

TOPOLOGICAL REPRESENTATION AND DIMENSIONALITY REDUCTION OF SINGLE  
CELL RNA SEQUENCING AND VIRAL PHYLOGENETIC DATA

By

Yuta Hozumi

A DISSERTATION

Submitted to  
Michigan State University  
in partial fulfillment of the requirements  
for the degree of

Applied Mathematics—Doctor of Philosophy

2024

## ABSTRACT

With the development and improvement in technology, we are now able to observe and collect large data, especially in the field of biology. Sequencing technology has rapidly advanced, allowing for large-scale sequencing with increased precision. This has resulted in large datasets with high dimensionality. One trade-off with high dimensionality is the increase in sparsity. There are multiple causes of sparsity, including sequencing errors, nonuniform noise, low read depth, etc. It is paramount that we find an effective way to reduce the dimensionality of the data and select important features for downstream analysis.

DNA sequencing is a vital aspect of computational biology. Analyzing the sequences gives insight into relationships between species as well as within species, such as evolution, genetic drift, mutation, common ancestry, and more. We gathered over 3.6 million SARS-CoV-2 sequences from all over the world and performed multiple sequence alignments to extract mutations. Utilizing the mutation data, we performed UMAP dimensionality reduction followed by clustering to analyze mutational trends in the US and the world. However, such alignment methods do come with limitations, namely in computational cost and assumptions. To this end, we developed *k*-mer topology, an alignment-free DNA sequencing method that utilizes techniques from topological data analysis. This results in a uniform set of features for all sequences, regardless of their sequence length.

Another new technology in DNA sequencing technology is single-cell RNA-sequencing (scRNA-seq), where gene expression profiles for each cell can be extracted. With current technology, roughly 10,000 cells and over 20,000 genes can be sequenced, which has caused problems due to the high dimensionality of the data. Most dimensionality reduction methods employ frequency domain representations obtained from matrix diagonalization and may not be efficient for large datasets with relatively high intrinsic dimensions. To address this challenge, Correlated Clustering and Projection (CCP) offers a novel data domain strategy that does not need to solve any matrix. CCP partitions high-dimensional features into correlated clusters and then projects correlated features in each cluster into a one-dimensional representation based on sample correlations. Additionally,

we proposed topological nonnegative matrix factorization (TNMF), a matrix decomposition algorithm which incorporates multiscale geometric and topological information in the form of persistent Laplacian. Due to the nonnegativity constraint of the method, the reduced features are interpretable. Both CCP and TNMF are verified on scRNA-seq data to show their superior performance in both clustering, classification, and visualization.

Traditional visualization methods require a dimensionality reduction to 2 or 3 dimensions. However, such aggressive reduction can lead to misleading conclusions. We, therefore, introduce Residue-Similarity (RS) plot, which is a visualization tool for data with dimensions greater than 3. The RS plot is constructed from two measures, residue (R) and similarity (S) scores. The R-score measures the interclass difference, and the S-score measures the intra-class similarity score. The Residue Similarity Index (RSI) was introduced to evaluate the R-score and S-score and is verified to correlate with clustering and classification accuracies across a variety of benchmark datasets.

Copyright by  
YUTA HOZUMI  
2024



This dissertation is dedicated to my parents, Eiji and Momoko.  
Thank you for always believing in me.

## ACKNOWLEDGEMENTS

I would like to begin by thanking Prof. Guo-Wei Wei for being the most supportive and encouraging advisor throughout my Ph.D. journey. During my first year, when I was struggling through the exams, Prof. Wei was the first one to reach out to me to offer advice. After I passed the exams, right when I was getting accustomed to research, we were hit by the COVID-19 pandemic. During this time, Prof. Wei called me regularly, making sure I was doing well, and always made himself available whenever I needed help. Even though I was still inexperienced, he invited me to join the COVID research group, which kick-started my research career. I was able to observe the motivation and brilliance of researchers at the forefront of research. Furthermore, Prof. Wei taught me the importance of the ‘big picture,’ which many students fail to see, and it has opened my eyes to mathematical biology. I genuinely appreciate him for providing me with numerous opportunities. Without Prof. Wei, I do not think my current goal of pursuing research would exist.

I would also like to thank my dissertation committee, Prof. Mark Iwen, Prof. Ekaterina Rapinchuk (Merkurjev), and Prof. Longxiu Huang for their encouragement, feedback, and comments. Every time I see them, they always offered encouragement and advices. Additionally, I would like to thank Prof. Changchuan Yin for introducing me to genomic analysis during the COVID projects. He introduced me to the field of bioinformatics, which became the basis of my dissertation. I would also like to thank the senior members of our group, Dr. Rui Wang, Prof. Jiahui Chen, and Prof. Duc Nguyen, for mentoring me throughout my Ph.D. I am especially thankful to Dr. Rui Wang for her guidance throughout my second and third years at MSU. I learned all kinds of skills, including coding, writing papers, topological data analysis, and more.

I am also thankful to my cohorts. Without their support, especially during my first year and COVID lockdown, I may not have been able to finish my Ph.D. I am especially thankful for Cullen Haselby, Luis Suarez, Quinn Minich and Chris Potvin for their encouragement. We are all in different fields of mathematics, but I am thankful for all the advises they have given me.

