably impute missing values from single-cell data. Our method consists of two modules. The first module performs data compression and clustering using deep neural networks. This compressed data is considered trustworthy information for imputation. The second module utilizes a z-test to detect genes that are highly impacted by dropouts. Then, the module imputes missing values affected by dropout events by learning the important features patterns in each cell group (identified in the first module). This strategy ensures that the true missing values are imputed by using only highly relevant information. In an extensive analysis using simulation and 8 real scRNA-seq datasets, we demonstrate that scISR improves the quality of single-cell data while preserving the transcriptome landscape.

Figure 3.4 shows the overall analysis pipeline of scIDS. The input of scIDS is an expression matrix in which rows represent genes and columns represent cells. The first module (Figure 3.4A) filters the genes and compresses the input data into a low-dimensional representation using two autoencoders. Given the compressed data, this module segregates the cells that share similar characteristics into different groups. The second module (Figure 3.4B) performs a z-score parametric measure to identify which genes need to be imputed. For each group of cells identified from the first module, a generalized linear model will learn from the compressed data to impute the missing data in the high dropout genes set.

3.3.1 Compressing data using autoencoders

The input of Module 1 is an already-normalized expression matrix in which rows represent cells while columns represent genes. Given the input matrix, we rescale the data to a range of 0 to 1 as follows:

$$X_{ij} = \frac{M_{ij} - min(M_{i.})}{max(M_{i.}) - min(M_{i.})} \tag{3.2}$$

where M is the input matrix and X is the normalized matrix.

After the rescaling, we further process the data using an 1-layer autoencoder to filter out genes that do not significantly contribute to differentiate cells. Autoencoder is a self-learning neural network that consists of two core components: an encoder and a

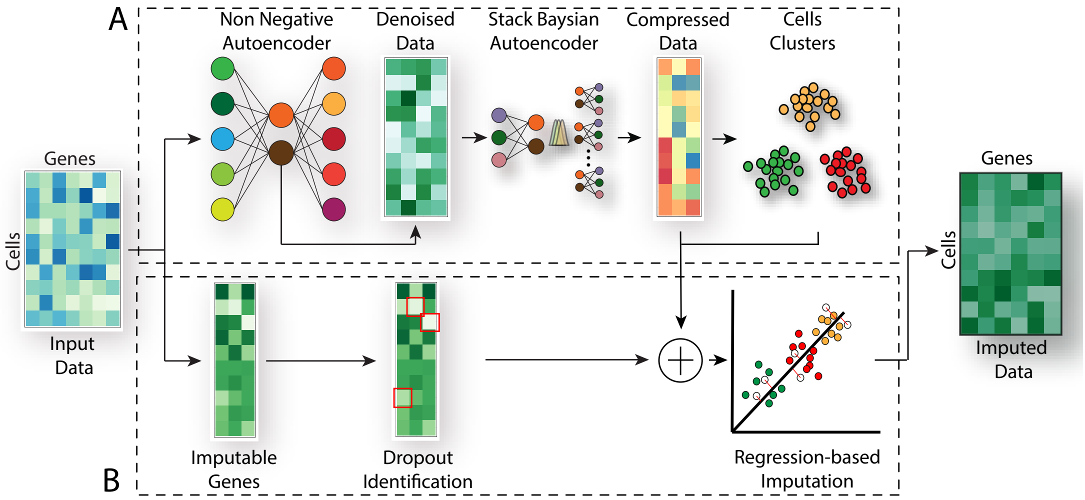
A
Non Negative Autoencoder
Denoised Data
Stack Baysian Autoencoder
Compressed Data
Cells Clusters
Genes
Cells
Input Data
Genes
Cells
Imputed Data
Imputable Genes
Dropout Identification
Regression-based Imputation
B

Figure 3.4: The overall analysis pipeline of scIDS. The input is a matrix in which rows represent cells and columns represent genes. In the first module (A), we perform features selection, data embedding, and cells clustering using two autoencoders. In the second module (B), we apply the z-test hypothesis testing to determine which genes need to be imputed. From the obtained genes set, we segregate cells into different groups using cluster assignment obtained from module A. For each group of cells, we adopt the generalized linear model to estimate the missing values using embedding data in module A. and we perform. The algorithm outputs the imputed matrix that has the same number of rows and columns as of the input data.

decoder. The encoder projects the input onto a lower-dimensional space (compressed data) while the decoder tries to reconstruct the original data from the compressed data. We also constrain the weights of the encoder to be non-negative. The non-negativity constraint ensures that each latent variable in the compressed space is a part-based, additive combination of the input. This technique shrinks the coefficients of less important features to zero while maintaining the non-negative coefficients of the significant features.

After the feature selection step, we obtain a denoised data matrix with the same number of cells that consists of important genes. Here, we further reduce the size of the data by conducting an additional step of dimensional reduction using a modified version of Variational Autoencoder (VAE) [84]. The VAE has the same structure as a standard autoencoder, which consists of an encoder and a decoder. The encoder (f_E) projects the input to a low-dimensional space while the decoder (f_D) reconstructs the original input from the compressed data. Given an expression profile of a cell x, we have $e = f_E(x)$, where e is the low-dimensional representation of x in the bottleneck layer. Instead of using e directly to reconstruct the data, VAE adds two transformations f_μ and f_σ to generate the parameters μ and σ. The new vector z is now sampled from the distribution $N(\mu, \sigma^2)$. The decoder uses z to reconstruct the data: $\bar{x} = f_D(z)$. Adding randomness to z will help the VAE model to avoid overfitting without losing the ability to learn a generalized representation of the input.

We call the second autoencoder a Stacked Variational Autoencoder because we modify the VAE model to generate multiple compressed spaces. Given a list of latent variables, we use a re-parameterization trick [84] to obtain multiple realizations of z as follows: $z = \mu + \sigma * N(0, 1)$. Given the list of latent variables, we use Weighted-based meta-clustering (wMetaC) to generate cells clusters and select the best latent variable as a compressed data M to be used for imputation.

3.3.2 Identifying dropouts and imputation

In this section, we aim to determine the set of genes that are likely to be impacted by dropouts. This is an important step to ensure that the missing data is correctly imputed without introducing false signals to the original data.

Our approach is based on the observation that for genes that are not impacted by dropouts, the log-transformed expression values are normally distributed [18]. Therefore, we use z-test to determine whether a zero value is observed by chance or by the impact of dropout events. For each gene g, we use the non-zero expression values to determine the parameters μ and σ of the Gaussian distribution. Next, we use z-test to estimate how likely a zero value occurs, given that the expression values follow the estimated Gaussian distribution. If the chance of observing a zero value is less than the significance threshold (0.05), we conclude that gene g is likely to be affected by dropout. By repeating this process for all genes, we can select a group of genes that are being affected by dropout and we call them as imputable set G.

After conducting the z-test, we obtain a new matrix with the same number of cells (rows), but the columns consist of genes that are highly impacted by dropout. Here, we perform imputation on imputable genes using the shared information within each cell group identified from the first module. For a gene $g_i \in G_i$ (imputable set) that belongs to the cell cluster i, let us denote y_i as the non-zero part of g_i. In the first step we calculate the Pearson correlation coefficient of y_i with the corresponding features in the compressed data M_i. We then select 5 features from M_i with the highest correlation coefficients. Denoting $\{m_{ij_1}, \ldots, m_{ij_5}\}$ as the selected features in M_i, we have $\{x_{ij_1}, \ldots, x_{ij_5}\}$ as the vectors obtained from $\{m_{ij_1}, \ldots, m_{ij_5}\}$ that are highly correlated with y_i. Note that each vector x_{ij_n} is a part of m_{ij_n}. We train the generalized linear model in which $\{x_{ij_1}, \ldots, x_{ij_5}\}$ are the predictor variables and y_i is the outcome variable. In our implementation, we adopt the *lm* function that is available in the *stats* package. Next, we use the trained linear model to estimate the missing values in g_i, using $\{m_{ij_1} \backslash x_{ij_1}, \ldots, m_{ij_5} \backslash x_{ij_5}\}$ as the predictors, where $m_{ij_n} \backslash x_{ij_n}$ is that part of m_{ij_n} that do not belong to x_{ij_n}.

We repeat this imputation process for all genes in each cells cluster generated by the first module. Given the already imputed genes, we merge them by the cells groups to obtain a new matrix that has the same size of the imputable set. Finally, we concatenate the set of good genes with the imputable set to obtain the final imputed data.

3.4 scRNA-seq data clustering using autoencoder and network fusion

Here we introduce scCAN, a single-cell clustering approach that consists of three modules: (1) a non-negative kernel autoencoder to filter out uninformative features, (2) a stacked, variational autoencoder to generate multiple low-dimensional representations of single-cell data, and finally (3) a graph-based technique to determine cell groups from multiple representations. In an extensive analysis using 28 scRNA-seq datasets, we demonstrate that scCAN significantly outperforms state-of-the-art methods in separating cells of different types. We further assess the clustering methods with regards to scalability and robustness against dropouts using simulated datasets. Overall, scCAN is the most robust and accurate method and can analyze most datasets in minutes.

The workflow of scCAN is shown in Figure 3.5. This workflow consists of three modules. The first module (Figure 3.5A) filters the genes and compresses the input data into a low-dimensional space using two autoencoders. Given the compressed data from module 1, the second module (Figure 3.5B) is used to cluster small data,

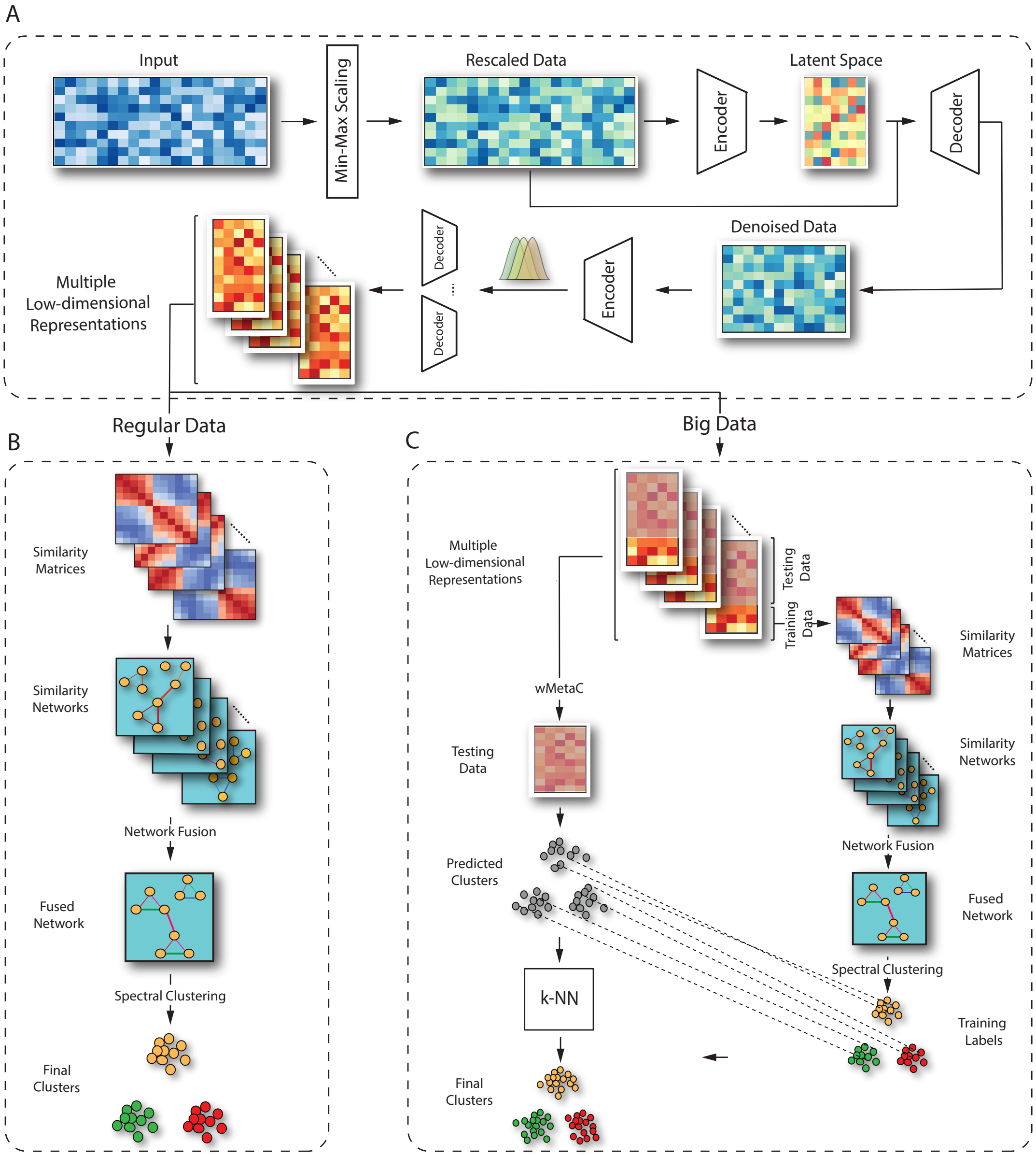
A
Input
Min-Max Scaling
Rescaled Data
Encoder
Latent Space
Decoder
Multiple
Low-dimensional
Representations
Decoder
Decoder
Encoder
Denoised Data
Regular Data
Big Data
B
Similarity
Matrices
Similarity
Networks
Network Fusion
Fused
Network
Spectral Clustering
Final
Clusters
C
Multiple
Low-dimensional
Representations
Testing
Data
Training
Data
Similarity
Matrices
wMetaC
Testing
Data
Similarity
Networks
Network Fusion
Predicted
Clusters
Fused
Network
Spectral Clustering
k-NN
Training
Labels
Final
Clusters

Figure 3.5: The overall analysis pipeline of scCAN consists of three modules. In the first module (A), we perform data normalization, gene filtering, and latent variables generation using two autoencoders. In the second module (B), we adopt the network fusion-based clustering method to segregate cell types for small data. The third module (C) aims at clustering big data using a combination of the network fusion approach and K nearest neighbors (k-NN) algorithm.

and the third module (Figure 3.5C) is used to cluster big data.

3.4.1 Data compression using autoencoders (Module 1)

Module 1 aims at compressing the original data into a compact representation. This module consists of three main steps: (1) data rescaling, (2) feature selection, and (3) multiple latent variables generation. The first step rescales the data, while the second step removes genes that are not informative. The third step transforms the data obtained from step 2 into a low-dimensional space using a stacked Bayesian autoencoder. The details of each step are presented in the following sections.

Min-max scaling

The input of Module 1 is an already-normalized expression matrix in which rows represent cells while columns represent genes. Given the input matrix, we rescale the data to a range of 0 to 1 as follows:

$$X_{ij} = \frac{M_{ij} - min(M_{i.})}{max(M_{i.}) - min(M_{i.})} \tag{3.3}$$

where M is the input matrix and X is the normalized matrix. Note that this min-max scaling is not a scRNA-seq normalization method. This min-max scaling added to our method is used on top of the already normalized data provided by users. Such scaling is frequently used in deep learning models [85–88] with the common purpose of reducing standard deviation and suppressing the effect of outliers without altering the transcriptome landscape.

Feature selection using non-negative-kernel autoencoder

After the rescaling, we further process the data using an 1-layer autoencoder to filter out genes that do not significantly contribute to differentiating cells. Autoencoder is a self-learning neural network that consists of two core components: an encoder and a decoder. The encoder projects the input onto a lower-dimensional space (compressed data) while the decoder tries to reconstruct the original data from the compressed data. Optimizing this process can theoretically result in a compact representation of the original data. By default, we set the dimension of the compressed data (bottleneck layer) to 50. The low number of dimensions ensures that the data obtained from the bottleneck layer is a compact representation of the original input, high-dimensional data.

We also constrain the weights of the encoder to be non-negative, so that each latent variable in the compressed space is a part-based, additive combination of the input. This technique shrinks the coefficients of less important features to zero while maintaining the non-negative coefficients of the significant features. From the weight distribution of the encoder, scCAN only keeps genes that have non-zero coefficients in the part-based representation. In essence, this set of genes can be considered the *optimal set* (sufficient and necessary) to represent the original data. This set is “necessary” because removing any gene from this set would greatly damage the reconstruction ability of the decoder. Concurrently, the set is “sufficient” because adding any other genes would not improve the quality of the compressed data. By default, scCAN selects the top 5,000 genes that have non-zero coefficients with the highest coefficient variances.

After this feature selection step, we obtain a new matrix with the same number of cells (rows), but the columns consist of only the optimal set of genes. This matrix serves as the input of another autoencoder to generate multiple low-dimensional representations of the data.

Dimensionality reduction using Stacked Variational Autoencoder

After the feature selection step, we obtain a denoised data matrix that consists of important genes. Although a significant number of genes have been removed, there are still thousands of genes. To reduce the computational resources required for clustering, we further reduce the size of the data by conducting an additional step of dimensional reduction using a modified version of Variational Autoencoder (VAE) [84]. We call it Stacked Variational Autoencoder because we generate multiple latent spaces instead of generating only one as in the original VAE.

The VAE has the same structure as a standard autoencoder, which consists of an encoder and a decoder. The encoder (f_E) projects the input to a low-dimensional space while the decoder (f_D) reconstructs the original input from the compressed data. Given an expression profile of a cell x, we have $e = f_E(x)$, where e is the low-dimensional representation of x in the bottleneck layer. Instead of using e directly to reconstruct the data, VAE adds two transformations f_μ and f_σ to generate the parameters μ and σ. The new vector z is now sampled from the distribution $N(\mu, \sigma^2)$. The decoder uses z to reconstruct the data: $\bar{x} = f_D(z)$. Adding randomness to z will help the VAE model to avoid overfitting without losing the ability of learning a generalized representation of the input.

Here we modify the VAE model to generate multiple compressed spaces with multiple realizations of z. The goal is to further diminish overfitting and to increase the robustness of the model. Given a list of latent variables, we use a re-parameterization trick [84] to obtain multiple realizations of z as follows: $z = \mu + \sigma * N(0, 1)$. This strategy ensures the VAE model can be back-propagated. In our model, we limit the size of the latent layer to a low number of dimensions ($d = 15$ by default). We keep d small to force the neural network to be as compressed as possible.

After finishing the training stage, the input data is processed through the encoder to generate multiple representative latent variables of the original data. As described in the next section, these compressed representations of the data are used for cell segregation (clustering).

3.4.2 Network fusion and spectral clustering for cell segregation (Module 2)

This section describes the workflow for analyzing datasets with a moderate number of cells ($n <= 5,000$ by default). When the number of samples is large (over 5,000 up to millions of cells), we use a different procedure (see Module 3 in Section 3.4.3).

The input of Module 2 is multiple low-dimensional representations (matrices) of the input data. We use a network fusion-based approach to cluster scRNA-seq data via multiple steps: (i) building a cell-similarity network for each of the representations, (ii) fusing the networks, and (iii) clustering using spectral clustering.

For each latent matrix, we construct a cell-similarity network $G = (V, E)$ where each vertex V corresponds to a cell and each edge E represents a link between two cells. Edges are weighted and stored in a $m \times m$ matrix W with W_{ij} represents the weight between cells x_i and x_j. To determine the weight for each pair of cells, we first compute the Euclidean distance ρ_{ij} between the cells x_i and x_j. Next, we calculate the average value of the distances between the cell x_i and its neighbors $\rho_{i\_} = \frac{\sum_{j=1 \ldots k}(\rho(x_i, n_j))}{k}$. We repeat this step for the cell x_j to obtain $\rho_{j\_}$. We keep the number of neighbors small ($k = 30$ by default) to preserve local cells relationship, but users are free to set their own values. We denote $\varepsilon_{ij} = \frac{\rho_{ij} + \rho_{i\_} + \rho_{j\_}}{3}$ as an average distance among cells x_i, x_j and neighbour cells to calculate $W_{ij} = exp\left(-\frac{\rho^2(x_i x_j)}{\mu \varepsilon_{ij}}\right)$ where μ is a Gaussian similarity kernel ($\sigma = 0.5$). Finally, we repeat this process for every pair of cells to obtain the similarity matrix W for the current latent matrix to obtain a similarity network. Here, each network is a graph representation of a single latent matrix.

Next, we perform network fusion to aggregate multiple similarity networks obtained from their corresponding latent matrices into a consolidated one. The network fusion approach is adapted from SNF method [89] by first calculating the full and sparse kernel for each vertex V in the network G. The full kernel is the normalized weight matrix P obtained from G. The sparse kernel S is a matrix that contains local affinity of cells and their neighbors. This step maintains the weights for cells in the same group while suppressing the weights of non-neighbouring cells to zero. That means cell similarities in the same community are more trustworthy than the remote ones. We repeatedly calculate the full and sparse kernel for all n similarity networks to get the lists of updated weight matrices and encoded neighbour similarity matrices. Then, those matrices are iteratively fused together to obtain the final fused network P as follows:

$$P^{(v)} = S^{(v)} \times \left(\frac{\sum_{k \neq v} P^{(k)}}{n-1} \right) \times (S^{(v)})^T, v = 1, 2, ..., n \tag{3.4}$$

Given the fused network P, we use the eigengap method [90] to determine the number of clusters. First, we compute adjacency matrix A and degree matrix D to get Laplacian matrix $L = D - A$. Here, eigen values (λ) and eigen vectors (x) are calculated by $Lx = \lambda x$. Next, eigengap is defined as $eigengap_i = \lambda_{i+1} - \lambda_i$ where λ_i is the i-th eigenvalue of the matrix L. In our method, i is user-control hyperparameter that is set from 2 to 15 by default. From the list of eigengap values, we sort them in ascending order and select the two highest eigengap values. Among those two, we select the eigengap that yields a minimum i to prevent overestimating the number of clusters. This i value is considered as the optimal number of clusters. Given the number of clusters, we use spectral clustering [91] to partition the cells in the fused network P.

3.4.3 Big data analysis (Module 3)

When the number of cells is large ($n > 5,000$), we split the cells into two sets: a training set of 5,000 randomly selected cells and a testing set that consists of the remaining cells. We then use the same procedure presented in Module 2 to cluster the *training data*. After this step, we obtain a *training cluster assignment*. We annotate the remaining cells in each latent matrix as *testing data*, and we aim to classify them using the cells labels obtained from the *training data*.

We perform the classification process on *testing data* in only one latent matrix among multiple ones obtained from Module 1. In order to do that, we select the best latent matrix that is a closed representation of other matrices. First, we use k-nearest neighbor adaption of spectral clustering algorithm (k-nn SC) to quickly get the clusters assignments for every latent matrices. Given the list of obtained clusters, we use weighted-based meta-clustering (wMetaC) implemented in SHARP [55] to determine the final cluster assignment. The wMetaC algorithm is conducted through 5 steps: (i) calculating cell-cell weighted similarity matrix W, $w_{ij} = s_{ij}(1 - s_{ij})$ where s_{ij} is the chance that cell i and j are in the same cluster, (ii) calculating cell weight, which is the sum of all cell-cell weights related to this cell, (iii) generating cluster-cluster similarity matrix $|C| \times |C|$, where C is the union of all the clusters obtained in each replicate, (iv) performing hierarchical clustering on cluster-cluster similarity matrix, and (v) determining final clustering result by voting scheme. One note of caution is that the final clustering results obtained from this step are only used to determine the best latent matrix. Then, we measure the adjusted Rand index (ARI) value between the final cluster and the cluster obtained from k-nn SC on each input latent. The latent matrix that yields the highest ARI value will be selected for classification.

Given the final latent matrix, we use k-NN algorithm to classify the remaining cells using cluster's labels obtained from the *training data*. Lastly, we merge the cluster assignments from the *training data* and the *testing data* to get the final clustering result.

Note that the default value of 5,000 allows us to have a sufficiently large sample size to properly determine the cell types which in turns will lead to a proper classification of the remaining cells. At the same time, 5,000 is a reasonable small number of samples that allows users to perform the analysis efficiently using personal computers. However, this default value might hinder the process of detecting rare cell types in large datasets. To enhance the method's capability to detect rare cell types, users can either increase the sample size or perform multi-stage clustering.

3.5 Validation

3.5.1 scRNA-seq data imputation validation

Benchmarking and validation for RIA

We will assess the performance of RIA using public scRNA-seq datasets that are available in NIH Gene Expression Omnibus (GEO) [79] and Array Express [92]. We will use five datasets: Biase's [93], Yan's [94], Goolam's [95], Deng's [96], and Zeisel's [40]. The processed data were downloaded from Hemberg lab's website (https://hemberg-lab.github.io/scRNA.seq.datasets). The details for each dataset (accession ID, number of cells, number of cell types, organism, and single-cell protocol) are described in Table 3.1. The first four studies, Biase [93], Yan's [94], Goolam [95] and Deng [96], measure the gene expression of embryonic cells at different stages, from zygote to the cells of the late blastocyst. Cell types of these datasets were labeled according to their developmental stages (timestamp). The fifth dataset, Zeisel [40], was obtained from a mouse brain tissue. The cell labels of this dataset were assigned based on expert knowledge of the underlying biology.

For each dataset, we will download the already processed expression data, in which genes are represented in rows and cells are in different columns. We only perform log_2 transformation to re-scale sc-RNAseq data, i.e., $log_2(\mathbf{A}+1)$ where $\mathbf{A}$ is the expression matrix. Genes that do not express across any cells will be removed.

Also, to validate our proposed approaches, we will assess the performance of our methods in comparison with two state-of-the-art methods for single-cell imputation: MAGIC [11] and scImpute [18]. Both methods are widely used and each represents a different imputation strategy. MAGIC uses Markov affinity matrix to smooth the data while scImpute is a statistical approach that models the data as a mixture of Gamma and Gaussian distributions.

For each of the five datasets described in Table 3.1, the cell types are known. We use this information *a posteriori* to assess how separable the cell populations are after imputation. For each dataset, we have a raw matrix that serves as the input of each imputation method. After imputation, we have four matrices: the raw data and three imputed matrices (from RIA, MAGIC, and scImpute). In order to assess how separable the cell types in each matrix, we use k-means [97] to cluster each matrix and then compare the obtained partitionings with the known cell types. We will use different metrics for comparing the obtained partitionings with the known types.

Here we will also aim to show that RIA improves the quality of the data without altering the transcriptomics landscapes. Since single-cell data are high-dimensional and are hard to interpret, it is desirable to visualize them in low dimensional space with two or three dimensions. Traditionally, researchers use t-distributed Stochastic Neighbourhood Embedding (t-SNE) [98, 99] for this purpose, which preserve local structure among cells. We first use Principal Component Analysis (PCA) [100] to reduce the number of dimensions to 20, and then use t-distributed Stochastic Neighbourhood Embedding (t-SNE) [101] to visualize the data. The purpose of using PCA is to reduce the running time of the visualization process. Then, we will use an evaluation metric to quantify the difference between original 2-D transcriptomics landscape produced from raw data and each of the method.

Lastly, we will measure the ability to recovers temporal trajectories in embryonic developmental stages for each method. We use the four embryonic datasets to demonstrate RIA's ability in recovering the temporal dynamics. The Biase dataset consists of 49 inter-blastomere cells from mouse embryonic stem cells (mESCs), including *zygote, 2-cell and 4-cell.* The Goolam dataset includes transcriptome data of 124 individual cells in mouse pre-implantation development stages: *2-cell, 4-cell, 8-cell, 16-cell and blast.* The Yan dataset consists of 90 cells from human pre-implantation embryos and human embryonic stem cells (hESCs). The Deng dataset includes the expression profiles of 268 individual cells of mouse pre-implantation embryos of mixed background.

Benchmarking and validation for scIDS

Similar to RIA, we will continue to use public datasets available from GEO, Array Express, and Broad Institute Single Cell Portal (`https://singlecell.broadinstitute.org/single_cell`) for validation. For this method, we will include datasets that contain higher number of cells making the total number of cells analyzed greater than 100,000. Table 3.2 shows the details of the eight single-cell datasets (accession ID, number of cells, number of cell types, organism, and single-cell protocol) used in our data analysis.

We will compare scIDS with the raw data and two widely used scRNA-seq imputation methods, knn-smoothing [43], and MAGIC [11] using eight scRNA-seq datasets mentioned above. For each of the eight datasets, we use a raw matrix as the input of

Table 3.1: Description of the eight single-cell datasets used to assess the performance of RIA

Dataset	**Accession ID**	**Size**	**K**	**Organism**	**Protocol**
Biase[93]	GSE57249	49	4	Mouse Embryo	SMARTer
Yan[94]	GSE36552	90	6	Human Embryo	Tang
Goolam[95]	E-MTAB-3321	124	5	Mouse Embryo	Smart-Seq2
Deng[96]	GSE45719	268	6	Mouse Embryo	Smart-Seq2
Zeisel[40]	GSE60361	3,005	9	Mouse Brain	STRT-Seq

Table 3.2: Description of the eight single-cell datasets used to assess the performance of scIDS.

Dataset	**Accession ID**	**Size**	**K**	**Organism**	**Protocol**
Pollen [102]	SRP041736	301	4	Human Tissues	SMARTer
Darmanis [103]	GSE67835	466	9	Human Brain	SMARTer
Usoskin [104]	E-MTAB-3321	124	3	Mouse Brain	STRT-Seq
Kolodziejczyk [5]	E-MTAB-2600	268	3	Mouse Embryo	SMARTer
Klein [105]	GSE65525	3,005	4	Mouse Embryo	inDrop
Baron [37]	GSE84133	3,005	14	Human Pancreas	inDrop
Hrvatin [106]	GSE102827	48,266	8	Mouse Visual Cortex	inDrop
Cao [107]	SCP454	90,579	7	Sea Squirt Embryos	10x Genomics

each imputation method. After imputation, we obtain four matrices: the raw data and three imputed matrices (from knn-smoothing, MAGIC, and scIDS). In order to assess the segregation of the cell types in each matrix, we use k-means to cluster each matrix and then compare the obtained cluster assignments with the known cell types. We will also use different metrics for comparing the obtained partitionings with the known types.

We will also measure the capability of scIDS in correctly imputing missing values without making a change to the transcriptomics landscapes. Preferably, life scientists impute the data in order to improve the quality of downstream analyses. At the same time, imputation should not completely change the data because of falsely introduced signals, leading to wrong or compromised findings. Since single-cell data are high-dimensional, the common practice is to project the high-dimensional data into a low dimensional space with two or three dimensions. The visualization in 2-D or 3-D helps researchers to interpret the single-cell data more efficiently. To reduce the running time, we first use a fast partial singular value decomposition method [108] to quickly reduce the number of features to 20. Then, we use t-SNE [99], and UMAP [109] to project the compact data into two-dimensional space for visualization.

To quantify the similarity between the imputed and original landscapes, we calculate the distance correlation index ($dCor$) [110] for each imputed landscape generated by t-SNE and UMAP. Given X and Y as the 2D representation of the raw and imputed data, $dCor$ is calculated as $dCor = \frac{dCov(X,Y)}{\sqrt{dVar(X)dVar(Y)}}$ where $dCov(X,Y)$ is the distance covariance between X and Y while $dVar(X)$ and $dVar(Y)$ are distance variances of X and Y. The $dCor$ coefficient takes value between 0 and 1, with the $dCor$ is expected to be 1 for a perfect similarity. Unlike Pearson correlation, $dCor$ measures both the linear and nonlinear associations between X and Y [110]. Especially, $dCor$ remains constant when we rotate the transcriptome landscape.

Benchmarking and validation for scISR

To assess the performance of imputation methods, we downloaded 25 publicly available scRNA-seq datasets available on NCBI, ArrayExpress, and Broad Institute Single Cell Portal (`https://singlecell.broadinstitute.org/single_cell`). The description of the datasets is shown in Table 3.3. The processed data of the first 15 datasets are also available at the Hemberg Lab's website (`https://hemberg-lab.github.io/scRNA.seq.datasets`). There are 14 plate-based datasets and 11 droplet-based datasets. Among these, 12 datasets are with UMI, and 13 datasets are with read counts. There are 7 datasets without normalization while the remaining 18 datasets were already normalized by the data providers: 3 CPM-, 3 TPM-, 4 RPKM-, 4 FPKM-, and 4 RPM-normalized.

We analyzed the data with minimal additional pre-processing steps. For datasets with the range of values larger than 100, we rescale the data using log transformation (base 2). We also remove genes that do not contribute to the analysis, including: (i) genes expressed in less than two cells; and (ii) genes that have less than one percent of non-zero-valued entries. In all 25 single-cell datasets, the cell types are known. However, these cell labels are not provided to any of the imputation methods. They are only used *a posteriori* to assess the quality of the imputed data.

To present a comprehensive simulation analysis, we generate a total of 116 datasets in four different scenarios: (1) uniform dropout distribution, (2) normal dropout distribution, 3 highly correlated cell groups, and (4) Splatter-based simulation [124].

In the first scenario, we generate 6 datasets by varying the number of cells from 100 to 10,000 and the number of genes from 300 to 10,000. The cells/genes combination setups are presented as follows: 100×300, 1,000×3,000, 3,000×9,000, 5,000×10,000, 7,000×10,000, and 10,000×10,000.

In each of the 6 datasets, the expression values follow a normal distribution

Table 3.3: Description of the 25 single-cell datasets used to assess the performance of scISR against other methods. The first three columns describe the name, accession ID, and tissue, while the following seven columns show the sequencing protocol, cell isolation technique, quantification scheme, normalized unit, dropout rate, number of cell types, and number of cells.

