Journal of High School Science

Exploring the effectiveness of spectral clustering on gene expression data

Warrier J[1], Mallery B[2]

Submitted: June 2, 2023, Revised: version 1, August 23, 2023, version 2, August 28, 2023
Accepted: September 5, 2023

Abstract

Analysis and understanding of gene expression data is essential to efficiently diagnose and treat diseases, without the need for high-cost screening tests. However, due to the complexity of the data, analyzing gene expression data requires the use of dimensionality reduction techniques to lower its complexity. While there have been several studies on the effectiveness of methods such as PCA, UMAP, and TSNE to gene expression data, this paper analyzed the effectiveness of spectral clustering, an algorithm that has been relatively unstudied in this field. This was achieved by optimizing the algorithm through two approaches: one that used several heuristic methods and another "experimental method" approach that employed quantitative clustering metrics. The performance of spectral clustering was then evaluated by comparing the results to those obtained with random assignment, K-Means, PCA, UMAP, and TSNE. Spectral clustering consistently performed better than random assignment and K-Means, indicating its potential in the field of gene expression. However, more work needs to be done to distinguish its performance from PCA, UMAP, and TSNE.

Keywords

Spectral clustering, Gene expression, Dimensionality reduction, Graph theory, Random projection, K-Nearest Neighbors, PCA, UMAP, TSNE, Eigengap

---

[1]Corresponding author, Jay Warrier, Khan Lab School, 1200 Villa St, Mountain View, CA 94041, USA. jayswarrier@gmail.com
[2]Brendan Mallery, Tufts University, 419 Boston Ave, Medford, MA 02155, USA. brendan.mallery@tufts.edu

I. Introduction

The advent of gene expression data poses great potential in the field of healthcare. Linking expression data with disease susceptibility could allow doctors to efficiently diagnose and administer treatment to an individual. The first part of this process is to create a co-expression network of genes, constructed from relationships found in the data. However, despite an abundance of such relationships, the complex structure of the dataset makes it difficult to infer them. One potential solution is to reduce the complexity of the data through dimensionality reduction. There have been several studies about the performance of DR methods such as PCA, UMAP, and TSNE on clustering gene expression data, all with relatively positive results (1-2). This paper explores a different type of dimensionality reduction method called Spectral Clustering. The ultimate goal was to study how Spectral Clustering assists in the understanding of how gene expression affects disease susceptibility. Towards this, one of the goals of this paper is an initial investigation of this method's effectiveness towards this classification task, with the hope that this analysis will help determine its effectiveness against other widely-used models in gene expression analysis, where performance is evaluated using standard metrics for unsupervised clustering. A main contribution is applying a comparative analysis to a novel dataset, on which such a comparison has yet to be applied.

There have been several papers and resources about the effectiveness of Spectral Clustering, with several examples of how it performs better than typical k-means clustering on data on irregular, non-convex shapes, such as concentric ellipses. This sets Spectral Clustering apart with more elementary methods such as k-means, which typically perform better when clusters are assumed to satisfy strong regularity properties, such as convexity (3). Figures 1.1-1.2 provide a demonstration of how Spectral Clustering outperforms k-means on an artificial, non-convex data set, namely a pair of concentric ellipses**.** Many popular accounts and implementations of Spectral Clustering exist (4-6), which mostly show the efficacy of Spectral Clustering on idealized, artificial data sets, or datasets with clearly separated groups. Spectral Clustering has also been applied to complex, high dimensional data sets, including gene expression datasets (7-9). This work adds to this body of work by applying and evaluating the performance of Spectral Clustering on the Genotype-Tissue Expression Project Version 9 data set (10), a state-of-the-art public repository of tissue-specific gene expression data.

Implementing Spectral Clustering requires the choice of a number of hyperparameters, and despite much work in the area there is no definitive understanding of how different choices of these hyperparameters affect the output clustering scheme, or how to choose hyperparameters for a given data set. A secondary goal of this paper is to explore how variation of hyperparameters affects qualitative and qualitative aspects of the obtained clustering. This was done through a systematic study of how varying hyperparameters affects quantitative and qualitative properties of the clustering schemes.

II. Methods

Spectral Clustering takes in an **n**-dimensional dataset and outputs a set of labels for the data-set, which are identified as clusters. At a high level, the Spectral Clustering algorithm operates by representing the dataset as a network, which is then used to construct a new embedding of the dataset into d-dimensional Euclidean space using the graph Laplacian of the network, which is then followed by performing a standard clustering method on this embedding (such as k-means). The user must specify several hyperparameters before implementing the algorithm, specifically the number of clusters and the dimension of the secondary embedding. Furthermore, the construction of the network from the dataset requires a user-specified subroutine, such as k-nearest or epsilon-nearest neighbors, which require their own choice of hyperparameters as well.

Spectral Clustering is designed to capture non-linear structure in datasets. Indeed, the construction of the secondary embedding using the graph Laplacian is motivated by a long line of work on non-linear embeddings of datasets using spectral properties of graphs (11)**.** This is exemplified by its performance in clustering datasets which appear to lie on non-linear manifolds, which can be challenging for more elementary methods. For instance, Figures 1.1-1.2 show that Spectral Clustering is able to separate a pair of nested concentric ellipses, while k-means is not. The key feature to notice is that k-means in Euclidean space separates clusters in a *linear* fashion, i.e. by finding suitable separating hyperplanes. In the example with the ellipses, one easily sees that no choice of hyperplane (in this case, no line) will be able to separate one ellipse from the other. As gene expression data is assumed to live on a highly non-linear manifold structure (as evidenced by e.g. (7)), it is natural to investigate how Spectral Clustering will perform on such data.