Lastly, I give my biggest thanks to my father Eiji Hozumi and mother Momoko Hozumi. They have been my biggest supporter from day 1 at MSU. I do sincerely thank them for their endless and

unrequited support and love. They were the first ones that I would reach out if anything happened, good or bad, and they would always listen till the very end.

Lastly, I would like to acknowledge that all of the work included in this thesis was supported in part the following grants: NIH grants R01GM126189, R01AI164266, R35GM148196; NSF grants DMS-1721024, DMS-1761320, IIS1900473, DMS-2052983, IIS-1900473; NASA grant 80NSSC21M0023; Michigan State University Research Foundation; Bristol-Myers Squibb 65109; Pfizer. Additionally, I would like to thank MSU's higher performance cluster computer, The IBM TJ Watson Research Center, The COVID-19 High Performance Computing Consortium, and NVIDIA for computational assistance.

## TABLE OF CONTENTS

CHAPTER 1	INTRODUCTION . . . . .	1
1.1	Dimensionality Reduction . . . . .	1
1.2	Topological Data Analysis . . . . .	5
1.3	Phylogenetic Analysis . . . . .	7
1.4	Single Cell RNA Sequencing (scRNA-seq) . . . . .	8
1.5	Outline . . . . .	11
	BIBLIOGRAPHY . . . . .	13
CHAPTER 2	BACKGROUND . . . . .	24
2.1	Overview of Dimensionality Reduction . . . . .	24
2.2	Clustering Algorithm . . . . .	28
2.3	Evaluation metrics for clustering and classification . . . . .	29
2.4	Topological Data Analysis . . . . .	32
2.5	Phylogenetic Analysis . . . . .	43
	BIBLIOGRAPHY . . . . .	47
CHAPTER 3	PHYLOGENETIC ANALYSIS . . . . .	50
3.1	UMAP-assisted $k$ -means clustering of large-scale SARS-CoV-2 mutation datasets . . . . .	50
3.2	$K$ -mer Topology for Whole Genome Analysis . . . . .	80
	BIBLIOGRAPHY . . . . .	100
APPENDIX 3A	ADDITIONAL MATERIALS FOR UMAP-ASSISTED $K$ -MEANS CLUSTERING OF LARGE-SCALE SARS-COV-2 MUTATION DATASETS . . . . .	105
APPENDIX 3B	ADDITIONAL MATERIALS FOR $K$ -MERS TOPOLOGY FOR ALIGNMENT-FREE SEQUENCE ANALYSIS . . . . .	113
CHAPTER 4	RESIDUE-SIMILARITY SCORES AND INDEXES . . . . .	120
4.1	Methods . . . . .	120
4.2	Results . . . . .	124
4.3	Discussion . . . . .	130
	BIBLIOGRAPHY . . . . .	133
CHAPTER 5	CORRELATED CLUSTERING AND PROJECTION . . . . .	135
5.1	Introduction . . . . .	135
5.2	Methods and Algorithms . . . . .	137
5.3	Results . . . . .	142
5.4	Discussion . . . . .	160
5.5	Concluding remarks . . . . .	167
	BIBLIOGRAPHY . . . . .	169
CHAPTER 6	TOPOLOGICAL NONNEGATIVE MATRIX FACTORIZATION . . . . .	172
6.1	Introduction . . . . .	172
6.2	Prior Work . . . . .	173
6.3	Topological NMF . . . . .	176

BIBLIOGRAPHY . . . . .	181
CHAPTER 7 APPLICATION IN SINGLE CELL RNA SEQUENCING . . . . .	182
7.1 Preprocessing of Single Cell RNA Sequencing data using Correlated Clustering and Projection . . . . .	182
7.2 Analyzing scRNA-seq data by CCP-assisted UMAP and t-SNE . . . . .	196
7.3 Topological Non-Negative Matrix Factorization for Single Cell RNA Sequencing Data . . . . .	215
BIBLIOGRAPHY . . . . .	229
APPENDIX 7A ADDITIONAL RESULTS FOR PREPROCESSING SINGLE CELL RNA SEQUENCING DATA USING CORRELATED CLUSTERING AND PROJECTION . . . . .	232
APPENDIX 7B ADDITIONAL MATERIALS FOR ANALYZING SCRNASQ DATA BY CCP-ASSISTED UMAP AND T-SNE . . . . .	241
APPENDIX 7C ALGEBRAIC CONNECTIVITY OF FRI IN CCP . . . . .	251
CHAPTER 8 DISSERTATION CONTRIBUTIONS AND FUTURE DIRECTIONS . .	253

# CHAPTER 1

## INTRODUCTION

### 1.1 Dimensionality Reduction

Technological advances have fueled exponential growth in high-dimensional data. In biological science, high-dimensional data are ubiquitous in genomics, epigenomics, transcriptomics, proteomics, metabolomics, and phenomics. In image science, an image of moderate size, i.e.,  $1024 \times 1024$ , gives rise to a 1,048,576-dimensional vector. The rapid increase in the size and complexity of scientific data has made the problem of the ‘curse of dimensionality’ more challenging than ever before in data sciences [1]. In machine learning, this problem is associated with the phenomenon that the average predictive power of a well-trained model first increases as the feature size increases but starts to deteriorate beyond a certain dimensionality [2, 3]. Moreover, data with an enormous volume of the feature space will become sparse, posing challenges for statistical analysis in determining statistical significance and principal variables. Furthermore, it is challenging to visualize data in high dimensions unless one can reduce the dimension to two or three. Therefore, it is desirable to reduce the dimensionality of high-dimensional data for the sake of prediction, analysis, and visualization. These challenges have been driving the development of many dimensionality reduction methods that can capture the intrinsic correlations in the original data in a low-dimensional representation [4].