Dataset	**Accession ID**	**Tissue**	**Sequencing Protocol**	**Cell Isolation**	**Quant. Scheme**	**Norm. Unit**	**Drop. Rate**	**Class**	**Size**
1. Fan [111]	GSE53386	Mouse Embryo	SUPeR-seq	Plate	Reads	FPKM	0.584	6	69
2. Treutlein [112]	GSE52583	Mouse Tissues	SMARTer	Plate	Reads	FPKM	0.902	5	80
3. Yan [94]	GSE36552	Human Embryo	Tang	Plate	Reads	RPKM	0.456	6	90
4. Goolam [95]	E-MTAB-3321	Mouse Embryo	Smart-Seq2	Plate	Reads	CPM	0.685	5	124
5. Deng [96]	GSE45719	Mouse Embryo	Smart-Seq	Plate	Reads	RPKM	0.605	6	268
6. Pollen [102]	SRP041736	Human Tissues	SMARTer	Plate	Reads	TPM	0.671	4	301
7. Darmanis [103]	GSE67835	Human Brain	SMARTer	Plate	Reads	CPM	0.808	9	466
8. Usoskin [104]	GSE59739	Mouse Brain	STRT-Seq	Plate	Reads	RPM	0.846	3	622
9. Camp [113]	GSE75140	Human Brain	SMARTer	Plate	Reads	FPKM	0.801	7	734
10. Klein [105]	GSE65525	Mouse Embryo	inDrop	Droplet	UMI	RPM	0.658	4	2,717
11. Romanov [114]	GSE74672	Human Brain	SMARTer	Plate	UMI	-	0.878	7	2,881
12. Segerstolpe [115]	E-MTAB-5061	Human Pancreas	Smart-Seq2	Plate	Reads	RPKM	0.823	15	3,514
13. Manno [39]	GSE76381	Human Brain	STRT-Seq	Plate	UMI	-	0.86	56	4,029
14. Marques [116]	GSE75330	Mouse Brain	Fluidigm C1	Plate	Reads	FPKM	0.891	13	5,053
15. Baron [37]	GSE84133	Human Pancreas	inDrop	Droplet	UMI	TPM	0.906	14	8,569
16. Sanderson [117]	SCP916	Mouse Tissues	10X Genomics	Droplet	Reads	-	0.764	11	12,648
17. Slyper	SCP345	Human Blood	10X Genomics	Droplet	UMI	-	0.956	8	13,316
18. Zilionis (Mouse) [118]	GSE127465	Mouse Lung	inDrop	Droplet	UMI	RPM	0.976	7	15,939
19. Tasic [119]	GSE115746	Mouse Visual Cortex	SMART-Seq	Plate	Reads	CPM	0.798	6	23,178
20. Zyl (Human) [120]	SCP780	Human Eye	inDrop	Droplet	UMI	-	0.913	19	24,023
21. Zilionis (Human) [118]	GSE127465	Human Lung	inDrop	Droplet	UMI	RPM	0.982	9	34,558
22. Wei [121]	SCP469	Human Synovium	10x Genomics	Droplet	UMI	TPM	0.915	9	41,565
23. Cao [107]	SCP454	Sea Squirt Embryos	10x Genomics	Droplet	UMI	-	0.821	7	90,579
24. Orozco [122]	GSE135133	Human Eye	10X Genomics	Droplet	UMI	RPKM	0.964	12	100,055
25. Darrah [123]	GSE139598	Human Blood	Drop-seq	Droplet	UMI	-	0.947	14	162,490

[1] UMI: Unique Molecular Identifier; CPM: Counts Per Million; RPM: Reads Per Million; RPKM: Reads Per Kilobase of transcript, per Million mapped reads; FPKM: Fragments Per Kilobase of transcript, per Million mapped reads.

$\mathcal{N}(\mu, \sigma)$. We set $\mu = 1$ and $\sigma = 0.15$. We slightly shift the mean of the cells and genes by adding a certain value to each group (-1, 0, 1, 1.5 for cell groups and -1, 0, 1 for gene groups) to create 4 different cell types and 3 gene groups – each cell type has an equal number of cells. We name this data as *complete data* and use the expression values as the ground truth for benchmarking. Next, we introduce the dropout events. We randomly select 40% of the genes and consider those as genes that are impacted by dropout events. We randomly assign 30% of the values of these genes to zero. We name this data as *masked data*.

3.5.2 scRNA-seq clustering analysis validation

Performance assessment

We will download 28 scRNA-seq datasets from public repositories to validate the clustering performance of scCAN. The Table 3.4 reports the Accession numbers, and Table 3.5 shows the specific link to each of the 28 datasets.

The datasets Guo, Kanton, Brann, and Miller were downloaded from the European Bioinformatics Institute (`https://www.ebi.ac.uk/gxa/sc/experiments/`). The datasets Slyper, Zilionis, Orozco, and Kozareva were downloaded from Broad Institute Single Cell Portal (`https://singlecell.broadinstitute.org/single_cell`). The datasets Montoro, Hrvatin, Darrah, and Cao were downloaded from NCBI [79]. The Brain 1.3M dataset was downloaded from the 10X Genomics website (`https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons`). The remaining 15 datasets were downloaded from Hemberg Group's website (`https://hemberg-lab.github.io/scRNA.seq.datasets`). In each dataset except Brain 1.3M, the cell populations labels stages are known. This information is only used *a posteriori* to assess the performance of each method in improving the identification of cell populations.

Table 3.4: Description of the 28 single-cell datasets used to assess the performance of scCAN. The first two columns describe the name and tissue while the next five columns show the number of cells, number of cell types, sequencing protocol, accession ID, and references. The first 27 datasets have true cell labels and can be used to assess the accuracy of the clustering methods.

Dataset	**Tissue**	**Size**	**Class**	**Protocol**	**Accession ID**	**Reference**
1. Pollen	Human Tissues	301	11	SMARTer	SRP041736	[102]
2. Patel	Human Tissues	430	5	Smart-Seq	GSE57872	[125]
3. Wang	Human Pancreas	457	7	SMARTer	GSE83139	[126]
4. Li	Human Tissues	561	9	SMARTer	GSE81861	[127]
5. Usoskin	Mouse Brain	622	4	STRT-Seq	GSE59739	[104]
6. Camp	Human Liver	777	7	SMARTer	GSE81252	[128]
7. Xin	Human Pancreas	1,600	8	SMARTer	GSE81608	[129]
8. Muraro	Human Pancreas	2,126	10	CEL-Seq2	GSE85241	[130]
9. Segerstolpe	Human Pancreas	2,209	14	Smart-Seq2	E-MTAB-5061	[115]
10. Romanov	Mouse Brain	2,881	7	SMARTer	GSE74672	[114]
11. Zeisel	Mouse Brain	3,005	9	STRT-Seq	GSE60361	[40]
12. Lake	Human Brain	3,042	16	Fluidigm C1	phs000833.v3.p1	[131]
13. Montoro	Human Pancreas	7,193	7	Smart-Seq2	GSE103354	[132]
14. Guo	Human Testis	7,416	7	10X Genomics	E-GEOD-134144	[133]
15. Baron	Human Pancreas	8,569	14	inDrop	GSE84133	[37]
16. Chen	Mouse Brain	12,089	46	Drop-seq	GSE87544	[38]
17. Slyper	Human Blood	13,316	8	10X Genomics	SCP345	
18. Kanton	Human Brain	17,542	14	Smart-Seq2	E-HCAD-5	[134]
19. Brann	Mouse Brain	26,766	46	10X Genomics	E-GEOD-151346	[135]
20. Zilionis	Human Lung	34,558	9	inDrop	GSE127465	[118]
21. Macosko	Mouse Retina	44,808	12	Drop-seq	GSE63473	[27]
22. Hrvatin	Mouse Visual Cortex	48,266	8	inDrop	GSE102827	[106]
23. Orozco	Human Eye	100,055	11	10X Genomics	GSE135133	[122]
24. Miller	Human Lung	142,523	11	10X Genomics	E-MTAB-8221	[136]
25. Darrah	Human Blood	162,490	14	Drop-seq	GSE139598	[123]
26. Kozareva	Mouse Cerebellum	611,034	18	10X Genomics	SCP795	[137]
27. Cao	Mouse Cerebellum	1,092,000	9	10X Genomics	GSE156793	[138]
28. Brain 1.3M	Mouse Brain	1,300,774	NA	10X Genomics	GSE93421	[139]

Table 3.5: Link to 28 single-cell datasets used to benchmark scCAN.

Dataset	Link	Reference
1. Pollen	https://hemberg-lab.github.io/scRNA.seq.datasets/human/tissues/#pollen	[102]
2. Patel	https://hemberg-lab.github.io/scRNA.seq.datasets/human/tissues/#patel	[125]
3. Wang	https://hemberg-lab.github.io/scRNA.seq.datasets/human/pancreas/#wang	[126]
4. Li	https://hemberg-lab.github.io/scRNA.seq.datasets/human/brain/#li	[127]
5. Usoskin	https://hemberg-lab.github.io/scRNA.seq.datasets/mouse/brain/#usoskin	[104]
6. Camp	https://hemberg-lab.github.io/scRNA.seq.datasets/human/liver/	[128]
7. Xin	https://hemberg-lab.github.io/scRNA.seq.datasets/human/pancreas/#xin	[129]
8. Muraro	https://hemberg-lab.github.io/scRNA.seq.datasets/human/pancreas/#muraro	[130]
9. Segerstolpe	https://hemberg-lab.github.io/scRNA.seq.datasets/human/pancreas/#segerstolpe	[115]
10. Romanov	https://hemberg-lab.github.io/scRNA.seq.datasets/mouse/brain/#romanov	[114]
11. Zeisel	https://hemberg-lab.github.io/scRNA.seq.datasets/mouse/brain/#zeisel	[40]
12. Lake	https://hemberg-lab.github.io/scRNA.seq.datasets/human/brain/#lake	[131]
13. Montoro	https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE103354	[132]
14. Guo	https://www.ebi.ac.uk/gxa/sc/experiments/E-GEOD-134144/	[133]
15. Baron	https://hemberg-lab.github.io/scRNA.seq.datasets/human/pancreas/	[37]
16. Chen	https://hemberg-lab.github.io/scRNA.seq.datasets/mouse/brain/#chen	[38]
17. Slyper	https://singlecell.broadinstitute.org/single_cell/study/SCP345/	
18. Kanton	https://www.ebi.ac.uk/gxa/sc/experiments/E-HCAD-5/	[134]
19. Brann	https://www.ebi.ac.uk/gxa/sc/experiments/E-GEOD-151346/	[135]
20. Zilionis	https://singlecell.broadinstitute.org/single_cell/study/SCP739/	[118]
21. Macosko	https://hemberg-lab.github.io/scRNA.seq.datasets/mouse/retina/	[27]
22. Hrvatin	https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE102827	[106]
23. Orozco	https://singlecell.broadinstitute.org/single_cell/study/SCP484/	[122]
24. Miller	https://www.ebi.ac.uk/gxa/sc/experiments/E-MTAB-8221/	[136]
25. Darrah	https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE139598	[123]
26. Kozareva	https://singlecell.broadinstitute.org/single_cell/study/SCP795/	[137]
27. Cao	https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE156793	[138]
28. Brain 1.3M	https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons	[139]

We compare scCAN with five state-of-the-art clustering methods that are widely used for single-cell analysis: CIDR [21], SEURAT3 [69], Monocle3 [71], SHARP [55], and SCANPY [70]. To compare scCAN with current methods, the following packages are used in the analysis: i) CIDR version 0.1.5 from GitHub (https://github.com/VCCRI/CIDR), ii) SEURAT3 version 3.2.3 from Github (https://github.com/satijalab/seurat/releases/tag/v3.2.3), iii) Monocle3 version 3.0 from Github (https://github.com/cole-trapnell-lab/monocle3), iv) SHARP version 1.1.0 from Github (https://github.com/shibiaowan/SHARP), and SCANPY version 1.4.4 from Anaconda. We carefully follow the instruction and tutorial provided by the authors of each package. We execute each method using default parameters suggested by the authors.

Validation metrics

We use three different metrics for comparing the obtained partitions with the known cell types: adjusted Rand index (ARI) [140], adjusted mutual information (AMI) [141], and V-measure [142]. To evaluate the capability of each method in predicting the true number of clusters, we use absolute log-modulus [143].

Rand index (RI) evaluates the similarity between predicted clusters and true cell types. Given P as a set of clusters and Q is a set of true cell types then RI is calculated as:

$$RI = \frac{t+u}{t+u+v+s} = \frac{t+u}{\binom{N}{2}} \tag{3.5}$$

where t is the number of pairs belonging to the same cell type in Q and are grouped together in the same cluster in P, u is the number of pairs of different cell types in Q and are grouped to different clusters in P, v is the number of pairs of the same cell types in Q and are grouped to different clusters in P, s is the number of pairs in different cell types in Q and are grouped together in the same cluster in P, N is the total number of cells, and $\binom{N}{2}$ is the number of possible pairs. In brief, RI measures the ratio of pairs that are clustered in the same way (either together or different) from two partitions (e.g. 0.80 means 80% of pairs are grouped in the same way). The Adjusted Rand Index (ARI) [140] is the corrected-for-chance version of the Rand Index. The ARI values ranged from -1 to 1 in which 0 indicates a random grouping. The ARI score is calculated as :

$$ARI = \frac{RI - exptected\_RI}{max(RI) - expected\_RI} \tag{3.6}$$

Adjusted mutual information (AMI) is an adjustment of the mutual information (MI) score to account for random partitioning. It accounts for the fact that the MI is generally higher for two clusters with a larger number of clusters, regardless of whether there is actually more information shared. The calculation of AMI is presented as follows:

Given a dataset of n cells with true partition $X = \{X_1, X_2, ..., X_R\}$ of R clusters and predicted partition $Y = \{Y_1, Y_2, ..., Y_C\}$ of C clusters. The mutual information of cluster overlap between X and Y can be summarized as a contingency table $M_{R\times C} = [n_{ij}]$, where $i = 1...R, j = 1...C$, and n_{ij} represents the number of common data point falls into cluster X_i is $p(i) = \frac{|x_i|}{n}$. The entropy associated with the clustering X is calculated as follows:

$$H(X) = \sum_{i=1}^{R} P(i) log P(i) \tag{3.7}$$

$H(X)$ gives outputs as non-negative values where 0 indicates that there is one cluster in the dataset. The mutual information (MI) between two clusters X and Y is calculated as follows:

$$MI(X,Y) = \sum_{i=1}^{R} \sum_{j=1}^{C} P(i,j) log \frac{P(i,j)}{P(i)P(j)} \frac{n_{ij}}{n} \tag{3.8}$$

where $P(i,j)$ is the cell that is classified to both clusters X_i in X and Y_j in Y. $P(i,j)$ is calculated as follows:

$$P(i,j) = \frac{|X_i \cap Y_j|}{n} \tag{3.9}$$

MI gives outputs as non-negative values bounded by the entropies $H(X)$ and $H(Y)$ and 0 indicates that there is no cell classified to the same cluster. To correct for the fact that two random clusterings do not give a constant value, and tends to be larger when the two partitions have a larger number of clusters. Therefore, AMI is defined as follows:

$$AMI(X,Y) = \frac{MI(X,Y) - E\{MI(X,Y)\}}{max\{H(X), H(Y)\} - E\{MI(X,Y)\}} \tag{3.10}$$

where $E\{MI(X,Y)\}$ is the expected mutual information between two random clusterings. The AMI takes value between 0 and 1 where 0 stands for random clustering and 1 represents a perfect partition.

V-Measure is the harmonic mean between two measures: homogeneity and completeness. Homogeneous clustering is when each cluster has data points belonging to the same class. Complete clustering is when all dat a points belonging to the same class are clustered into the same cluster. Given a set of classes $C = \{C_1, C_2, ..., C_l\}$, a set of cluster $K = \{K_1, K_2, ..., K_m\}$ and the conditional entropy of the class distribution given the identified clustering is computed as $H(C|K)$. The homogeneity is defined as follows:

$$h = \begin{cases} 1 & if H(C|K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & otherwise \end{cases} \tag{3.11}$$

The completeness is symmetrical to homogeneity. To measure the completeness, the distribution of cluster assignments within each class is assessed. In a perfect clustering, each of these distributions will be completely skewed to a single cluster. Given the homogeneity h and completeness c, the V-measure is computed as the weighted harmonic mean β between homogeneity and completeness:

$$V - measure = \frac{1 + \beta * h * c)}{(\beta * h) + c} \tag{3.12}$$

if β is greater than 1, completeness is weighted more strongly in the calculation. If β is less than 1, homogeneity is weighted more strongly. Since the computations of homogeneity, completeness and V-measure are completely independent of the number of classes, the number of clusters, the size of the dataset and the clustering algorithm, these measures can be employed for evaluating any clustering solution.

To evaluate the accuracy of methods in estimating the correct number of clusters, we used absolute symmetric log-modulus [143] transformation defined as follows:

$$L(x) = |sign(x) * log10(|x| + 1)| \tag{3.13}$$

where x is the difference between the estimated number of clusters and the true number of cell types in a given dataset. The higher values of absolute log-modulus transformation mean the number of estimated clusters is more different from the number of true cell types. x equals to zero denotes the perfect estimation.

Chapter 4

Method Validation and Results

In this chapter, we will focus on the presentation and exploration of our findings, offering a comprehensive view of the results derived from our imputation and clustering analysis. These results are derived from the public data previously introduced in the experimental design chapter.

4.1 Results of imputation analysis using RIA

Here we assess the performance of RIA using five single-cell datasets that are available in NIH Gene Expression Omnibus (GEO) [79] and Array Express [92]: Biase [93], Yan [94], Goolam [95], Deng [96], and Zeisel [40]. The processed data were downloaded from Hemberg lab's website (https://hemberg-lab.github.io/scRNA.seq.datasets).

In each dataset, the cell populations and developmental stages are known. This informatinom are only used *a posteriori* to assess the performance of each method in improving the identification of cell populations and the recovery of temporal trajectories. We compare our method with two state-of-the-art methods for single-cell imputation: MAGIC [11] and scImpute [18]. Both methods are widely used and each represents a different imputation strategy. MAGIC uses Markov affinity matrix to smooth the data while scImpute is a statistical approach that models the data as a mixture of Gamma and Gaussian distributions.

For each dataset, we downloaded the already processed expression data, in which genes are represented in rows and cells are in different columns. We only perform log_2 transformation to re-scale sc-RNAseq data, i.e., $log_2(\mathbf{A}+1)$ where $\mathbf{A}$ is the expression matrix. Genes that do not express across any cells will be removed.

4.1.1 RIA improves the identification of sub-populations while preserving the biological landscape

For each of the five datasets described in Table 3.1, the cell types are known. We use this information *a posteriori* to assess how separable the cell populations are after imputation. For each dataset, we have a raw matrix that serves as the input of each imputation method. After imputation, we have four matrices: the raw data and three imputed matrices (from RIA, MAGIC, and scImpute). In order to assess how separable the cell types in each matrix, we use k-means [97] to cluster each matrix and then compare the obtained partitionings with the known cell types. We use three different metrics for comparing the obtained partitionings with the known types: adjusted Rand index (ARI) [140], Jaccard index [144] and Purity [145].

Table 4.1: Comparisons of RIA performance against other methods using adjusted Rand index (ARI).

Dataset	**Adjusted Rand Index**			
	Raw	**RIA**	**scImpute**	**MAGIC**
Biase	0.558	**0.711**	-0.009	0.154
Yan	0.558	**0.573**	0.507	0.029
Goolam	0.501	**0.914**	0.321	0.197
Deng	0.549	**0.815**	0.229	0.483
Zeisel	0.738	**0.768**	0.689	0.289

Table 4.1 shows the ARI values obtained for each method and for the raw data. For each row, cells highlighted in bold have the highest ARI values. For each of the five datasets analyzed, the ARI values obtained for RIA are substantially higher than those of scImpute and MAGIC, demonstrating the superiority of the developed method over existing approaches. More importantly, the ARI values for RIA are higher than those obtained for raw data, demonstrating the ability of RIA in recovering the true expression of missing values due to dropout events. At the same time, it also demonstrates that RIA do not introduce false signals. In contrast, the ARI values obtained for scImpute and MAGIC are consistently lower than those obtained for raw data. There might be two reasons. First, these methods rely on sophisticated models that are prone to overfitting. Second, they lack of an efficient mechanism to verify whether a low expression value is due to sequencing limitation (i.e., dropout) or indeed due to biological phenomena. Therefore, they are likely to add false signals to the imputed data.

Tables 4.2 and 4.3 show the Jaccard index and Purity values obtained for raw data and imputed data using RIA, scImpute, and MAGIC. Again, these metrics confirm that RIA is the best among the competing methods. All of the three benchmarking metrics show that RIA consistently outperforms scImpute and MAGIC in every single analysis.

Here we will also demonstrate that RIA improves the quality of the data without altering the transcriptomics landscapes. Since single-cell data are high-dimensional and are hard to interpret, it is desirable to visualize them in low dimensional space with two or three dimensions. Traditionally, researchers use t-distributed Stochastic Neighbourhood Embedding (t-SNE) [98, 99] for this purpose, which preserve local structure among cells. We first use Principal Component Analysis (PCA) [100] to reduce the number of dimensions to 20, and then use t-distributed Stochastic Neigh-

Table 4.2: Comparisons of RIA performance against other methods using Jaccard Index

Dataset	**Jaccard Index**			
	Raw	**RIA**	**scImpute**	**MAGIC**
Biase	0.589	**0.708**	0.339	0.289
Yan	0.498	**0.498**	0.473	0.146
Goolam	0.496	**0.892**	0.375	0.312
Deng	0.524	**0.781**	0.395	0.518
Zeisel	0.651	**0.683**	0.605	0.285

Table 4.3: Comparisons of RIA performance against other methods using Purity Index

Dataset	**Purity Index**			
	Raw	**RIA**	**scImpute**	**MAGIC**
Biase	0.795	**0.836**	0.449	0.612
Yan	0.711	**0.778**	0.733	0.467
Goolam	0.822	**0.952**	0.693	0.621
Deng	0.805	**0.839**	0.627	0.750
Zeisel	0.876	**0.893**	0.840	0.668

bourhood Embedding (t-SNE) [101] to visualize the data. The purpose of using PCA is to reduce the running time of the visualization process.

Figures 4.1 and 4.2 show the visualization of the raw data and the imputed data. For all of the five datasets, the transcriptomics landscape of RIA is similar to that of the original data, demonstrating that RIA did not alter the transcriptomics landscape. On the contrary, the transcriptomics landscapes obtained from scImpute and MAGIC are very different from the those of the original data.

Regarding time complexity, both MAGIC and RIA are extremely fast. These two methods are able to analyze any of the five datasets in minutes. On the other hand, scImpute is slow because it needs to iteratively estimate the mixture parameters for every single gene across the genome. It takes scImpute an hour to analyze the Zeisel datasets using 20 cores.

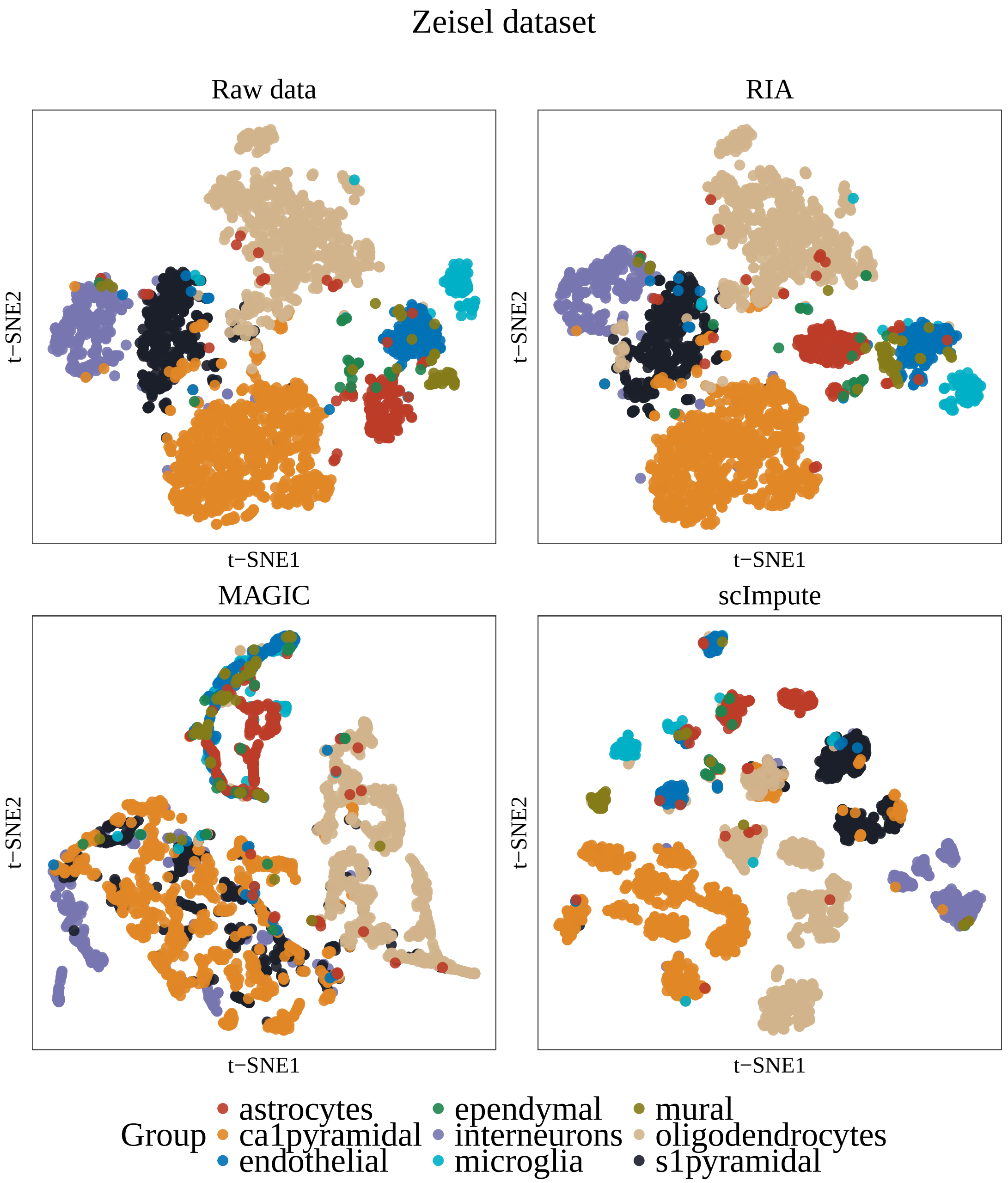

Figure 4.1: Transcriptomics landscape of the Zeisel dataset. The scatter plot shows first two principle components calculated by t-SNE for raw and imputation data using RIA, scImpute, and MAGIC. RIA preserve the transcriptomics landscape of the data whereas scImpute and MAGIC introduces artificial signals and complete change the landscape.

4.1.2 RIA recovers temporal trajectories in embryonic developmental stages

We use the four embryonic datasets to demonstrate RIA's ability in recovering the temporal dynamics. The Biase dataset consists of 49 inter-blastomere cells from mouse embryonic stem cells (mESCs), including *zygote, 2-cell and 4-cell.* The Goolam dataset includes transcriptome data of 124 individual cells in mouse pre-implantation development stages: *2-cell, 4-cell, 8-cell, 16-cell and blast.* The Yan dataset consists of 90 cells from human pre-implantation embryos and human embryonic stem cells (hESCs). The Deng dataset includes the expression profiles of 268 individual cells of mouse pre-implantation embryos of mixed background.

Figure 4.2 shows the transcriptomics landscape and temporal development stages using the raw data and imputation data produced by RIA, MAGIC, and scImpute. The lines in each scatter plot connect cell groups in consecutive developmental stages. For example, for the Biase dataset, the zygote group is directly connected with the 2-cell class while the 2-cell class is connected with the 4-cell class. For this dataset, raw data and data imputed by any of the three imputation methods clearly distinguish cells at different time points. The pseudotime ordering is consistent with the time labels. For the Goolam dataset, the landscapes of the raw data and data imputed by RIA and scImpute have similar pattern. On the contrary, the transcriptomics landscape of MAGIC is very different from the rest.

For the Yan and Deng datasets, the data imputed by RIA better distinguish cell groups of different time points. The pseudotime ordering for RIA accurately reflects the transcriptome dynamics along the time course. On the contrary, the raw data and data imputed by MAGIC and scImpute fail to depict a clear time trajectory. Overall, RIA better recovers temporal trajectories than existing state-of-the-art imputation methods.

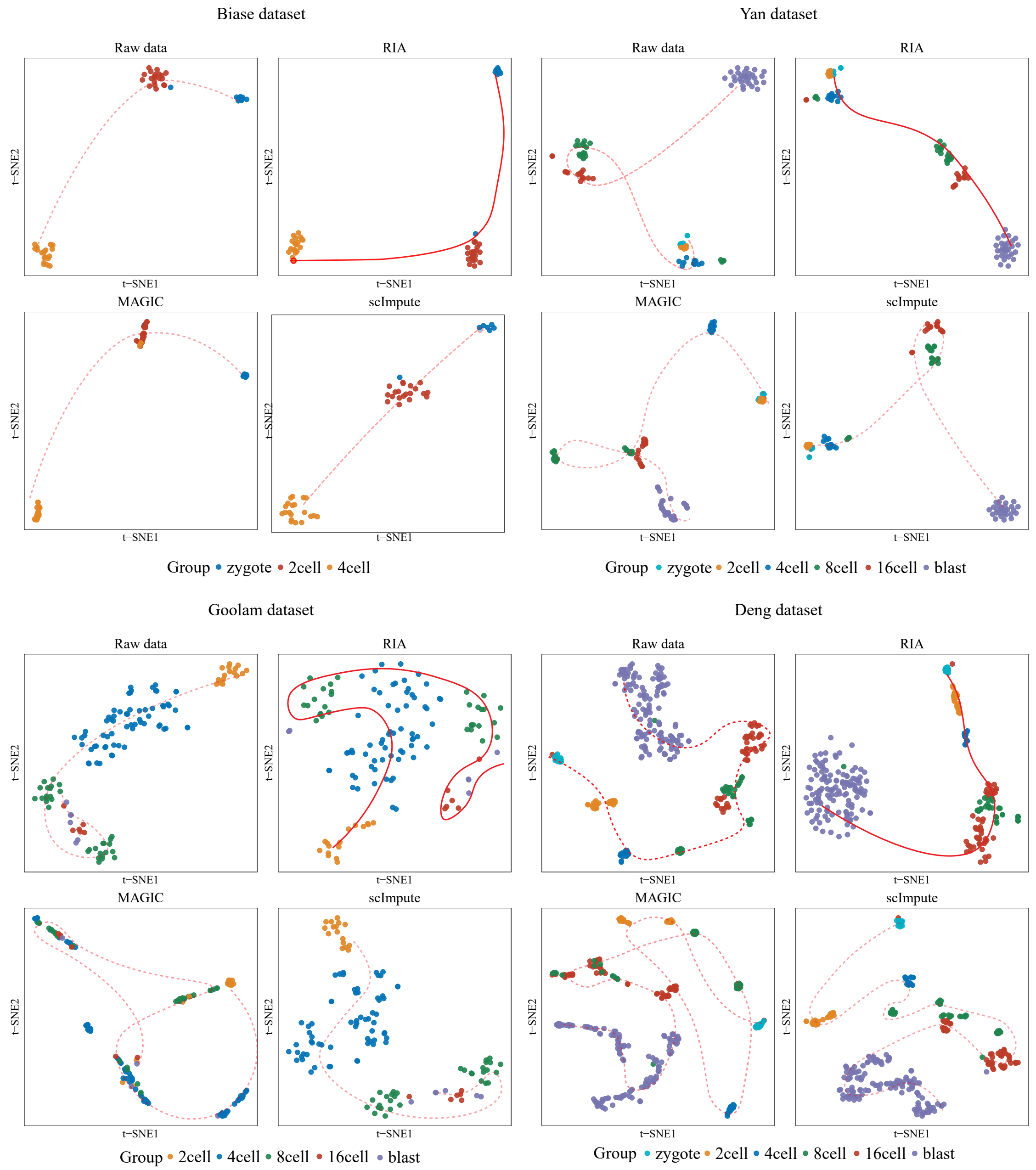

Figure 4.2: Transcriptomics landscape and temporal development stages. The scatter plots show the first two dimensions of the t-SNE results calculated from Biase, Yan, Goolam, and Deng datasets. Due to dropouts, it is difficult to recognize different temporal dynamics of cells. The raw data and imputed data using scImpute and DrImpute do not show clear patterns. On the contrary, RIA significantly elucidates the cell lineage identification such that it is clearly recognized in the 2-D scatter plots.

4.2 Results of imputation analysis using scIDS

In this section, we assess the performance of scIDS in the following capabilities: (1) improving the quality of cluster analysis, (2) preserving the cell transcriptome landscape. We compare scIDS with the raw data and two widely used scRNA-seq imputation methods, knn-smoothing [43], and MAGIC [11] using eight scRNA-seq datasets.

We use Cao dataset was downloaded from Broad Institute Single Cell Portal (`https://singlecell.broadinstitute.org/single_cell`). The Hrvatin dataset was downloaded from NCBI [79]. The remaining six datasets were downloaded from Hemberg Group's website (`https://hemberg-lab.github.io/scRNA.seq.datasets`). In each of these datasets, the true cell type (labels) is known. This information will be used *a posteriori* to assess the performance of each clustering method. We apply a log transformation (base 2) to rescale the data if the maximum expression value of the data is larger than 100, and we remove the genes that do not express across in any cells.

4.2.1 scIDS improves the identification of cells population.