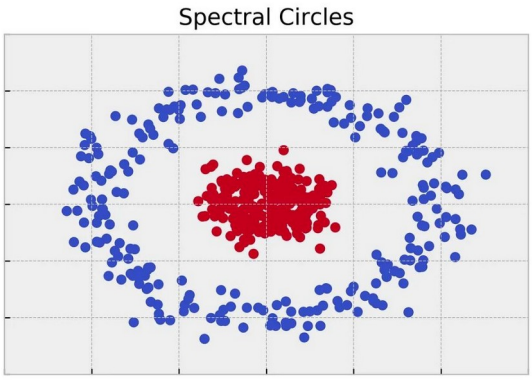

Figure 1.1-Spectral clustering on concentric circles

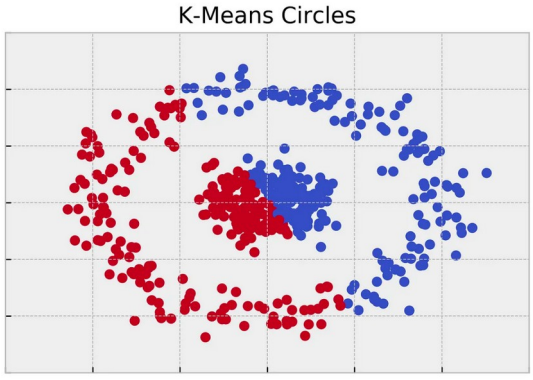

Figure 1.2-K-means clustering on concentric circles

A detailed overview of the Spectral Clustering algorithm is now presented: The first step is creating a network of the data points. The most common method is a k-nearest neighbors graph (12), where each point is connected to its k-closest neighbors, where k is a user chosen parameter and closest is in terms of Euclidean distance in n-dimensions. Alternatively, one may apply k-nearest neighbors using an alternative definition of "closest'', for instance using a radial basis function applied to the Euclidean distance between points, as is done in the Sci-kit Learn (sklearn) implementation of Spectral Clustering (13). Once this network is constructed, the algorithm then creates an affinity (or similarity) matrix, where rows and columns represent nodes, and individual values represent whether or not there is a connection. For example, if the entry in the third row and first column is 1, points 1 and 3 are connected. It also creates a degree matrix, a diagonal matrix representing the number of connections of each point, so the value in the first row and first column represents the number of connections to the first point. All non-diagonal values in the degree matrix are zero.

With these two matrices, the algorithm computes the graph Laplacian by subtracting the adjacency from the degree matrix. The graph Laplacian is an important tool in the field of graph theory, as it contains significant information about the structure of a graph (14, Section 3). The eigenvectors of the graph Laplacian are then used to compute a new embedding of the data set (15). The number of eigenvectors **s** used is a user-specified parameter, and the algorithm uses the eigenvectors with the **s** smallest eigenvalues, indicating this choice allows one to reduce the dimension of the dataset from **n** dimensions to **s** dimensions. The algorithm then applies an additional clustering algorithm, for instance k-means, to cluster the data. This discussion is summarized in Figure 2, where Spectral Clustering is formally defined.

The Genotype-Tissue Expression Project, launched in 2012 (16), has produced several gene-expression datasets for analysis. In May 2022, they published their ninth version (10), allowing users to model expression data across multiple tissues, as opposed to the single-tissue datasets created before. The dataset was annotated (17), meaning it contained several components of gene and sample information that were unnecessary for analysis (Figure 3). Because of this, the raw, numerical expression data, or ".X", was singled out, and the other components were removed. This resulted in a dataset of 200,000 genes and 30,000 columns. In order to make the dataset more tractable for analysis, the dimension of the data was reduced using random projection (18), a dimensionality reduction technique to simplify the data at the cost of a small part of the variation. This was done using the random projection function in Sci-Kit Learn (19), which is a standard dimensionality reduction technique used to lower the dimension of a dataset while preserving most of the information. The number of columns in the dataset was reduced to 10,000, while retaining at least 90% accuracy of the relative distances of the data points (for more information on the error of random projection, see (20)). As the dataset was still too large to naively run the analysis, a subsampling routine was built into the method,

which sampled 1000 uniformly random genes. Because a small portion of the data was subsampled, each trial involved 200 iterations of 1000 randomly sampled genes, which covered the whole dataset in the best case scenario.

Algorithm 1: Spectral clustering (with Euclidean distance similarity)

Data: $m \times n$ data matrix X, nearest neighbor parameter, $k > 0$, embedding dimension d, number of clusters K.
Compute Euclidean distance matrix for $\{x_1, x_2, ..., x_m\}$ (rows of X);
Compute k-nearest neighbor graph Γ;
Compute A_Γ, the graph Laplacian of Γ;
for $1 \leq i \leq d$ **do**
  $y_i \leftarrow i$'th eigenvector of A_Γ;
end
$Y_d \leftarrow [y_1, y_2, ..., y_d]$;
for $1 \leq j \leq n$ **do**
  $x'_i \leftarrow j$'th row of Y_d;
end
Compute K-means clustering on $\{x'_1, x'_2, ..., x'_n\}$;
for $1 \leq j \leq n$ **do**
  Label x_i with label of x'_i;
end
Result: Labeling of dataset m data points in X into K clusters

Figure 2 – Outline of spectral clustering

As mentioned previously, Spectral Clustering requires the choice of numerous hyperparameters, namely the number of neighbors for the adjacency matrix, the number of eigenvectors to take from the Laplacian, and the number of clusters for the final k-means portion, finding ones that lead to the best clustering. However, since the goal of this project is to discover new clusters of genes, unlabeled clustering was performed, meaning there were no ground truth values available to evaluate performance. To remedy this, two methods were employed: a heuristic approach, and an optimization based approach.