Dimensionality reduction can be achieved through various deep neural networks (DNN), such as graph neural networks, autoencoders, transformers, etc. However, most DNN methods may not work well with excessively high-dimensional data. Commonly used dimensionality reduction algorithms fall into two categories: linear and nonlinear with respect to a certain distance metric. Principle component analysis (PCA) [5] is a basic linear DR algorithm that focuses on finding the principal components by creating new uncorrelated variables that successively maximize variances [6]. Specifically, the first principal component is a vector that maximizes the variance of the projected data, while the  $i$ th principal component is a vector that is orthogonal to the first  $(i - 1)$  principal components, leading to the maximization of the variance of the projected data. Linear

discriminant analysis (LDA) is another linear dimensionality reduction method proposed by Sir Ronald Fisher in 1936 [7]. As a generalization of Fisher’s linear discriminant, LDA aims to find a linear combination of features that maximizes the separability of classes and minimizes the inter-class variance for the multi-class classification problem [8]. Nonnegative matrix factorization (NMF) is another linear dimensionality reduction method that is often employed for data with nonnegative entries, such as count matrix in single cell RNA-sequencing (scRNA-seq) data. NMF decomposes a nonnegative matrix into 2 nonnegative factor matrices, and the basis is a nonnegative combination of the original features, leading to a more interpretable result [9]. Another category of dimensionality reduction methods contains many nonlinear algorithms, which can be classified into two groups: those that favor the preservation of the global pairwise distance and those that seek to retain local distance instead of global distance. Algorithms such as kernel principal component analysis (kernel PCA), Sammon mapping, and spectral embedding fall within the former category, while Isomap, LargeVis, Laplacian eigenmaps, locally linear embedding (LLE), diffusion maps [10, 11], t-SNE, and UMAP fall into the latter category. Kernel PCA [12] is an extension of PCA. Standard PCA typically has poor performance if the data has complicated algebraic structures that cannot be well-represented in a linear space. Therefore, kernel PCA is designed by applying kernel functions in a reproducing kernel Hilbert space in 1998. In 1969, John W. Sammon first proposed the Sammon mapping [13], which aims to conserve the structure of inter-point distances by minimizing Sammon’s error and attempts to ensure the mapping does not affect the underlying topology [14]. Spectral embedding computes the full Laplacian graph and uses graph eigenvectors, which allows for the preservation of the original global graph structure in the lower-dimensional space. Although kernel PCA, Sammon mapping, and spectral embedding preserve the pairwise distance structure among all the data, they fail to capture the local relationship between data points. Therefore, nonlinear algorithms are essential to incorporate the local structure in low-dimensional space and better describe the local information of the original data.

A quantitative survey of dimensionality reduction techniques is provided in Ref. [15]. Several widely used nonlinear DR algorithms are briefly discussed in the following. Isomap [16] is

a nonlinear method aiming to preserve the geodesic distance between samples while reducing dimensionality. It's an extension of multidimensional scaling (MDS) [17], replacing the Euclidean distance in MDS with geodesic distance (estimated by Dijkstra's distance in graph theory). Isomap is a local method, estimating the intrinsic geometry of a data manifold by roughly estimating each sample's neighbors, ensuring its efficiency [18]. Laplacian Eigenmap (LE), introduced in 2003 [19], is another unsupervised nonlinear algorithm preserving the local properties of a weighted graph Laplacian. LE constructs a neighborhood graph where each data point is linked to its nearest neighbors. Then, edge weights are estimated using the Gaussian kernel function. After solving the eigenvectors of the generalized weighted neighborhood graph matrix, eigenvectors associated with zero eigenvalues are discarded, and subsequent eigenvectors (smallest) are used for embedding in a  $k$ -dimensional space. Moreover, t-Distributed Stochastic Neighbor Embedding (t-SNE) [20, 21] is a nonlinear, manifold-based method well-suited for reducing high-dimensional data into two- or three-dimensional space for visualization. T-SNE represents similarities for every pair of data by constructing a conditional probability distribution over pairs of data. It then applies the 'student t-distribution' to obtain the probability distribution in the embedded space. By minimizing the Kullback-Leibler (KL) divergence between these two probabilities in the original and embedded space, t-SNE preserves the significant structure of the data for analysis and visualization [22]. Furthermore, a state-of-art nonlinear dimensionality reduction algorithm is uniform manifold approximation and projection (UMAP) [23], a graph-based algorithm that builds on the Laplacian eigenmaps and performs great visualization and feature extraction. Three assumptions make UMAP stand out among the other dimensionality reduction algorithms: (1) Data is uniformly distributed on a Riemann manifold, (2) Riemannian metric is locally constant, and (3) The manifold is locally connected. UMAP creates  $k$ -dimensional weighted graph representations based on the  $k$ -nearest neighborhood searching and intends to minimize the edge-wise cross-entropy between the embedded low-dimensional weighted graph representation in terms of a fuzzy set cross-entropy loss function via the stochastic gradient descent. Specifically, UMAP constructs a weighted directed adjacency matrix  $A$ , where  $A(i, j)$  represents the connection between the  $i$ th node and the



$j$ th node when the  $j$ th node is one of the  $k$  nearest neighbors. Next, a normalized sparse Laplacian matrix can be derived from  $A$  with the implementation of the cross-entropy loss involved, and the  $k$ -dimensional eigenvectors of this normalized Laplacian will be used to represent each of the original data points in a low-dimension space.

