Title: Comparing and Evaluating Data Mining Techniques on RNA-Seq

Authors: Anirban Chowdhury, Daniel Nason

Department of Statistics and Data Science, Carnegie Mellon University

achowdh1@andrew.cmu.edu, dnason@andrew.cmu.edu

1 Introduction

A subset of molecular biology, gene expression is the process of how cells use information encoded within DNA to selectively activate a set of genes. DNA is transcribed into RNA and then translated into proteins, although the steps in this process are varied across different cell types as a result of gene regulation. Gene regulation determines when genes switch off and on, thus controlling cell differentiation and gene expression. Understanding gene expression allows for comparison of gene expression profiles from different tissues or the conditions that are needed to identify genes that impact how a phenotype is determined (Finotello 2015). Such information can provide insights about the genetic variables that reveal the state of a cell or the mechanisms underlying diseases.

The development of RNA sequencing (RNA-seq) has provided researchers with a direct way to measure gene expression. RNA-seq sequences short strings ("reads") from random positions in the RNAs present in the sample and maps them to a reference genome with the count of each gene to measure its expression (Finotello 2015). Zeisel (2015) utilized this technology to "perform a molecular census of the primary somatosensory cortex (S1) and the hippocampal CA1 region, based on 3005 single-cell transcriptomes." That is, the study utilized clustering techniques to discover distinct classes of cells based on data collected from 3,005 mouse brain cells. The authors developed BackSPIN (divisive biclustering method based on sorting points into neighborhoods (SPIN)) since standard hierarchical clustering methods contributed mostly noise to the clusters (Zeisel 2015). Seven distinct classes were identified: ependymal astrocytes, endothelial mural cells, interneurons, microglia, oligodendrocytes, pyramidal CA1 and SS cells (Zeisel 2015). 47 distinct subclasses were identified in these classes, and these were identified in multiple mice (Zeisel 2015). The findings illustrate the utility of single cell RNA-seq and how it can be employed to study the brain and other cell types.

In this paper, we compare various data mining techniques to determine what methodology best determines cell-type information using the Zeisel (2015) dataset with an emphasis on transferable applications. We approach the cell clustering problem from both a supervised and unsupervised approach. By exploring 3 well-studied dimensionality reduction techniques (Principal Component Analysis, sparse Principal Component Analysis, and Student's t-distributed Stochastic Neighbor Embedding) and 4 clustering techniques that have seen success in cell-type identification problems (k-Means Clustering, Expectation Maximization, Hierarchical Clustering, and Seurat), we determine what unsupervised methodology leads to the minimum misclustering rate. We also explore 3 common supervised statistical learning algorithms (logistic regression, LASSO logistic regression, and XGBoost) to predict cell type from RNA-seq count data. Our findings suggest possible applications in transfer learning, such as biologists using these models to automatically label cells from different RNA-seq data instead of labeling cell type by hand.

2 Methods

We used the dataset from Ziesel (2015) for this analysis, which consists of single-cell RNA-seq data from 3005 transcriptomes of various cell types in the primary somatosensory cortex and CA1 region of the hippocampus of mice. Counts of RNA molecules as a metric for gene expression were produced for 1000 genes for each transcriptome. The entries in the dataset represent the molecular count of the relevant RNA molecule for a specific gene and sample. The authors also released approximate labels for each sample. Before the data was used, we performed some normalization and scaling operations in order to reduce experimental errors and reduce skewness. To reduce any measurement or experimentation errors (Li 2020), we first performed library size normalization by dividing each count by the sum of counts of its row and multiplying by 10^4. Then, to reduce any skewness and center the data, we took a log base 2 transformation of the normalized count data (adding 1 to all values in the dataset before taking the log to avoid any numerical instability issues). The columns were then centered to have a mean of 0 and a standard deviation of 1.