For each of the eight datasets, we use a raw matrix as the input of each imputation method. After imputation, we obtain four matrices: the raw data and three imputed matrices (from knn-smoothing, MAGIC, and scIDS). In order to assess the segregation of the cell types in each matrix, we use k-means to cluster each matrix and then compare the obtained cluster assignments with the known cell types. We use three different metrics to quantify the quality of the clustering result: adjusted Rand index (ARI) [140], adjusted mutual information (AMI) [141] and V-measure [142].

Table 4.4 shows the ARI values obtained for each method and for the raw data from eight datasets. For each row, the values highlighted in bold indicate the highest ARI value. The cell with "N/A" indicates out of memory or error. In this analysis,

Table 4.4: Comparisons of scIDS performance against other methods using adjusted Rand index (ARI).

Dataset	**Adjusted Rand Index**			
	Raw	**knn-smooth**	**MAGIC**	**scIDS**
Pollen	0.955	0.577	0.564	**0.959**
Darmanis	0.612	0.194	0.298	**0.702**
Usoskin	0.736	0.035	0.276	**0.741**
Kolodziejczyk	0.727	0.203	0.163	**0.996**
Klein	0.984	**0.991**	0.451	0.984
Baron	0.557	0.568	**0.578**	0.559
Hrvatin	0.713	0.822	0.821	**0.832**
Cao	0.376	N/A	0.378	**0.434**
Mean	0.708	0.484	0.441	**0.776**

scIDS consistently outperforms all comparing methods by maintaining the highest average ARI value of 0.776. This is the highest value compared to 0.708, 0.484, and 441 of raw's, knn-smoothing's, and MAGIC's. More importantly, the ARI values obtained from scDIS are always higher than those obtained from raw data. This vast improvement demonstrates the ability of scIDS in imputing the dropouts without introducing false signals. Unlike scIDS, knn-smoothing and MAGIC have ARI values that are lower than the raw in 5 and 4 datasets. These methods rely on sophisticated models that might lead to overfitting. Moreover, knn-smoothing and MAGIC do not have an efficient mechanism to distinguish whether a low expression value is due to sequencing limitation (i.e., dropout) or indeed due to biological phenomena. Therefore, they are likely to add false signals to the imputed data.

Tables 4.5 and 4.6 show the adjusted mutual information and V-measure values obtained for raw data and imputed data using knn-smoothing, MAGIC, and scIDS. Again, the result is similar to the analysis using ARI. scIDS has the highest AMI values in 6 out of 8 datasets with an average AMI value of 0.808 while the average AMI values of raw data, knn-smoothing, and MAGIC are 0.761, 0.577, and 0.581, respectively. The same trend can be seen for V-measure values in Table 4.5. scIDS has the highest average of V-measure value (0.825). All of the three benchmarking metrics show that scIDS consistently outperforms knn-smoothing and MAGIC in all analyses.

Table 4.5: Comparisons of scIDS performance against other methods using adjusted mutual information (AMI).

Dataset	**Adjusted Mutual Information**			
	Raw	**knn-smooth**	**MAGIC**	**scIDS**
Pollen	0.95	0.788	0.79	**0.95**
Darmanis	0.722	0.411	0.532	**0.738**
Usoskin	0.716	0.069	0.404	**0.722**
Kolodziejczyk	0.774	0.304	0.296	**0.991**
Klein	0.97	**0.981**	0.579	0.97
Baron	0.681	0.65	**0.701**	0.683
Hrvatin	0.775	0.839	0.848	**0.862**
Cao	0.498	N/A	0.501	**0.551**
Mean	0.761	0.577	0.581	**0.808**

Table 4.6: Comparisons of scIDS performance against other methods using V-measure.

Dataset	**V-Measure Index**			
	Raw	**knn-smooth**	**MAGIC**	**scIDS**
Pollen	0.953	0.802	0.799	**0.954**
Darmanis	0.723	0.469	0.563	**0.742**
Usoskin	0.721	0.1	0.473	**0.726**
Kolodziejczyk	0.784	0.364	0.354	**0.992**
Klein	0.973	**0.981**	0.625	0.973
Baron	0.767	0.715	**0.787**	0.769
Hrvatin	0.828	0.845	0.858	**0.877**
Cao	0.515	N/A	0.518	**0.567**
Mean	0.783	0.611	0.622	**0.825**

4.2.2 scIDS preserves the biological landscape.

In this section, we show that scIDS has a capability of correctly imputing missing values without making a change to the transcriptomics landscapes. Preferably, life scientists impute the data in order to improve the quality of downstream analyses. At the same time, imputation should not completely change the data because of falsely introduced signals, leading to wrong or compromised findings. Since single-cell data are high-dimensional, the common practice is to project the high-dimensional data into a low dimensional space with two or three dimensions. The visualization in 2-D or 3-D helps researchers to interpret the single-cell data more efficiently. To reduce the running time, we first use a fast partial singular value decomposition method [108] to quickly reduce the number of features to 20. Then, we use t-SNE [99], and UMAP [109] to project the compact data into two-dimensional space for visualization.

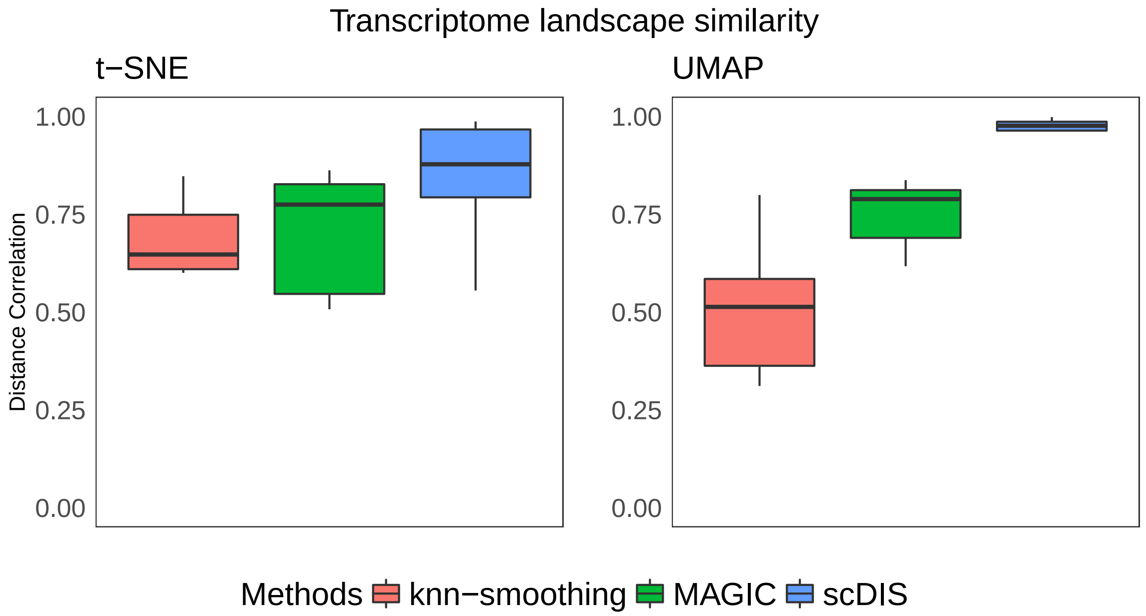

Figure 4.3: The similarity between the imputed and original landscapes.

To quantify the similarity between the imputed and original landscapes, we calculate the distance correlation index ($dCor$) [110] for each imputed landscape generated by t-SNE and UMAP. Given X and Y as the 2D representation of the raw and imputed data, $dCor$ is calculated as $dCor = \frac{dCov(X,Y)}{\sqrt{dVar(X)dVar(Y)}}$ where $dCov(X,Y)$ is the distance covariance between X and Y while $dVar(X)$ and $dVar(Y)$ are distance variances of X and Y. The $dCor$ coefficient takes value between 0 and 1, with the $dCor$ is expected to be 1 for a perfect similarity. Unlike Pearson correlation, $dCor$ measures both the linear and nonlinear associations between X and Y [110]. Especially, $dCor$ remains constant when we rotate the transcriptome landscape. Figure 4.3 shows the distribution of $dCor$ values for all eight analyzed datasets. In this figure, the left panel shows the values obtained from t-SNE while the right panel shows the values obtained from UMAP representations. The bar plot shows that scIDS has the highest $dCor$ values. A Wilcoxon test also confirms that the correlation $dCor$ obtained from scIDS are significantly higher than the rest ($p = 1.1 \times 10^{-2}$ and 6.11×10^{-5} for t-SNE and UMAP, respectively).

Figure 4.4 shows the visualization of the raw data and the imputed data for the Baron dataset. Among three comparing methods, the transcriptomics landscape of scIDS is similar to that of the original data (raw), demonstrating that scIDS did not alter the transcriptomics landscape. On the contrary, the transcriptomics landscapes obtained from knn-smoothing and MAGIC are very different from the original data.

4.3 Results of imputation analysis using scISR

In this section, we assess the performance of scISR, we use both real scRNA-seq data and simulation. We compare scISR with five popular methods, MAGIC [11], scImpute [18], SAVER [17], scScope [50], and scGNN [146]. SAVER and scImpute are statistical approaches that impute the missing values using mixture models; MAGIC is a mathematical approach that relies on Markov transition to estimate the missing values. scScope uses a recurrent network layer to iteratively perform imputations

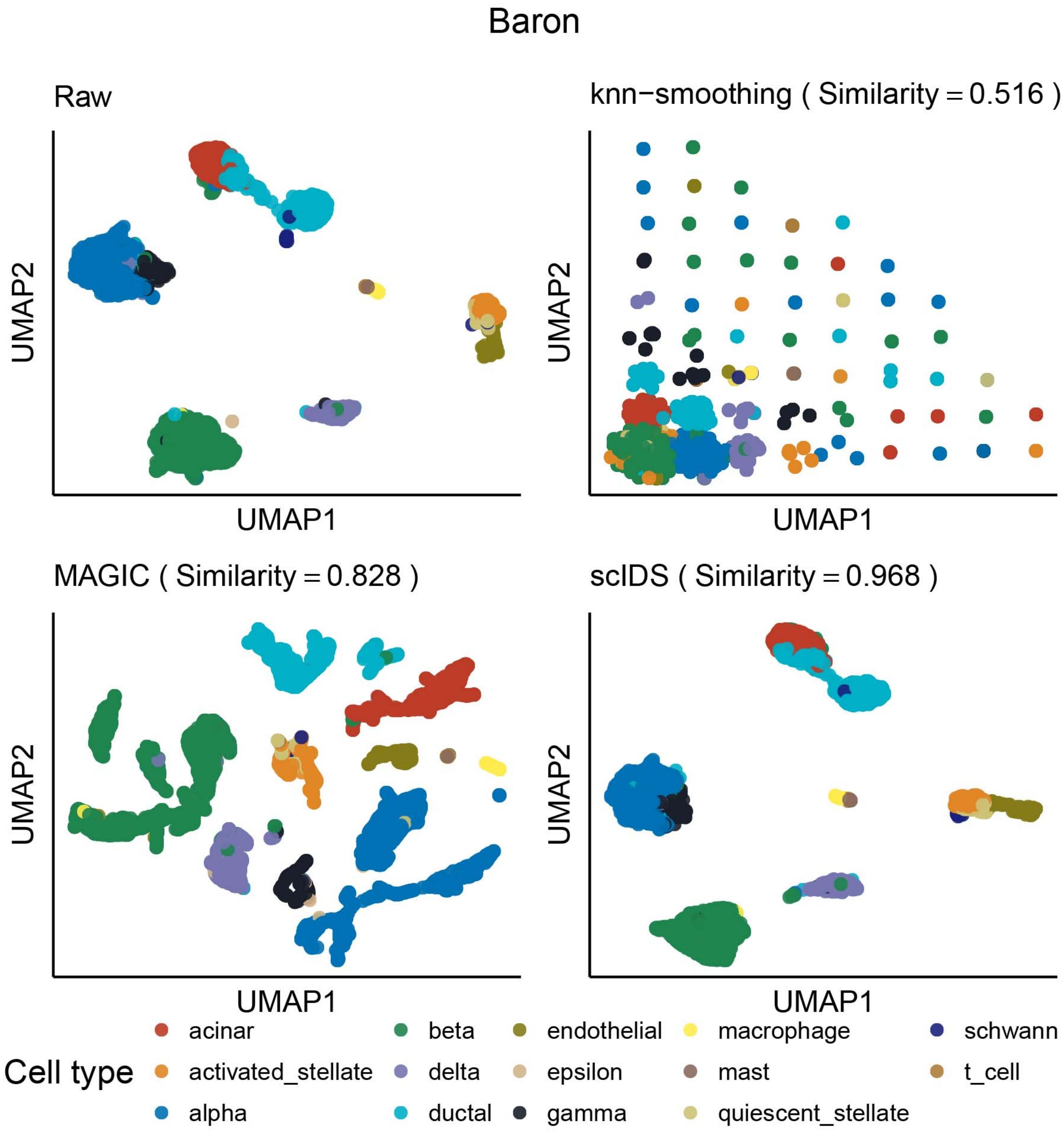

Figure 4.4: The visualization of Baron dataset.

on zero-valued entries of input scRNA-seq data. scGNN formulates and aggregates cell–cell relationships with graph neural networks and models heterogeneous gene expression patterns using a left-truncated mixture Gaussian model. scGNN uses the cell-cell relationships to impute the dropouts.

First, we apply the six methods on 25 real scRNA-seq datasets with known cell types presented in Table 3.3. The cell labels are only used *a posteriori* to assess whether the imputation enhances the cell segregation, i.e., making the cell types more separable without drastically altering the transcriptome landscape. Second, we simulate 116 single-cell expression datasets whose values follow different distributions and dropout rates. We then apply the six imputation methods, scISR, MAGIC, scImpute, SAVER, scScope, and scGNN on the masked dataset to recover the missing values. Since we know exactly the missing entries and values, we can accurately assess the reliability of each method in terms of both sensitivity and specificity.

4.3.1 Cluster analysis of 25 scRNA-seq datasets

We use the known cell types of the 25 scRNA-seq datasets to assess whether the imputation helps separate cells of different types in cluster analysis. We compare scISR against MAGIC, scImpute, SAVER, scScope, and scGNN using three assessment metrics: Adjusted Rand Index (ARI) [140], Jaccard Index (JI) [144], and Purity Index (PI) [145].

Given a dataset (raw data), we use k-means to cluster the cells using the true number of cell types k as the number of clusters. We calculate the Adjusted Rand Index (ARI) [140] to compare k-means partitioning against the known cell labels. Rand Index (RI) measures the agreement between a given clustering and the ground truth. The ARI is the corrected-for-chance version of the RI. The ARI takes values from -1 to 1, with the ARI expected to be 1 for a perfect agreement, and 0 for random partitionings. Next, we apply each of the six imputation methods to the raw data to obtain the imputed data. Again, we use k-means to partition the imputed data and calculate the ARI values using the true cell labels. We expect that by imputing the raw data, we obtain better data in which the cells of different types are more separable. Therefore, we assess the performance of each method by comparing the

ARI of the imputed data against the ARI obtained from the raw data. We repeat the whole procedure for all 25 datasets to assess how well each imputation method performs.

Table 4.7 and Figure 4.6 show the ARI values obtained for the 25 datasets. For each row, a cell of a method is highlighted in green if the imputed ARI is higher than the raw ARI. The maximum memory permitted for each analysis was set to 100 GB of RAM. scISR and MAGIC are the only methods able to analyze all datasets. scImpute runs out of memory when analyzing datasets with 23,178 cells (Tasic) or larger. SAVER crashes when analyzing the Tasic dataset, and it runs out of memory when analyzing datasets with 90,579 cells (Cao) or larger. scScope runs out of memory when analyze the biggest dataset (Darrah). scGNN ran out of memory when analyzing the datasets Cao, Orozco, and Darrah. We report the running time of imputation methods on 25 single-cell datasets in Figure **??**. Overall, scISR is the fastest method and can complete the imputation for the largest dataset (Darrah) in 50 minutes. For 25 real datasets, scISR is able to improve the ARI values 21 out of 25. The average ARI value of scISR is 0.571, which is the highest compared to those of raw data and data imputed by MAGIC, scImpute, SAVER, scScope, and scGNN (0.504, 0.461, 0.286, 0.423, 0.165, and 0.279, respectively). Overall, scISR increases the ARI values by 13.3% across all datasets. For the two datasets Zyl (Human) (24,023 cells) and Zilionis (Human) (34,558 cells), scISR increases the ARI values significantly (11.3% and 14.5%, respectively). For Orozco and Darrah datasets with more than 100,000 cells, scISR increases the ARI values by 13.6% and 77.2%, respectively. A one-sided Wilcoxon test also confirms that the ARI values of scISR are significantly higher than those of raw data ($p = 3.2 \times 10^{-5}$) and of other imputation methods ($p = 9.8 \times 10^{-6}$).

To perform a more comprehensive analysis, we also compare the methods using two other metrics: Jaccard Index (JI) [144] and Purity Index (PI) [145]. The detailed

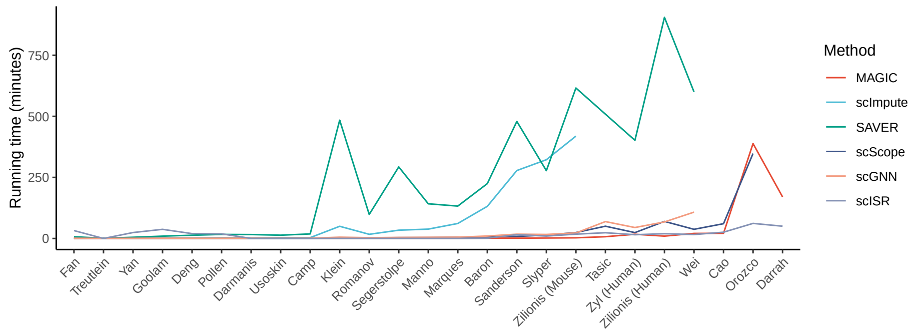

Figure 4.5: Running time of the six imputation methods on 25 real scRNA-seq datasets. scISR is the fastest and can impute the Darrah dataset in 50 minutes.

results for each dataset and method are reported in Table 4.1 and Supplementary Tables S2–S3. Overall, scISR is the only method that has better clustering accuracy on average when comparing with using the raw data. The results are similar for analyses using JI and PI. Among all methods, scISR has the highest average JI values (Supplementary Table S2). Its average JI value is 0.531, compare to 0.468, 0.453, 0.276, 0.403, 0.243 and 0.273 of the raw data, MAGIC's, scImpute's, SAVER's, scScope's, and scGNN's. A one-sided Wilcoxon test also confirms that the JI values of scISR are significantly higher than those of raw data ($p = 3.2 \times 10^{-5}$) and of all other methods ($p = 4.8 \times 10^{-5}$). Supplementary Table S3 shows the PI values obtained from raw and imputed data. It is the only method that has the average PI value higher than that of the raw data. All other methods have an average PI less than that of the raw data. scISR improves cluster analysis by having PI values higher than those of the raw data in 15 out of 25 datasets. A one-sided Wilcoxon test also confirms that the PI values of scISR are significantly higher than those of raw data ($p = 0.007$) and of all other methods ($p = 9.9 \times 10^{-5}$).

We also report the gene level normalized intra dispersion, which is the ratio between the intra-cell-type standard deviation and the gene's standard deviation, in Supplementary Figure S2. The median dispersion of scISR is 3.6×10^{-3} which is much lower compared to 2×10^{-1}, 1.1×10^{2}, 2.4×10^{-1}, 1.3×10^{-1}, 2.3×10^{-2}, and 5.4×10^{1} of raw data and data imputed by MAGIC, scImpute, SAVER, scScope and scGNN, respectively.

To further assess the performance of imputation methods, we perform an additional clustering analysis using Seurat [23]. This method can automatically determine the number of cell types from the input data. We first used Seurat to cluster the raw and imputed data of the 25 real scRNA-seq datasets. We then compared the clustering results against true cell types using Adjusted Rand Index (ARI). Supplementary Figure S3 and Table S4 show the ARI values obtained from the raw data and the data obtained from the six imputation methods. scISR is able to improve the cluster analysis in 14 out of 25 datasets. MAGIC, scImpute, SAVER, scScope, and scGNN improve the cluster analysis in 5, 3, 5, 4, and 5 datasets, respectively. The mean ARI value of scISR is 0.499 which is higher than the mean ARI values of all other methods (the mean ARI values for MAGIC, scImpute, SAVER, scScope, and scGNN are 0.315, 0.283, 0.324, 0.155, and 0.186, respectively). scISR is the only method that has mean ARI higher than that of the raw data.

Next, to assess the performance of each method with respect to different cell isolation techniques, quantitative schemes, and normalized units, we divide the datasets into multiple overlapping groups: (1) 14 plate-based and 11 droplet-based datasets; (2) 12 with UMI and 13 with read count; and (3) 7 without normalization, 11 with transcript length-normalization (RPKM/FPKM/TPM), and 7 with transcript-depth normalization (CPM/RPM). Figure 4.6 shows the ARI values obtained for raw data and data imputed by four imputation methods. The ARI values of scISR are consistently higher than those of raw data and of other methods in each grouping. Interestingly, the ARI values of raw data are comparable across quantification schemes (UMI/read) but differ greatly across different normalization units (Figure 4.7A). Well-

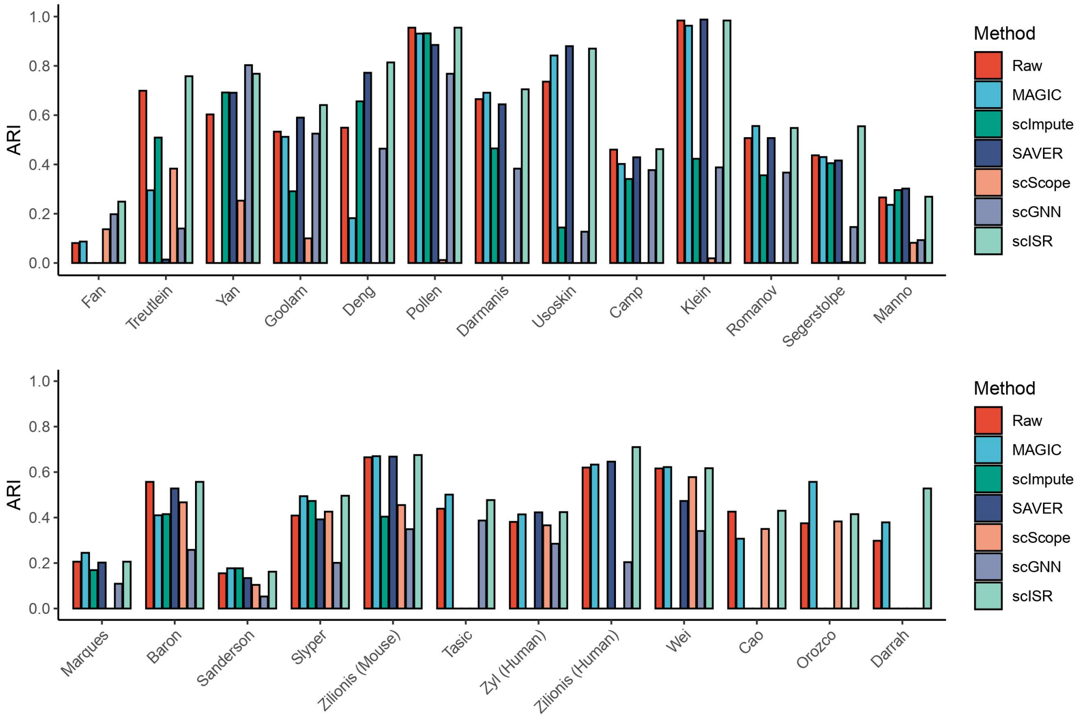

Figure 4.6: Adjusted Rand Index (ARI) obtained from raw and imputed data. The x-axis shows the names of the datasets while the y-axis shows ARI value of each method. scISR improves cluster analysis by having ARI values higher than those of the raw data in 21 out of 25 datasets.

known normalization techniques developed for bulk RNA-seq (RPKM/FPKM/TPM) improve raw data's cluster analysis (better than no normalization), but they have apparent disadvantages compared to CPM/RPM. The ARI values of scISR follow the same trend but are always higher than those of raw data. Similarly, Figures 4.7B and Figure 4.7C show the JI and PI values obtained for the cluster analysis. Regardless of the assessment metrics, cluster analysis in conjunction with scISR has a notable advantage over other imputation methods.

To understand the impact of data scaling on the performance of the imputation methods, we also perform the same analysis without log transformation applied to the input data. Supplementary Figure S4 shows the overall results of the analysis while Supplementary Tables S5–S7 show the detailed results for each dataset and

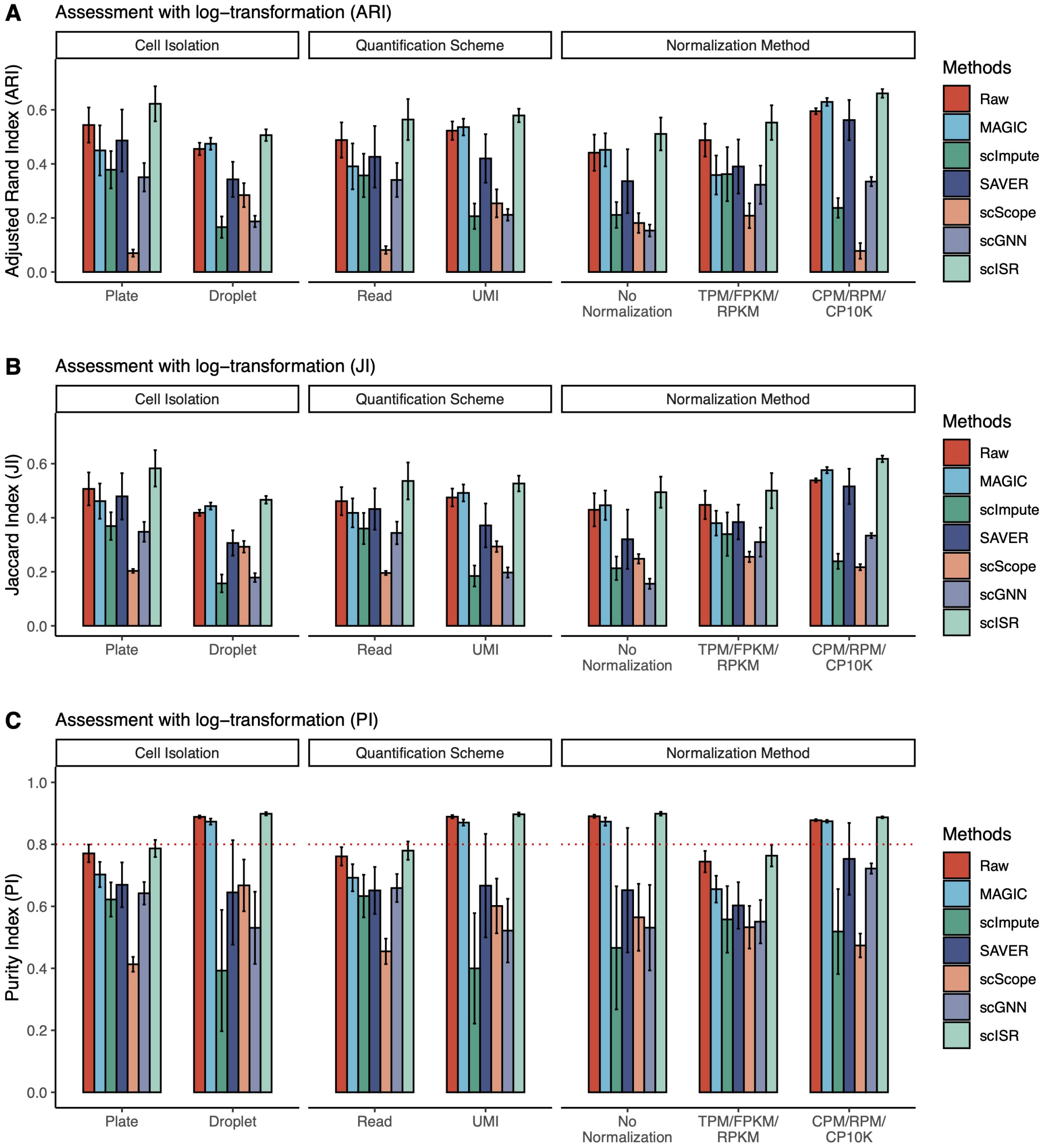

Figure 4.7: Assessment results of each imputation method with respect to cell isolation techniques, quantification schemes, or normalized units. The analysis is performed with a log transformation of the data. Panel (A) shows the results using Adjusted Rand Index (ARI), while panels (B) and (C) show the results using Jaccard Index (JI) and Purity Index (PI). scISR consistently outperforms other methods in every grouping by having the highest ARI, JI, and PI values.

Table 4.7: Adjusted Rand Index (ARI) obtained from raw and imputed data. In each row, a cell is highlighted in green if the ARI value is higher than that of the raw data. scISR improves cluster analysis by having ARI values higher than those of the raw data in 21 out of 25 datasets. A one-sided Wilcoxon test also confirms that the ARI values of scISR are significantly higher than those of raw data ($p = 3.2 \times 10^{-5}$) and of all other methods ($p = 9.8 \times 10^{-6}$).

Dataset	Size	Raw	MAGIC	scImpute	SAVER	scScope	scGNN	scISR
Fan	69	0.081	0.087	0.000	0.000	0.137	0.198	0.249
Treutlein	80	0.699	0.295	0.509	0.014	0.383	0.140	0.758
Yan	90	0.603	0.000	0.692	0.691	0.253	0.803	0.768
Goolam	124	0.533	0.512	0.291	0.590	0.1	0.525	0.641
Deng	268	0.549	0.182	0.656	0.772	0	0.464	0.814
Pollen	301	0.955	0.931	0.932	0.885	0.012	0.768	0.955
Darmanis	466	0.665	0.691	0.465	0.644	0	0.383	0.705
Usoskin	622	0.736	0.842	0.144	0.880	0	0.127	0.870
Camp	734	0.460	0.402	0.341	0.429	0	0.377	0.462
Klein	2,717	0.984	0.963	0.423	0.988	0.019	0.388	0.984
Romanov	2,881	0.507	0.556	0.356	0.507	0	0.367	0.548
Segerstolpe	3,514	0.437	0.430	0.405	0.576	0.004	0.146	0.555
Manno	4,029	0.266	0.236	0.296	0.302	0.082	0.093	0.269
Marques	5,053	0.206	0.245	0.169	0.202	0	0.109	0.206
Baron	8,569	0.557	0.410	0.415	0.528	0.467	0.258	0.557
Sanderson	12,648	0.155	0.177	0.177	0.134	0.104	0.053	0.162
Slyper	13,316	0.409	0.494	0.473	0.392	0.426	0.201	0.496
Zilionis (Mouse)	15,939	0.665	0.670	0.404	0.668	0.455	0.349	0.675
Tasic	23,178	0.439	0.501	N/A	N/A	0	0.387	0.477
Zyl (Human)	24,023	0.381	0.414	N/A	0.423	0.366	0.285	0.424
Zilionis (Human)	34,558	0.620	0.633	N/A	0.646	0	0.204	0.710
Wei	41,565	0.616	0.622	N/A	0.473	0.578	0.341	0.617
Cao	90,579	0.426	0.307	N/A	N/A	0.35	N/A	0.430
Orozco	100,055	0.375	0.557	N/A	N/A	0.383	N/A	0.415
Darrah	162,490	0.298	0.379	N/A	N/A	N/A	N/A	0.528
Mean ARI		0.504	0.461	0.286	0.423	0.165	0.279	0.571

[1] N/A: Out of memory or error.

method. With the exception of scISR, a decrease in performance is observed for all imputation methods due to the dominance of genes with large values. This leads to a wider accuracy gap between scISR and other imputation methods.

4.3.2 Preservation of the transcriptome landscape

The purpose of this analysis is to assess whether the imputation alters the transcriptome landscape. Preferably, life scientists impute the data in order to improve the quality of downstream analyses. At the same time, imputation should not completely change the data because of falsely introduced signals, leading to wrong or compromised findings. In the above sections, we have demonstrated that scISR significantly improves the quality of downstream analyses (e.g., cluster analysis). In this section, we will demonstrate that scISR preserves the transcriptome landscape of the data as well. For this purpose, we will visualize the transcriptome landscape of the raw and imputed data using t-SNE [99] and UMAP [109]. We will also quantify the similarity between the imputed and original landscapes using the distance correlation index [110].

First, we use t-SNE [99] to generate the 2D transcriptome landscapes of the raw and imputed data. The 2D visualizations of the 25 datasets are shown in Supplementary Figures S6–S10. Overall, MAGIC, SAVER, and scISR produce landscapes that are similar to those of the raw data for every single dataset analyzed. The same cannot be said about scImpute, scScope, and scGNN. For the Manno dataset (last row in Supplementary Figure S8), scImpute, scScope, and scGNN completely alter the landscape. scImpute tends to split cells into smaller groups while scScope and scGNN mix cells from different cell types together. This can be clearly observed in datasets such as Camp, Segerstolpe, Manno (Human).

To perform a more comprehensive analysis, we also generate the 2D transcriptome landscapes of the 25 datasets using UMAP [109]. The visualizations are shown in Supplementary Figures S11–S15. Again, except for scImpute, scScope, and scGNN, other methods preserve the landscape very well. For scImpute, scScope, and scGNN, the difference between the original and imputed landscape becomes more obvious in UMAP visualization.