All of the dimensionality reduction algorithms mentioned above have broad applications in science and technology. However, they often rely on frequency domain representations obtained from matrix diagonalization. Typically, the computational complexity of eigenvalue decomposition for a full matrix is  $O(M^3)$ , where  $M$  is the number of samples forming an  $M \times M$  matrix. While fast solvers exist, they may sacrifice accuracy, especially for datasets with relatively high intrinsic dimensions [15]. Moreover, for datasets with a large number of features  $I$ , where  $M \ll I$ , the reliance on matrix diagonalization limits the performance of these algorithms. Additionally, many methods depend on computing distances between data entries (samples), which can be problematic in high dimensions. Particularly, methods that utilize nearest neighbors, such as UMAP and t-SNE, may suffer from instability in datasets with moderately high intrinsic dimensions, as outlined in the ‘curse of dimensionality’ [1].

A somewhat related but distinct problem is tensor-based dimensionality reduction [24, 25], which addresses data with internal structures of geometric, topological, algebraic, and/or physical origins. Methods dealing with tensorial structures, such as Tucker decomposition [26, 27], are often employed in addition to the aforementioned dimensionality reduction approaches. These methods find applications in various domains such as videos, X-ray Computed Tomography (X-ray CT), and Magnetic Resonance Imaging (MRI) data.

Other related issues pertain to feature evaluation, ranking, clustering, extraction, and selection, particularly for unlabeled data. Feature evaluation and ranking can be accomplished through filtering or embedding techniques, while feature clustering and selection can be carried out using methods such as  $k$ -means,  $k$ -means++,  $k$ -medoids, among others. These methods often serve as preprocessing steps for dimensionality reduction. For labeled data, various supervised learning methods can be employed for feature selection or extraction.

In this dissertation, we introduce two novel dimensionality reduction methods: Correlated Clustering and Projection (CCP) and Topological Nonnegative Matrix Factorization (TNMF). CCP is a data domain dimensionality reduction method consisting of two steps. First, features are clustered based on their similarities. Then, the clustered features are nonlinearly projected into a single descriptor. TNMF, on the other hand, is a persistent Laplacian regularized NMF method. Unlike standard manifold regularization, TNMF can capture the geometrical and topological shape of the data at multiscale. We evaluate CCP and TNMF on standard benchmark datasets and apply these methods to single-cell RNA sequencing (scRNA-seq) data. Additionally, we propose a novel visualization method called the Residue-Similarity (RS) plot. The RS plot comprises two components. Given a set of labels for each sample, the Residue (R) score measures the intraclass similarity score, while the Similarity (S) score measures the interclass similarity score. We find that the RS plot can effectively visualize data with dimensions greater than 3, and the RS score correlates with the accuracy of classification tasks.

## 1.2 Topological Data Analysis

In recent years, there has been an explosion of data across multiple disciplines, with biology experiencing a significant influx of new data from technological advances such as genomics, single-cell RNA sequencing (scRNA-seq) [28], spatial transcriptomics (ST) [29], epigenomics [30], and DNA sequencing [31]. The data in many fields is often high-dimensional, noisy, and lacks uniformity in size or features. However, traditional data analysis tools, including dimensionality reduction techniques, often struggle to capture the intricate structures and relationships within such complex datasets. Moreover, traditional approaches are ill-equipped to handle nonuniform data distributions. To address these limitations, topological data analysis (TDA) has gained popularity in recent years due to its ability to extract meaningful insights from complex data using properties from algebraic topology [32, 33, 34, 35].

At the heart of TDA lies persistent homology, a branch of algebraic topology that serves as a powerful tool for characterizing the intrinsic structure in datasets, regardless of the inherent noise present in the data [36, 37, 38]. Persistent homology differs from traditional dimensionality reduc-

tion techniques because they rely on the topology of data rather than its geometric properties and traditional metrics. In essence, persistent homology represents the dataset as a topological space and using tools from algebraic topology to extract meaningful features, namely the topological invariants. Additionally, by introducing filtration, we can keep track of the changes in the topological invariants to understand the geometric and topological shape of the data, which is termed persistence. Through the lens of persistent homology, researchers can uncover hidden patterns, identify relevant structures, and gain a deeper understanding of the datasets' intrinsic characteristics.

Spectral graph theory has also been introduced to TDA in the recent years, with the introduction of topological Laplacian. Eckmann et al. [39] introduced simplicial complexes to the graph Laplacian defined on point cloud data, leading to the combinatorial Laplacian. This can be viewed as a discrete counterpart of the de Rham-Hodge Laplacian on manifolds. Both the Hodge Laplacian and the combinatorial Laplacian are topological Laplacians that give rise to topological invariants in their kernel space, specifically the harmonic spectra. However, the nonharmonic spectra contain algebraic connectivity that cannot be revealed by the topological invariants from the standard persistent homology[40].