The first step in the heuristic approach was choosing the appropriate number of neighbors for the Laplacian. One key feature of the graph Laplacian is that its number of zero-eigenvalues (eigenvalues with a value of 0) correspond to the number of connected components in the network (14, Section 3.1, Proposition 2). The minimum number of neighbors that formed one connected component was chosen, meaning that no data points would be isolated from the main network. Equivalently, the minimal number of neighbors such that the connectivity matrix had only one zero-eigenvalue was selected, as shown in Figure 4. This was done by calculating the mean number of zero eigenvalues, or connected components, of 200 graph Laplacians with each neighbor amount from 10 to 400 (in increments of 10), and finding the first neighbor amount where the number of connected components fell below 1.5, which indicated that a majority of the Laplacians had one connected component.

Next, to determine the optimal number of eigenvectors to take from the Laplacian, the eigengap heuristic was implemented, which relies on matrix perturbation theory: Perturbation theory studies how the eigenvectors and eigenvalues of matrices change under small perturbations by other matrices. A hallmark theorem is the Davis-Kahan theorem (14, Theorem 7), which controls the distance between eigenspaces of matrices that differ by a perturbation, in terms of the size of the perturbation and the spectral information about the original (unperturbed) matrix. In the context of Spectral Clustering, this gives rise to the following heuristic for eigenvector selection: Suppose that a data set D has k “true” clusters. Then the first k-largest eigenvectors of the "true" graph Laplacian L obtained from D should correspond to the indicator vectors of these clusters. If one assumes that, due to noise in the data, or perhaps errors introduced in the construction of the similarity matrix, the Laplacian is instead L'=L+R for some noise matrix R. If R is a small perturbation, it is expected that the first k eigenvectors of L' are similar to the first k eigenvectors of L, and hence should be close to the indicator vectors for the underlying clusters. The Davis-Kahan theorem says that there is a suitable notion of distance (the "canonical angles", see (21)) between the eigenspaces spanned by the first k eigenvectors of L and the first k eigenvectors of L' that is bounded above by $\frac{\vee R \vee}{\lambda_{k+1} - \lambda_k \vee}$. Here, ||R|| denotes the Frobenius norm of the perturbation R, and $\lambda_{k+1} - \lambda_k \vee$ is the gap between the k and (k+1) largest eigenvalues of L. Furthermore, if k clusters are actually present, then the first k eigenvalues $\lambda_1, \ldots, \lambda_k = 0$. The combination of these two observations is that, when working with L', one should choose k so that the gaps between $\lambda_1, \ldots, \lambda_k$ are small and the gap between λ_k and $\lambda_{k+1}$is large, which by the Davis-Kahan theorem would give an eigenspace that is close to the eigenspace spanned by the "true" number of clusters. Simply put, the number of eigenvectors before the large gap represents both the number of clusters and eigenvectors to use.

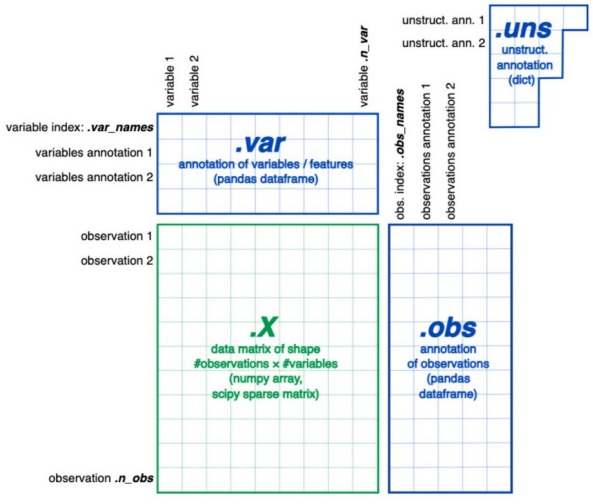

Figure 3 - Layout of the annotated data

While this is relatively easy to determine visually (Figure 4), since the algorithm requires 200 trials, a quantitative method was needed to determine the eigengap. Since the gap strongly resembles an inflection point, where the second derivative changes from positive to negative, this could be accomplished using a second-derivative approximation algorithm (Algorithm 2). This is done by looking at the absolute difference between adjacent eigenvalues, and finding the point where it decreases. A visual representation can be seen in Figure 5.

An alternative heuristic method was used to determine the optimal number of clusters, and implemented the elbow method (22), where the ideal number of clusters is where the plot for a cost function (in this case, the Within-Cluster Sum of Squares, or WCSS) starts to decrease in a linear fashion. This was done by running k-means 200 times at every cluster amount from 2 to 20, and plotting the mean WCSS per cluster amount to find the elbow (Figure 6).

The experimental method was performed by testing every combination of hyperparameters to find optimal choices. Since optimizing for all three parameters was computationally expensive, the number of neighbors was fixed using the aforementioned approach in the heuristic method.

Due to the lack of ground-truth labeling, each model was evaluated through their Silhouette, Calinski-Harabasz (CH), and Davies-Bouldin (DB) scores, metrics which primarily operate on comparing within-cluster to between-cluster variation (further described in the results section).

To find the optimal number of clusters and components, 200 iterations of each combination of clusters and components from 2-16 were run, and the mean of each of the three scores was calculated (Figures 7.1-7.3). Since the metrics were on different scales, z-score normalization was applied to each of the tables. Subsequently, the tables were summed up to create an aggregate z-score table (Table 1), with the optimal combination being the cell with the highest score.

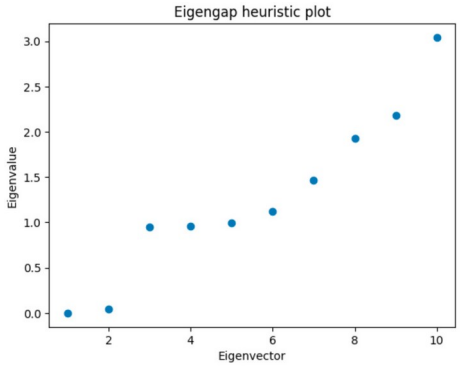

Figure 4 - First 10 eigenvalues of a Laplacian created with 385 neighbors - Note 1 zero-eigenvalue and a significant gap after the first 2 eigenvalues.