We explored 3 different dimension reduction techniques in this work: PCA, Sparse PCA, and tSNE. The baseline method we experimented with was Principal Component Analysis (PCA). This method determines a set of orthogonal vectors that capture the most variation in the original data. The first step in the procedure is to obtain the sample correlation matrix for the dataset of interest (after centering scaling). Next, this matrix is represented in terms of its eigenvalues and eigenvectors. The eigenvector that corresponds to the largest eigenvalue defines the axis of the first principal component; the eigenvector that corresponds to the second largest eigenvalue defines the axis of the second principal component, and so on. This allows us to project the original dataset into a subspace of k dimensions that maximizes the variance in the projected axes (and in turn minimizes the reconstruction error when projecting back into the original space) by taking a matrix multiplication with the top k principal components. To determine the appropriate value of k for the Ziesel dataset, we created a Scree plot, which plots the square root of each eigenvalue against its index (ordered from largest to smallest) and visually inspected the graphic in order to determine what number of principal components contained most of the variation in the original data. Upon inspection, we determined that k=10 was most appropriate. Thus, we projected our original 1000-dimensional data into a subspace of only 10 dimensions to capture a large proportion of the variance in the original data.

One issue with PCA is that it can be susceptible to noise in the dataset. Each principal component generated by PCA can be expressed as a linear combination of the columns of the original dataset with weights determining how important each feature is to that component (called the factor loading). In practice, PCs often have similar noisy loadings assigned to all features in the data rather than a few large loadings indicating a small number of important features and many loadings near or exactly 0. Thus, PCA is sometimes unable to uncover hidden structure in the dataset due to the noise in the factor loadings. To remedy this, we employed a screening step to first eliminate columns of our dataset that were uninformative using Penalized Matrix Decomposition (PMD) (Witten 2009), and then repeated the PCA process above on the screened data. The PMD process relies on a penalty parameter chosen through 5-fold cross validation. Again, we examined a scree plot and decided to use the first 10 PCs.

Both PCA and sPCA are limited in that they only construct linear projections, and are thus unable to capture any nonlinear variation that could be lost in dimension reduction. Thus, we also explored a nonlinear dimension-reduction approach known as Student's t-distributed Stochastic Neighbor Embedding (tSNE). This approach learns a low dimensional space to transform the original data by first transforming the distances between points in the original space into conditional probabilities, i.e. the distance between two points is mapped to the probability that the points are neighbors. It then assumes that these probabilities are t-distributed both in the original space and in the smaller space that the method learns. Finally, the algorithm learns a set of points in a smaller space that minimizes the distance between the distribution of neighbor probabilities from the original space (in terms of KL-Divergence). The procedure takes in a perplexity parameter relating to the separation of groupings that the algorithm learns, with smaller perplexity values leading to more separated groupings. We used a perplexity parameter of 30, the default value, as initial experimentation with other values did not lead to improved results. This procedure is very computationally intensive and can only learn an output of at most 3 dimensions. To aid this process, we also included an initial PCA step to reduce the number of dimensions to 50 before applying the tSNE manifold learning procedure to arrive at a final 3 dimensions.

After first reducing the dimensions of the dataset using each of the three methods above, we attempted clustering with four different methods to compare performance and determine which best suited the dataset. We first experimented with k-means clustering, a simple, flexible approach to generate k clusters from the input data. Since we have 7 different cell types, we generated k=7 clusters. The algorithm proceeds by first randomly selecting k initial clusters and cluster centroids, then iteratively reassigning points to their closest centroids and updating centroids until convergence.

The EM algorithm is a model-based clustering algorithm in which data points are assumed to follow a multivariate normal distribution unique to each cluster, where cluster membership is a latent variable learned by the procedure. The algorithm proceeds by iteratively repeating an Expectation (E) step, in which probabilities of cluster membership conditional on Gaussian parameters are computed, and a Maximization (M) step, where the model parameters are updated to maximize the likelihood of cluster assignments using the values computed in the E step. The implementation of the EM algorithm for model based clustering used optimizes BIC, a function of log likelihood with a complexity penalty. It also takes in a parameter for the number of clusters to model.