A significant development in topological Laplacians occurred in 2019 with the introduction of persistent topological Laplacians. Specifically, evolutionary de Rham theory was introduced to obtain persistent Hodge Laplacians on manifolds [41]. Meanwhile, persistent combinatorial Laplacian [42], also known as the persistent spectral graph or persistent Laplacian (PL), was introduced for point cloud data. These methods have spurred numerous theoretical developments [43, 44, 45, 46, 47] and package [48], as well as remarkable applications in various fields, including protein engineering [49], forecasting emerging SARS-CoV-2 variants BA.4/BA.5 [50], and predicting protein-ligand binding affinity [51]. Recently, PL has been shown to improve PCA performance [52].

In this dissertation, we utilize TDA as a visualization tool for high dimensional data. Additionally, we utilize persistent homology to develop a novel algorithm called k-mers topology, which is utilized to analyze viral nucleotide sequence. Lastly, we develop topological nonnegative matrix

factorization (TNMF) utilizing PL to analyze single cell RNA sequencing data.

### 1.3 Phylogenetic Analysis

Phylogenetic analysis of genetic sequences is crucial for elucidating the evolutionary relationships both between and within species [53]. This analysis entails the comparison of two or more DNA or RNA sequences to discern similarities, differences, and patterns within the genetic code. By identifying similarities and differences among sequences, researchers can gain insights into evolutionary relationships, mutations, genetic drifts, genome assembly, gene annotation, and more. Having a robust and scalable method that enables comparisons within species and across species is essential for laying the groundwork of genomic analysis.

Traditional phylogenetic analysis utilizes sequence alignment, where pairs of sequences are aligned to obtain similarity scores. These methods include sequence similarity scores, like BLAST [54] and FASTA [55], sequence profile search, like PSI-BLAST [54] and HMMER [56], whole-genome comparison, like Mauve [57] and TBA [58], and multiple sequence alignment, like ClustalW [59], MAFFT [60], and Muscle [61]. In recent years, the sequence of SARS-CoV-2 has been heavily studied using alignment-based methods to analyze mutational trends and predict future mutations [62, 63, 50, 64, 65]. An advantage of sequence alignment methods is their ability to extract mutation sites, which can then be utilized for downstream analysis. These alignment-based methods assume that the segments of the sequences can be categorized into conserved and non-conserved segments. The effectiveness of the alignment can then be measured based on the amount of conserved region penalized by the amount of non-conserved segments, or gaps. However, such methods can fail if these conserved segments are not properly arranged in the sequence or if there is not a sufficient amount of conserved segments, which often occurs in real-world data [66, 67, 68]. Additionally, alignment-based methods are time-consuming and require a large amount of memory, which makes large-scale comparison difficult. Other cases where alignment-based methods can fail can be found in [69]. To combat these issues, alignment-free methods have been developed, which do not assume anything about the sequences.

One of the most common alignment-free methods is the  $k$ -mers-based approach. In the original

$k$ -mers methods developed by Blaisdell in 1986 [70], the frequency of words or motifs is counted, and the sequences are represented as a count-based vector. Then, a metric can be defined to compare sequences for phylogenetic analysis. Numerous extensions have been proposed to improve the  $k$ -mers method, such as those by Wu et al. [71, 72, 73], Korf et al. [74], and Jun et al. [75]. Because the counting is linear with respect to the sequence length, it is extremely computationally efficient, allowing for whole-genome analysis. However, these methods do not capture the relationship between the  $k$ -mers or the positional information about the sequence.

Several alternative alignment-free methods have been proposed as well. The Natural Vector Method (NVM) computes the moments of the  $k$ -mers position [76, 77], incorporating the relationship between the  $k$ -mers within the sequence, thus overcoming limitations in the  $k$ -mers based method. In information theory-based methods, the distance between sequences is obtained by computing the amount of information shared between them [78, 79, 80, 81, 82]. The Chaos Game Representation (CGR), originally proposed by Jeffory in 1990 [83], represents the sequence as an iterative function, allowing the DNA sequence to be visualized as an image [84]. The CGR representation was further extended in [85, 86, 87, 88] for sequence analysis. Other methods, such as the discrete Fourier power spectrum method [89, 90], fuzzy integral similarity method [91], and more [92], have been developed for phylogenetic analysis.

In this dissertation, we employ alignment-based methods to study the mutational patterns in SARS-CoV-2. After aligning the sequences, we extract mutations and utilize UMAP to reduce the dimensionality for clustering. Additionally, we introduce a novel alignment-free method called  $k$ -mer topology, which utilizes tools from persistent homology to extract patterns within the sequence. We benchmark our method on virus classification tasks and apply it to standard phylogenetic analysis.

## **1.4 Single Cell RNA Sequencing (scRNA-seq)**

Single-cell RNA sequencing (scRNA-seq) reveals heterogeneity within cell types, leading to an understanding of cell-cell communication, cell differentiation, and differential gene expression. With current technology and protocols, more than 20,000 genes can be identified. Additionally,

numerous data analysis pipelines have been developed to assist in analyzing such complex data [93, 94, 95, 96, 97, 98]. Despite improvements in technology that enable more accurate gene readings, analyzing these readings remains challenging. Causes of this challenge include dropout event-induced zero expression counts, low sequencing depth resulting in fewer readings, general noise, and the high dimensionality of the original data [28]. As a result, dimensionality reduction and feature selection become important for downstream analysis.