Algorithm 2: Second-derivative approximation

Data: $n \times m$ data array $\mathcal{D}$;
Subsample count k;
Number of neighbors R;
Iterations p.
$Y \leftarrow$ empty array
for $1 \leq j \leq p$ **do**
  Uniformly sample $X_1, X_2, ..., X_k \sim \mathcal{D}$
  $X \leftarrow [X_1, X_2, ..., X_k]$
  $\mathcal{G} \leftarrow R$-nearest neighbor graph of X
  Compute Laplacian $\Delta_{\mathcal{G}}$
  Compute (sorted) eigenvalues $\{\lambda_1, \lambda_2, ..., \lambda_k\}$ of $\Delta_{\mathcal{G}}$ and associated eigenvectors $\{v_1, ..., v_k\}$.
  for $1 \leq \ell \leq k$ **do**
    $\delta_i \leftarrow \lambda_{i+1} - \lambda_i$
    if $\delta_{\ell+1} < \delta_\ell$ **then**
      # components$\leftarrow \ell$
      # components appended to Y.
    end
  end
end
Average $\mathcal{A} \leftarrow \frac{1}{|Y|} \sum_i Y_i$
Data: Result: $\mathcal{A}$.

Figure 5 – Outline of Algorithm 2

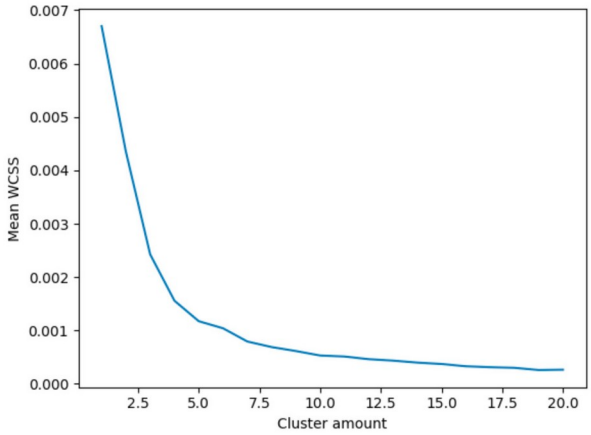

Figure 6 - Mean WCSS plot of 200 K-Means algorithms, clustering the first 2 eigenvectors of a Laplacian created with 385 neighbors

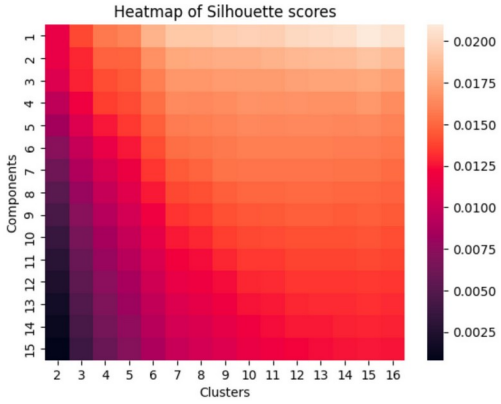

Figure 7.1 - Mean Silhouette Coefficient of 200 Spectral Clustering algorithms vs. cluster amount and number of components (Laplacian was created with 385 neighbors)

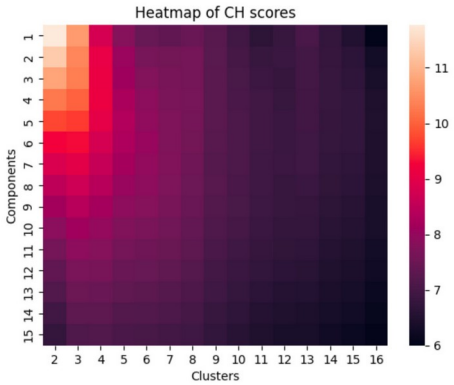

Figure 7.2 - Mean CH Coefficient of 200 Spectral Clustering algorithms vs. cluster amount and number of components (Laplacian was created with 385 neighbors)

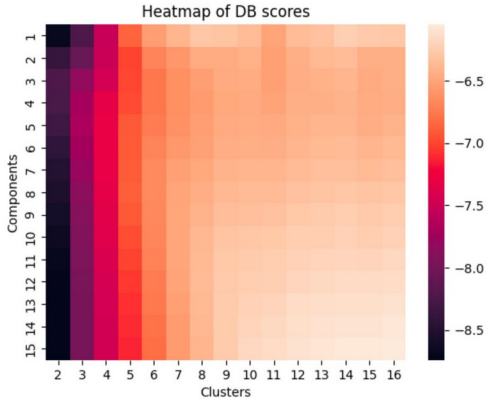

Figure 7.3 - Mean DB Coefficient of 200 Spectral Clustering algorithms vs. cluster amount and number of components. The scale is inverted since lower scores indicate better performance. (Laplacian was created with 385 neighbors)

Table 1 - Aggregate Z-Scores for Spectral Clustering. Each algorithm had an input of 385 neighbors