Hierarchical clustering is a general agglomerative algorithm that begins with individual points as clusters and iteratively merges the two closest clusters together until only one cluster remains. How the clustering occurs depends both on how the distance is measured between clusters and the agglomeration method used to cluster the data. The distance matrix computations considered were Euclidean, Maximum, Manhattan, Canberra, Binary, and Minkowski; the agglomeration methods applied to these distance matrix computations included Ward’s method, single linkage, complete linkage, average linkage, Mcquitty’s method, median linkage, and centroid linkage. From the analysis we found that Ward’s method applied to the Manhattan distance matrix minimized the misclustering rate for all three of the dimension reduction techniques. The Manhattan distance is based on the sum of the absolute value of the distance between points (Black 2019), which makes it less sensitive to relatively large distances between two clusters compared to other methods that use squared distances between points . Ward’s method greedily chooses to minimize the distance between two clusters, which is defined as how much the sum of the squared distances between these clusters increases when they are merged(Ward 1963).

The final clustering method considered was a graph-based approach implemented in the Seurat software package. This algorithm works by constructing a nearest neighbors graph in the PCA space and adjusting edge weights to account for overlap in neighborhoods, i.e. edge weights are increased if two points share very close neighbors and decreased if this is not the case. Finally, a graph clustering algorithm (the Louvain method, discussed in Hoffman 2021) is used to generate cluster labels from the nearest neighbors graph. This algorithm takes in a resolution parameter that determines the granularity of the clusters and thus affects the resulting number of clusters. After some experimentation we found that a resolution of 0.12 allowed us to generate 7 clusters, which is the target number of cell types.

In addition to performing unsupervised learning, we also attempted supervised learning on the scaled Zeisel data set after applying each of the three dimension reduction techniques (PCA, sPCA, and tSNE). Because the response variable (the seven cell types) has multiple discrete values, classification analysis was performed from models developed utilizing logistic regression, LASSO logistic regression, and XGBoost.

Logistic regression is a type of generalized linear model that utilizes predictor variables to model the probability of a certain class or event existing. The more generalized version of the logistic regression model, multinomial logistic regression, is utilized since the response variable (cell types) has more than two values to account for these additional categories (Greene 2012). Logistic regression assumes that the log odds of a response variable being in a given class relative to the base class is linearly related to the explanatory variables in the model. The multinomial logit function is used to link these explanatory variables to the log odds of the response variable for a given category. Since logistic regression is not a closed-form expression, maximum likelihood estimation (MLE) is used to estimate the parameters in the model.

Penalized logistic regression was also used in the analysis to classify cell types. While the assumptions of this model are similar to those of logistic regression, penalized logistic regression also includes a constraint in the model which enforces a penalty for including additional terms based on the value of a tuning parameter. This penalty shrinks all coefficients in the model toward zero as the penalty increases to eliminate the effect of the variables that contribute less information toward predicting the outcome variable. The penalty associated with Least Absolute Shrinkage and Selection Operator (LASSO), also known as the L1-norm, is utilized for the regression analysis. Cross-validation was also utilized on the training data to choose the tuning parameter for the penalty. Since this is a random process, the tuning parameter selected is 1 standard error larger than the minimum value calculated for the parameter to avoid capitalization on chance.

In addition to logistic regression, we also employed XGBoost to classify the cell types in the data. XGBoost is a decision tree-based ensemble Machine Learning algorithm that uses weak learners (a single node decision tree that performs slightly better than random guessing) to predict the errors of other weak learners and combines these trees. This process is performed iteratively based on the values of various tuning parameters in order to make a strong learner model while avoiding overfitting the data in the model by minimizing the bias-variance tradeoff. These tuning parameters include the maximum number of boosting iterations and the maximum depth of a decision tree. Cross validation was also utilized on the training data to determine the optimal number of boosting iterations for the algorithm to perform in order to avoid overfitting the training data.