Numerous dimensionality reduction and feature selection methods have been proposed for scRNA-seq data. One such method is scRNA by non-negative and low-rank representation (SinLRR), which assumes that scRNA-seq has an inherently low rank, and attempts to find the smallest rank matrix that captures the original data [99]. Additionally, various non-negative matrix factorization (NMF) methods with different constraints have been developed. In these methods, the low-dimensional representation of scRNA-seq data is achieved through a linear combination of the original genes, which are called meta-genes [100, 101, 102, 103, 104, 105]. Single-cell interpretation via multikernel learning (SIMLR) utilizes multiple kernels to learn a cell-cell similarity metric that generalizes to different biological experiments and experimental procedures [106]. In addition, more traditional approaches, such as principal component analysis (PCA) [107] and its derivatives [108, 109], and visualization techniques, such as uniform manifold approximation and projection (UMAP) [23] and t-distributed stochastic neighbor embedding (t-SNE) [110], have been heavily utilized for scRNA-seq data. Furthermore, deep learning has also been used for dimensionality reduction [111, 112, 113, 114, 115, 116].

Deep learning and ensemble methods are another class of approaches that have become popular for single cell RNA-seq analysis. Single-cell variational inference (scVI) utilizes deep neural networks to obtain information from similar cells and genes to approximate the distribution of underlying gene expression values [111]. Single-cell cluster using marker genes (SCMcluster) utilizes known marker genes to guide feature selection and perform ensemble clustering [117]. AutoCell [118] utilizes variational autoencoding network that combines the Gaussian mixture model and graph embedding to model the high dimensional scRNA-seq data. Diffusion models

[119, 120, 121, 122], generative adversarial network (GAN) [121], language models [123, 124, 125], transformers [126, 127, 128], ensemble methods [106, 129, 130] and more [131, 132] have also been used for scRNA-seq analysis. Though these methods have great performance, they rely on careful curation of data and often require large amount of data for pretraining.

Although numerous techniques have been developed, PCA is the most commonly used method for downstream analysis of scRNA-seq data [133]. PCA is a linear dimensionality reduction method, where its goal is to compute the principal components as new features that maximize the variance. The first principal component is a feature that maximizes the variance of the projected data, and each  $i$ th principal component is orthogonal to the  $i - 1$  principal component that maximizes the variance of the projected data [6]. Single-cell consensus clustering (SC3) [129] utilizes PCA and the eigenvectors of the graph Laplacian induced by Euclidean, Pearson, and Spearman distances, and performs a consensus on k-means results obtained from different dimensions using the CSPA algorithm to obtain the final cell clustering result. CellChat [134] utilizes the low-dimensional representation of scRNA-seq alongside known interactions between ligands, receptors, and cofactors to predict cell-cell communication, and the user can perform dimensionality reduction prior to utilizing CellChat. DEEPsc [135] is a deep learning method that predicts the probability of a cell belonging to a reference atlas by projecting scRNA-seq to the PCA space of the reference atlas, which can then be used to predict cell types. A popular package Seurat [136] utilizes supervised PCA (sPCA) which finds the projection that captures the weighted nearest neighbor graph of the reference dataset for its downstream analysis. In addition to cell clustering, semi-supervised and supervised learning methods have been used to classify cell types according to their reference cells by projecting unknown cells to the PCA space of the reference cells [137, 138].

Utilizing PCA has many advantages, such as computational efficiency and ease in projecting new data into principal components. However, PCA lacks concrete interpretability and loses the non-negativity of read-count data. In contrast, the components of NMF are all positive and can be considered as metagenes, where metagenes are linear combinations of the original genes. Non-linear dimensionality reduction methods, such as UMAP, t-SNE, and Isomap, have great

performance for low dimensionality that can capture the local structure of the data, but they also lack interpretability due to matrix diagonalization. Moreover, both PCA and traditional nonlinear reduction methods are unstable when the data is reduced to higher dimensions, which is unfavorable for machine learning and deep learning tasks that typically require a large number of features.

In this dissertation, we apply Correlated Clustering and Projection (CCP) and Topological Nonnegative Matrix Factorization (TNMF) to scRNA-seq data. We found that CCP outperforms PCA in both classification and clustering tasks when the number of components is higher than 50. Additionally, CCP improves both UMAP and t-SNE visualization of scRNA-seq data. Lastly, TNMF outperforms other NMF methods in clustering tasks of scRNA-seq data.

## 1.5 Outline

The outline of the dissertation is as followed. In Chapter 2, we introduce background information that will be utilized throughout this dissertation, such as evaluation metrics, clustering algorithm, and dimensionality reduction algorithms. In Chapter 3, we introduce a large scale clustering algorithm, UMAP-assisted  $k$ -Means, and apply the technique to SARS-CoV-2 single nucleotide polymorphism (SNP) data. Furthermore, we introduce a novel alignment-free DNA sequence analysis method called  $k$ -mers topology. In Chapter 4, we introduce Residue-Similarity scores and indexes, which present a novel approach to evaluating classification and clustering problems. In Chapter 4, we introduce Correlated clustering and Projection (CCP), a nonlinear data-domain dimensionality reduction, and we compare its performance to standard benchmark data. In Chapter 5, we introduce topological nonnegative matrix factorization, a persistent Laplacian (PL) regularized nonnegative matrix factorization (NMF). Compared to traditional graph Laplacian, the PL utilizes filtration to capture multiscale interaction, which a traditional Laplacian cannot. Additionally, PL captures standard topological information and homotopic shape evolution, which gives a comprehensive view of the overall shape of the data. In Chapter 6, CCP and TNMF are applied to single-cell RNA sequencing (scRNA-seq) data. CCP improves other dimensionality reduction methods in not only clustering and classification tasks, but also improves visualization. Additionally, RS analysis is performed on scRNA-seq data, which has been shown to be both ef-



fective for visualizing and evaluating performance. TNMF was shown to improve in adjusted rand index (ARI), normalized mutual information (NMI), purity, and ACC against other NMF methods for clustering.