Components /Clusters	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	1.34	1.47	1.00	1.03	1.64	2.02	2.32	2.11	1.80	1.27	1.78	2.09	1.99	1.78	1.18
3	1.28	1.19	1.05	0.82	1.24	1.63	1.84	1.51	1.32	1.03	1.24	1.52	1.39	1.19	0.73
4	0.85	1.40	1.04	0.75	1.11	1.43	1.50	1.30	1.11	0.83	0.92	1.17	1.05	0.84	0.49
5	-0.15	1.04	1.11	0.81	1.06	1.36	1.35	1.14	1.01	0.78	0.83	0.94	0.82	0.75	0.36
6	-1.10	0.45	0.75	0.87	1.04	1.27	1.25	1.03	0.92	0.73	0.74	0.88	0.68	0.59	0.29
7	-2.00	-0.27	0.25	0.58	1.01	1.17	1.11	0.95	0.87	0.69	0.69	0.78	0.60	0.50	0.22
8	-2.80	-0.95	-0.22	0.28	0.85	1.09	0.99	0.88	0.81	0.63	0.64	0.71	0.52	0.43	0.16
9	-3.52	-1.59	-0.62	-0.02	0.59	0.92	0.86	0.77	0.71	0.53	0.57	0.60	0.44	0.35	0.12
10	-4.15	-2.15	-1.07	-0.39	0.28	0.69	0.71	0.66	0.59	0.45	0.47	0.50	0.35	0.28	0.06
11	-4.72	-2.64	-1.46	-0.77	-0.04	0.45	0.52	0.49	0.46	0.34	0.37	0.38	0.25	0.19	-0.02
12	-5.21	-3.07	-1.85	-1.15	-0.37	0.17	0.29	0.32	0.32	0.21	0.25	0.25	0.13	0.07	-0.11
13	-5.67	-3.47	-2.22	-1.51	-0.69	-0.13	0.04	0.10	0.15	0.08	0.12	0.13	0.01	-0.04	-0.21
14	-6.07	-3.82	-2.56	-1.84	-1.00	-0.43	-0.20	-0.11	-0.03	-0.07	0.00	-0.01	-0.10	-0.15	-0.31
15	-6.42	-4.14	-2.85	-2.14	-1.30	-0.71	-0.45	-0.33	-0.21	-0.20	-0.13	-0.15	-0.21	-0.27	-0.41
16	-6.72	-4.45	-3.13	-2.43	-1.59	-0.97	-0.69	-0.55	-0.38	-0.36	-0.28	-0.28	-0.35	-0.39	-0.53

A comparison between Spectral Clustering and other industry-standard algorithms was performed, namely random assignment (which served as a control), k-means, PCA, UMAP, and TSNE. For each algorithm, the mean of each score from 200 trials at each cluster level from 2 through16 was calculated. To find the optimal number of clusters, the same aggregate z-score method that was implemented in Spectral Clustering was used (Tables 2.1-2.4). The final evaluation table consists of each method's Silhouette, CH, DB, and aggregate z-scores (Table 3).

Table 2.1 - Evaluation Metrics of PCA

Clusters/Metric	Silhouette	CH	DB	Aggregate Z-Score
2	0.007	12.38	7.87	1.90
3	0.009	11.61	7.97	1.50
4	0.012	10.75	7.80	1.59
5	0.012	10.68	7.47	2.19
6	0.013	10.97	7.70	2.25
7	0.015	10.21	7.48	2.29
8	0.015	9.22	6.83	2.30
9	0.016	8.21	6.74	1.56
10	0.015	8.70	7.69	0.61
11	0.015	8.24	6.64	1.54
12	0.015	7.85	6.60	1.14
13	0.015	7.44	6.56	0.76
14	0.013	7.07	6.58	-0.06
15	0.013	6.73	6.61	-0.57
16	0.012	6.48	6.53	-0.79

Table 2.2 - Evaluation Metrics of UMAP

Clusters/Metric	Silhouette	CH	DB	Aggregate Z-Score
2	0.010	10.11	9.69	-2.25
3	0.009	9.84	8.53	-1.08
4	0.010	9.40	7.85	-0.24
5	0.013	9.46	7.30	1.24
6	0.015	9.24	6.93	2.10
7	0.017	8.63	6.64	2.39
8	0.018	7.81	6.54	2.06
9	0.019	7.66	6.43	2.35
10	0.019	7.92	6.50	2.44
11	0.019	7.52	6.41	2.18
12	0.019	7.15	6.42	1.72
13	0.018	6.82	6.45	1.17
14	0.017	6.49	6.44	0.47
15	0.016	6.27	6.41	0.15
16	0.015	6.06	6.40	-0.48

Table 2.3 - Evaluation Metrics of TSNE

Clusters/Metric	Silhouette	CH	DB	Aggregate Z-score
2	0.010	10.77	9.59	-1.39
3	0.010	9.72	8.65	-1.22
4	0.011	9.33	7.95	-0.17
5	0.013	9.28	7.32	1.25
6	0.015	9.13	6.90	1.99
7	0.016	8.62	6.77	2.13
8	0.016	7.92	6.41	1.88
9	0.017	7.36	6.31	1.61
10	0.017	7.58	6.35	1.76
11	0.017	7.45	6.40	1.49
12	0.017	7.16	6.40	1.20
13	0.016	6.82	6.34	0.89
14	0.017	6.49	6.39	0.54
15	0.017	6.28	6.25	0.56
16	0.017	6.04	6.32	0.22

Table 2.4 - Evaluation Metrics of k-means

Clusters/Metric	Silhouette	CH	DB	Aggregate Z-score
2	0.0063	11.23	7.33	1.43
3	0.0088	10.73	7.60	1.17
4	0.0114	8.75	7.83	-0.58
5	0.0134	9.72	7.47	1.48
6	0.0140	7.99	7.16	0.27
7	0.0155	8.34	6.96	1.32
8	0.0155	8.29	6.80	1.49
9	0.0162	8.18	6.80	1.56
10	0.0149	8.67	6.83	1.71
11	0.0145	8.30	6.89	1.13
12	0.0158	7.85	6.63	1.37
13	0.0140	7.41	6.66	0.38
14	0.0139	7.08	6.45	0.35
15	0.0126	6.82	6.60	-0.48
16	0.0121	6.52	6.56	-0.85