To quantify the similarity between the imputed and original landscapes, we calculate the distance correlation index ($dCor$) [110] for each imputed landscape generated by t-SNE and UMAP. Given X and Y as the 2D representation of the raw and imputed data, $dCor$ is calculated as $dCor = \frac{dCov(X,Y)}{\sqrt{dVar(X)dVar(Y)}}$ where $dCov(X,Y)$ is the distance covariance between X and Y while $dVar(X)$ and $dVar(Y)$ are distance variances of X and Y. Specifically, the method first calculates the pair-wise distances for X by computing the distance between each pair of cells, resulting in a square matrix. Second, it calculates the pair-wise distances for Y. Finally, it compares the two matrices using the formula described above to obtain the distance correlation. The $dCor$ coefficient takes a value between 0 and 1, with the $dCor$ is expected to be 1 for a perfect similarity. In our analysis, when we rotate the transcriptome landscape, $dCor$ does not change. In contrast to Pearson correlation, this metric measures both the linear and nonlinear associations between X and Y [110].

The $dCor$ values are displayed in each panel in Supplementary Figures S6–S15. We also plot the $dCor$ distributions in Figure 4.8. In this figure, the left panel shows the values obtained from t-SNE while the right panel shows the values obtained from UMAP representations. The mean correlations using t-SNE for MAGIC, scImpute, SAVER, scScope, scGNN, and scISR are 0.78, 0.46, 0.68, 0.36, 0.48, and 0.88, respectively. The bar plot shows that scISR has the highest mean correlation, as well as the smallest variance. This demonstrates that scISR consistently preserves the transcriptome landscape of the datasets analyzed. MAGIC is the second-best method in this analysis. Using UMAP, scISR obtains a mean correlation of 0.86 compared to those of 0.8, 0.5, 0.7, 0.4, and 0.57, for MAGIC, scImpute, SAVER, scScope, and scGNN, respectively. A one-sided Wilcoxon test also confirms that the correlation values obtained from scISR are significantly higher than the rest ($p = 3 \times 10^{-9}$ and 2.8×10^{-7} for t-SNE and UMAP, respectively).

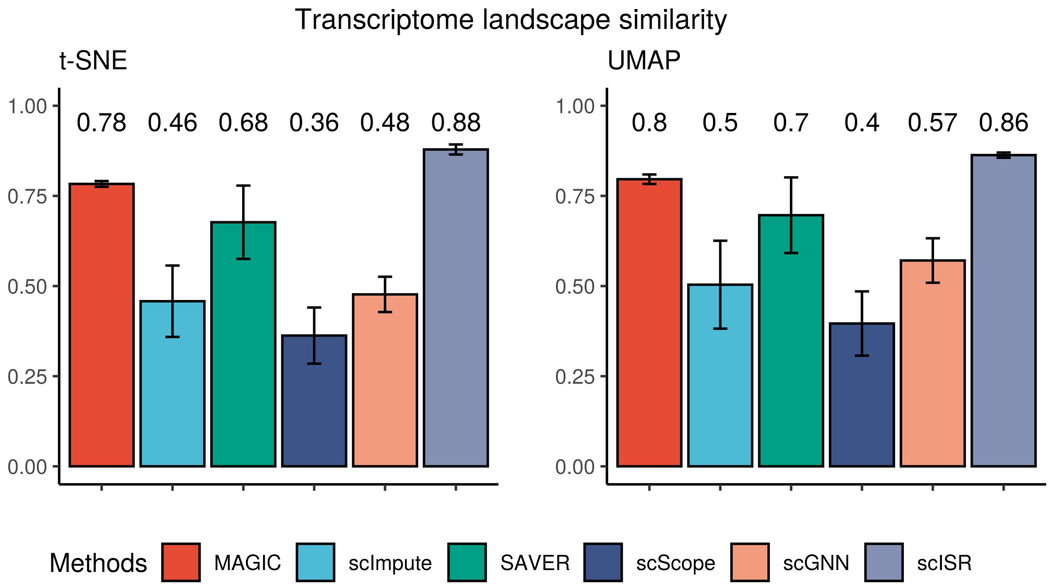

Figure 4.8: The distance correlation between raw data and imputed data using the first two components obtained from t-SNE and UMAP. Higher correlation values indicate more similarity between the imputed and original landscapes. Different colors represent different imputation methods. scISR has the highest mean correlation with the smallest variance. A one-sided Wilcoxon test indicates that the correlation values obtained from scISR are significantly higher than the rest ($p = 3 \times 10^{-9}$ and 2.8×10^{-7} for t-SNE and UMAP, respectively).

4.3.3 Simulation studies

To present a comprehensive simulation analysis, we generate a number of simulations by varying the number of cells from 100 to 10,000 and the number of genes from 300 to 10,000. The cells/genes combination setups are presented as follows: 100×300, 1,000×3,000, 3,000×9,000, 5,000×10,000, 7,000×10,000, and 10,000×10,000.

In each of the 6 datasets, the expression values follow a normal distribution $\mathcal{N}(\mu, \sigma)$. We set $\mu = 1$ and $\sigma = 0.15$. We slightly shift the mean of the cells and genes to create 4 different cell types and 3 gene groups – each cell type has an equal number of cells. We name this data as *complete data* and use the expression values as the ground truth for benchmarking. Next, we introduce the dropout events. We randomly select 40% of the genes and consider those as genes that are impacted by dropout events. We randomly assign 30% of the values of these genes to zero. We name this data as *masked data*.

In these case studies, we present a detailed simulation results for 3 datasets: 100×300, 1,000×3,000 and 10,000×10,000. Panels A and B in Figures 4.9, 4.10 and 4.11 show the simulation data for the setting of 100×300, 1,000×3,000 and 10,000×10,000, respectively. In each figure, panel A shows the transcriptome landscape of the complete data and panel B shows the masked data. In each dataset, the transcriptome landscape and gene-cell heatmap of the *complete data* clearly show the presence of three cell types and four gene groups. With *masked data*, dropout events clearly alter the cells' transcriptome landscape, making it difficult to separate the cell types. The ultimate goal of imputation is to infer the masked (dropout) values in order to recover the original transcriptome landscape and expression profile.

We apply the six imputation methods on the *masked data* and assess the quality of the imputed data by comparing them against the ground truth. Panels C, D, E, F, G, and H in Figures 4.9, 4.10 and 4.11 show the data imputed by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR, respectively.

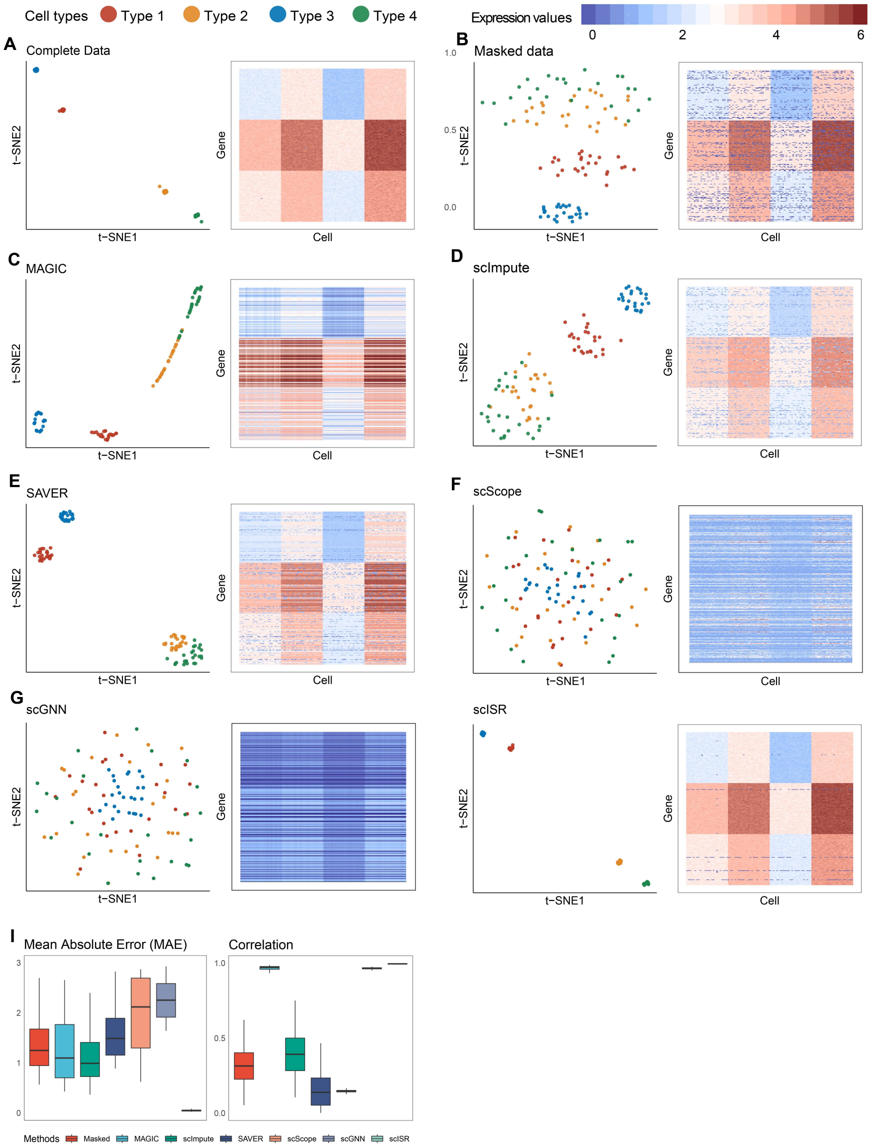

Figure 4.9: Assessment of MAGIC, scImpute, SAVER, and scISR using simulation (100 cells and 300 genes). (A) – (H) The visualization of the *complete data*, *masked data* and *imputed data* recovered by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR. In each subfigure, the left panel shows the transcriptome landscape using t-SNE while the right panel shows the gene-cell heatmap. (I) Mean Absolute Error (MAE) and correlation coefficients obtained by comparing masked/imputed data with the complete data. We calculate the MAE and correlation values for each gene and then plot the distributions of each metric using boxplot. The transcriptome landscapes and heatmaps show that scISR comes closest to recovering the complete data. scISR also has significantly smaller MAE values as well as significantly higher correlation coefficients than other methods with p-values 1.6×10^{-64} and 9.2×10^{-63}, respectively (Wilcoxon test).

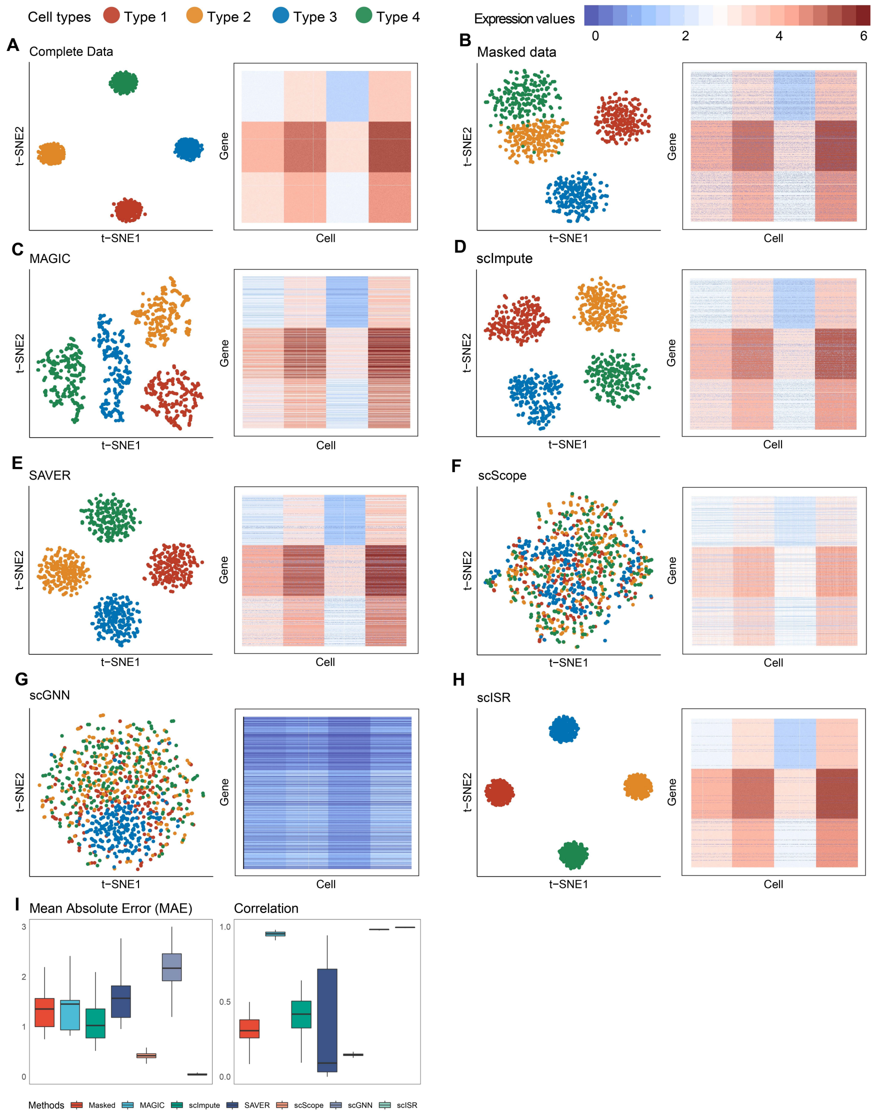

Figure 4.10: Assessment of MAGIC, scImpute, SAVER, and scISR using simulation of 1,000 cells. (A) – (H) The visualization of the *complete data*, *masked data* and *imputed data* recovered by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR. In each subfigure, the left panel shows the transcriptome landscape using t-SNE while the right panel shows the gene-cell heatmap. (I) Mean Absolute Error (MAE) and correlation coefficients obtained by comparing masked/imputed data with the complete data. We calculate the MAE and correlation values for each gene and then plot the distributions of each metric using boxplot. The transcriptome landscapes and heatmaps show that scISR comes closest to recovering the complete data. scISR also has significantly smaller MAE values as well as significantly higher correlation coefficients than other methods with p-values $< 10^{-100}$ and $< 10^{-100}$, respectively (using Wilcoxon test).

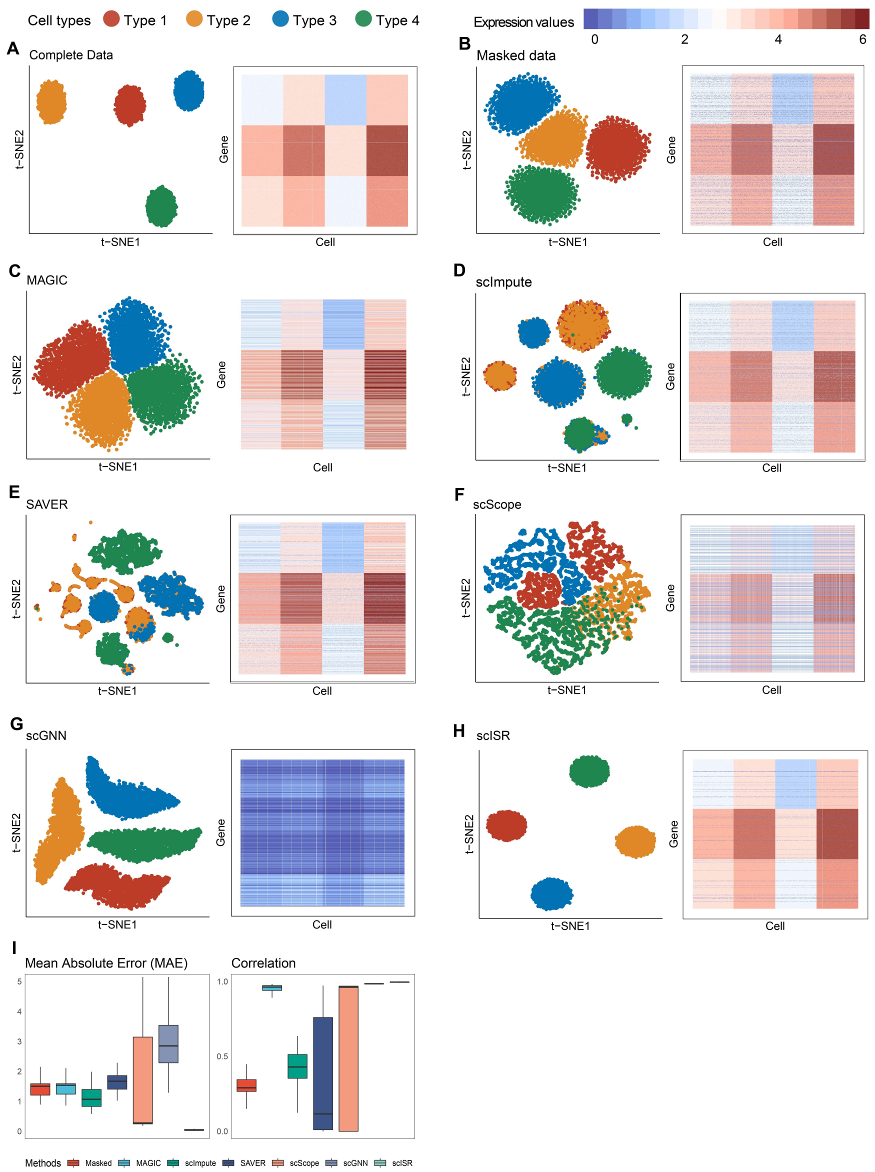

Figure 4.11: Assessment of MAGIC, scImpute, SAVER, and scISR using simulation of 10,000 cells. (A) – (H) The visualization of the *complete data*, *masked data* and *imputed data* recovered by MAGIC, scImpute, SAVER, scScope, scGNN, and scISR. In each subfigure, the left panel shows the transcriptome landscape using t-SNE while the right panel shows the gene-cell heatmap. (I) Mean Absolute Error (MAE) and correlation coefficients obtained by comparing masked/imputed data with the complete data. We calculate the MAE and correlation values for each gene and then plot the distributions of each metric using boxplot. The transcriptome landscapes and heatmaps show that scISR comes closest to recovering the complete data. scISR also has significantly smaller MAE values as well as significantly higher correlation coefficients than other methods with p-values $< 10^{-100}$ and $< 10^{-100}$, respectively (using Wilcoxon test).

These case studies show that MAGIC imputes the missing values by smoothing the expression values. Many expression values, including non-zero-valued entries, were altered by MAGIC, making the landscape of the imputed data very different from those of both *complete* and *masked data.* scImpute improves the quality of the data but is still not able to separate some cell types. In addition, scImpute also alters the values of non-zero entries to make the data better fit into the assumed mixture model. SAVER further improves the transcriptome landscape and separates the 4 cell types. However, data imputed by SAVER does not entirely match with the *complete data*, in which many dropout values remain uncorrected many other dropout entries imputed with wrong values. scScope and scGNN oversmooth the imputed data such that it merges all the cells in four types together. The heatmaps clearly show that many expression values, including non-zero-valued entries, were altered by scScope and scGNN.

Using the true expression values of the complete data in all 6 datasets, we calculate the mean absolute error (MAE) and correlation between the imputed data and the ground truth for the genes that were impacted by dropout events. Figure 4.12 displays the mean absolute error (MAE) (left panel) and correlation values (right panel) for each method and each cell/gene combination. scISR is the best method in recovering the gene expression values with the smallest MAE and the highest correlation values.

In the second scenario, we generate in total 40 datasets resulted from the combination of 2 different dropout distributions: uniform and normal, 4 different dropout rates: 60%, 70%, 80%, and 90%, and 5 different sizes of data with the number of cells×genes are: 1,000×3,000, 3,000×9,000, 5,000×10,000, 7,000×10,000, and 10,000×10,000. Since scISR uses the hypergeometric test, which can be less accurate when the dropout probability does not follow a uniform distribution, we use this simulation to assess the stability of scISR when imputing data with different dropout distributions.

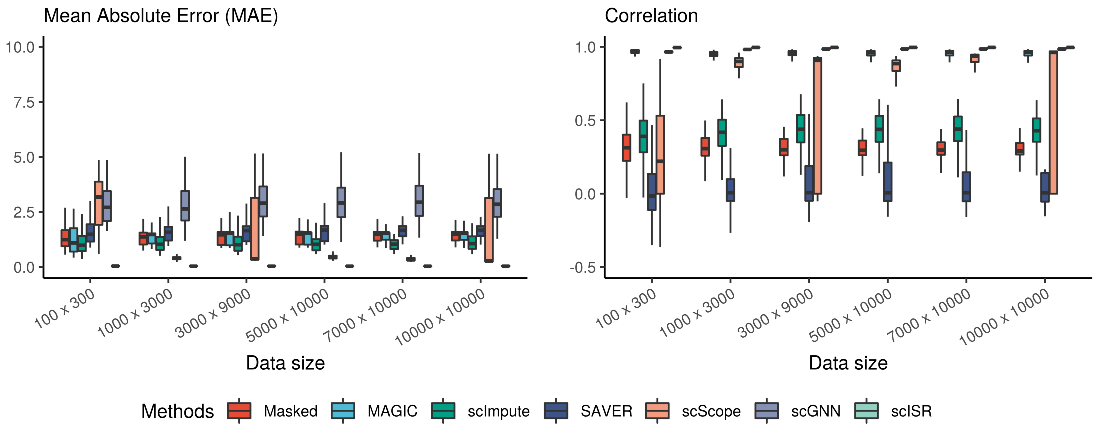

Figure 4.12: Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using simulation studies. Mean Absolute Error (MAE) and correlation coefficients were obtained by comparing imputed data with the complete data. In each analysis, scISR has smaller MAE values and higher correlation coefficients than other methods.

To generate datasets of a certain size (e.g., 1,000×3,000), we first generate an expression matrix whose values follow a normal distribution $N(\mu, \sigma)$ where $\mu = 1$ and $\sigma = 0.15$. We then slightly shift the mean of the cells and genes by adding a certain value to each group (-1, 0, 1, 1.5 for cell groups and -1, 0, 1 for gene groups) to create 4 different cell types. We name this as *complete data*. Next, we randomly assign dropout values to the data in two different cases. In the first case, the dropout probability is uniformly distributed. In the second case, the dropout probability follows a normal distribution. For example, at 60% dropout rate, the dropout probability follows a distribution of $N(0.6, 0.1)$. We then vary the dropout rate from 60% to 90%. We name the data with dropouts as *masked data*. Next, we impute the *masked data* using imputation methods to obtain the *imputed data*. Finally, to assess the performance of imputation methods, we compare the imputed data against the complete data using Mean Absolute Error (MAE) and correlation coefficients. The detailed results are presented in Supplementary Figure S19.

Overall, when the dropout probability is uniformly distributed, in all datasets, scISR is able to recover most of the dropout values, resulting in a median MAE close to zero and correlation coefficients close to one at any dropout rate. When the dropout probability is normally distributed, in all datasets, scISR still performs as well at 60% to 80% dropout. When the dropout rate is 90%, for the dataset of size 1,000×3,000, scISR can recover only a part of the data (median MAE of approximately 2.11 compared to 3.65 of masked data). However, the results clearly show that the bigger the size of the data, the better scISR can recover the missing values. The reason for such improvement is that with the same dropout rate, larger datasets provide us with more data to learn from, leading to improved hypothesis testing (hypergeometric test) and prediction (linear regression). For datasets with 7,000 cells or more, the median MAE is close to zero for both uniform and normal distributions at any dropout rate. In summary, scISR (using hypergeometric test) performs well for large datasets with high dropout rates even when the dropout probability is not uniformly distributed. Moreover, scISR also outperforms other methods in recovering the missing data by having the lowest median MAE and highest median correlation.

In the next scenario, we generate 40 new simulated datasets, in which the cells of the same cell type have high correlation. We use the same combinations of number of cells, dropout rates, and dropout distributions as in the second scenario (see Supplementary Section 4.2 for the details of the simulation). Supplementary Figure S20 shows the results obtained from the 40 new simulated datasets. scISR outperforms other methods by having the lowest mean absolute errors and highest correlations in every analysis performed.

In the last scenario, we perform additional simulation with negative binomial distribution as noise model using Splatter. We set the number of genes to 15,000 and the number of cell types to 3. We generated 30 datasets with different cell numbers: 5,000, 10,000, 25,000, 50,000, 100,000 and 200,000. For each sample size, we varied the sparsity levels by adjusting the *dropout.mid* parameters (midpoint parameter for dropout logistic function of Splatter). We set *dropout.mid* to 2.5, 3, 3.5, 4, and 4.5, which led to sparsity levels of 84%, 87%, 89%, 91%, 93%, respectively.

We used the mean absolute error (MAE) values and correlation coefficients between the ground truth expression and imputed expression data to assess the performance of imputation methods. Supplementary Figure S22 shows the results, in which scISR and scScope are the only methods that can perform imputation on the biggest dataset. MAGIC, SAVER, scImpute, and scGNN cannot analyze datasets with are more than 100,000, 10,000, 10,000, and 50,000 cells, respectively. Overall, MAGIC, SAVER, scScope, and scGNN are unable to correctly recover the missing values, which leads to MAE values that are even higher than the masked data (data without imputation). scImpute has good results in small datasets but is unable to impute datasets with more than 10,000 cells. Even in datasets with 10,000 cells, scImpute returns errors when the dropout rate increases (91% and 93%). In contrast, scISR is able to improve the quality of the dropout data in all scenarios. We also report the running time for these simulation studies in Supplementary Figure S23. scISR and scScope are the only methods that can perform imputation on dataset with 200,000 cells. Both methods can analyze the largest dataset with 200,000 cells in approximately 100 to 200 minutes. Other methods either run out of memory or are unable to finish in a reasonable amount of time, which was set to one day.

4.3.4 Robustness of scISR against non-uniform dropout probability

To further investigate the robustness of the hypergeometric test embedded in scISR, we have also performed additional simulation studies with different sample sizes and dropout scenarios. In these simulations, we know the ground truth and the underlying probability distributions. Therefore, we can properly assess the reliability of scISR when the dropout probability is not uniformly distributed.

First, we generate a new dataset that consists of 1,000 samples and 3,000 genes – all expression values follow a normal distribution $N(\mu, \sigma)$ where $\mu = 1$ and $\sigma = 0.15$. We slightly shift the mean of the cells and genes to create 4 different cell types. We name this as *complete data*. Next, we randomly assign dropout values to the data in two different scenarios. In the first scenario, the dropout probability is uniformly distributed. In the second scenario, the dropout probability follows a normal distribution. For example, at 60% dropout rate, the dropout probability follows a distribution of $N(0.6, 0.1)$. To make the simulation more general, we vary the number of cells (from 1,000 to 10,000), the number of genes (from 3,000 to 10,000), and the dropout rate (from 60% to 90%). We name the data with dropouts as *masked data*. Next, we impute the *masked data* using six imputation methods to obtain the *imputed data*. Finally, to assess the performance of imputation methods, we compare the imputed data against the complete data using Mean Absolute Error (MAE) and correlation coefficients.

The top left panel in Figure 4.13 shows the MAE values obtained for datasets with 1,000 cells and 3,000 genes. In this panel, the left side displays the results obtained for uniform distributions while the right side shows the results for the normal distributions. When the dropout probability is uniformly distributed, scISR is able to recover most of the dropout values, resulting in a median MAE close to zero at any dropout rate. When the dropout probability is normally distributed, scISR still performs as well at 60% to 80% dropout but it becomes less accurate at 90% rate. At 90% dropout rate, scISR recovers only a part of the data (median MAE of approximately 2.11 compared to 3.65 of masked data). Assessment results using correlation coefficient (top right panel) also confirm our finding. However, as seen in Figure 4.13, the result of scISR is still much better than other imputation methods.

The next two panels (second row) in Figure 4.13 show the results obtained for datasets with 3,000 cells and 9,000 genes. scISR is more accurate (lower MAE and higher correlation) for these datasets compared to datasets with 1,000 cells. At dropout rates of 60%, 70%, and 80%, scISR performs consistently well for uniform and normal distributions alike (median MAE value close to zero). At 90% rate, the median MAE of scISR for normal distributions is now 1.61 (compared to 2.11 for datasets with 1,000 cells and 3,000 genes). The reason for such improvement is that with the same dropout rate, larger datasets provide us with more data to learn from, leading to improved hypothesis testing (hypergeometric test) and prediction (linear regression). For datasets with 7,000 cells or more, the median MAE is close to zero for both uniform and normal distributions at any dropout rate. In summary, scISR (using hypergeometric test) performs well for large datasets with high dropout rates even when the dropout probability is not uniformly distributed. Moreover, similar to dataset with 1,000 cells, scISR also outperforms other methods in recovering the missing values in bigger datasets.

Next, we investigate performance of scISR using simulated datasets in which the cells of the same cell type have high correlation. Denote m as the number of genes. We first generated four different vectors: i) $(\frac{0}{m}, \frac{1}{m}, \ldots, \frac{m}{m})$, ii) $(\frac{0}{m}, -\frac{1}{m}, \ldots, -\frac{m}{m})$ iii) $(\frac{0}{m}, \frac{2}{m}, \ldots, \frac{m}{m}, \frac{0}{m}, \frac{2}{m}, \ldots, \frac{m}{m})$, and iv) $(\frac{0}{m}, -\frac{2}{m}, \ldots, -\frac{m}{m}, \frac{0}{m}, -\frac{2}{m}, \ldots, -\frac{m}{m})$. Each vector was used to simulated a cell type. Instead of shifting the mean of a cell type, we added the first vector to the expression of the first cell type. Similarly, we added the second, third, and fourth vectors to the second, third, and fourth cell types, respectively. By doing so, cells of the same type will have high correlation. Similar to the above simulation, we added dropouts with various rates (60%, 70%, 80%, and 90%) and distributions (uniform and normal). We also simulated datasets of different numbers of cells: 1,000, 3,000, 5,000, 7,000, and 10,000.

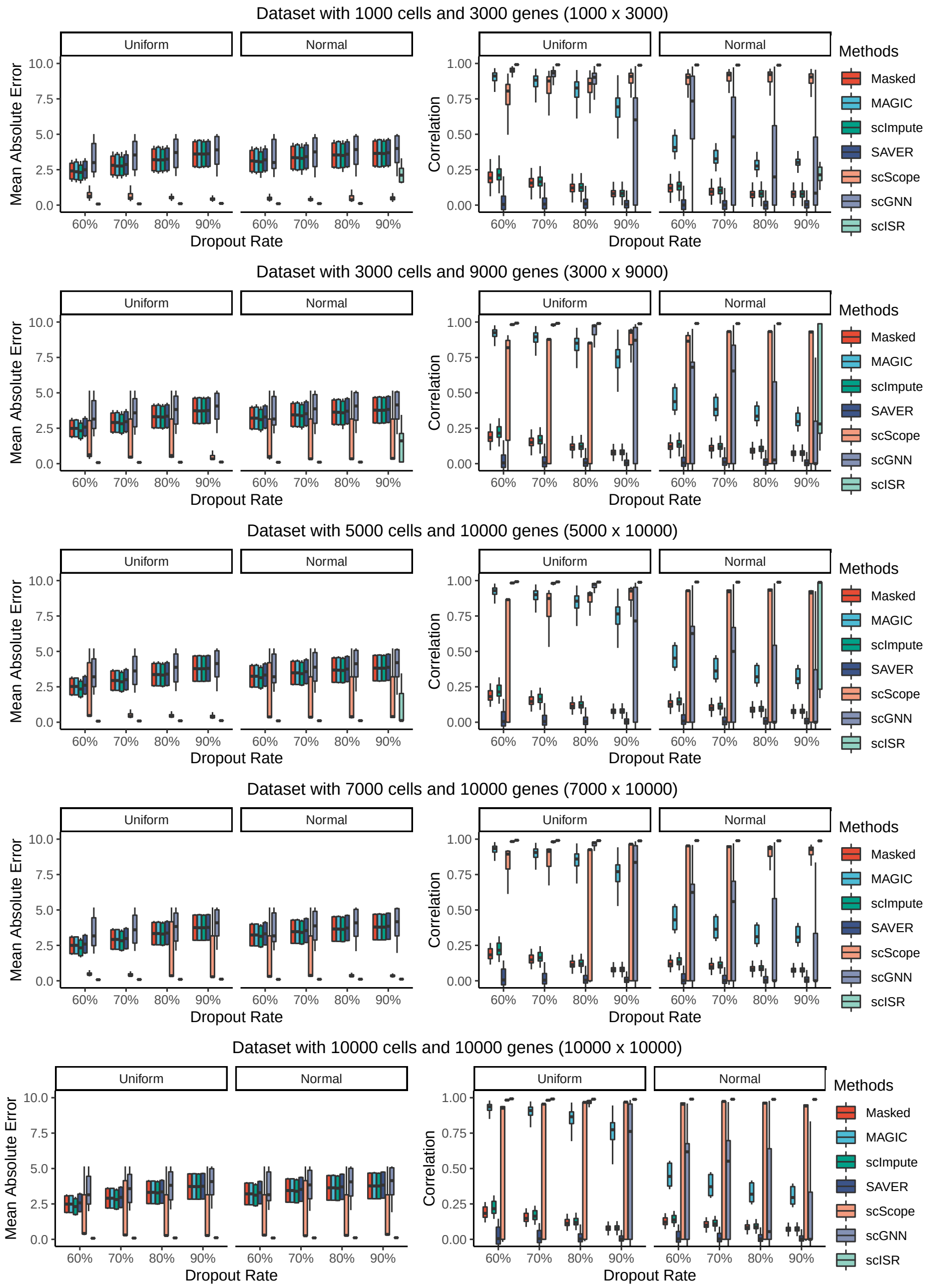

Figure 4.13: Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using simulated datasets with different dropout distributions and sample sizes. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. In each panel, the left side shows the results for uniform distributions while the right side shows the results for normal distributions. For small datasets (e.g., datasets with 1,000 cells) with high dropout rates, scISR is less accurate when the dropout probability is normally distributed. When the sample size increases, scISR becomes more accurate. For datasets with 7,000 cells or more, scISR performs well for both uniform and normal distributions alike across all dropout rates. For most of the dataset sizes and dropout rates, scISR have a much better median MAE and correlation compared to other methods.