## BIBLIOGRAPHY

- [1] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.
- [2] Gerard V Trunk. A problem of dimensionality: A simple example. *IEEE Transactions on pattern analysis and machine intelligence*, (3):306–307, 1979.
- [3] B Chandrasekaran and Anil K Jain. Quantization complexity and independent measurements. *IEEE Transactions on Computers*, 100(1):102–106, 1974.
- [4] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.
- [5] George H Dunteman. *Principal components analysis*. Number 69. Sage, 1989.
- [6] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.
- [7] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.
- [8] Petros Xanthopoulos, Panos M Pardalos, and Theodore B Trafalis. Linear discriminant analysis. In *Robust data mining*, pages 27–33. Springer, 2013.
- [9] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. *IEEE Transactions on knowledge and data engineering*, 25(6):1336–1353, 2012.
- [10] Ronald R Coifman, Stephane Lafon, Ann B Lee, Mauro Maggioni, Boaz Nadler, Frederick Warner, and Steven W Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences*, 102(21):7426–7431, 2005.
- [11] Ketson R Dos Santos, Dimitrios G Giovanis, and Michael D Shields. Grassmannian diffusion maps–based dimension reduction and classification for high-dimensional data. *SIAM Journal on Scientific Computing*, 44(2):B250–B274, 2022.
- [12] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.
- [13] John W Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 100(5):401–409, 1969.
- [14] Paul Henderson. Sammon mapping. *Pattern Recognit. Lett*, 18(11-13):1307–1316, 1997.
- [15] Mateus Espadoto, Rafael M Martins, Andreas Kerren, Nina ST Hirata, and Alexandru C

- Telea. Toward a quantitative survey of dimension reduction techniques. *IEEE transactions on visualization and computer graphics*, 27(3):2153–2173, 2019.
- [16] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.
  - [17] Al Mead. Review of the development of multidimensional scaling methods. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 41(1):27–39, 1992.
  - [18] Farzana Anowar, Samira Sadaoui, and Bassant Selim. Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne). *Computer Science Review*, 40:100378, 2021.
  - [19] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.
  - [20] Geoffrey Hinton and Sam T Roweis. Stochastic neighbor embedding. In *NIPS*, volume 15, pages 833–840. Citeseer, 2002.
  - [21] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
  - [22] Bo Li, Yan-Rui Li, and Xiao-Long Zhang. A survey on laplacian eigenmaps based manifold learning methods. *Neurocomputing*, 335:336–351, 2019.
  - [23] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
  - [24] Jiun-Hung Chen and Linda G Shapiro. Pca vs. tensor-based dimension reduction methods: An empirical comparison on active shape models of organs. In *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 5838–5841. IEEE, 2009.
  - [25] Hongcheng Wang and Narendra Ahuja. A tensor approximation approach to dimensionality reduction. *International Journal of Computer Vision*, 76(3):217–229, 2008.
  - [26] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.
  - [27] Xutao Li, Michael K Ng, Gao Cong, Yunming Ye, and Qingyao Wu. Mr-ntd: Manifold regularization nonnegative tucker decomposition for tensor data dimension reduction and representation. *IEEE transactions on neural networks and learning systems*, 28(8):1787–1800, 2016.
  - [28] David Lähnemann, Johannes Köster, Ewa Szczurek, Davis J McCarthy, Stephanie C Hicks,

- Mark D Robinson, Catalina A Vallejos, Kieran R Campbell, Niko Beerenwinkel, Ahmed Mahfouz, et al. Eleven grand challenges in single-cell data science. *Genome biology*, 21(1):1–35, 2020.
- [29] Cameron G Williams, Hyun Jae Lee, Takahiro Asatsuma, Roser Vento-Tormo, and Ashraf Haque. An introduction to spatial transcriptomics for biomedical research. *Genome Medicine*, 14(1):68, 2022.
  - [30] Christoph Bock and Thomas Lengauer. Computational epigenetics. *Bioinformatics*, 24(1):1–10, 2008.
  - [31] Susana Vinga and Jonas Almeida. Alignment-free sequence comparison—a review. *Bioinformatics*, 19(4):513–523, 2003.
  - [32] David Cohen-Steiner, Herbert Edelsbrunner, John Harer, and Yuriy Mileyko. Lipschitz functions have  $l_p$ -stable persistence. *Foundations of computational mathematics*, 10(2):127–139, 2010.
  - [33] Paul Bendich, Herbert Edelsbrunner, and Michael Kerber. Computing robustness and persistence for images. *IEEE transactions on visualization and computer graphics*, 16(6):1251–1260, 2010.
  - [34] Robert Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.
  - [35] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.
  - [36] Afra Zomorodian and Gunnar Carlsson. Localized homology. *Computational Geometry*, 41(3):126–148, 2008.
  - [37] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 347–356, 2004.
  - [38] Edelsbrunner, Letscher, and Zomorodian. Topological persistence and simplification. *Discrete & computational geometry*, 28:511–533, 2002.
  - [39] Beno Eckmann. Harmonische funktionen und randwertaufgaben in einem komplex. *Commentarii Mathematici Helvetici*, 17(1):240–255, 1944.
  - [40] Danijela Horak and Jürgen Jost. Spectra of combinatorial laplace operators on simplicial complexes. *Advances in Mathematics*, 244:303–336, 2013.
  - [41] Jiahui Chen, Rundong Zhao, Yiyang Tong, and Guo-Wei Wei. Evolutionary de rham-hodge method. *Discrete and continuous dynamical systems. Series B*, 26(7):3785, 2021.