Table 3 - Comparison of the highest z-scores of each model

Method/Metric	Silhouette	CH	DB	Aggregate Z-score
Spectral Clustering (Experimental, 385 neighbors, 2 components, 8 clusters)	0.0192	7.45	6.30	2.32
Spectral Clustering (Heuristic Method #1, 385 neighbors, 2 components, 2 clusters)	0.0112	7.85	7.60	-1.23
Spectral Clustering (Heuristic Method #2, 385 neighbors, 2 components, 7 clusters)	0.0194	7.43	6.33	2.30
PCA (8 clusters)	0.015	9.22	6.83	2.30
UMAP (10 clusters)	0.019	7.92	6.50	2.44
TSNE (7 clusters)	0.016	8.62	6.77	2.13
k-means (10 clusters)	0.0149	8.67	6.83	1.71
Random Assignment	-0.001	1.23	10.89	-16.08

III. Results and Discussion:

The first experiment performed was an evaluation of Spectral Clustering's performance with hyperparameters chosen via commonly used heuristics. To determine the number of neighbors, the one connected component heuristic was used (14, Section 3.1, Proposition 2), and calculated the mean number of connected components (over 200 trials) for each chosen number of neighbors, which ranged from 0 to 400 in multiples of 10. Then the lowest number of neighbors such that the mean number of connected components was lower than 1.5, which indicated that a majority

of the graphs constructed had a single connected component, was selected. This threshold was crossed between 380 (1.53 connected components) and 390 (1.46 connected components) neighbors, so 385 neighbors was chosen as the optimal value. To choose the number of components and clusters, the eigengap heuristic was applied, as defined in the Methods section. To locate the eigengap, the second derivative test was applied. Out of 200 Laplacians, the average position of the eigenvector immediately preceding the eigengap was 2.39, which was rounded to 2, the nearest whole number. An example of the eigenvalues from a randomly generated Laplacian with 385 neighbors can be seen in Figure 4, and it demonstrates both of the properties mentioned above, having one connected component (zero eigenvalue) and a significant gap between the 2nd and 3rd eigenvalue. This indicated that the optimal number of eigenvectors and clusters was 2, but this ended up performing relatively poorly with respect to the metrics (results shown in Table 1 and elaborated further later in the section). Because of this, the cluster amount was re-determined using the elbow method (22), by summing up the WCSS of 200 k-means algorithms for each cluster amount from 1 to 20, and looking for the point when the decrease in WCSS appears to become linear. Figure 6 shows that the decrease in WCSS appears to become linear at 7 clusters, and hence this was taken to be the optimal number.

Next, the hyperparameters were optimizing experimentally. As mentioned in the Methods section, the hyperparameters were evaluated using three metrics: the Silhouette coefficient, the Calinski-Harabasz (CH) index, and the Davies-Bouldin (DB) index (23). The Silhouette Coefficient calculates the difference of the within-cluster and between-cluster Euclidean distance of each point. The coefficient ranges from -1 to 1, with a higher number representing dense and well-separated clusters. The CH Index computes the ratio between within-cluster and between-cluster variation. Similar to the Silhouette Coefficient, a higher number indicates a better model. Finally, the Davies-Bouldin Index (DB), compares similarity between clusters with the size of the clusters themselves. Unlike the first two metrics, the DB index should be minimized, as a lower score means that there is lower similarity between clusters, indicating the clusters are better separated.

To significantly reduce the number of computations, 385 neighbors was fixed as the k-value for consistency with the heuristic method. Then, the average Silhouette, DB, and CH scores for varying numbers of clusters and components was calculated. Interestingly, the three metrics provided conflicting results. The Silhouette score was at its highest value with a high number of clusters and low number of components, as shown in Figure 7.1, where the score peaked at 2 components and 15 clusters. On the other hand, the CH score was at its highest value with a low number of both components and clusters, as shown in Figure 7.2, where the score peaked at 2 clusters and 2 components. Finally, as shown by Figure 7.3, the DB scores improve as the number of clusters increases, but no correlation is found between the score and the number of components. Since all three metrics point to

different levels of hyperparameters, it was impossible to determine the precise combination of clusters and components solely through visualization. To address this, all three metrics were incorporated through an aggregate score: Since the three metrics were on different scales, Z-score normalization was first applied to each set of scores (24). After this, the three tables were summed to give a score for each choice of hyperparameters, which are displayed in Table 1. The choice of parameters that performed the best according to this aggregate score was selected, which was 2 clusters and 8 components.

The results of these two methods were then compared to the industry standard ones. For each method, the mean of 200 Silhouette, CH, and DB scores for each cluster amount ranging from 2-16 was calculated and converted to an aggregate z-score to find the optimal number of clusters (Figures 2.1-2.4). Since the fine results of the experimental test could be attributed to randomness, the aggregate z-score of the algorithm with the heuristic parameters (2 components, 7 clusters) was recalculated.

Multiple conclusions can be drawn from the final evaluation table (Table 3). First, all algorithms performed significantly better than random assignment, indicating there is some relationship found in the data. Next, the eigengap heuristic was used to determine the number of eigenvectors and the elbow method to determine the number of clusters, Spectral Clustering had nearly identical heuristic (2 components and 7 clusters), and experimental (2 components and 8 clusters) parameters, which corresponded to very similar aggregate z-scores. In contrast, the algorithm whose number of clusters and components were both determined through the eigengap heuristic (2 components and 2 clusters) performed significantly worse. Third, k-means performed noticeably worse than the other four models, which all had similar aggregate z-scores. Finally, UMAP and Spectral Clustering had higher Silhouette scores, while k-means, TSNE, and PCA had higher CH scores.

In this section future work is discussed. As shown in Figures 8.1-8.6, each sample typically consists of a main section, consisting of the majority of the identified clusters, and then a set of outliers. Although this is dependent on the chosen 2D-linear embedding, it indicates that there may be multiple levels of structure present in the dataset. It would be interesting to apply hierarchical clustering methods to separate the outliers and compare each algorithm's performance on each component. Another important next step is to diversify and contextualize the chosen evaluation metrics. Currently, each cluster is being evaluated through quantitative metrics, where better scores are assumed to correlate to better results. However, the metrics give conflicting results, and when aggregated via Z-scores, the results showed very similar performance to one another, making it hard to come to a definitive conclusion on which models performed better. A more rigorous way to evaluate the model is to test its accuracy in clustering genes. The dataset provided contains a UMAP visualization for each gene's expression across all samples (Figure 9). With these UMAPs, low overall expression, such as KLHL17 and SAMD11, can be removed, and the remaining

can be grouped based on where they peak (genes with high expression in similar areas would be grouped together). This provides a definite ground truth to compare Spectral Clustering's ability against the industry standard models.