To assess the performance of these models, the data was split into 2 parts: 70% of the data was used to train the model and the remaining 30% was utilized to test model performance. We split the data into these subgroups to avoid overfitting the model to the noise of the original dataset and determine how the model performed when using data that it had never seen before. Models were assessed by examining their misclassification rates on the testing data. Misclassification rate is defined as the number of times the model incorrectly classified a cell type divided by the number of classifications the model performed. Since each of the three models were applied to the data using each of the three dimension reduction techniques, 9 total misclassification rates were calculated and compared to determine the model that performed the best at classifying cell types. We assessed the most important features of the data by inspecting the factor loadings of the PCs that were identified utilizing feature importance as indicated by the best performing XGBoost model.

3 Results

The first step in our analysis was to employ dimension reduction to determine the appropriate number of features to use. For PCA and sPCA, we used Scree plots to determine the number of principal components to include. For tSNE, we used 3 dimensions, as the computational complexity of the algorithm limits the number of possible output dimensions.

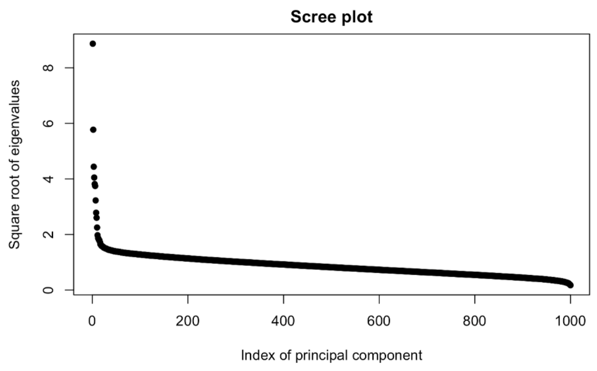

Figure 1: Scree plot of square root of eigenvalues for the principal components

This plot shows that the square roots of the first 10 eigenvalues seem much higher than the rest. So, for the PCA portion of the supervised and unsupervised learning analyses, we proceed with 10 principal components.

Additionally, we visualized the true cell types against the first two principal components to determine if they contain enough information to appropriately separate the data.

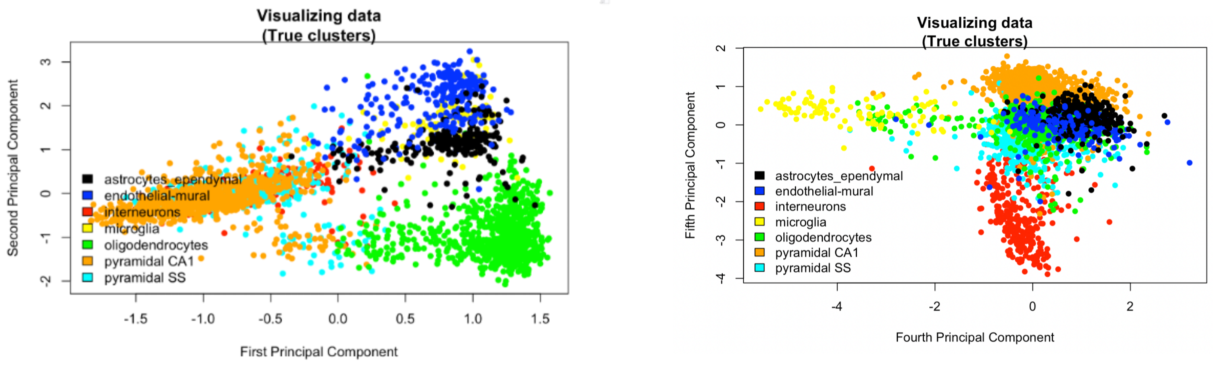

Figure 2: Visualization of Cell Type in PC Space

From the left plot above, it seems like the first two components do not contain enough information to accurately separate the different cell types. In particular, Pyramidal CA1 and Pyramidal SS seem to overlap almost entirely in this space. However, if we visualize the data against the fourth and fifth components, these two cell types do seem to form different clusters. Thus, it appears that the later principal components contain information that is not contained in the first two which could be useful in clustering. This supports our decision to use 10 of the top PCs rather than just the first few.