Figure 4.14 shows the results obtained from the 40 new simulated datasets. We also used the same metrics to assess the similarity between imputed and the complete data: (1) mean absolute error (the smaller the better), and (2) correlation (the higher the better). scISR outperforms other methods by having the lowest mean absolute errors and highest correlations in every analysis performed.

To measure the accuracy of the hyper-geometric test as a standalone module, we compared the altered zero values against the ground truth (in which we know which zero is true zero and which is dropout). We define the following terms: 1) TP (a dropout value is altered by scISR), 2) FN (a dropout value not altered), 3) FP (a true zero value altered), and 4) TN (a true zero value not altered). For assessment purpose, we used the F-score to measure the accuracy of the hypothesis testing. Note that F-score is calculated based on precision and recall: $F1 = 2 \times \frac{precision \times recall}{precision + recall}$ or $F1 = \frac{TP}{TP + \frac{1}{2}(FP+FN)}$. In the ideal case, F-score equals to 1 if both precision and recall equal to 1 (i.e., $FP = FN = 0$). Figure 4.15 shows the F-score values obtained from the 40 simulated datasets (5 cell numbers × 4 dropout rates × 2 distributions). When the dropout probability is uniformly distributed, the median F-scores are close to 1 in all settings. When the dropout probability is normally distributed, the median values are less than 1 for small datasets with high dropout rates. However, as the sample size increases, the results improve. For datasets with 7,000 cells or more, the median F-scores are close to 1 for both uniform and normal distributions.

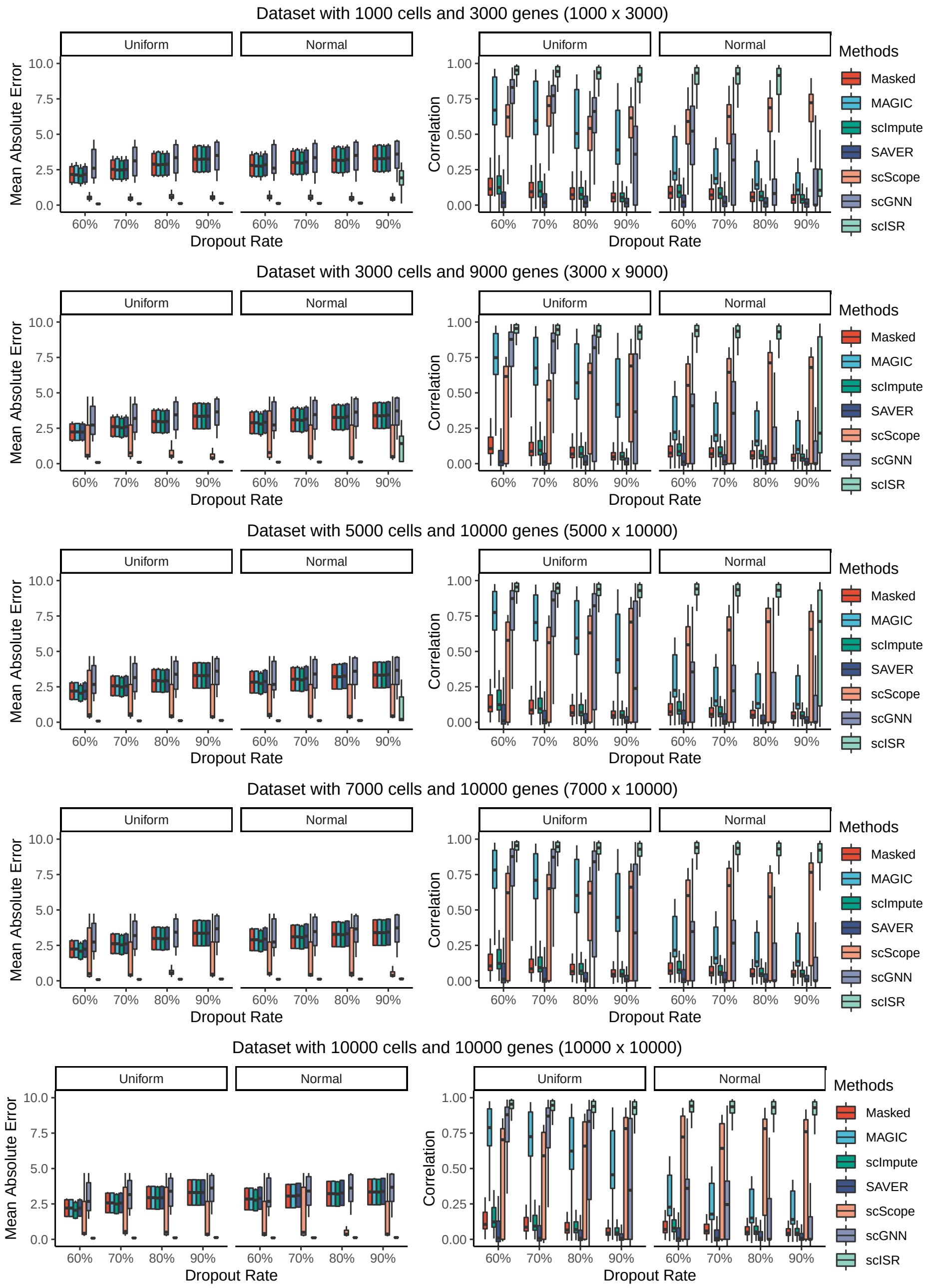

Figure 4.14: Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using simulated datasets with different dropout distributions and sample sizes. In each dataset, cells of the same type have high correlation and cells of different types have low correlation. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. In each panel, the left side shows the results for uniform distributions while the right side shows the results for normal distributions. For small datasets (e.g., datasets with 1,000 cells) with high dropout rates, scISR is less accurate when the dropout probability is normally distributed. When the sample size increases, scISR becomes more accurate. For datasets with 7,000 cells or more, scISR performs well for both uniform and normal distributions alike across all dropout rates. For most of the dataset sizes and dropout rates, scISR have a much better median MAE and correlation compared to other methods.

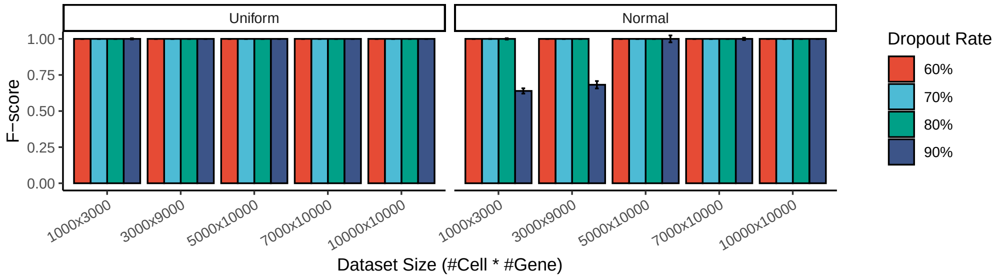

Figure 4.15: The accuracy of scISR hypothesis testing using F-score. The F-score measures how well the algorithm distinguish between true zero values and dropouts. The left panel shows the F-scores for datasets with uniform distribution while the right panel shows the F-scores for datasets with normal distribution. For datasets with 7,000 cells or more, the median F-scores are close to 1 for both uniform and normal distributions alike across all dropout rates. In other words, scISR accurately identifies the zero values that need to be imputed.

4.3.5 Simulation studies using Splatter package

Using Splatter R package [124], we perform additional simulation with negative binomial distribution as noise model. We set the number of genes to 15,000 and the number of cell types to 3. We generated 30 datasets with different cell numbers: 5,000, 10,000, 25,000, 50,000, 100,000 and 200,000. For each sample size, we varied the sparsity levels by adjusting the *dropout.mid* parameters (midpoint parameter for dropout logistic function of Splatter). We set *dropout.mid* to 2.5, 3, 3.5, 4, and 4.5, which led to sparsity levels of 84%, 87%, 89%, 91%, 93%, respectively. Both sample size (hundreds of thousands of cells) and dropout rates (84%–93%) are often expected from current scRNA-seq datasets. In total, we simulated 30 new datasets using Splatter (6 cell numbers $\times$ 5 sparsity levels).

We used the mean absolute error (MAE) values and correlation coefficients between the ground truth expression and imputed expression data to assess the performance of imputation methods. Figure 4.16 shows the results, in which scISR and scScope are the only methods that can perform imputation on the biggest dataset. MAGIC, SAVER, scImpute, and scGNN cannot analyze datasets with are more than 100,000, 10,000, 10,000, and 50,000 cells, respectively. For large datasets, these methods either returned error, ran out of memory (the memory limit on our machine is 128 GB), or could not finish the analysis in a reasonable amount of time (more than one day).

Overall, MAGIC, SAVER, scScope, and scGNN are unable to correctly recover the missing values, which leads to MAE values that are even higher than the masked data (data without imputation). scImpute has good results in small datasets but is unable to impute datasets with more than 10,000 cells. Even in datasets with 10,000 cells, scImpute returns errors when the dropout rate increases (91% and 93%). In contrast, scISR is able to improve the quality of the dropout data in all scenarios.

We also report the running time for these simulation studies. As seen in Figure 4.17, scISR and scScope are the only methods that can perform imputation on dataset with 200,000 cells. The reason scScope can analyze the biggest dataset in this simulation is because the number of genes is set to 15,000, which is lower than that of real datasets. Both methods can analyze the largest dataset with 200,000 cells in approximately 100 to 200 minutes. Other methods either run out of memory or are unable to finish in a reasonable amount of time, which was set to one day.

4.3.6 Robustness of scISR against batch effect

To investigate the effect of batch effect on scISR, we tested our approach using simulated datasets generated by Splatter package. We used the following parameters: the number of genes is set to 15,000; 5 sparsity levels are generated with zero ratios ranging from 84% to 93%; the number of cells is fixed to 25,000; batch effect is either enable or disable. Splatter simulates batch effect by generating a small scaling factor for each gene in each batch. We generated a total of 10 datasets using these parameters. As seen in Figure 4.18, batch effects do not have a significant impact on the performance of scISR.

4.4 Results of clustering analysis using scCAN

In this section, we assess the performance of scCAN in the following capabilities: (1) correct estimation of the number of cell types, (2) proper segregation of cells of different types, (3) robustness against dropout events, and (4) scalability against the increasing number of cell types. For this purpose, we analyze 28 real scRNA-seq datasets and simulation in various scenarios. We compare scCAN with five state-of-the-art clustering methods that are widely used for single-cell analysis: CIDR [21],

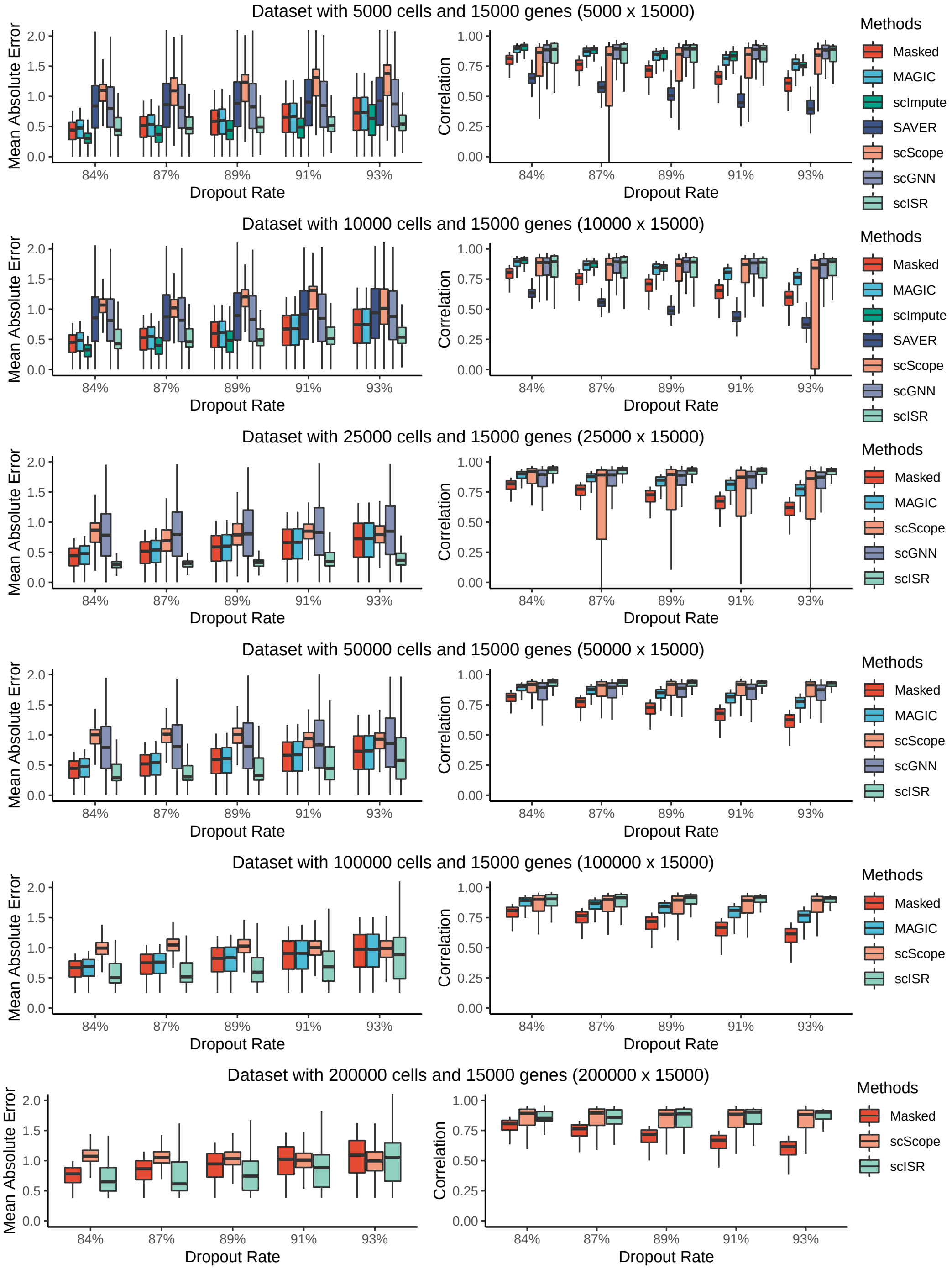

Figure 4.16: Assessment of MAGIC, scImpute, SAVER, scScope, scGNN, and scISR using datasets simulated by Splatter. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. scISR and scScope are the only methods that can perform imputation on the biggest dataset, while MAGIC, SAVER, scImpute, and scGNN stop working with datasets bigger than 100,000, 10,000, 10,000, and 50,000 cells, respectively. scISR is the only method that can improve the dropout data in all scenarios.

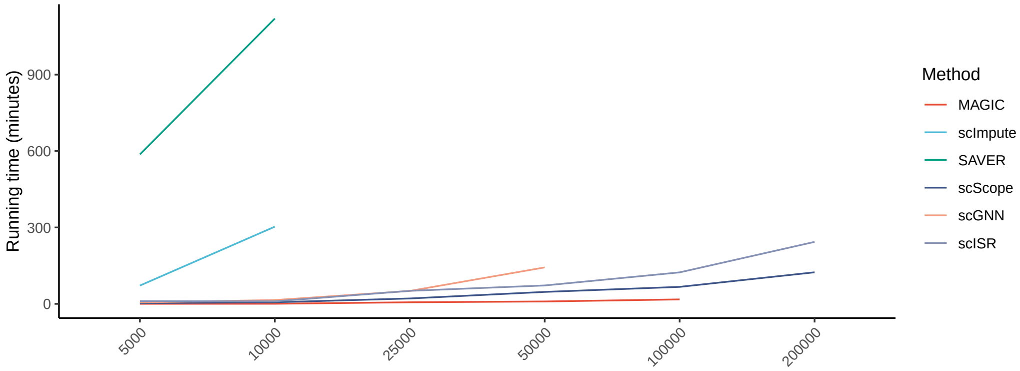

Figure 4.17: Running time of the six imputation methods on simulated datasets. These datasets have 15,000 cells and varying number of cells (5,000 to 200,000). scISR and scScope are the only methods that can analyze all datasets. The two methods can finish the analysis of 200,000 cells in 200 and 100 minutes, respectively.

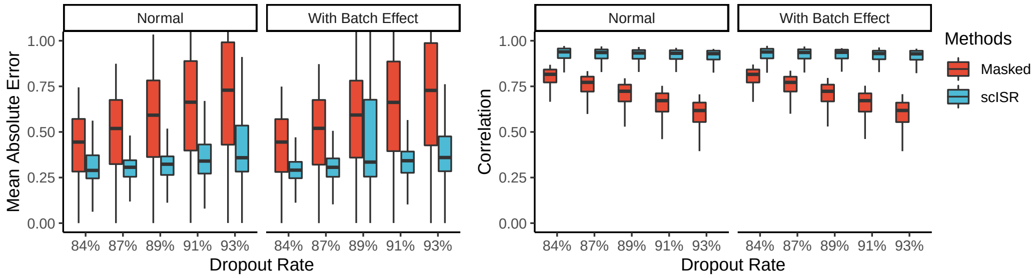

Figure 4.18: Impact of batch effects on scISR. The left panels show the Mean Absolute Error (MAE) values while the right panels show the correlation coefficients. In each panel, the left 10 boxes show the results for data without batch effects while the right 10 boxes show the results for data with batch effects. Overall, batch effects do not have a significant impact on the performance of scISR.

SEURAT3 [69], Monocle3 [71], SHARP [55], and SCANPY [70].

4.4.1 Estimating the number of true cell types

We use CIDR [21], SEURAT3 [69], Monocle3 [71], SHARP [55], SCANPY [70], and scCAN to partition each of the 27 real scRNA-seq datasets. To evaluate how well each method estimates the number of cell types, we compare the number of clusters produced by each method against the number of true cell types using the absolute log-modulus [143]: $L(x) = |sign(x) * log10(|x| + 1)|$ where x is the difference between

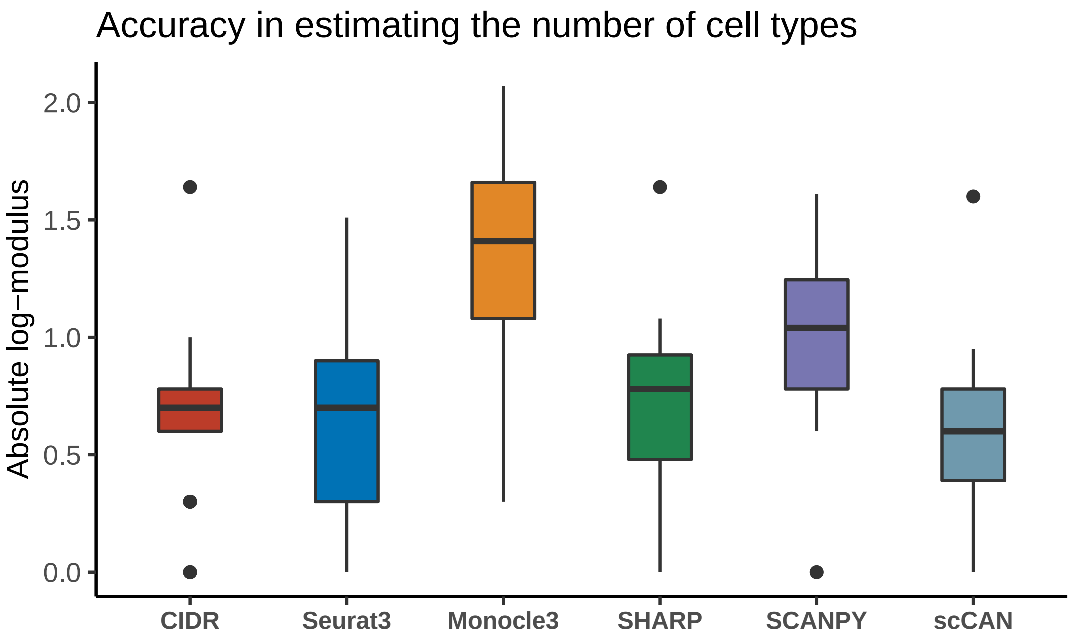

Figure 4.19: Absolute log-modulus values obtained from CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN for 27 real scRNA-seq datasets. This metric measures the difference between the number of clusters and the number of true cell types. The average log modulus of scCAN is 0.59 while those of Monocle3, SCANPY, SHARP, SEURAT3, and CIDR are 1.35, 1, 0.72, 0.64, and 0.63, respectively. scCAN significantly outperforms other methods by having the smallest absolute log-modulus values (Wilcoxon p-value of $p = 8.6 \times 10^{-4}$). Note that the dataset Brain 1.3M was excluded from this analysis because it does not have true cell type information.

the number of clusters and the number of cell types. The lower the $L(x)$ value, the more similar the number of clusters and the true number of cell types. $L(x)$ equals to zero denotes a perfect estimation.

Figure 4.19 shows the absolute log-modulus values obtained using the six clustering methods. Each box represents the absolute log-modulus values across 27 scRNA-seq datasets for a method. We observe that Monocle3 and SCANPY frequently overestimate the number of clusters. Both methods have the highest absolute log-modulus values. Overall, scCAN is the best method in estimating the number of true cell types. The average log modulus of scCAN is 0.59 whereas those of Monocle3, SCANPY, SHARP, SEURAT3, and CIDR are 1.35, 1, 0.72, 0.64, and 0.63, respectively. A one-sided Wilcoxon test also confirms that the absolute log-modulus values obtained from scCAN are significantly smaller than other methods with a p-value of 9×10^{-4}. We report the absolute log-modulus values for each method and each dataset in Table 4.8.

4.4.2 Segregating cells of different types

To assess the accuracy of each clustering method, we also compare the clustering results against the true cell labels. For this purpose, we use three evaluation metrics: adjusted Rand index (ARI) [140], adjusted mutual information (AMI) [141], and V-measure [142]. Details of each metric are provided in Supplementary Section 3.

Figure 4.20A shows the ARI values obtained from the six clustering methods. Each box represents the ARI values across 27 datasets for a method. The results show that scCAN significantly outperforms other state-of-the-art methods by having the highest ARI values ($p = 6 \times 10^{-12}$ using Wilcoxon test). The average ARI values of scCAN is 0.81 which is substantially higher than those of other methods (0.50, 0.55, 0.23, 0.41, and 0.40 for CIDR, Seurat3, Monocle3, SHARP, and SCANPY, respectively). More importantly, scCAN has the highest ARI values in 24 out of 27 datasets. The details can be seen in Table4.9

Figure 4.20B shows the AMI values of each method. The AMI values of scCAN are significantly higher than those of other methods ($p = 9 \times 10^{-10}$ using Wilcoxon test). The average AMI value of scCAN is 0.77 while the average AMI values of CIDR, Seurat3, Monocle3, SHARP, and SCANPY are 0.52, 0.64, 0.43, 0.41 and 0.55, respectively. scCAN also has the highest AMI values in 23 out of 27 datasets. The details can be seen in Table 4.10

Figure 4.20C shows a similar trend using V-measure. The V-measure values of scCAN are significantly higher than those of other methods ($p = 2 \times 10^{-8}$). The

Table 4.8: Estimation of the number of cell types of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by absolute log-modulus values. Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). The average absolute log-modulus value of scCAN is 0.59, which are smaller than the rest.

Datasets	**CIDR**	**Seurat3**	**Monocle3**	**SHARP**	**SCANPY**	**scCAN**
Pollen	0.60	0.30	0.30	0.70	0.60	0.60
Patel	0.30	0.30	0.60	0.60	0.70	0.00
Wang	0.00	0.00	0.78	0.60	0.60	0.48
Li	0.78	0.48	0.30	1.00	0.00	0.48
Usoskin	0.30	0.48	0.90	0.30	1.04	0.00
Camp	0.60	0.48	1.00	0.00	0.78	0.78
Xin	0.85	0.30	1.08	0.48	0.60	0.60
Muraro	0.70	0.90	1.26	0.48	1.18	0.70
Segerstolpe	0.60	0.95	1.51	0.00	1.41	0.70
Romanov	0.00	0.90	1.32	0.48	1.26	0.30
Zeisel	0.70	0.95	1.40	0.78	1.30	0.30
Lake	0.70	0.00	1.40	0.95	1.04	0.30
Montoro	0.78	0.30	1.41	1.04	0.60	0.90
Guo	0.30	0.78	1.52	0.85	1.23	0.30
Baron	0.70	0.70	1.57	0.85	1.15	0.30
Chen	1.00	0.30	1.58	0.90	1.11	0.78
Slyper	0.60	0.85	1.58	0.48	1.08	0.48
Kanton	0.78	0.30	1.66	1.08	0.85	0.78
Brann	1.64	1.51	1.20	1.64	1.51	1.60
Zilionis	0.70	1.00	1.68	0.60	1.08	0.48
Macosko	0.60	0.90	1.78	0.78	0.78	0.70
Hrvatin	NA	0.78	1.83	0.90	1.00	0.48
Orozco	NA	1.28	2.07	1.08	1.61	0.85
Miller	NA	NA	2.03	NA	1.32	0.95
Darrah	NA	NA	1.93	NA	1.36	0.48
Kozareva	NA	NA	NA	NA	0.85	0.95
Cao	NA	NA	NA	NA	1.04	0.60
Mean	0.63	0.64	1.35	0.72	1.00	0.59

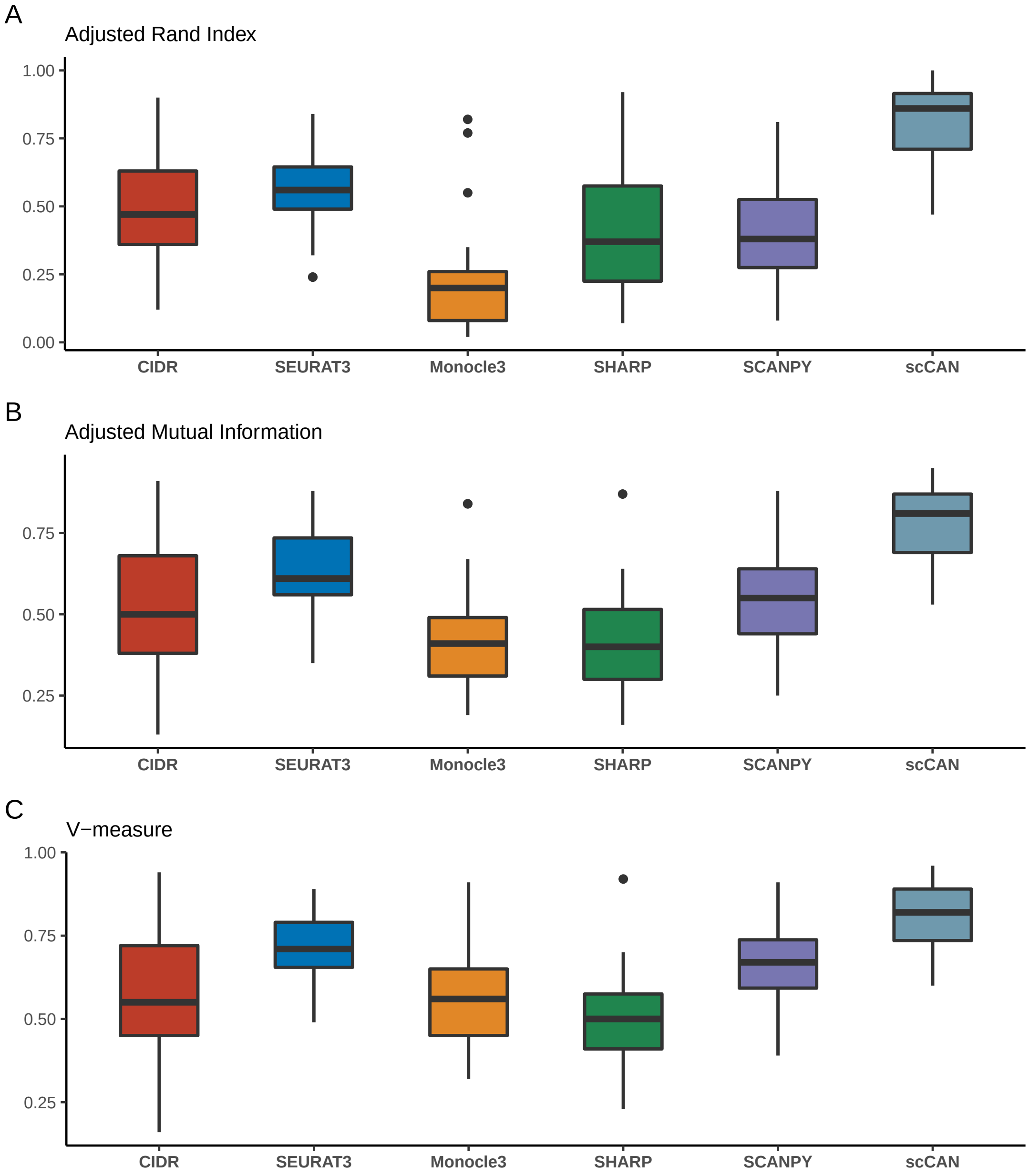

Figure 4.20: Accuracy assessment of the six clustering methods using adjusted Rand index (ARI), adjusted mutual information (AMI), and V-measure. scCAN consistently and substantially outperforms other methods in every assessment by having the highest ARI, AMI, and V-measure values across 27 real scRNA-seq datasets.

Table 4.9: Performance of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by Adjusted Rand Index (ARI). Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). Cells highlighted in bold have the highest ARI values. The average ARI of scCAN is 0.81, which is much higher than the rest (SEURAT3 is the second best with an average ARI of 0.55). In addition, scCAN has the highest ARI values in all but three datasets (Camp, Montoro and Hrvatin).

Dataset	#Cells	CIDR	SEURAT3	Monocle3	SHARP	SCANPY	scCAN
Pollen	301	0.90	0.73	0.82	0.09	0.77	**0.92**
Patel	430	0.45	0.82	0.26	0.09	0.66	**0.86**
Wang	457	0.63	0.56	0.28	0.41	0.62	**0.83**
Li	561	0.62	0.84	0.77	0.19	0.81	**0.94**
Usoskin	622	0.82	0.56	0.35	0.07	0.34	**0.93**
Camp	777	0.61	**0.65**	0.55	0.44	0.61	0.61
Xin	1,600	0.57	0.50	0.15	0.56	0.29	**0.98**
Muraro	2,126	0.22	0.64	0.30	0.31	0.43	**0.91**
Segerstolpe	2,209	0.37	0.60	0.20	0.33	0.31	**0.95**
Romanov	2,881	0.32	0.48	0.19	0.59	0.30	**0.63**
Zeisel	3,005	0.37	0.50	0.24	0.46	0.32	**0.86**
Lake	3,042	0.47	0.51	0.23	0.21	0.43	**0.58**
Montoro	7,193	0.30	0.24	0.08	**0.80**	0.20	0.70
Guo	7,416	0.75	0.62	0.23	0.24	0.46	**0.86**
Baron	8,569	0.73	0.56	0.21	0.36	0.46	**0.94**
Chen	12,089	0.36	0.69	0.25	0.59	0.62	**0.72**
Slyper	13,316	0.63	0.24	0.06	0.39	0.26	**0.67**
Kanton	17,542	0.47	0.40	0.13	0.31	0.47	**0.67**
Brann	26,766	0.12	0.32	0.06	0.76	0.32	**0.86**
Zilionis	34,558	0.53	NA	0.12	0.37	0.38	**0.89**
Macosko	44,808	0.17	NA	0.04	0.71	0.23	**0.85**
Hrvatin	48,266	NA	NA	0.13	**0.92**	0.57	0.78
Orozco	100,055	NA	NA	0.04	0.20	0.22	**0.77**
Miller	142,523	NA	NA	0.04	NA	0.16	**0.90**
Darrah	162,490	NA	NA	0.02	NA	0.08	**0.47**
Kozareva	611,034	NA	NA	NA	NA	0.12	**1.00**
Cao	1,092,000	NA	NA	NA	NA	0.48	**0.89**
Mean		0.50	0.55	0.23	0.41	0.40	**0.81**

Table 4.10: Performance of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by Adjusted Mutual Information (AMI). Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). Cells highlighted in bold have the highest AMI values. The average AMI of scCAN is 0.77, which is much higher than the rest (SEURAT3 is the second best with an average AMI of 0.64). In addition, scCAN has the highest AMI values in all but four datasets (Camp, Montoro, Chen and Hrvatin).