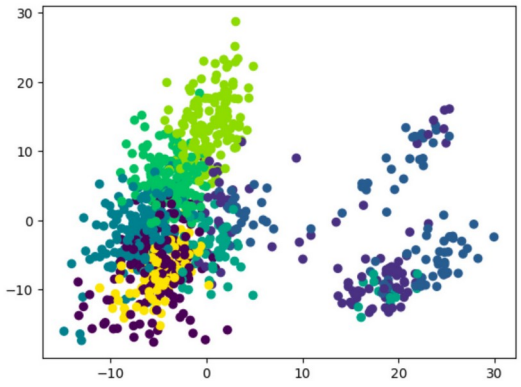

Figure 8.1 - Spectral Clustering with 2 components and 8 clusters

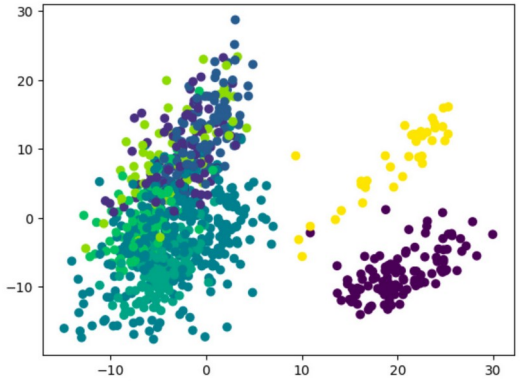

Figure 8.2 - PCA with 8 clusters

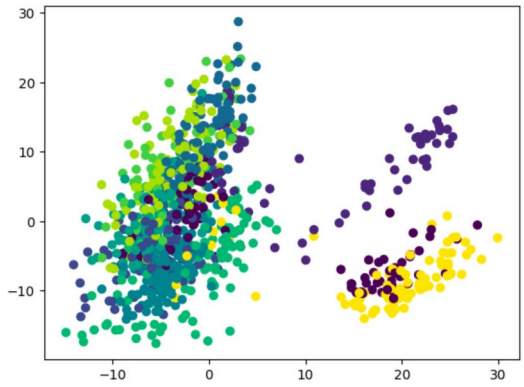

Figure 8.3 - UMAP with 10 clusters

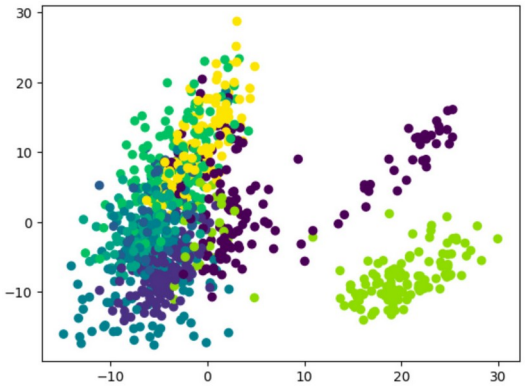

Figure 8.4 - TSNE with 8 clusters

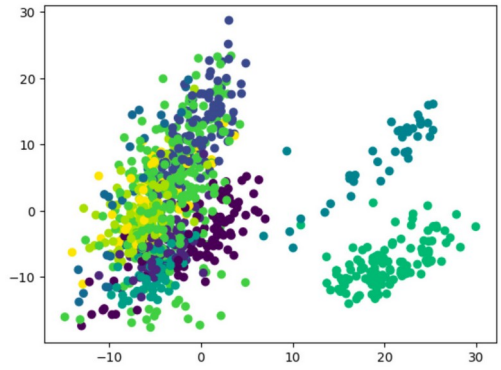

Figure 8.5 - KMeans with 10 clusters

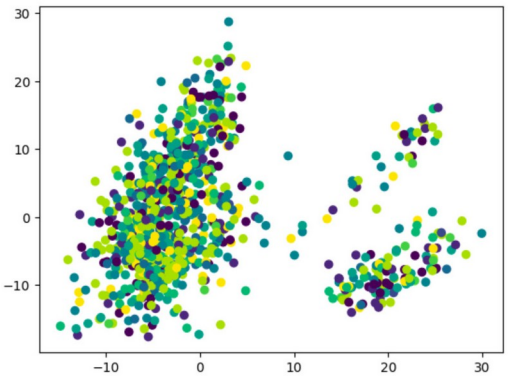

Figure 8.6 - Random Assignment with 8 clusters

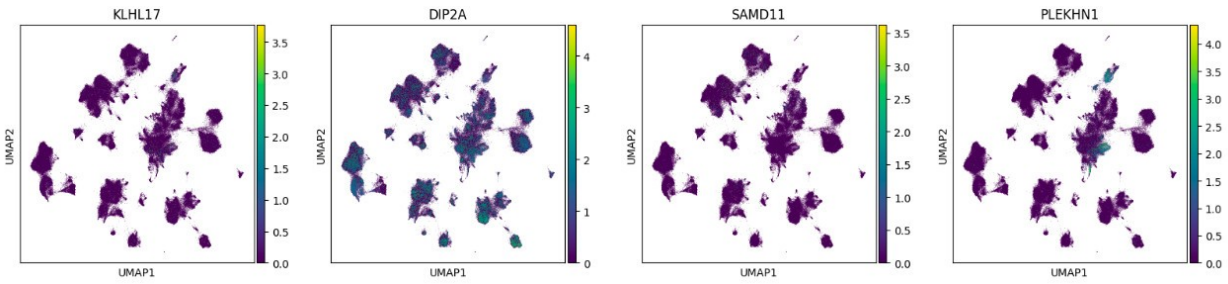

Figure 9 - UMAP visualizations for 4 randomly selected genes (a higher value indicates higher expression of that gene at that location)