In order to determine if there was any noise present in the dataset that was impacting the clustering capacity of PCA, we tried sPCA and examined if any more relevant information was mined. We started with a Scree plot on the screened data after removing unimportant columns and again found the first 10 components to explain much more variation relative to the others. When visualizing the true cell types against the sPCA components we find the same pattern where the first 2 PCs do not separate the two Pyramidal cell types, but the fourth and fifth PCs do separate them. Thus, we can make the same conclusion that we did from PCA. Furthermore, this suggests that there was little noise in the dataset to negatively impact PCA, as the sPCA visualizations look very similar (omitted for conciseness).

In order to determine if there was any nonlinear relationship that could help us reduce dimension, we also employed tSNE to reduce the size of the data to 3 dimensions.

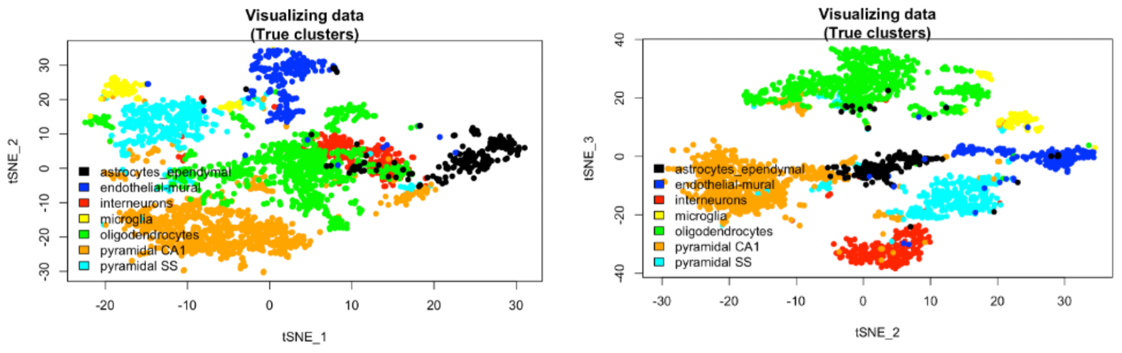

Figure 3: Visualization of Cell Type in tSNE Space

The first two tSNE components seem to separate the data slightly better than the first two PCA components, as the two Pyramidal cell types are now in different groupings. However, there are still a few instances of overlap. For example, the interneurons and oligodendrocytes seem to occupy the same areas of the plot. Overall, the separation is slightly improved compared to PCA. We can similarly visualize the second and third tSNE components, as shown in the right plot above. In this plot, the separation based on cell type looks more reasonable, with less general overlap. Overall, it appears that tSNE is more appropriate for dimension reduction as it visually appears to naturally separate the cells into groupings along all 3 axes. The next step in the analysis is to quantitatively evaluate these methods using both unsupervised and supervised learning algorithms and determine error rates.

Table 1: Misclustering Rates by Clustering Method & Dimension Reduction Technique

	k-means*	EM*	Hierarchical**	Seurat
PCA	0.238985	0.3148153	0.1810316	0.191015
Sparse PCA	0.236782	0.3175973	0.18203	N/A
tSNE	0.2456173	0.125198	0.1161398	N/A
*Misclustering rates calculated over 100 simulated trials.				
**Misclustering rates calculated using Manhattan distance and Ward's Method.				