Dataset	**#Cells**	**CIDR**	**SEURAT3**	**Monocle3**	**SHARP**	**SCANPY**	**scCAN**
Pollen	301	0.91	0.80	0.84	0.20	0.88	**0.93**
Patel	430	0.55	0.77	0.29	0.16	0.64	**0.84**
Wang	457	0.66	0.60	0.42	0.40	0.64	**0.75**
Li	561	0.69	0.88	0.84	0.27	0.84	**0.95**
Usoskin	622	0.76	0.61	0.48	0.19	0.48	**0.88**
Camp	777	0.72	**0.77**	0.67	0.59	0.72	0.72
Xin	1,600	0.51	0.57	0.35	0.50	0.44	**0.91**
Muraro	2,126	0.41	0.72	0.53	0.31	0.60	**0.87**
Segerstolpe	2,209	0.42	0.72	0.47	0.33	0.55	**0.88**
Romanov	2,881	0.33	0.55	0.37	0.52	0.44	**0.61**
Zeisel	3,005	0.38	0.58	0.46	0.46	0.50	**0.81**
Lake	3,042	0.47	0.65	0.53	0.22	0.67	**0.74**
Montoro	7,193	0.35	0.35	0.25	**0.64**	0.33	0.58
Guo	7,416	0.76	0.71	0.49	0.51	0.59	**0.87**
Baron	8,569	0.65	0.69	0.49	0.40	0.64	**0.87**
Chen	12,089	0.37	0.75	0.59	0.52	**0.75**	0.55
Slyper	13,316	0.68	0.46	0.31	0.30	0.46	**0.73**
Kanton	17,542	0.49	0.53	0.39	0.30	0.57	**0.64**
Brann	26,766	0.13	0.53	0.33	0.52	0.54	**0.72**
Zilionis	34,558	0.50	NA	0.40	0.41	0.53	**0.84**
Macosko	44,808	0.27	NA	0.26	0.41	0.42	**0.66**
Hrvatin	48,266	NA	NA	0.41	**0.87**	0.64	0.76
Orozco	100,055	NA	NA	0.29	0.32	0.43	**0.65**
Miller	142,523	NA	NA	0.23	NA	0.33	**0.82**
Darrah	162,490	NA	NA	0.19	NA	0.25	**0.53**
Kozareva	611,034	NA	NA	NA	NA	0.39	**0.94**
Cao	1,092,000	NA	NA	NA	NA	0.61	**0.84**
Mean		0.52	0.64	0.43	0.41	0.55	**0.77**

average V-measure value of scCAN is 0.81 while the average AMI values of CIDR, Seurat3, Monocle3, SHARP, and SCANPY are 0.57, 0.72, 0.56, 0.50 and 0.66, respectively. scCAN also has the highest V-measure values in 23 out of 27 datasets. The details can be seen in 4.11.

The visualizations of cell transcriptomic landscape for 27 datasets using original cell types and cluster assignments generated by scCAN are shown at Supplementary Figures S1–S5 and Supplementary Figures S6–S10 available from [147].

Robustness against dropouts

One of the prominent challenges in single-cell data analysis is the prevalence of dropouts. To assess how robust each method is against dropouts, we simulate a number of datasets. There are a number of tools that generate simulated data, including Splatter [124] and SymSim [148]. Though powerful, these tools cannot completely emulate real-world situations. The simulators do not preserve expression levels and gene correlation structure of real genes [149]. Therefore, instead of generating completely new expression values, we simulate different dropout scenarios using the 27 real datasets listed above. For each dataset, we gradually increase the number of dropouts by randomly replace non-zero expression values with zeros. The dropout rates are set to 50%, 55%, 60%, 65%, 70%, 75%, 80%, 85% and 90%. In summary, we generate 243 simulated datasets (27 real datasets with 9 different dropout rates per dataset).

For each dataset, the true cell label of each cell is known and thus can be used *a posteriori* to assess the robustness of each clustering method. We analyze each of the 243 datasets using the six clustering methods and then calculate the ARI values. Figure 4.21 shows the ARI values for each method across datasets of varying dropout rates. Overall, scCAN consistently outperforms other methods in clustering

Table 4.11: Performance of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN on 27 single-cell datasets measured by V-measure. Cells with NA values indicate that the method was not able to analyze the dataset (crashed or out-of-memory). Cells highlighted in bold have the highest V-measure values. The average V-measure of scCAN is 0.81, which is much higher than the rest (SEURAT3 is the second best with an average V-measure of 0.72). In addition, scCAN has the highest V-measure values in all but four datasets (Romanov, Montoro, Chen and Kanton).

Dataset	#Cells	CIDR	SEURAT3	Monocle3	SHARP	SCANPY	scCAN
Pollen	301	0.94	0.89	0.91	0.33	0.91	**0.96**
Patel	430	0.57	0.79	0.33	0.26	0.72	**0.84**
Wang	457	0.71	0.65	0.52	0.53	0.72	**0.81**
Li	561	0.77	0.89	0.90	0.41	0.87	**0.96**
Usoskin	622	0.80	0.71	0.62	0.23	0.63	**0.93**
Camp	777	0.79	**0.82**	0.79	0.66	**0.82**	**0.82**
Xin	1,600	0.55	0.68	0.50	0.50	0.58	**0.92**
Muraro	2,126	0.43	0.79	0.66	0.46	0.72	**0.87**
Segerstolpe	2,209	0.45	0.77	0.62	0.42	0.69	**0.92**
Romanov	2,881	0.34	**0.66**	0.49	0.56	0.58	0.62
Zeisel	3,005	0.47	0.67	0.60	0.59	0.63	**0.82**
Lake	3,042	0.54	0.69	0.63	0.35	0.73	**0.75**
Montoro	7,193	0.46	0.49	0.38	**0.70**	0.47	0.65
Guo	7,416	0.79	0.81	0.65	0.52	0.73	**0.89**
Baron	8,569	0.72	0.77	0.65	0.55	0.76	**0.89**
Chen	12,089	0.42	**0.78**	0.69	0.65	0.77	0.60
Slyper	13,316	0.70	0.59	0.45	0.42	0.59	**0.73**
Kanton	17,542	0.49	0.60	0.52	0.41	**0.65**	0.64
Brann	26,766	0.16	0.64	0.48	0.65	0.65	**0.80**
Zilionis	34,558	0.58	NA	0.56	0.52	0.65	**0.89**
Macosko	44,808	0.33	NA	0.41	0.49	0.56	**0.70**
Hrvatin	48,266	NA	NA	0.58	**0.92**	0.78	0.82
Orozco	100,055	NA	NA	0.44	0.41	0.60	**0.75**
Miller	142,523	NA	NA	0.37	NA	0.49	**0.88**
Darrah	162,490	NA	NA	0.32	NA	0.39	**0.63**
Kozareva	611,034	NA	NA	NA	NA	0.56	**0.96**
Cao	1,092,000	NA	NA	NA	NA	0.74	**0.90**
Mean		0.57	0.72	0.56	0.50	0.67	**0.81**

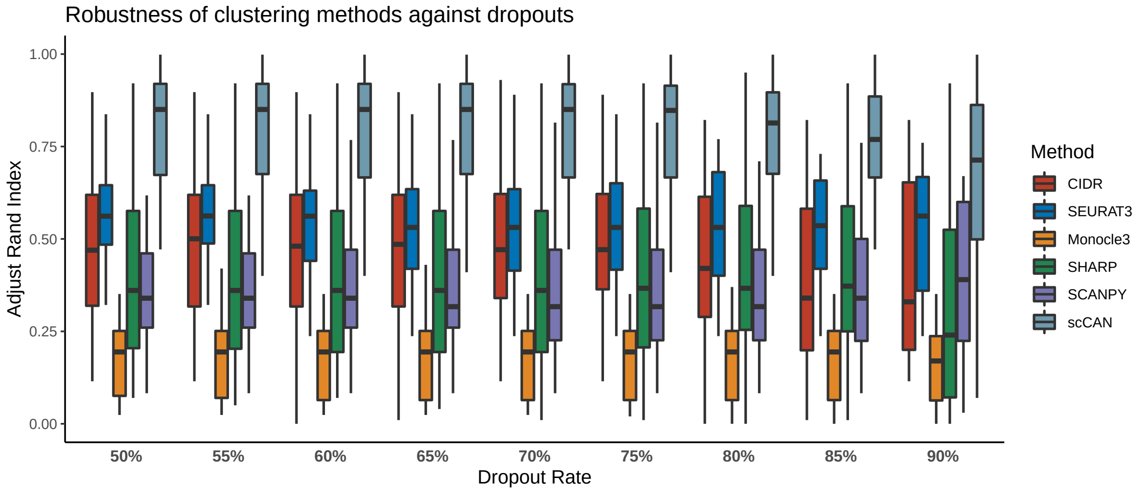

Figure 4.21: Assessment of CIDR, SEURAT3, Monocle3, SHARP, SCANPY and, scCAN against dropouts. Simulations were obtained by varying the number of zeros in each of 27 real biological datasets from 50% to about 90%, respectively. Each box plot shows the ARI values obtained from each method for a specific dropout portion. Wilcoxon test shows that the ARI values obtained from scCAN are significantly higher than CIDR, SEURAT3, Monocle3, SHARP, SCANPY ($p < 2.2 \times 10^{-16}$).

cell populations regardless of dropout rates. A one-sided Wilcoxon test also confirms that the ARI values obtained from scCAN are significantly higher than those of CIDR, SEURAT3, Monocle3, SHARP, SCANPY ($p < 2.2 \times 10^{-16}$).

4.4.3 Time and space complexity

In order to assess the scalability of the clustering methods, we record the running time that each method uses to analyze the 28 real datasets. Figure 4.22 shows the running time of the methods with varying numbers of cells. The time complexity of CIDR increases exponentially with respect to sample size. Supplementary Table S7 shows the detailed running time of each method for all 28 datasets. The cell with “NA” indicates out of memory or error. The memory of our machine is limited to 256 GB. scCAN and SCANPY can cluster all datasets in minutes.

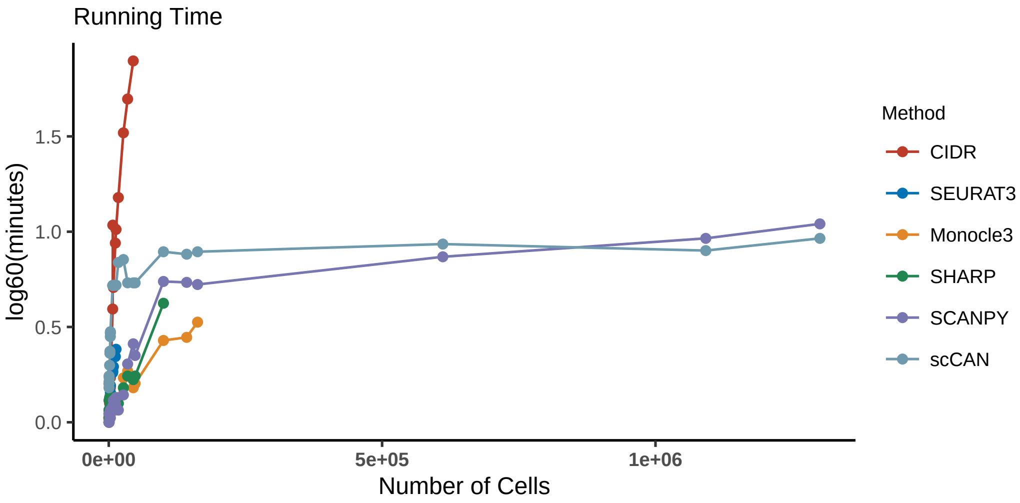

Figure 4.22: Running time of CIDR, SEURAT3, Monocle3, SHARP, SCANPY, and scCAN for the analysis of 28 real scRNA-seq datasets. The horizontal axis shows the number of cells while the vertical axis shows the running time in the log scale (base 60) of minutes. scCAN and SCANPY are the only two methods that can analyze datasets with more than 200,000 cells.

4.4.4 Comparison of the clustering methods used in Modules 2 and 3

The first method (core method) is more accurate but it requires more computational power and memory. Therefore, we developed the second method that allows users to analyze large datasets faster and using less memory. If the input dataset is small (by default 5,000 cells or less), both methods will be the same and thus produce the same results. When the dataset is large (5,000 cells or more), we use the first method to analyze a subset of the data to determine the cell types and then assign the the remaining cells to the determined cell types (second method).

Note that the default value of 5,000 allows us to have a sufficiently large sample size to properly determine the cell types which in turns will lead to a proper classification of the remaining cells. At the same time, 5,000 is a reasonable small number of samples that allows users to perform the analysis efficiently using personal computers. Users can also change this parameter to use the first method even for large datasets, if they have more memory and are willing to wait longer for their results. In the following text, as requested, we will provide a direct comparison between the two methods in terms of both accuracy and running time.

Table 4.12 shows a direct comparison of the two methods in terms of both accuracy and running time using the same server (with 200 GB of RAM). Consistent with the previous submission, we used adjusted Rand index (ARI), adjusted mutual information (AMI), and V-measure to assess the performance of each method. Cells with NA values indicate that a method was not able to analyze the dataset (out-of-memory). Cells highlighted in bold have the higher accuracy (ARI, AMI, and V-measure) and lower running time.

Overall, the first method can only analyze the first 21 datasets. It returns NA for the last seven datasets with 44,808 cells or more (out of memory). The second method can analyze all datasets, even for the Cao dataset with more than a million cells.

Regarding running time, the second method is substantially faster than the first method. For example, the second method was able to analyze the Zilionis dataset in 18 minutes while it takes the first method method almost 3 days. For the Cao dataset with a million cells, the second method finished the analysis in less than 40 minutes whereas the first method ran out of memory and could not analyze the data.

Regarding the accuracy, the first method is slightly more accurate (when they can analyze the data) but the difference between the two methods is marginal. For example, the first method has a higher ARI in three dataset (Guo, Chen, and Slyper) but has lower ARI in three other datasets (Montoro, Kanton, and Zilionis). Similarly, the two methods have comparable AMI and V-measure values.

In summary, the first method is slightly more accurate but the second method is capable of analyzing large datasets and requires less memory and running time. Therefore, the scCAN software automatically switches to the second method when analyzing datasets with 5,000 cells or more. Users can adjust this parameter if they wish to run the first method for larger datasets, given that they have sufficient memory and are willing to wait longer for the results.

Table 4.12: Performance of the two clustering methods used in Module 2 (method 1) and Module 3 (method 2) on single-cell datasets measured by adjusted Rand index (ARI), adjusted mutual information (AMI), V-measure and running time (minutes). Cells with NA values indicate that the method was not able to analyze the dataset (out-of-memory). Cells highlighted in bold have the higher accuracy (ARI, AMI, and V-measure) or lower running time.

Datasets	**#Cells**	**ARI**		**AMI**		**V-measure**		**Running Time**	
		Method 1	**Method 2**	**Method 1**	**Method 2**	**Method 1**	**Method 2**	**Method 1**	**Method 2**
Pollen	301	0.92	0.92	0.93	0.93	0.96	0.96	1.3	1.3
Patel	430	0.86	0.86	0.84	0.84	0.84	0.84	1.1	1.1
Wang	457	0.83	0.83	0.75	0.75	0.81	0.81	1.3	1.3
Li	561	0.94	0.94	0.95	0.95	0.96	0.96	1.7	1.7
Usoskin	622	0.93	0.93	0.88	0.88	0.93	0.93	1.4	1.4
Camp	777	0.61	0.61	0.72	0.72	0.82	0.82	1.6	1.6
Xin	1,600	0.98	0.98	0.91	0.91	0.92	0.92	2.4	2.4
Muraro	2,126	0.91	0.91	0.87	0.87	0.87	0.87	3.4	3.4
Segerstolpe	2,209	0.95	0.95	0.88	0.88	0.92	0.92	3.6	3.6
Romanov	2,881	0.63	0.63	0.61	0.61	0.62	0.62	5.5	5.5
Zeisel	3,005	0.86	0.86	0.81	0.81	0.82	0.82	5.9	5.9
Lake	3,042	0.58	0.58	0.74	0.74	0.75	0.75	6.1	6.1
Montoro	7,193	0.68	**0.70**	0.54	**0.58**	0.63	**0.65**	163.9	**17.9**
Guo	7,416	**0.88**	0.86	**0.88**	0.87	**0.90**	0.89	192.8	**17.9**
Baron	8,569	0.94	0.94	**0.88**	0.87	**0.90**	0.89	280.0	**17.9**
Chen	12,089	**0.85**	0.72	**0.69**	0.55	**0.77**	0.60	674.9	**17.9**
Slyper	13,316	**0.75**	0.67	**0.78**	0.73	**0.76**	0.73	777.7	**17.9**
Kanton	17,542	0.29	**0.67**	0.31	**0.64**	0.42	**0.64**	1,349	**17.9**
Brann	26,766	0.86	0.86	**0.73**	0.72	0.80	0.80	1,728	**17.9**
Zilionis	34,558	0.87	**0.89**	0.84	0.84	0.85	**0.89**	3,834	**18.5**
Macosko	44,808	NA	**0.89**	NA	**0.66**	NA	**0.70**	NA	**18.5**
Hrvatin	48,266	NA	**0.78**	NA	**0.76**	NA	**0.82**	NA	**18.6**
Orozco	100,055	NA	**0.77**	NA	**0.65**	NA	**0.75**	NA	**37.6**
Miller	142,523	NA	**0.90**	NA	**0.82**	NA	**0.88**	NA	**36.0**
Darrah	162,490	NA	**0.47**	NA	**0.53**	NA	**0.63**	NA	**37.9**
Kozareva	611,034	NA	**1.00**	NA	**0.94**	NA	**0.96**	NA	**45.0**
Cao	1,092,000	NA	**0.89**	NA	**0.84**	NA	**0.90**	NA	**39.0**

4.4.5 Effects of min-max scaling

The min-max scaling is not a scRNA-seq normalization method and it is not intended to do so. We leave the step of data processinga and normalization completely up to the users. This min-max scaling added to our method is used on top of the already normalized data provided by users. Such scaling is frequently used in deep learning models [85–88] with the common purpose of reducing standard deviation and suppressing the effect of outliers. Below, we will demonstrate that the min-max scaling step improves the clustering performance without altering the transcriptome landscape.

To demonstrate the usefulness of this min-max scaling on clustering, we re-analyzed all single-cell datasets using scCAN without applying the min-max scaling step. Figure 4.23 shows the ARI values obtained from scCAN in two scenarios: scCAN with and without the scaling step. Overall, the min-max scaling makes the analysis more robust (lower variance) and more accurate (higher ARI). This result demonstrates the usefulness of the min-max scaling in improving the performance of scCAN.

To further demonstrate that the min-max scaling does not alter the transcriptome landscape of the data, we calculated the distance correlation index ($dCor$) [110] between the two dimensional representation of scaling and non-scaling data generated by t-SNE. Given X and Y as the 2D representation of the scaling and non-scaling data, $dCor$ is calculated as $dCor = \frac{dCov(X,Y)}{\sqrt{dVar(X)dVar(Y)}}$ where $dCov(X, Y)$ is the distance covariance between X and Y while $dVar(X)$ and $dVar(Y)$ are distance variances of X and Y. Specifically, $dCor$ first calculates the pair-wise distances for X by computing the distance between each pair of cells, resulting in a square matrix. Second, it calculates the pair-wise distances for Y. Finally, it compares the two matrices using the formula described above to obtain the distance correlation. The $dCor$ coefficient has values ranging from 0 to 1, with the $dCor$ is expected to be 1 for a perfect sim-

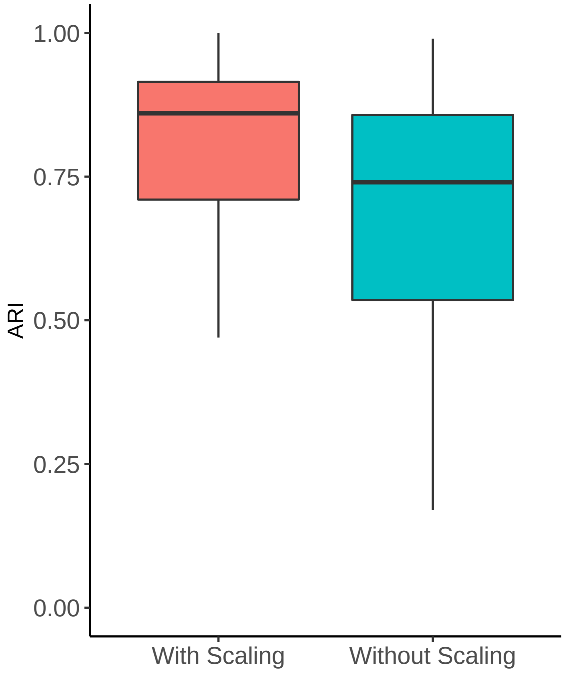

Figure 4.23: Impact of min-max scaling on scCAN. The analysis without scaling has higher variability and lower ARI values.

ilarity. In our analysis, when we rotate the transcriptome landscape, $dCor$ does not change. We re-analyzed the single-cell datasets and calculate the distance correlation for each dataset. Overall, the $dCor$ values obtained from all datasets are very high (median $dCor$ of 0.99 and variance of 0.01). This confirms that the min-max scaling does not alter the transcriptome landscape of the data while improving the clustering results.

4.4.6 Rare cell types detection

The sampling process is necessary to reduce both time and space complexity, but it can alter the capability of detecting rare cell types. By selecting 5,000 cells from a large dataset, we might end up with insufficient number of rare cells, and therefore reduce the chance of detecting them.

In addition, we have developed two strategies to enhance the method's capability of detecting rare cell types. First, we now allow users to change the parameter $samp.size$ so that they can increase the sample size, thus boosting the method's capability in detecting rare cell types. Second, we provide an instruction to perform multi-state clustering, i.e., further splitting the clustering results. When a cell type has too few cells, these cells are often mistakenly grouped with other cell types. By further splitting each clusters, we are able to detect rare cell types that would not be possible by performing one-stage clustering.

To demonstrate the efficiency of both solutions, we have tested them on the Zilionis dataset. The Zilionis dataset has 34,558 cells and 9 cell types. The transcriptome landscape and the cell types of the dataset are shown in Figure 4.24A. Among the 9 cell types, the tRBC cell type has only 108 cells (0.3%). A sub-sample of 5,000 cells is expected to have approximately 19 tRBC cells, which might be insufficient for many clustering method to detect them. Indeed, as show in Figure 4.24B, scCAN mistakenly grouped tRBC cells with tPlasma cells when we used the default setting of $samp.size = 5,000$.

To demonstrate the efficiency of the first strategy, we set $samp.size = 10,000$. The clustering results using the new parameter is shown in Figure 4.24C. With a sample size of 10,000, the method can properly separate tRBC cells and assigned them to cluster 2. To quantify how well the method separates tRBC cells from other cells, we calculated the F1 score [150]. Briefly, $F1 = 2 * \dfrac{precison * recall}{precison + recall} = \dfrac{TP}{TP + \frac{1}{2}(FP + FN)}$ where: i) TP are tRBC cells that were correctly assigned to cluster 2, ii) FP are cells of other cell types that were mistakenly assigned to cluster 2, iii) and FN are tRBC cells but were not assigned to cluster 2. In the ideal case, FP=FN=0 which leads to F1=1. In the analysis shown in Figure 4.24C, F1 score is 0.9 which indicates that scCAN properly separated tRBC from the rest. The method is expected to perform even better if we further increase the sample size.

To demonstrate the efficiency of the second strategy, we performed a two-stage clustering using the the default setting of $samp.size = 5,000$. In stage one, we partitioned the data using scCAN and obtained the clustering results as shown in Figure 4.24B. In stage two, we further partitioned each cluster obtained from stage one using the same method scCAN. The results of stage two are shown in Figure 4.24D. Cluster 2 were further split into two sub-clusters: 2_1 and 2_2. The tRBC cells were completely separated from the rest (cluster 2_2) with an F1 score of 1. This demonstrates that users can efficiently detect rare cell types using multi-stage clustering even with the default parameter $samp.size = 5,000$.

4.4.7 Scalability of scCAN

To demonstrate the scalability of scCAN, we downloaded and analyzed the Brain 1.3M dataset (`https://genomebiology.biomedcentral.com/articles/10.1186/s13059-017-1382-0`). Only scCAN and SCANPY were able to analyze this dataset of 1.3 million of cells. The clustering results of the two methods are shown in Figure 4.25. scCAN partitioned the data into 19 cluster whereas SCANPY partitioned the data into 20 clusters. The running time of scCAN and SCANPY were 51 minutes and 70 minutes, respectively. Note that we could not assess the accuracy of the two methods using this particular dataset because it does not have true cell type information.

Second, we downloaded the Cao dataset [138] that contains 1,092,000 cells sequenced from the human cerebellum with known cell types. Again, only scCAN and SCANPY were able to analyze this dataset. Figure 4.26A shows the visualization of 2D t-SNE embedding data generated from raw data with original cells annotations while Figure 4.26B–C show the visualizations of Cao dataset using clusters generated from SCANPY and scCAN. SCANPY can cluster the whole dataset in 51 minutes

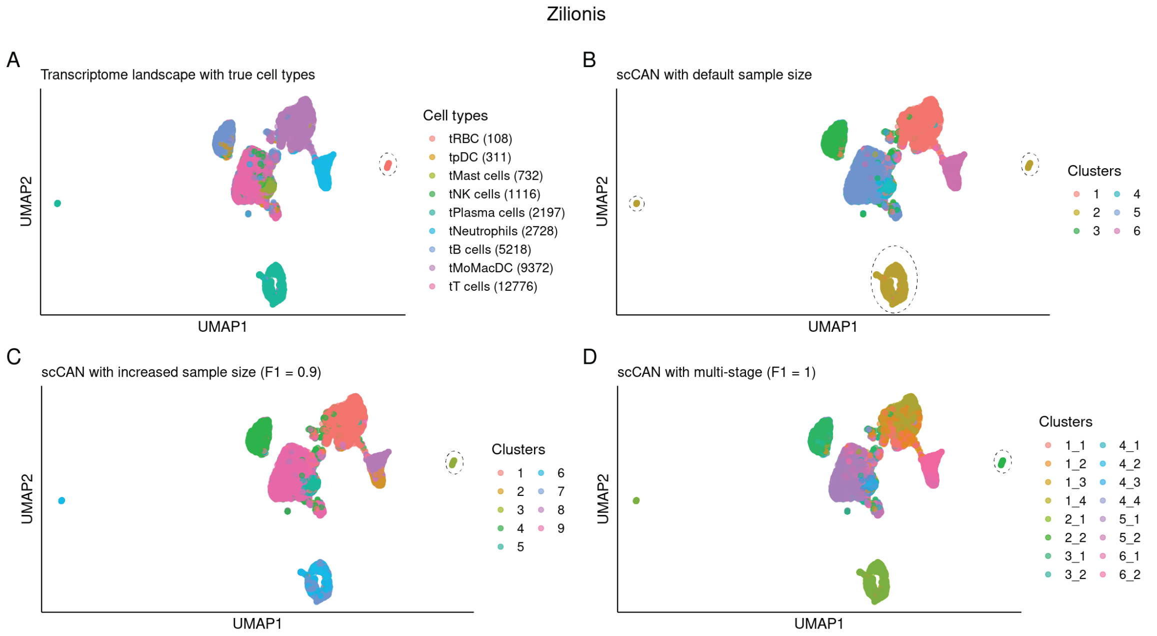

Figure 4.24: Rare cell type detection using the Zilionis dataset as example. The dataset has a total of 34,558 cells, in which there are 108 tRBC cells (rare cell type with 0.3% prevalence). (A) Transciptome landscape and true cell types. (B) Clustering results using scCAN with default sample size ($samp.size = 5,000$), in which tRBC are mistakenly grouped with tPlasma cells. (C) Clustering results with sample size of 10,000 ($samp.size = 10,000$). In this case, scCAN properly separates tRBC cells in cluster 2 with an F1 score of 0.9. (D) Clustering results using two-stage strategy and default sample size ($samp.size = 5,000$). scCAN properly separates tRBC cells in cluster 2 with a perfect F1 score of 1.

with the ARI of 0.48 (Figure 4.26B), while scCAN takes 39 minutes to partition cells with the ARI of 0.89 (Figure 4.26C). We have updated the analysis results for the Brain 1.3M and Cao dataset to the main Manuscript and Supplementary Material.

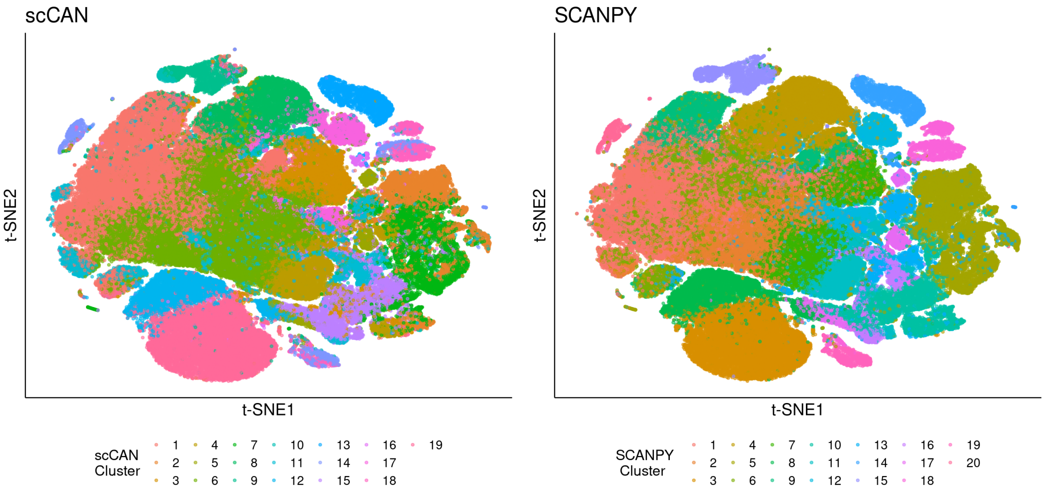

Figure 4.25: Clustering results of the Brain 1.3M dataset using scCAN and SCANPY. The left panel shows cell annotation of 20 clusters discovered by SCANPY. The right panel shows the cell partitions of 19 clusters identified from scCAN.

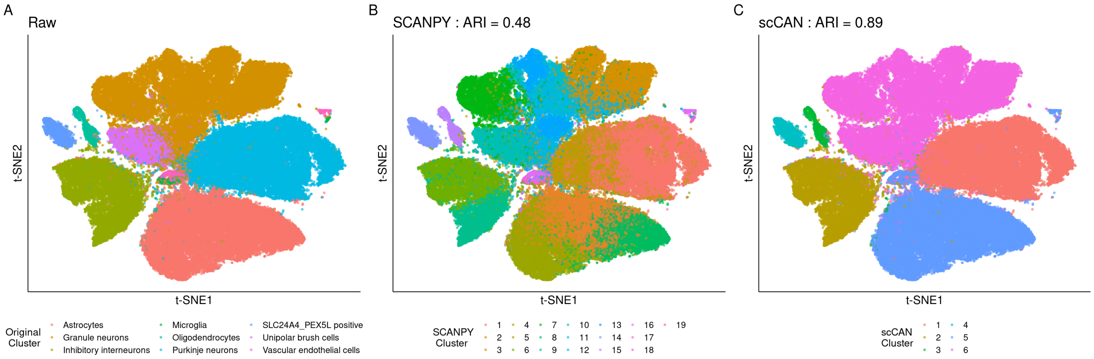

Figure 4.26: Visualizing of the Cao dataset using t-SNE. (A) Transcriptome landscape with true cell type information. (B) Transcriptome landscape of the clusters identified by SCANPY. (C) Transciptome landscape of clusters identified by scCAN. scCAN outperforms SCANPY by having a higher ARI value.

Chapter 5

Conclusions and Future Research

Single-cell RNA sequencing (scRNA-seq) technology has emerged as an invaluable asset in the world of genomics and molecular biology, enabling us to probe the biological mechanisms of diseases at an unprecedented level of granularity. This technology revolutionized our understanding of cellular heterogeneity and dynamics, providing detailed insights into the transcriptome of individual cells, hence contributing to advancements in diverse fields such as developmental biology, oncology, and immunology. Despite the tremendous strides made, current scRNA-seq methods are not without limitations. Issues like dropouts, noise in the data, and the difficulty in analyzing sparse matrices present substantial challenges, which hamper our ability to fully exploit this technology's potential. Addressing these limitations, the need for sophisticated scRNA-seq data imputation and clustering techniques has become crucial. Data imputation aids in filling the gaps in the dataset, reducing the impact of dropouts and noise, while efficient clustering methodologies can help delineate distinct cell populations and states from the scRNA-seq data, thereby unraveling the underlying biological phenomena.

First, we presented RIA, a novel technique that can accurately impute missing values from single-cell data. Our approach is divided into two components. The first module uses hypothesis testing to determine which variables are likely to be influenced by dropout occurrences. The second module uses a robust regression technique to predict the missing value. The data itself is used to learn all of the parameters. The method is put to the test with five benchmarking datasets totaling 3,535 cells. We show that RIA outperforms existing imputation approaches for identifying cell populations and temporal trajectories.

Second, we developed scIDS, which can impute missing values from single-cell data with high accuracy. Our approach is divided into two parts. The first module employs deep neural networks to compress and cluster data. This compressed data is regarded as reliable data for imputation. The second module does a z-test to identify genes that have been heavily influenced by dropouts. The module then learns the essential feature patterns in each cell group (identified in the previous module) and imputes missing values caused by dropout occurrences. Using only highly relevant information, this technique guarantees that the genuine missing values are imputed. We show that scIDS increases the quality of single-cell data while retaining the transcriptome landscape in an exhaustive study that includes simulation and 8 actual scRNA-seq datasets.

Third, we introduced scISR, a new method for imputation that involves subspace regression. This method uses a statistical technique to predict outcomes based on a subset of variables or ‘subspace’ of the complete dataset. The subspace regression might be used to focus on a subset of features that are most relevant to the outcome, which can be particularly useful in high-dimensional datasets. This method can be particularly beneficial in a single-cell genomics context because it might allow for more accurate prediction of gene expression values.