IV Conclusion

This work investigated the efficacy of the Spectral Clustering method on the Genotype-Tissue Expression Project V.9 dataset. It was investigated how choices of the hyperparameters affected the quality of clustering, where the choices included the use of several popular heuristic methods and an alternative optimization based method using clustering metrics. When compared to a null hypothesis and other commonly used algorithms in the field, it was shown that Spectral Clustering performed significantly better than k-means clustering and random assignment, but more work is needed to be done to distinguish Spectral Clustering from PCA, UMAP, and TSNE. To the author's knowledge, this is the first systematic investigation of how these choices of hyperparameter affect clustering on this dataset.

V. References:

1. Martins Ribiero, Sara Carolina. A Comparative Evaluation of Dimensionality Reduction Methods on Large-Scale Gene Expression Datasets, 2020, https://repositorio-aberto.up.pt/bitstream/10216/131934/2/440960.pdf

2. Eraslan, G., Drokhlyansky, E., Anand, S., Fiskin, E., Subramanian, A., Slyper, M., et al. (2022). Single-nucleus cross-tissue molecular reference maps toward understanding disease gene function. *Science*, *376*(6594). https://doi.org/10.1126/science.abl4290

3. Doshi, Neerja. "Spectral clustering. The intuition and math behind how it… | by Neerja Doshi." *Towards Data Science*, https://towardsdatascience.com/spectral-clustering-82d3cff3d3b7

4. Ding, Shifei, Liwen Zhang, and Yu Zhang. “Research on spectral clustering algorithms and prospects.” 2010 2nd International Conference on Computer Engineering and Technology. Vol. 6. IEEE, 2010. https://ieeexplore.ieee.org/abstract/document/5486345

5. Guide, Step. “Spectral Clustering | What, why and how of Spectral Clustering!” *Analytics Vidhya*, 24 May 2021, https://www.analyticsvidhya.com/blog/2021/05/what-why-and-how-of-spectral-clustering/

6. Ellis, Christina. “When to use spectral clustering.” *Crunching the Data*, https://crunchingthedata.com/when-to-use-spectral-clustering/

7. Lee, George, Carlos Rodriguez, and Anant Madabhushi. "Investigating the efficacy of nonlinear dimensionality reduction schemes in classifying gene and protein expression studies." *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 5.3 (2008): 368-384, https://pubmed.ncbi.nlm.nih.gov/18670041/

8. Huang, Grace T et al. “Spectral clustering strategies for heterogeneous disease expression data.” Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing (2013): 212-23. https://pubmed.ncbi.nlm.nih.gov/23424126/

9. Romero, Miguel, et al. “Feature Extraction with Spectral Clustering for Gene Function Prediction Using Hierarchical Multi-Label Classification - Applied Network Science.” SpringerOpen, Springer International Publishing, 10 May 2022, https://doi.org/10.1007/s41109-022-00468-w

10. “Genotype-Tissue Expression Project (Gtex) portal.” Genome.Gov, https://www.gtexportal.org/home/datasets

11. Luo, Bin, Richard C. Wilson, and Edwin R. Hancock. "Spectral embedding of graphs." *Pattern recognition* 36.10 (2003): 2213-2230. https://ui.adsabs.harvard.edu/abs/2003PatRe..36.2213L/abstract

12. scikit-learn developers. “sklearn.manifold.kneighbors_graph — scikit-learn 1.3.0 documentation.” *Scikit-learn*, https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.kneighbors_graph.html

13. scikit-learn developers. “sklearn.cluster.SpectralClustering — scikit-learn 1.2.2 documentation.” *Scikit-learn*, https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html

14. Luxburg, Ulrike von. A Tutorial on Spectral Clustering - Arxiv.Org, https://doi.org/10.48550/arXiv.0711.0189

15. scikit-learn developers. "sklearn.manifold.spectral_embedding — scikit-learn 1.3.0 documentation." *Scikit-learn*, https://scikit-learn.org/stable/modules/generated/sklearn.manifold.spectral_embedding.html#sklearn-manifold-spectral-embedding

16. "Genotype-Tissue Expression Project (Gtex)." *Genome.Gov*, www.genome.gov/Funded-Programs-Projects/Genotype-Tissue-Expression-Project

17. "Usage Principles." Usage Principles - Scanpy 1.9.3 Documentation, https://scanpy.readthedocs.io/en/stable/usage-principles.html

18. Bingham, Ella, and Heikki Mannila. "Random projection in dimensionality reduction: applications to image and text data." *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01)*, 2001. https://dl.acm.org/doi/10.1145/502512.502546

19. scikit-learn developers. "6.6. Random Projection — scikit-learn 1.2.2 documentation." *Scikit-learn*, https://scikit-learn.org/stable/modules/random_projection.html

20. "Johnson–Lindenstrauss lemma." *Wikipedia*, https://en.wikipedia.org/wiki/Johnson%E2%80%93Lindenstrauss_lemma

21. Stewart, Gilbert W., and Ji-guang Sun. Matrix Perturbation Theory. Academic Press, 1990. https://catalog.princeton.edu/catalog/996702193506421

22. Humaira, H, and R Rasyidah. Determining the Appropriate Cluster Number Using Elbow Method for K-Means algorithm. EUDL, http://doi.org/10.4108/eai.24-1-2018.2292388

23. Wei, Haitian. "How to measure clustering performances when there are no ground truth?" *Medium*, 2020, https://medium.com/@haataa/how-to-measure-clustering-performances-when-there-are-no-ground-truth-db027e9a871c

24. "Standard score", *Wikipedia,* https://en.wikipedia.org/wiki/Standard_score