Table 1 presents the misclustering rate for each combination of dimension reduction method and clustering method. Seurat's graph algorithm is only applicable on PCA, so the other cells in the table for this column are omitted. Additionally, since both k-means and EM are sensitive to initial values, their misclustering rates were averaged over 100 trials. For hierarchical clustering, the misclustering rates were calculated using Manhattan Distance and Ward's method to cluster the data points. From these results we see that EM performs the worst for both PCA and sPCA. However, it performs much better than k-means and comparably well to hierarchical clustering for tSNE. The most performant clustering method was Hierarchical, as it had the lowest rate for all dimensions. Seurat performed comparably well in the PCA space, so it was also able to capture cluster information from only the linear features from PCA. In terms of dimension reduction, the most performant technique seems to be tSNE. Although it had less features after reduction (3 as opposed to 10 for both PCA and sPCA), EM and hierarchical clustering both performed the best on the tSNE data, and k-means performed comparably well on tSNE as opposed to PCA and sPCA. In summary, the best performing method is tSNE dimension reduction and Hierarchical clustering.

Table 2: Misclassification Rates by Classification Method & Dimension Reduction Technique on Testing Data

	Logistic Regression	LASSO Logistic Regression	Gradient Boosting
PCA	0.03436807	0.04101996	0.0388027
Sparse PCA	0.03547672	0.03769401	0.0421286
tSNE	0.07649667	0.08203991	0.0521064

Table 2 presents the supervised misclassification rates for all model/dimension reduction pairs. The first observation we can make is that all models, even the baseline logistic regression model, perform extremely well. However, for tSNE, the models perform slightly worse, even though the clusters that the tSNE components generated looked more reasonable than the ones from PCA or sPCA. Our XGBoost model also allows us to extract information about the importance of certain genes in the prediction, i.e. we can determine what genes give the most information about cell type. Our best performing XGBoost model used PCA features, and upon inspection of the importance it assigned to all features, we found that PCs 1 and 5 were the most important. Upon further inspection of the factor loadings for these PCs, we determined that the top 5 genes with the highest absolute factor loadings for PC1 were Matr3, Tmem30a, Ube2b, Tro, Astn1, and Gpi1, and the top 5 for PC2 were Slc32a1, Dnm3, Sema3c, Rbms3, Npas1, and Ubash3b.

In summary, our analysis leads us to the conclusion that for supervised learning tasks, PCA and sPCA are more appropriate, and all modeling techniques are applicable because the statistical signal to determine cell type from gene expression seems straightforward to capture. One apparent discrepancy in the results is that tSNE seems to perform better for unsupervised tasks but slightly worse for supervised tasks.

4 Discussion

From our analysis, we found that after applying the dimension reduction techniques to the data, the misclassification rates were consistently lower than the misclustering rates. This is unsurprising, since the classification algorithms have access to the cell labels when performing their analyses unlike the clustering methods applied. Examining the clustering results, we find that in general, hierarchical clustering outperforms each of the other clustering methods regardless of the dimension reduction techniques used. This relatively strong performance compared to the other techniques, suggesting that it is an appropriate technique for single cell RNA-seq. However, it is important to note that the misclustering rates for hierarchical clustering are only smallest when using Manhattan distance and Ward's method to cluster the data. Using other combinations of distance and linkage methods resulted in higher misclustering rates in the data across each of the dimension reduction techniques.

Interestingly, EM performs remarkably better for tSNE compared to PCA and sPCA. This could be because the groupings that tSNE generates in its low dimensional space are closer to a gaussian distribution than the PCA or sPCA components. K-means also performs relatively poorly for all dimension reduction techniques; this may be because it is a rather naive clustering algorithm that does not accurately capture intricate groupings in multidimensional data, even though the dimensions are still small. Another noteworthy aspect of the clustering results is the relatively better performance of tSNE compared to PCA and sPCA across each of the clustering methods. This could be due to nonlinearity in the data that is better captured by the tSNE manifold but neglected by PCA and sPCA. The results could be a possible avenue for research extensions to determine whether tSNE features better fit the EM gaussian assumptions as opposed to PCA or sPCA, as well as its utility beyond generating clustering visualizations for unsupervised learning.