Finally, we presented scCAN, a single-cell clustering approach comprised of three modules: (1) a non-negative kernel autoencoder for filtering out uninformative features, (2) a stacked, variational autoencoder for generating multiple low-dimensional representations of single-cell data, and (3) a graph-based technique for determining cell groups from multiple representations. We show that scCAN greatly outperforms state-of-the-art approaches in sorting cells of various kinds in a comprehensive evaluation utilizing 28 scRNA-seq datasets. Using simulated datasets, we further evaluate the clustering algorithms in terms of scalability and resistance to dropouts. Overall, the most robust and reliable approach is scCAN, which can evaluate most datasets in minutes.

In conclusion, our scRNA-seq imputation and clustering methods have the potential of integration with existing data analysis pipelines to enhance the quality and reliability of downstream research endeavors. The seamless integration of these innovative imputation and clustering approaches into current scRNA-seq data analysis workflows not only empowers researchers to tackle increasingly complex biological questions but also contributes to the advancement of various scientific domains, including systems-level analysis [151–163], meta-analysis [164–169], cancer subtype discovery [81, 82, 170–182], single-cell analysis [83, 147, 183–190], and other important research areas [191–204]. By continuously refining and expanding these computational tools, the scientific community can harness the full potential of single-cell transcriptomics and accelerate the discovery of novel biological insights, ultimately benefiting human health and well-being.

In terms of future endeavors, we have the potential to enhance the methods we've developed and expand their application to a broader range of data types and other phenotypes, opening the door to new biological findings. Several projects are lined up as a direct continuation of the work we've outlined above. These projects encompass:

Data imputation, the technique of estimating missing values in a dataset, has vast potential for application across various types of data including miRNA, mRNA, and clinical data. In the field of miRNA and mRNA data, imputation can be incredibly valuable. MicroRNAs (miRNAs) and messenger RNAs (mRNAs) are types of RNA molecules that play crucial roles in gene regulation, and their expression levels can be critical indicators of various biological processes and diseases. However, due to the inherent complexities of RNA sequencing technologies, these datasets often suffer from missing values or dropouts, which can obscure the underlying biological signals. Applying data imputation to these datasets can help fill in these gaps and improve the accuracy and reliability of downstream analyses, such as differential expression analysis and gene network inference.

Similarly, in clinical data, missing values are a common issue. Clinical datasets often contain missing values due to reasons such as patient dropouts, missing visits, or incomplete medical records. This can pose significant challenges in data analysis and may lead to biased results if not handled properly. Data imputation techniques can be used to estimate these missing values based on the observed data, enabling more robust and reliable analyses. Moreover, imputing missing clinical data can lead to more complete datasets, which can improve the power and accuracy of statistical analyses, support more informed decision-making in clinical practice, and ultimately contribute to better patient outcomes.

Furthermore, as we continue to develop and refine imputation techniques, it's possible to customize these methods for different data types and application scenarios. For instance, methods may be tailored to account for the specific characteristics and structures of miRNA, mRNA, and clinical data. Such specialized imputation methods could potentially outperform general-purpose methods and provide more accurate and biologically meaningful imputations. In conclusion, applying data imputation to various data types, including miRNA, mRNA, and clinical data, holds great promise for improving the quality of data and facilitating new biological and clinical discoveries.

References

[1] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B Tuch, Asim Siddiqui, Kaiquin Lao, and Azim Surani. mRNA-Seq whole-transcriptome analysis of a single cell. *Nature Methods*, 6(5):377–382, 2009.

[2] Serena Liu and Cole Trapnell. Single-cell transcriptome sequencing: recent advances and remaining challenges. *F1000Research*, 5, 2016.

[3] Christoph Ziegenhain, Beate Vieth, Swati Parekh, Björn Reinius, Amy Guillaumet-Adkins, Martha Smets, Heinrich Leonhardt, Holger Heyn, Ines Hellmann, and Wolfgang Enard. Comparative analysis of single-cell RNA sequencing methods. *Molecular Cell*, 65(4):631–643, 2017.

[4] Charles A. Herring, Bob Chen, Eliot T. McKinley, and Ken S. Lau. Single-cell computational strategies for lineage reconstruction in tissue systems. *Cellular and Molecular Gastroenterology and Hepatology*, 5(4):539–548, 2018.

[5] Philip Brennecke, Simon Anders, Jong Kyoung Kim, Aleksandra A Kolodziejczyk, Xiuwei Zhang, Valentina Proserpio, Bianka Baying, Vladimir Benes, Sarah A Teichmann, John C Marioni, and Marcus G Heisler. Accounting for technical noise in single-cell RNA-seq experiments. *Nature Methods*, 10 (11):1093–1095, 2013.

[6] Florian Buettner, Kedar N Natarajan, F Paolo Casale, Valentina Proserpio, Antonio Scialdone, Fabian J Theis, Sarah A Teichmann, John C Marioni, and Oliver Stegle. Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. *Nature Biotechnology*, 33(2):155–160, 2015.

[7] Peter V Kharchenko, Lev Silberstein, and David T Scadden. Bayesian approach to single-cell differential expression analysis. *Nature Methods*, 11(7):740–742, 2014.

[8] Simone Rizzetto, Auda A Eltahla, Peijie Lin, Rowena Bull, Andrew R Lloyd, Joshua WK Ho, Vanessa Venturi, and Fabio Luciani. Impact of sequencing depth and read length on single cell RNA sequencing data of T cells. *Scientific Reports*, 7:12781, 2017.

[9] Swati Parekh, Christoph Ziegenhain, Beate Vieth, Wolfgang Enard, and Ines Hellmann. The impact of amplification on differential expression analyses by RNA-seq. *Scientific Reports*, 6:25533, 2016.

[10] Ashraful Haque, Jessica Engel, Sarah A Teichmann, and Tapio Lönnberg. A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications. *Genome Medicine*, 9(1):75, 2017.

[11] David Van Dijk, Roshan Sharma, Juozas Nainys, Kristina Yim, Pooja Kathail, Ambrose J Carr, Cassandra Burdziak, Kevin R Moon, Christine L Chaffer, Diwakar Pattabiraman, Brian Bierie, Linas Mazutis, Guy Wolf, Smita Krishnaswamy, and Dana Pe'er. Recovering gene interactions from single-cell data using data diffusion. *Cell*, 174(3):716–729, 2018.

[12] Zdravko I Botev, Joseph F Grotowski, Dirk P Kroese, et al. Kernel density estimation via diffusion. *The Annals of Statistics*, 38(5):2916–2957, 2010.

[13] Wuming Gong, Il-Youp Kwak, Pruthvi Pota, Naoko Koyano-Nakagawa, and Daniel J Garry. DrImpute: imputing dropout events in single cell RNA sequencing data. *BMC Bioinformatics*, 19:220, 2018.

[14] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[15] Stefano Monti, Pablo Tamayo, Jill Mesirov, and Todd Golub. Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52(1-2):91–118, 2003.

[16] Vladimir Yu Kiselev, Kristina Kirschner, Michael T Schaub, Tallulah Andrews, Andrew Yiu, Tamir Chandra, Kedar N Natarajan, Wolf Reik, Mauricio Barahona, Anthony R Green, and Martin Hamberg. SC3: consensus clustering of single-cell RNA-seq data. *Nature Methods*, 14:483–486, 2017.

[17] Mo Huang, Jingshu Wang, Eduardo Torre, Hannah Dueck, Sydney Shaffer, Roberto Bonasio, John I Murray, Arjun Raj, Mingyao Li, and Nancy R Zhang. SAVER: gene expression recovery for single-cell RNA sequencing. *Nature Methods*, 15(7):539–542, 2018.

[18] Wei Vivian Li and Jingyi Jessica Li. An accurate and robust imputation method scImpute for single-cell RNA-seq data. *Nature Communications*, 9:997, 2018.

[19] Elham Azizi, Sandhya Prabhakaran, Ambrose Carr, and Dana Pe'er. Bayesian inference for single-cell clustering and imputing. *Genomics and Computational Biology*, 3(1):e46–e46, 2017.

[20] Dilan Görür and Carl Edward Rasmussen. Dirichlet process gaussian mixture models: Choice of the base distribution. *Journal of Computer Science and Technology*, 25(4):653–664, 2010.

[21] Peijie Lin, Michael Troup, and Joshua W. K. Ho. CIDR: Ultrafast and accurate clustering through imputation for single-cell RNA-seq data. *Genome Biology*, 18:59, 2017.

[22] Justina žurauskienė and Christopher Yau. pcaReduce: hierarchical clustering of single cell transcriptional profiles. *BMC Bioinformatics*, 17(1):1–11, 2016.

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

[24] Bo Wang, Daniele Ramazzotti, Luca De Sano, Junjie Zhu, Emma Pierson, and Serafim Batzoglou. SIMLR: a tool for large-scale genomic analyses by multi-kernel learning. *Proteomics*, 18(2):1700232, 2018.

[25] Orit Rozenblatt-Rosen, Michael JT Stubbington, Aviv Regev, and Sarah A Teichmann. The Human Cell Atlas: From vision to reality. *Nature*, 550(7677): 451–453, 2017.

[26] Itai Yanai and Tamar Hashimshony. CEL-Seq2—Single-cell RNA sequencing by multiplexed linear amplification. *Single Cell Methods: Sequencing and Proteomics*, pages 45–56, 2019.

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

[28] Simone Picelli, Omid R Faridani, Åsa K Björklund, Gösta Winberg, Sven Sagasser, and Rickard Sandberg. Full-length RNA-seq from single cells using Smart-seq2. *Nature Protocols*, 9(1):171–181, 2014.

[29] Paul Datlinger, André F Rendeiro, Thorina Boenke, Thomas Krausgruber, Daniele Barreca, and Christoph Bock. Ultra-high throughput single-cell RNA sequencing by combinatorial fluidic indexing. *BioRxiv*, pages 2019–12, 2019.

[30] Andrew McDavid, Greg Finak, Pratip K Chattopadyay, Maria Dominguez, Laurie Lamoreaux, Steven S Ma, Mario Roederer, and Raphael Gottardo. Data exploration, quality control and testing in single-cell qPCR-based gene expression experiments. *Bioinformatics*, 29(4):461–467, 2013.

[31] Stephanie C Hicks, F William Townes, Mingxiang Teng, and Rafael A Irizarry. Missing data and technical variability in single-cell rna-sequencing experiments. *Biostatistics*, 19(4):562–578, 2018.

[32] Mihriban Karaayvaz, Simona Cristea, Shawn M Gillespie, Anoop P Patel, Ravindra Mylvaganam, Christina C Luo, Michelle C Specht, Bradley E Bernstein, Franziska Michor, and Leif W Ellisen. Unravelling subclonal heterogeneity and aggressive disease states in TNBC through single-cell RNA-seq. *Nature Communications*, 9(1):3588, 2018.

[33] Jingshu Wang, Divyansh Agarwal, Mo Huang, Gang Hu, Zilu Zhou, Chengzhong Ye, and Nancy R Zhang. Data denoising with transfer learning in single-cell transcriptomics. *Nature Methods*, 16(9):875–878, 2019.

[34] Wenhao Tang, François Bertaux, Philipp Thomas, Claire Stefanelli, Malika Saint, Samuel Marguerat, and Vahid Shahrezaei. bayNorm: Bayesian gene ex-

pression recovery, imputation and normalization for single-cell RNA-sequencing data. *Bioinformatics*, 36(4):1174–1181, 2020.

[35] Zhun Miao, Jiaqi Li, and Xuegong Zhang. scRecover: Discriminating true and false zeros in single-cell RNA-seq data for imputation. *bioRxiv*, page 665323, 2019.

[36] Mengjie Chen and Xiang Zhou. VIPER: variability-preserving imputation for accurate gene expression recovery in single-cell RNA sequencing studies. *Genome Biology*, 19(1):1–15, 2018.

[37] Maayan Baron, Adrian Veres, Samuel L. Wolock, Aubrey L. Faust, Renaud Gaujoux, Amedeo Vetere, Jennifer Hyoje Ryu, Bridget K. Wagner, Shai S. Shen-Orr, Allon M. Klein, Douglas A. Melton, and Itai Yanai. A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure. *Cell Systems*, 3(4):346–360, 2016.

[38] Renchao Chen, Xiaoji Wu, Lan Jiang, and Yi Zhang. Single-cell RNA-seq reveals hypothalamic cell diversity. *Cell Reports*, 18(13):3227–3241, 2017.

[39] Gioele La Manno, Daniel Gyllborg, Simone Codeluppi, Kaneyasu Nishimura, Carmen Salto, Amit Zeisel, Lars E Borm, Simon RW Stott, Enrique M Toledo, J Carlos Villaescusa, Peter Lönnerberg, Jesper Ryge, Roger A Barker, Ernest Arenas, and Sten Linnarsson. Molecular diversity of midbrain development in mouse, human, and stem cells. *Cell*, 167(2):566–580, 2016.

[40] Amit Zeisel, Ana B Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, Charlotte Rolny, Gonçalo Castelo-Branco, Jens Hjerling-

Leffler, and Sten Linnarsson. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. *Science*, 347(6226):1138–1142, 2015.

[41] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B.*, 39:1–39, 1977.

[42] Mary Kathryn Cowles and Bradley P Carlin. Markov chain monte carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91(434):883–904, 1996.

[43] Florian Wagner, Yun Yan, and Itai Yanai. K-nearest neighbor smoothing for high-throughput single-cell rna-seq data. *BioRxiv*, page 217737, 2017.

[44] Jonathan Ronen and Altuna Akalin. netSmooth: Network-smoothing based imputation for single cell RNA-seq. *bioRxiv*, page 234021, 2017.

[45] Weimiao Wu, Yunqing Liu, Qile Dai, Xiting Yan, and Zuoheng Wang. 2S3: A gene graph-based imputation method for single-cell RNA sequencing data. *PLOS Computational Biology*, 17(5):e1009029, 2021.

[46] Divyanshu Talwar, Aanchal Mongia, Debarka Sengupta, and Angshul Majumdar. AutoImpute: Autoencoder based imputation of single-cell RNA-seq data. *Scientific Reports*, 8:16329, 2018.

[47] Gökcen Eraslan, Lukas M Simon, Maria Mircea, Nikola S Mueller, and Fabian J Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature Communications*, 10:390, 2019.

[48] Cédric Arisdakessian, Olivier Poirion, Breck Yunits, Xun Zhu, and Lana X Garmire. DeepImpute: an accurate, fast, and scalable deep neural network method to impute single-cell RNA-seq data. *Genome Biology*, 20(1):1–14, 2019.

[49] Matthew Amodio, David Van Dijk, Krishnan Srinivasan, William S Chen, Hussein Mohsen, Kevin R Moon, Allison Campbell, Yujiao Zhao, Xiaomei Wang, Manjunatha Venkataswamy, Anita Desai, V Ravi Priti Kumar, Ruth Montgomery, Guy Wolf, and Smita Krishnaswamy. Exploring single-cell data with deep multitasking neural networks. *Nature Methods*, 16(11):1139–1145, 2019.

[50] Yue Deng, Feng Bao, Qionghai Dai, Lani F Wu, and Steven J Altschuler. Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning. *Nature Methods*, 16(4):311–314, 2019.

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

[52] George C Linderman, Jun Zhao, Manolis Roulis, Piotr Bielecki, Richard A Flavell, Boaz Nadler, and Yuval Kluger. Zero-preserving imputation of single-cell RNA-seq data. *Nature Communications*, 13(1):192, 2022.

[53] Aanchal Mongia, Debarka Sengupta, and Angshul Majumdar. Mcimpute: matrix completion based imputation for single cell rna-seq data. *Frontiers in Genetics*, 10:9, 2019.

[54] Lihua Zhang and Shihua Zhang. Pblr: an accurate single cell rna-seq data imputation tool considering cell heterogeneity and prior expression level of dropouts. *bioRxiv*, page 379883, 2018.

[55] Shibiao Wan, Junil Kim, and Kyoung Jae Won. SHARP: hyperfast and accurate processing of single-cell RNA-seq data via ensemble random projection. *Genome Research*, 30(2):205–213, 2020.

[56] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[57] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.

[58] Leland McInnes, John Healy, and James Melville. Umap: uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[59] Tian Tian, Ji Wan, Qi Song, and Zhi Wei. Clustering single-cell RNA-seq data with a model-based deep learning approach. *Nature Machine Intelligence*, 1(4): 191–198, 2019.

[60] Kaikun Xie, Yu Huang, Feng Zeng, Zehua Liu, and Ting Chen. scAIDE: clustering of large-scale single-cell RNA-seq data reveals putative and rare cell types. *NAR Genomics and Bioinformatics*, 2(4):lqaa082, 2020.

[61] Luca Alessandri, Francesca Cordero, Marco Beccuti, Nicola Licheri, Maddalena Arigoni, Martina Olivero, Maria Flavia Di Renzo, Anna Sapino, and Raffaele Calogero. Sparsely-connected autoencoder (SCA) for single cell RNAseq data mining. *NPJ Systems Biology and Applications*, 7(1):1–10, 2021.

[62] Yulun Wu, Yanming Guo, Yandong Xiao, and Songyang Lao. AAE-SC: A scRNA-seq clustering framework based on adversarial autoencoder. *IEEE Access*, 8:178962–178975, 2020.

[63] Bin Yu, Chen Chen, Ren Qi, Ruiqing Zheng, Patrick J Skillman-Lawrence, Xiaolin Wang, Anjun Ma, and Haiming Gu. scGMAI: a Gaussian mixture model

for clustering single-cell RNA-Seq data based on deep autoencoder. *Briefings in Bioinformatics*, 2020.

[64] Elisabetta Mereu, Atefeh Lafzi, Catia Moutinho, Christoph Ziegenhain, Davis J McCarthy, Adrián Álvarez-Varela, Eduard Batlle, Dominic Grün, Julia K Lau, Stéphane C Boutet, Chad Sanada, Aik Ooi, Robert C. Jones, Kelly Kaihara, Chris Brampton, Yasha Talaga, Yohei Sasagawa, Kaori Tanaka, Tetsutaro Hayashi, Caroline Braeuning, Cornelius Fischer, Sascha Sauer, Timo Trefzer, Christian Conrad, Xian Adiconis, Lan T. Nguyen, Aviv Regev, Joshua Z. Levin, Swati Parekh, Aleksandar Janjic, Lucas E. Wange, Johannes W. Bagnoli, Wolfgang Enard, Marta Gut, Rickard Sandberg, Itoshi Nikaido, Ivo Gut, Oliver Stegle, and Holger Heyn. Benchmarking single-cell RNA-sequencing protocols for cell atlas projects. *Nature Biotechnology*, 38(6):747–755, 2020.

[65] Lu Yang, Jiancheng Liu, Qiang Lu, Arthur D Riggs, and Xiwei Wu. SAIC: an iterative clustering approach for analysis of single cell RNA-seq data. *BMC Genomics*, 18(6):9–17, 2017.

[66] Ming-Wen Hu, Dong Won Kim, Sheng Liu, Donald J Zack, Seth Blackshaw, and Jiang Qian. PanoView: An iterative clustering method for single-cell RNA sequencing data. *PLoS Computational Biology*, 15(8):e1007040, 2019.

[67] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[68] Vincent A. Traag, Ludo Waltman, and Nees Jan van Eck. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1):1–12, 2019.

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

[70] F. Alexander Wolf, Philipp Angerer, and Fabian J. Theis. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology*, 19:15, 2018.

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

[72] Yuchen Yang, Ruth Huh, Houston W Culpepper, Yuan Lin, Michael I Love, and Yun Li. SAFE-clustering: single-cell aggregated (from ensemble) clustering for single-cell RNA-seq data. *Bioinformatics*, 35(8):1269–1277, 2019.

[73] Ruth Huh, Yuchen Yang, Yuchao Jiang, Yin Shen, and Yun Li. SAME-clustering: S ingle-cell A ggregated Clustering via M ixture Model E nsemble. *Nucleic Acids Research*, 48(1):86–95, 2020.

[74] Xiaoshu Zhu, Jian Li, Hong-Dong Li, Miao Xie, and Jianxin Wang. Sc-GPE: A Graph Partitioning-Based Cluster Ensemble Method for Single-Cell. *Frontiers in Genetics*, 11:1606, 2020.

[75] Chen Xu and Zhengchang Su. Identification of cell types from single-cell transcriptomes using a novel clustering method. *Bioinformatics*, page btv088, 2015.

[76] Xiaoshu Zhu, Jie Zhang, Yunpei Xu, Jianxin Wang, Xiaoqing Peng, and Hong-Dong Li. Single-cell clustering based on shared nearest neighbor and graph

partitioning. *Interdisciplinary Sciences: Computational Life Sciences*, pages 1–14, 2020.

[77] Xiaoshu Zhu, Hong-Dong Li, Lilu Guo, Fang-Xiang Wu, and Jianxin Wang. Analysis of single-cell RNA-seq data by clustering approaches. *Current Bioinformatics*, 14(4):314–322, 2019.

[78] Tahani Alqurashi and Wenjia Wang. Clustering ensemble method. *International Journal of Machine Learning and Cybernetics*, 10(6):1227–1246, 2019.

[79] Tanya Barrett, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Michelle Holko, Andrey Yefanov, Hyeseung Lee, Naigong Zhang, Cynthia L. Robertson, Nadezhda Serova, Sean Davis, and Alexandra Soboleva. NCBI GEO: archive for functional genomics data sets–update. *Nucleic Acids Research*, 41(D1):D991–D995, 2013.

[80] Bengtsson, Martin and Ståhlberg, Anders and Rorsman, Patrik and Kubista, Mikael. Gene expression profiling in single cells from the pancreatic islets of langerhans reveals lognormal distribution of mrna levels. *Genome Research*, 15 (10):1388–1392, 2005.

[81] Tin Nguyen, Rebecca Tagett, Diana Diaz, and Sorin Draghici. A novel approach for data integration and disease subtyping. *Genome Research*, 27:2025–2039, 2017.

[82] Hung Nguyen, Sangam Shrestha, Sorin Draghici, and Tin Nguyen. PINSPlus: A tool for tumor subtype discovery in integrated genomic data. *Bioinformatics*, 35(16):2843–2846, 2019.

[83] Bang Tran, Duc Tran, Hung Nguyen, Nam Sy Vo, and Tin Nguyen. Ria: a novel regression-based imputation approach for single-cell rna sequencing. In *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–9. IEEE, 2019.

[84] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114*, 2013.

[85] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[86] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, 2017.

[87] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[88] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 6105–6114, Long Beach, California, USA, 2019.

[89] Bo Wang, Aziz M. Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature Methods*, 11(3): 333–337, 2014.

[90] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[91] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2:849–856, 2002.

[92] Gabriella Rustici, Nikolay Kolesnikov, Marco Brandizi, Tony Burdett, Miroslaw Dylag, Ibrahim Emam, Anna Farne, Emma Hastings, Jon Ison, Maria Keays, Natalja Kurbatova, James Malone, Roby Mani, Annalisa Mupo, Rui Pedro Pereira, Ekaterina Pilicheva, Johan Rung, Anjan Sharma, Y. Amy Tang, Tobias Ternent, Andrew Tikhonov, Danielle Welter, Eleanor Williams, Alvis Brazma, Helen Parkinson, and Ugis Sarkans. ArrayExpress update–trends in database growth and links to data analysis tools. *Nucleic Acids Research*, 41(D1):D987–D990, 2013.

[93] Fernando H Biase, Xiaoyi Cao, and Sheng Zhong. Cell fate inclination within 2-cell and 4-cell mouse embryos revealed by single-cell RNA sequencing. *Genome Research*, 24(11):1787–1796, 2014.

[94] Liying Yan, Mingyu Yang, Hongshan Guo, Lu Yang, Jun Wu, Rong Li, Ping Liu, Ying Lian, Xiaoying Zheng, Jie Yan, Jin Huang, Ming Li, Xinglong Wu, Lu Wen, Kaiqin Lao, Ruiqiang Li, Jie Qiao, and Fuchou Tang. Single-cell RNA-seq profiling of human preimplantation embryos and embryonic stem cells. *Nature Structural & Molecular Biology*, 20:1131–1139, 2013.

[95] Mubeen Goolam, Antonio Scialdone, Sarah JL Graham, Iain C Macaulay, Agnieszka Jedrusik, Anna Hupalowska, Thierry Voet, John C Marioni, and Magdalena Zernicka-Goetz. Heterogeneity in Oct4 and Sox2 targets biases cell fate in 4-cell mouse embryos. *Cell*, 165(1):61–74, 2016.

[96] Qiaolin Deng, Daniel Ramsköld, Björn Reinius, and Rickard Sandberg. Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells. *Science*, 343(6167):193–196, 2014.

[97] John A Hartigan and Manchek A Wong. Algorithm AS 136: A k-means clustering algorithm. *Applied Statistics*, pages 100–108, 1979.

[98] Laurens Van Der Maaten. Accelerating t-SNE using tree-based algorithms. *The Journal of Machine Learning Research*, 15(1):3221–3245, 2014.

[99] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[100] Baglama, James and Reichel, Lothar. Augmented implicitly restarted Lanczos bidiagonalization methods. *SIAM Journal on Scientific Computing*, 27(1):19–42, 2005.

[101] JH Krijthe. Rtsne: T-distributed stochastic neighbor embedding using barnes-hut implementation. *R package version 0.13, URL https://github.com/jkrijthe/Rtsne*, 2015.

[102] Alex A. Pollen, Tomasz J. Nowakowski, Joe Shuga, Xiaohui Wang, Anne A. Leyrat, Jan H. Lui, Nianzhen Li, Lukasz Szpankowski, Brian Fowler, Peilin Chen, Naveen Ramalingam, Gang Sun, Myo Thu, Michael Norris, Ronald Lebofsky, Dominique Toppani, Darnell W. Kemp Ii, Michael Wong, Barry Clerkson, Brittnee N. Jones, Shiquan Wu, Lawrence Knutsson, Beatriz Alvarado, Jing Wang, Lesley S. Weaver, Andrew P. May, Robert C. Jones, Marc A. Unger, Arnold R. Kriegstein, and Jay A. A. West. Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. *Nature Biotechnology*, 32:1053–1058, 2014.

[103] Spyros Darmanis, Steven A Sloan, Ye Zhang, Martin Enge, Christine Caneda, Lawrence M Shuer, Melanie G Hayden Gephart, Ben A Barres, and Stephen R Quake. A survey of human brain transcriptome diversity at the single cell level. *Proceedings of the National Academy of Sciences of the United States of America*, 112(23):7285–7290, 2015.

[104] Dmitry Usoskin, Alessandro Furlan, Saiful Islam, Hind Abdo, Peter Lönnerberg, Daohua Lou, Jens Hjerling-Leffler, Jesper Haeggström, Olga Kharchenko, Peter V Kharchenko, Sten Linnarson, and Patrik Ernfors. Unbiased classification of sensory neuron types by large-scale single-cell RNA sequencing. *Nature Neuroscience*, 18:145–153, 2015.

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

[106] Sinisa Hrvatin, Daniel R. Hochbaum, M. Aurel Nagy, Marcelo Cicconet, Keiramarie Robertson, Lucas Cheadle, Rapolas Zilionis, Alex Ratner, Rebeca Borges-Monroy, Allon M. Klein, Bernardo L. Sabatini, and Michael E. Greenberg. Single-cell analysis of experience-dependent transcriptomic states in the mouse visual cortex. *Nature Neuroscience*, 21(1):120–129, 2018.

[107] Chen Cao, Laurence A. Lemaire, Wei Wang, Peter H. Yoon, Yoolim A. Choi, Lance R. Parsons, John C. Matese, Michael Levine, and Kai Chen. Comprehensive single-cell transcriptome lineages of a proto-vertebrate. *Nature*, 571(7765): 349–354, 2019.

[108] Jim Baglama, Lothar Reichel, and B. W. Lewis. *irlba: Fast Truncated Singular Value Decomposition and Principal Components Analysis for Large Dense and*

Sparse Matrices, 2018. URL `https://CRAN.R-project.org/package=irlba`. R package version 2.3.2.

[109] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel WH Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W Newell. Dimensionality reduction for visualizing single-cell data using UMAP. *Nature Biotechnology*, 37:38–44, 2019.

[110] Gábor J Székely, Maria L Rizzo, and Nail K Bakirov. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769–2794, 2007.

[111] Xiaoying Fan, Xiannian Zhang, Xinglong Wu, Hongshan Guo, Yuqiong Hu, Fuchou Tang, and Yanyi Huang. Single-cell RNA-seq transcriptome analysis of linear and circular RNAs in mouse preimplantation embryos. *Genome Biology*, 16(1):148, 2015.

[112] Barbara Treutlein, Doug G Brownfield, Angela R Wu, Norma F Neff, Gary L Mantalas, F Hernan Espinoza, Tushar J Desai, Mark A Krasnow, and Stephen R Quake. Reconstructing lineage hierarchies of the distal lung epithelium using single-cell RNA-seq. *Nature*, 509:371–375, 2014.

[113] J Gray Camp, Farhath Badsha, Marta Florio, Sabina Kanton, Tobias Gerber, Michaela Wilsch-Bräuninger, Eric Lewitus, Alex Sykes, Wulf Hevers, Madeline Lancaster, Juergen A Knoblich, Robert Lachmann, Svante Pääbo, Wieland Huttner, and Barbara Treutlein. Human cerebral organoids recapitulate gene expression programs of fetal neocortex development. *Proceedings of the National Academy of Sciences of the United States of America*, 112(51):15672–15677, 2015.

[114] Roman A Romanov, Amit Zeisel, Joanne Bakker, Fatima Girach, Arash Hellysaz, Raju Tomer, Alán Alpár, Jan Mulder, Frédéric Clotman, Erik Keimpema, Brian Hsueh, Ailey K Crow, Henrik Martens, Christian Schwindling, Daniela Calvigioni, Jaideep S Bains, Zoltán Máté, Gábor Szabó, Yuchio Yanagawa, Ming-Dong Zhang, Andre Rendeiro, Matthias Farlik, Mathias Uhlén, Peer Wulff, Christop Bock, Christian Broberger, Karl Deisseroth, Tomas Hökfelt, Sten Linnarsson, Tamas L Horvath, and Tibor Harkany. Molecular interrogation of hypothalamic organization reveals distinct dopamine neuronal subtypes. *Nature Neuroscience*, 20(2):176–188, 2017.

[115] Åsa Segerstolpe, Athanasia Palasantza, Pernilla Eliasson, Eva-Marie Andersson, Anne-Christine Andréasson, Xiaoyan Sun, Simone Picelli, Alan Sabirsh, Maryam Clausen, Magnus K. Bjursell, David M. Smith, Maria Kasper, Carina Ämmälä, and Rickard Sandberg. Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes. *Cell Metabolism*, 24(4):593–607, 2016.

[116] Sueli Marques, Amit Zeisel, Simone Codeluppi, David van Bruggen, Ana Mendanha Falcão, Lin Xiao, Huiliang Li, Martin Häring, Hannah Hochgerner, Roman A Romanov, Hannah Hochgerner, Roman A Romanov, Daniel Gyllborg, Ana B Muñoz-Manchado, Jens Hjerling-Leffler, Tibor Harkany, William D Richardson, Sten Linnarsson, and Gonçalo Castelo-Branco. Oligodendrocyte heterogeneity in the mouse juvenile and adult central nervous system. *Science*, 352(6291):1326–1329, 2016.

[117] Sydney M Sanderson, Zhengtao Xiao, Amy J Wisdom, Shree Bose, Maria V Liberti, Michael A Reid, Emily Hocke, Simon G Gregory, David G Kirsch,

and Jason W Locasale. The Na+/K+ atpase regulates glycolysis and defines immunometabolism in tumors. *bioRxiv*, 2020. doi: 10.1101/2020.03.31.018739.

[118] Rapolas Zilionis, Camilla Engblom, Christina Pfirschke, Virginia Savova, David Zemmour, Hatice D Saatcioglu, Indira Krishnan, Giorgia Maroni, Claire V Meyerovitz, Clara M Kerwin, Sun Choi, William G Richards, Assunta De Rienzo, Daniel G Tenen, Raphael Bueno, Elena Levantini, and Allon M Pittet, Mikael J amd Klein. Single-cell transcriptomics of human and mouse lung cancers reveals conserved myeloid populations across individuals and species. *Immunity*, 50(5):1317–1334, 2019.

[119] Bosiljka Tasic, Zizhen Yao, Lucas T. Graybuck, Kimberly A. Smith, Thuc Nghi Nguyen, Darren Bertagnolli, Jeff Goldy, Emma Garren, Michael N. Economo, Sarada Viswanathan, Osnat Penn, Trygve Bakken, Vilas Menon, Jeremy Miller, Olivia Fong, Karla E. Hirokawa, Kanan Lathia, Christine Rimorin, Michael Tieu, Rachael Larsen, Tamara Casper, Eliza Barkan, Matthew Kroll, Sheana Parry, Nadiya V. Shapovalova, Daniel Hirschstein, Julie Pendergraft, Heather A. Sullivan, Tae Kyung Kim, Aaron Szafer, Nick Dee, Peter Groblewski, Ian Wickersham, Ali Cetin, Julie A. Harris, Susan M. Levi, Boaz P.and Sunkin, Linda Madisen, Tanya L. Daigle, Loren Looger, Amy Bernard, John Phillips, Ed Lein, Michael Hawrylycz, Karel Svoboda, Allan R. Jones, Christof Koch, and Hongkui Zeng. Shared and distinct transcriptomic cell types across neocortical areas. *Nature*, 563(7729):72–78, 2018.