The classification results illustrate that all of the models tested performed very well in classifying the different cell types. Regardless of the dimension reduction technique applied, the misclassification rate did not exceed 10% for any of the models. This could indicate that the cell type has a very simple, possibly linear, statistical signal from gene expression, as all models performed comparably well when conducting analyses on both PCA and sPCA. However, the relatively poor performance of the models on tSNE suggests that there could be some overlap between clusters that could not be assessed visually. One possible explanation for this is that tSNE embeddings are mostly used for visualization rather than modeling, so they could be intricate and more difficult for models to capture (as evidenced by the fact that XGBoost performed better than logistic regression or LASSO logistic regression). It could also be the case that the difference in performance between the supervised learning methods on tSNE versus PCA and sPCA is due to noise rather than meaningful differences in performance, as the change in the classification error from tSNE to PCA or sPCA is still only around 3%. Relevant information could also be contained in the later PCs, since tSNE only included 3 dimensions as opposed to PCA and sPCA which each included 10.

There were two primary limitations of our work. The first is that we did not explore all possible dimension reduction techniques. In particular, Uniform Manifold Approximation and Projection (UMAP) is known to have computational benefits over tSNE. It is both faster to compute, allowing for more than 3 dimensions in the output space, and preserves structure of the original data better. It is possible that this method would allow us to extract more meaningful groupings from the data than the other methods we explored. Additionally, we only explored these methods on the Ziesel 2015 dataset, not any other RNA-seq data. We therefore cannot make broad conclusions about the efficacy of our models or approaches on all possible RNA-seq data. To get a better sense of how these methods perform in general, we could apply them to various other RNA-seq datasets to determine if the same conclusions (e.g. Manhattan/Ward hierarchical clustering works best, EM works best on tSNE embeddings, etc.) can be made on this other data.

One possible external application of our work is in transfer learning. This field of machine learning focuses on training an algorithm on one dataset and applying it to another. In context, we could apply any of our supervised learning models to new data as a labelling service. Biologists could, instead of labelling similarly collected cells by hand or using approximate methods, use our models on their new data to automatically generate labels with near-perfect accuracy. This could save both time and computational resources for researchers.

5 References

Francesca Finotello, Barbara Di Camillo, Measuring differential gene expression with RNA-seq: challenges and strategies for data analysis, Briefings in Functional Genomics, Volume 14, Issue 2, March 2015, Pages 130–142, https://doi.org/10.1093/bfgp/elu035

Greene, William H. (2012). Econometric Analysis (Seventh ed.). Boston: Pearson Education. pp. 803–806. ISBN 978-0-273-75356-8.

Hoffman P et al (2021), Seurat - A Guided Cell Clustering Tutorial. Satija Lab. Available from:https://satijalab.org/seurat/archive/v3.1/pbmc3k_tutorial.html

Li, X., Cooper, N.G.F., O'Toole, T.E. *et al.* Choice of library size normalization and statistical methods for differential gene expression analysis in balanced two-group comparisons for RNA-seq studies. *BMC Genomics* 21, 75 (2020). https://doi.org/10.1186/s12864-020-6502-7

Paul E. Black, "Manhattan distance", in Dictionary of Algorithms and Data Structures [online], Paul E. Black, ed. 11 February 2019. (accessed 12/12/2021) Available from: https://www.nist.gov/dads/HTML/manhattanDistance.html

Ward, J. H., Jr. (1963), "Hierarchical Grouping to Optimize an Objective Function", Journal of the American Statistical Association, 58, 236–244.

Witten D, Tibshirani R, Hassle T (2009), "A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis", Biostatistics, 3, 515-34.

Zeisel, A., Muñoz-Manchado, A. B., Codeluppi, S., Lönnerberg, P., La Manno, G., Juréus, A., ... & Linnarsson, S. (2015). Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. Science, 347(6226), 1138-1142.