[120] Tavé van Zyl, Wenjun Yan, Alexi McAdams, Yi-Rong Peng, Karthik Shekhar, Aviv Regev, Dejan Juric, and Joshua R. Sanes. Cell atlas of aqueous humor outflow pathways in eyes of humans and four model species provides insight

into glaucoma pathogenesis. *Proceedings of the National Academy of Sciences*, 117(19):10339–10349, 2020.

[121] Kevin Wei, Ilya Korsunsky, Jennifer L. Marshall, Anqi Gao, Gerald FM. Watts, Triin Major, Adam P. Croft, Jordan Watts, Philip E. Blazar, Jeffrey K. Lange, Thomas S. Thornhill, Andrew Filer, Karim Raza, Laura T. Donlin, Accelerating Medicines Partnership Rheumatoid Arthritis, Systemic Lupus Erythematosus (AMP RA/SLE) Consortium, Christian W. Siebel, Christopher D. Buckley, Soumya Raychaudhuri, and Michael B. Brenner. Notch signalling drives synovial fibroblast identity and arthritis pathology. *Nature*, 582:259–264, 2020.

[122] Luz D Orozco, Hsu-Hsin Chen, Christian Cox, Kenneth J Katschke Jr, Rommel Arceo, Carmina Espiritu, Patrick Caplazi, Sarajane Saturnio Nghiem, Ying-Jiun Chen, Zora Modrusan, Amy Dressen, Leonard D Goldstein, Christine Clarke, Tushar Bhangale, Brian Yaspan, Marion Jeanne, Michael J Townsend, Menno van Lookeren Campagne, and Jason A Hackney. Integration of eQTL and a single-cell atlas in the human eye identifies causal genes for age-related macular degeneration. *Cell Reports*, 30(4):1246–1259, 2020.

[123] Patricia A. Darrah, Joseph J. Zeppa, Pauline Maiello, Joshua A. Hackney, Marc H. Wadsworth, Travis K. Hughes, Supriya Pokkali, Phillip A. Swanson, Nicole L. Grant, Mark A. Rodgers, Megha Kamath, Chelsea M. Causgrove, Dominick J. Laddy, Aurelio Bonavia, Danilo Casimiro, Philana Ling Lin, Edwin Klein, Alexander G. White, Charles A. Scanga, Alex K. Shalek, Mario Roederer, JoAnne L. Flynn, and Robert A. Seder. Prevention of tuberculosis in macaques after intravenous BCG immunization. *Nature*, 577(7788):95–102, 2020.

[124] Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: Simulation of single-cell RNA sequencing data. *Genome Biology*, 18:1–15, 2017.

[125] Anoop P Patel, Itay Tirosh, John J Trombetta, Alex K Shalek, Shawn M Gillespie, Hiroaki Wakimoto, Daniel P Cahill, Brian V Nahed, William T Curry, Robert L Martuza, David N. Louis, Orit Rozenblatt-Rosen, Mario L. Suvà, Aviv Regev, and Bradley E. Bernstein. Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma. *Science*, 344(6190):1396–1401, 2014.

[126] Yue J. Wang, Jonathan Schug, Kyoung-Jae Won, Chengyang Liu, Ali Naji, Dana Avrahami, Maria L. Golson, and Klaus H. Kaestner. Single-cell transcriptomics of the human endocrine pancreas. *Diabetes*, 65(10):3028–3038, 2016.

[127] Huipeng Li, Elise T Courtois, Debarka Sengupta, Yuliana Tan, Kok Hao Chen, Jolene Jie Lin Goh, Say Li Kong, Clarinda Chua, Lim Kiat Hon, Wah Siew Tan, et al. Reference component analysis of single-cell transcriptomes elucidates cellular heterogeneity in human colorectal tumors. *Nature Genetics*, 49(5):708, 2017.

[128] J. Gray Camp, Keisuke Sekine, Tobias Gerber, Henry Loeffler-Wirth, Hans Binder, Malgorzata Gac, Sabina Kanton, Jorge Kageyama, Georg Damm, Daniel Seehofer, Lenka Belicova, Marc Bickle, Rico Barsacchi, Ryo Okuda, Emi Yoshizawa, Masaki Kimura, Hiroaki Ayabe, Hideki Taniguchi, Takanori Takebe, and Barbara Treutlein. Multilineage communication regulates human liver bud development from pluripotency. *Nature*, 546(7659):533–538, 2017.

[129] Yurong Xin, Jinrang Kim, Haruka Okamoto, Min Ni, Yi Wei, Christina Adler, Andrew J. Murphy, George D. Yancopoulos, Calvin Lin, and Jesper Gromada. RNA sequencing of single human islet cells reveals type 2 diabetes genes. *Cell Metabolism*, 24(4):608–615, 2016.

[130] Mauro J. Muraro, Gitanjali Dharmadhikari, Dominic Grün, Nathalie Groen, Tim Dielen, Erik Jansen, Leon van Gurp, Marten A. Engelse, Francoise Carlotti, Eelco J.P. de Koning, and Alexander van Oudenaarden. A Single-Cell Transcriptome Atlas of the Human Pancreas. *Cell Systems*, 3(4):385–394.e3, 2016. ISSN 2405-4712.

[131] Blue B Lake, Rizi Ai, Gwendolyn E Kaeser, Neeraj S Salathia, Yun C Yung, Rui Liu, Andre Wildberg, Derek Gao, Ho-Lim Fung, Song Chen, et al. Neuronal subtypes and diversity revealed by single-nucleus RNA sequencing of the human brain. *Science*, 352(6293):1586–1590, 2016.

[132] Daniel T. Montoro, Adam L. Haber, Moshe Biton, Vladimir Vinarsky, Brian Lin, Susan E. Birket, Feng Yuan, Sijia Chen, Hui Min Leung, Jorge Villoria, Noga Rogel, Grace Burgin, Alexander M. Tsankov, Avinash Waghray, Michal Slyper, Julia Waldman, Lan Nguyen, Danielle Dionne, Orit Rozenblatt-Rosen, Purushothama Rao Tata, Hongmei Mou, Manjunatha Shivaraju, Hermann Bihler, Martin Mense, Guillermo J. Tearney, Steven M. Rowe, John F. Engelhardt, Aviv Regev, and Jayaraj Rajagopal. A revised airway epithelial hierarchy includes CFTR-expressing ionocytes. *Nature*, 560(7718):319, 2018. ISSN 1476-4687. doi: 10.1038/s41586-018-0393-7.

[133] Jingtao Guo, Xichen Nie, Maria Giebler, Hana Mlcochova, Yueqi Wang, Edward J. Grow, Robin Kim, Melissa Tharmalingam, Gabriele Matilionyte, Cecilia Lindskog, Douglas T. Carrell, Rod T. Mitchell, Anne Goriely, James M. Hotaling, and Bradley R. Cairns. The dynamic transcriptional cell atlas of testis development during human puberty. *Cell Stem Cell*, 26(2):262–276, 2020.

[134] Sabina Kanton, Michael James Boyle, Zhisong He, Malgorzata Santel, Anne Weigert, Fátima Sanchís-Calleja, Patricia Guijarro, Leila Sidow, Jonas Simon

Fleck, Dingding Han, Zhengzong Qian, Michael Heide, Wieland B. Huttner, Philipp Khaitovich, Svante Pääbo, Barbara Treutlein, and J. Gray Camp. Organoid single-cell genomic atlas uncovers human-specific features of brain development. *Nature*, 574(7778):418–422, 2019.

[135] David H Brann, Tatsuya Tsukahara, Caleb Weinreb, Marcela Lipovsek, Koen Van den Berge, Boying Gong, Rebecca Chance, Iain C Macaulay, Hsin-Jung Chou, Russell B Fletcher, Diya Das, Kelly Street, Hector Roux de Bezieux, Yoon-Gi Choi, Davide Risso, Sandrine Dudoit, Elizabeth Purdom, Jonathan Mill1, Ralph Abi Hachem, Hiroaki Matsunami, Darren W. Logan, Bradley J. Goldstein, Matthew S. Grub, John Ngai, and Sandeep Robert Datta. Non-neuronal expression of SARS-CoV-2 entry genes in the olfactory system suggests mechanisms underlying COVID-19-associated anosmia. *Science Advances*, 6 (31):eabc5801, 2020.

[136] Alyssa J Miller, Qianhui Yu, Michael Czerwinski, Yu-Hwai Tsai, Renee F Conway, Angeline Wu, Emily M Holloway, Taylor Walker, Ian A Glass, Barbara Treutlein, Grey Camp, and Jason R. Spence. In vitro and in vivo development of the human airway at single-cell resolution. *Developmental Cell*, 53(1): 117–128, 2020.

[137] Velina Kozareva, Caroline Martin, Tomas Osorno, Stephanie Rudolph, Chong Guo, Charles Vanderburg, Naeem M Nadaf, Aviv Regev, Wade Regehr, and Evan Macosko. A transcriptomic atlas of the mouse cerebellum reveals regional specializations and novel cell types. *bioRxiv*, 2020.

[138] Junyue Cao, Diana R. ODay, Hannah A. Pliner, Paul D. Kingsley, Mei Deng, Riza M. Daza, Michael A. Zager, Kimberly A. Aldinger, Ronnie Blecher-Gonen, Fan Zhang, Malte Spielmann, James Palis, Dan Doherty, Frank J. Steemers,

Ian A. Glass, Cole Trapnell, and Jay Shendure. A human cell atlas of fetal gene expression. *Science*, 370(6518):eaba7721, 2020.

[139] Grace X. Y. Zheng, Jessica M. Terry, Phillip Belgrader, Paul Ryvkin, Zachary W. Bent, Ryan Wilson, Solongo B. Ziraldo, Tobias D. Wheeler, Geoff P. McDermott, Junjie Zhu, Mark T. Gregory, Joe Shuga, Luz Montesclaros, Jason G. Underwood, Donald A. Masquelier, Stefanie Y. Nishimura, Michael Schnall-Levin, Paul W. Wyatt, Christopher M. Hindson, Rajiv Bharadwaj, Alexander Wong, Kevin D. Ness, Lan W. Beppu, H. Joachim Deeg, Christopher McFarland, Keith R. Loeb, William J. Valente, Nolan G. Ericson, Emily A. Stevens, Jerald P. Radich, Tarjei S. Mikkelsen, Benjamin J. Hindson, and Jason H. Bielas. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8:14049, Jan 2017.

[140] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[141] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010.

[142] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, 2007.

[143] J. A. John and Norman R. Draper. An alternative family of transformations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 29(2): 190–197, 1980.

[144] Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901.

[145] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

[146] Juexin Wang, Anjun Ma, Yuzhou Chang, Jianting Gong, Yuexu Jiang, Ren Qi, Cankun Wang, Hongjun Fu, Qin Ma, and Dong Xu. scGNN is a novel graph neural network framework for single-cell RNA-Seq analyses. *Nature Communications*, 12(1):1882, 2021.

[147] Bang Tran, Duc Tran, Hung Nguyen, Seungil Ro, and Tin Nguyen. scCAN: single-cell clustering using autoencoder and network fusion. *Scientific Reports*, 12:10267, 2022.

[148] Xiuwei Zhang, Chenling Xu, and Nir Yosef. Simulating multiple faceted variability in single cell RNA sequencing. *Nature Communications*, 10(1):1–16, 2019.

[149] Tianyi Sun, Dongyuan Song, Wei Vivian Li, and Jingyi Jessica Li. scDesign2: a transparent simulator that generates high-fidelity single-cell gene expression count data with gene correlations captured. *Genome Biology*, 22(1):1–37, 2021.

[150] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer, 2005.

[151] Zeynab Maghsoudi, Ha Nguyen, Alireza Tavakkoli, and Tin Nguyen. A comprehensive survey of the approaches for pathway analysis using multi-omics data integration. *Briefings in Bioinformatics*, 23(6):bbac435, 2022.

[152] Hung Nguyen, Duc Tran, Jonathan M. Galazka, Sylvain V. Costes, Afshin Beheshti, Sorin Draghici, and Tin Nguyen. CPA: A web-based platform for consensus pathway analysis and interactive visualization. *Nucleic Acids Research*, 49(W1):W114–W124, 2021.

[153] Hung Nguyen, Duc Tran, Bang Tran, Bahadir Pehlivan, and Tin Nguyen. A comprehensive survey of regulatory network inference methods using single-cell RNA sequencing data. *Briefings in Bioinformatics*, 22(3):1–15, 2021.

[154] Jovan Tanevski, Thin Nguyen, Buu Truong, Nikos Karaiskos, Mehmet Eren Ahsen, Xinyu Zhang, Chang Shu, Ke Xu, Xiaoyu Liang, Ying Hu, Hoang VV Pham, Li Xiaomei, Thuc D Le, Adi L Tarca, Gaurav Bhatti, Roberto Romero, Nestoras Karathanasis, Phillipe Loher, Yang Chen, Zhengqing Ouyang, Disheng Mao, Yuping Zhang, Maryam Zand, Jianhua Ruan, Christoph Hafemeister, Peng Qiu, Duc Tran, Tin Nguyen, Attila Gabor, Thomas Yu, Justin Guinney, Enrico Glaab, Roland Krause, Peter Banda, DREAM SCTC Consortium, Gustavo Stolovitzky, Nikolaus Rajewsky, Julio Saez-Rodriguez, and Pablo Meyer. Gene selection for optimal prediction of cell position in tissues from single-cell transcriptomics data. *Life Science Alliance*, 3(11), 2020. doi: 10.26508/lsa.202000867.

[155] Tuan-Minh Nguyen, Adib Shafi, Tin Nguyen, and Sorin Draghici. Identifying significantly impacted pathways: a comprehensive review and assessment. *Genome Biology*, 20:203, 2019.

[156] Adib Shafi, Tin Nguyen, Azam Peyvandipour, Hung Nguyen, and Sorin Draghici. A multi-cohort and multi-omics meta-analysis framework to identify network-based gene signatures. *Frontiers in Genetics*, 10:159, 2019.

[157] Adib Shafi, Tin Nguyen, Azam Peyvandipour, and Sorin Draghici. GSMA:

an approach to identify robust global and test Gene Signatures using Meta-Analysis. *Bioinformatics*, 36(2):487–495, 2019.

[158] Hung Nguyen, Sangam Shrestha, Duc Tran, Adib Shafi, Sorin Draghici, and Tin Nguyen. A comprehensive survey of tools and software for active subnetwork identification. *Frontiers in Genetics*, 10:155, 2019.

[159] Tin Nguyen, Cristina Mitrea, and Sorin Draghici. Network-based approaches for pathway level analysis. *Current Protocols in Bioinformatics*, 61(1):8–25, 2018.

[160] Edward Cruz, Hung Nguyen, Tin Nguyen, and Ian Wallace. Functional analysis tools for post-translational modification: a post-translational modification database for analysis of proteins and metabolic pathways. *The Plant Journal*, 99(5):1003–1013, 2019.

[161] Diana Diaz, Tin Nguyen, and Sorin Draghici. A systems biology approach for unsupervised clustering of high-dimensional data. In *The Second International Workshop on Machine Learning, Optimization and Big Data*, pages 193–203, 2016.

[162] Diana Diaz, Michele Donato, Tin Nguyen, and Sorin Draghici. MicroRNA-augmented pathways (mirAP) and their applications to pathway analysis and disease subtyping. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, volume 22, pages 390–401, New Jersey, 2017. World Scientific.

[163] Thair Judeh, Tin Chi Nguyen, and Dongxiao Zhu. QSEA for fuzzy subgraph querying of KEGG pathways. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pages 474–481, 2012.

[164] Tin Nguyen, Adib Shafi, Tuan-Minh Nguyen, A. Grant Schissler, and Sorin Draghici. NBIA: a network-based integrative analysis framework–applied to pathway analysis. *Scientific Reports*, 10:4188, 2020.

[165] Brian Marks, Nina Hees, Hung Nguyen, and Tin Nguyen. MIA: A Multi-cohort Integrated Analysis for biomarker identification. In *Proceedings of the 9th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*. ACM, 2018.

[166] Tin Nguyen, Cristina Mitrea, Rebecca Tagett, and Sorin Draghici. DANUBE: Data-driven meta-ANalysis using UnBiased Empirical distributions - applied to biological pathway analysis. *Proceedings of the IEEE*, 105(3):496–515, 2017.

[167] Tin Nguyen, Diana Diaz, Rebecca Tagett, and Sorin Draghici. Overcoming the matched-sample bottleneck: an orthogonal approach to integrate omic data. *Scientific Reports*, 6:29251, 2016. doi: 10.1038/srep29251.

[168] Tin Nguyen, Diana Diaz, and Sorin Draghici. TOMAS: A novel TOpology-aware Meta-Analysis approach applied to System biology. In *Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 13–22. ACM, 2016.

[169] Tin Nguyen, Rebecca Tagett, Michele Donato, Cristina Mitrea, and Sorin Draghici. A novel bi-level meta-analysis approach-applied to biological pathway analysis. *Bioinformatics*, 32(3):409–416, 2016.

[170] Rebecca J. Austin-Datta, Carlo La Vecchia, Thomas J George, Faheez Mohamed, Paolo Boffetta, Sean P. Dineen, Daniel Q. Huang, Thanh-Huyen T Vu, Tin C. Nguyen, Jennifer B Permuth, and Hung N. Luu. A call for standardized reporting of early-onset colorectal peritoneal metastases. *European Journal*

of Cancer Prevention: the Official Journal of the European Cancer Prevention Organisation (ECP), DOI: 10.1097/CEJ.0000000000000816, 2023.

[171] Quang-Huy Nguyen, Tin Nguyen, and Duc-Hau Le. Identification and validation of a novel three hub long noncoding RNAs with m6A modification signature in low-grade gliomas. *Frontiers in Molecular Biosciences*, 9:801931, 2022.

[172] Quang-Huy Nguyen, Tin Nguyen, and Duc-Hau Le. DrGA: cancer driver gene analysis in a simpler manner. *BMC Bioinformatics*, 23:86, 2022.

[173] Bashir Dabo, Claudio Pelucchi, Matteo Rota, Harshonnati Jain, Paola Bertuccio, Rossella Bonzi, Domenico Palli, Monica Ferraroni, Zuo-Feng Zhang, Aurora Sanchez-Anguiano, Yenh Thi-Hai Pham, Chi Thi-Du Tran, Anh Gia Pham, Guo-Pei Yu, Tin C. Nguyen, Joshua Muscat, Shoichiro Tsugane, Akihisa Hidaka, Gerson S. Hamada, David Zaridze, Dmitry Maximovitch, Manolis Kogevinas, Nerea Fernàndez de Larrea, Stefania Boccia, Robert C. Pastorino, Robertav; Kurtz, Areti Lagiou, Pagona Lagiou, Jesus Vioque, M. Constanza Camargo, Maria Paula Curado, Nuno Lunet, Paolo Boffetta, Eva Negri, Carlo La Vecchia, and Hung N. Luu. The association between diabetes and gastric cancer: results from the stomach cancer pooling project consortium. *European Journal of Cancer Prevention*, 31(3):260, 2022.

[174] Hung N. Luu, Pedram Paragomi, Renwei Wang, Joyce Y. Huang, Jennifer Adams-Haduch, Øivind Midttun, Arve Ulvik, Tin C. Nguyen, Randall E. Brand, Yutang Gao, Per Magne Ueland, and Jian-Min Yuan. The association between serum serine and glycine and related-metabolites with pancreatic cancer in a prospective cohort study. *Cancers*, 14(9):2199, 2022.

[175] Hung Nguyen, Duc Tran, Bang Tran, Monikrishna Roy, Adam Cassell, Sergiu Dascalu, Sorin Draghici, and Tin Nguyen. SMRT: Randomized data transfor-

mation for cancer subtyping and big data analysis. *Frontiers in Oncology*, 11: 725133, 2021.

[176] Thi Hai Yen Nguyen, Tin Nguyen, Quang-Huy Nguyen, and Duc-Hau Le. Re-identification of patient subgroups in uveal melanoma. *Frontiers in Oncology*, 11:731548, 2021.

[177] Duc Tran, Hung Nguyen, Uyen Le, George Bebis, Hung N. Luu, and Tin Nguyen. A novel method for cancer subtyping and risk prediction using consensus factor analysis. *Frontiers in Oncology*, 10:1052, 2020.

[178] Quang-Huy Nguyen, Hung Nguyen, Tin Nguyen, and Duc-Hau Le. Multi-omics analysis detects novel prognostic subgroups of breast cancer. *Frontiers in Genetics*, 11:1265, 2020. ISSN 1664-8021. doi: 10.3389/fgene.2020.574661.

[179] Suzan Arslanturk, Sorin Draghici, and Tin Nguyen. Integrated cancer subtyping using heterogeneous genome-scale molecular datasets. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, volume 25, page 551. World Scientific, 2020.

[180] Hung Nguyen, Bang Tran, Duc Tran, Quang-Huy Nguyen, Duc-Hau Le, and Tin Nguyen. Disease subtyping using community detection from consensus networks. In *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*, pages 318–323. IEEE, 2020.

[181] Hung Nguyen, Sushil J Louis, and Tin Nguyen. MGKA: A genetic algorithm-based clustering technique for genomic data. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 103–110. IEEE, 2019.

[182] Yan Yan, Tin Nguyen, Bobby Bryant, and Frederick C Harris Jr. Robust

fuzzy cluster ensemble on cancer gene expression data. In *Proceedings of 11th International Conference*, volume 60, pages 120–128, 2019.

[183] Yifan Zhang, Duc Tran, Tin Nguyen, Sergiu M Dascalu, and Frederick C. Harris. A robust and accurate single-cell data trajectory inference method using ensemble pseudotime. *BMC Bioinformatics*, 24(1):1–21, 2023.

[184] Duc Tran, Ha Nguyen, Hung Nguyen, and Tin Nguyen. Dwen: A novel method for accurate estimation of cell type compositions from bulk data samples. In *2022 14th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–6. IEEE, 2022.

[185] Duc Tran, Hung Nguyen, Bang Tran, Carlo La Vecchia, Hung N. Luu, and Tin Nguyen. Fast and precise single-cell data analysis using hierarchical autoencoder. *Nature Communications*, 12:1029, 2021.

[186] Attila Gabor, Marco Tognetti, Alice Driessen, Jovan Tanevski, Baosen Guo, Wencai Cao, He Shen, Thomas Yu, Verena Chung, Bernd Bodenmiller, Julio Saez-Rodriguez, Augustinas Prusokas, Alidivinas Prusokas, Renata Retkute, Anand Rajasekar, Karthik Raman, Malvika Sudhakar, Raghunathan Rengaswamy, Edward S.C. Shih, Min jeong Kim, Changje Cho, Dohyang Kim, Hyeju Oh, Jinseub Hwang, Kim Jongtae, Yeongeun Nam, Sanghoo Yoon, Taeyong Kwon, Kyeongjun Lee, Sarika Chaudhary, Nehal Sharma, Shreya Bande, Gao Gao fan zhu Cankut Cubuk, Pelin Gundogdu, Joaquin Dopazo, Kinza Rian, Carlos Loucera, Matias M Falco, Martin Garrido-Rodriguez, Maria Peña-Chilet, Huiyuan Chen, Gabor Turu, Laszlo Hunyadi, Adam Misak, Baosen Guo, Wencai Cao, He Shen, Lisheng Zhou, Xiaoqing Jiang, Pieta Zhang, Aakansha Rai, Rintu Kutum, Sadhna Rana, Rajgopal Srinivasan, Swatantra Pradhan, James Li, Vladimir Bajic, Christophe Van Neste, Didier Barradas-bautista, So-

mayah Abdullah Albarade, Igor Nikolskiy, Musalula Sinkala, Duc Tran, Hung Nguyen, Tin Nguyen, Alexander Wu, Benjamin DeMeo, Brian Hie, Rohit Singh, Jiwei Liu, Xueer Chen, Leonor Saiz, Jose M. G Vilar, Peng Qiu, Akash Gosain, Anjali Dhall, Dinesh Bajaj, Harpreet Kaur, Krishna Bagaria, Mayank Chauhan, Neelam Sharma, Gajendra Raghava, Sumeet Patiyal, Jianye Hao, Jiajie Peng, Shangyi Ning, Yi Ma, Zhongyu Wei, Atte Aalto, Jorge Goncalves, Laurent Mombaerts, Xinnan Dai, Jie Zheng, Piyushkumar Mundra, Fan Xu, Jie Wang, Krishna Kant Singh, and Mingyu Lee. Cell-to-cell and type-to-type heterogeneity of signaling networks: insights from the crowd. *Molecular Systems Biology*, 17:e10402, 2021. doi: 10.15252/msb.202110402. URL `https://doi.org/10.15252/msb.202110402`.

[187] Duc Tran, Bang Tran, Hung Nguyen, and Tin Nguyen. A novel method for single-cell data imputation using subspace regression. *Scientific Reports*, 12: 2697, 2022.

[188] Bang Tran, Quyen Nguyen, Sangam Shrestha, and Tin Nguyen. scids: Single-cell imputation by combining deep autoencoder neural networks and subspace regression. In *2021 13th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–8, 2021. doi: 10.1109/KSE53942.2021.9648664.

[189] Duc Tran, Frederick C Harris, Bang Tran, Nam Sy Vo, Hung Nguyen, and Tin Nguyen. Single-cell RNA sequencing data imputation using deep neural network. In *ITNG 2021 18th International Conference on Information Technology-New Generations*, pages 403–410. Springer, 2021.

[190] Duc Tran, Hung Nguyen, Frederick C. Harris, and Tin Nguyen. Single-cell rna sequencing data imputation using similarity preserving network. In *2021 13th*

International Conference on Knowledge and Systems Engineering (KSE), pages 1–6. IEEE, 2021.

[191] Amira Alotaibi, Tarik Alafif, Faris Alkhilaiwi, Yasser Alatawi, Hassan Althobaiti, Abdulmajeed Alrefaei, Yousef Hawsawi, and Tin Nguyen. Vit-deit: An ensemble model for breast cancer histopathological images classification. In *2023 1st International Conference on Advanced Innovations in Smart Cities (ICAISC)*, pages 1–6. IEEE, 2023.

[192] Benjamin T. Caswell, Caio C. de Carvalho, Hung Nguyen, Monikrishna Roy, Tin Nguyen, and David C. Cantu. Thioesterase enzyme families: Functions, structures, and mechanisms. *Protein Science*, 31(3):652–676, 2022.

[193] Evagelia C. Laiakis, Maisa Pinheiro, Tin Nguyen, Hung Nguyen, Afshin Beheshti, Sucharita M. Dutta, William K. Russell, Mark R. Emmett, and Richard Britten. Quantitative proteomic analytic approaches to identify metabolic changes in the medial prefrontal cortex of rats exposed to space radiation. *Frontiers in Physiology*, DOI: 10.3389/fphys.2022.971282, 2022.

[194] Amruta Kale, Tin Nguyen, Frederick C. Harris Jr, Chenhao Li, Jiyin Zhang, and Xiaogang Ma. Provenance documentation to enable explainable and trustworthy AI: A literature review. *Data Intelligence*, pages 1–41, 2022.

[195] Egle Cekanaviciute, Duc Tran, Hung Nguyen, Alejandra Lopez Macha, Eloise Pariset, Sasha Langley, Giulia Babbi, Sherina Malkani, Sébastien Penninckx, Jonathan C. Schisler, Tin Nguyen, Gary H. Karpen, and Sylvain V. Costes. Mouse genomic associations with in vitro sensitivity to simulated space radiation. *Life Sciences in Space Research*, DOI: 10.1016/j.lssr.2022.07.006, 2022.

[196] Quang Tran, Nam Sy Vo, Eric Hicks, Tin Nguyen, and Vinhthuy Phan. Anal-

ysis of short-read aligners using genome sequence complexity. In *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*, pages 312–317. IEEE, 2020.

[197] Michael P. Menden, Dennis Wang, Mike J. Mason, Bence Szalai, Krishna C. Bulusu, Yuanfang Guan, Thomas Yu, Jaewoo Kang, Minji Jeon, Russ Wolfinger, Tin Nguyen, Mikhail Zaslavskiy, AstraZeneca-Sanger Drug Combination DREAM Consortium, In Sock Jang, Zara Ghazoui, Mehmet E. Ahsen, Robert Vogel, Elias C. Neto, Thea Norman, Eric K. Y. Tang, Mathew J. Garnett, Giovanni Y. Di Veroli, Christian Zwaan, Stephen Fawell, Gustavo Stolovitzky, Justin Guinney, Jonathan R. Dry, and Julio Saez-Rodriguez. Community assessment to advance computational prediction of cancer drug combinations in a pharmacogenomic screen. *Nature Communications*, 10:2674, 2019.

[198] John C Stansfield, Duc Tran, Tin Nguyen, and Mikhail G Dozmorov. R tutorial: Detection of differentially interacting chromatin regions from multiple Hi-C datasets. *Current Protocols in Bioinformatics*, 66(1):e76–e76, 2019.

[199] Alfred G. Schissler, Hung Nguyen, Tin Nguyen, Juli Petereit, and Vincent Gardeux. *Statistical Software*, volume 10.1002/9781118445112.stat00527.pub2, pages 1–11. American Cancer Society, 2019.

[200] Adib Shafi, Cristina Mitrea, Tin Nguyen, and Sorin Draghici. A survey of the approaches for identifying differential methylation using bisulfite sequencing data. *Briefings in Bioinformatics*, 19(5):737–753, 2018.

[201] Tin Chi Nguyen and Dongxiao Zhu. Markovbin: An algorithm to cluster metagenomic reads using a mixture modeling of hierarchical distributions. In *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, pages 115–123, 2013.

[202] Zhiyu Zhao, Tin Chi Nguyen, Nan Deng, Kristen Marie Johnson, and Dongxiao Zhu. SPATA: a seeding and patching algorithm for de novo transcriptome assembly. In *2011 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW)*, pages 26–33. IEEE, 2011.

[203] Tin Chi Nguyen, Zhiyu Zhao, and Dongxiao Zhu. SPATA: A highly accurate GUI tool for de novo transcriptome assembly. In *2011 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW)*, pages 1051–1053. IEEE, 2011.

[204] Tin Chi Nguyen, Nan Deng, Guorong Xu, Zhansheng Duan, and Dongxiao Zhu. iQuant: A fast yet accurate GUI tool for transcript quantification. In *2011 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW)*, pages 1048–1050. IEEE, 2011.

Appendix A

Publication list

A.1 Journal articles

1. **Bang Tran**, Duc Tran, Hung Nguyen, Seungil Ro, and Tin Nguyen. scCAN: single-cell clustering using autoencoder and network fusion. *Scientific Reports*, 2022. DOI: 10.1038/s41598-022-06500-4.

2. Duc Tran, **Bang Tran**, Hung Nguyen, and Tin Nguyen. A novel method for single-cell data imputation using subspace regression. *Scientific Reports*, 2022. DOI: 10.1038/s41598-022-06500-4.

3. Duc Tran, Hung Nguyen, **Bang Tran**, Carlo La Vecchia, Hung N. Luu, and Tin Nguyen. Fast and precise single-cell data analysis using hierarchical autoencoder. *Nature Communications*. 2021. DOI: 10.1038/s41467-021-21312-2.

4. Hung Nguyen, Duc Tran, **Bang Tran**, Monikrishna Roy, Adam Cassell, Sergiu M Dascalu, Sorin Draghici, Tin Nguyen. SMRT: Randomized data transformation for cancer subtyping and big data analysis. *Frontiers in Oncology*, 2021. DOI: 10.3389/fonc.2021.725133.

5. Hung Nguyen, Duc Tran, **Bang Tran**, Bahadir Pehlivan, and Tin Nguyen. A comprehensive survey of regulatory network inference methods using single-cell RNA sequencing data. *Briefings in Bioinformatics*, 2020. DOI: 10.1093/bib/bbaa190.

A.2 Conference proceedings

1. Duc Tran, **Bang Tran**, Hung Nguyen, Frederick C. Harris, Nam Sy Vo, and Tin Nguyen. Single-cell RNA sequencing data imputation using deep neural network. In *Proceedings of the 18th International Conference on Information Technology-New Generations (ITNG)*, 2021.

2. **Bang Tran**, Quyen Nguyen, Sangam Shrestha, and Tin Nguyen. scIDS: Single-cell Imputation by combining Deep autoencoder neural networks and Subspace regression. In *Proceedings of the 13th International Conference on Knowledge and Systems Engineering (KSE)*, 2021.

3. Hung Nguyen, **Bang Tran**, Duc Tran, Quang-Huy Nguyen, Duc-Hau Le, and Tin Nguyen. Disease subtyping using community detection from consensus networks. In *Proceedings of the 12th International Conference on Knowledge and Systems Engineering (KSE)*, 2020.

4. **Bang Tran**, Duc Tran, Hung Nguyen, Nam Sy Vo, and Tin Nguyen. RIA: a novel regression-based imputation approach for single-cell RNA sequencing. In *Proceedings of the 11th International Conference on Knowledge and Systems Engineering (KSE)*, 2019.

ProQuest Number: 30572919

INFORMATION TO ALL USERS
The quality and completeness of this reproduction is dependent on the quality
and completeness of the copy made available to ProQuest.

Distributed by ProQuest LLC (2023).
Copyright of the Dissertation is held by the Author unless otherwise noted.

This work may be used in accordance with the terms of the Creative Commons license
or other rights statement, as indicated in the copyright statement or in the metadata
associated with this work. Unless otherwise specified in the copyright statement
or the metadata, all rights are reserved by the copyright holder.

This work is protected against unauthorized copying under Title 17,
United States Code and other applicable copyright laws.

Microform Edition where available © ProQuest LLC. No reproduction or digitization
of the Microform Edition is authorized without permission of ProQuest LLC.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346 USA