- [42] Rui Wang, Duc Duy Nguyen, and Guo-Wei Wei. Persistent spectral graph. *International journal for numerical methods in biomedical engineering*, 36(9):e3376, 2020.
- [43] Facundo Mémoli, Zhengchao Wan, and Yusu Wang. Persistent laplacians: Properties, algorithms and implications. *SIAM Journal on Mathematics of Data Science*, 4(2):858–884, 2022.
- [44] Jian Liu, Jingyan Li, and Jie Wu. The algebraic stability for persistent laplacians. *arXiv preprint arXiv:2302.03902*, 2023.
- [45] Xiaoqi Wei and Guo-Wei Wei. Persistent sheaf laplacians. *arXiv preprint arXiv:2112.10906*, 2021.
- [46] Rui Wang and Guo-Wei Wei. Persistent path laplacian. *Foundations of Data Science*, 5:26–55, 2023.
- [47] Dong Chen, Jian Liu, Jie Wu, and Guo-Wei Wei. Persistent hyperdigraph homology and persistent hyperdigraph laplacians. *Foundations of Data Science*, doi: 10.3934/fods.2023010, 2023.
- [48] Rui Wang, Rundong Zhao, Emily Ribando-Gros, Jiahui Chen, Yiyong Tong, and Guo-Wei Wei. Hermes: Persistent spectral graph software. *Foundations of data science (Springfield, Mo.)*, 3(1):67, 2021.
- [49] Yuchi Qiu and Guo-Wei Wei. Persistent spectral theory-guided protein engineering. *Nature Computational Science*, 3(2):149–163, 2023.
- [50] Jiahui Chen, Yuchi Qiu, Rui Wang, and Guo-Wei Wei. Persistent laplacian projected omicron ba. 4 and ba. 5 to become new dominating variants. *Computers in Biology and Medicine*, 151:106262, 2022.
- [51] Zhenyu Meng and Kelin Xia. Persistent spectral-based machine learning (perspect ml) for protein-ligand binding affinity prediction. *Science advances*, 7(19):eabc5329, 2021.
- [52] Sean Cottrell, Rui Wang, and Guowei Wei. PLPCA: Persistent Laplacian enhanced-PCA for microarray data analysis. *Journal of Chemical Information and Modeling*, doi.org/10.1021/acs.jcim.3c01023, 2023.
- [53] Masatoshi Nei. Phylogenetic analysis in molecular evolutionary genetics. *Annual review of genetics*, 30(1):371–403, 1996.
- [54] Stephen F Altschul, Thomas L Madden, Alejandro A Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997.

- [55] William R Pearson and David J Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences*, 85(8):2444–2448, 1988.
- [56] Robert D Finn, Alex Bateman, Jody Clements, Penelope Coghill, Ruth Y Eberhardt, Sean R Eddy, Andreas Heger, Kirstie Hetherington, Liisa Holm, Jaina Mistry, et al. Pfam: the protein families database. *Nucleic acids research*, 42(D1):D222–D230, 2014.
- [57] Aaron E Darling, Bob Mau, and Nicole T Perna. progressivemauve: multiple genome alignment with gene gain, loss and rearrangement. *PloS one*, 5(6):e11147, 2010.
- [58] Mathieu Blanchette, W James Kent, Cathy Riemer, Laura Elnitski, Arian FA Smit, Krishna M Roskin, Robert Baertsch, Kate Rosenbloom, Hiram Clawson, Eric D Green, et al. Aligning multiple genomic sequences with the threaded blockset aligner. *Genome research*, 14(4):708–715, 2004.
- [59] Julie D Thompson, Desmond G Higgins, and Toby J Gibson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic acids research*, 22(22):4673–4680, 1994.
- [60] Kazutaka Katoh, Kazuharu Misawa, Kei-ichi Kuma, and Takashi Miyata. Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic acids research*, 30(14):3059–3066, 2002.
- [61] Robert C Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–1797, 2004.
- [62] Yuta Hozumi, Rui Wang, Changchuan Yin, and Guo-Wei Wei. Umap-assisted k-means clustering of large-scale sars-cov-2 mutation datasets. *Computers in biology and medicine*, 131:104264, 2021.
- [63] Jiahui Chen and Guo-Wei Wei. Omicron ba. 2 (b. 1.1. 529.2): high potential for becoming the next dominant variant. *The journal of physical chemistry letters*, 13(17):3840–3849, 2022.
- [64] Michael Bleher, Lukas Hahn, Maximilian Neumann, Juan Angel Patino-Galindo, Mathieu Carriere, Ulrich Bauer, Raul Rabadan, and Andreas Ott. Topological data analysis identifies emerging adaptive mutations in sars-cov-2. *arXiv preprint arXiv:2106.07292*, 2021.
- [65] Juan Ángel Patiño-Galindo, Ioan Filip, Ratul Chowdhury, Costas D Maranas, Peter K Sorger, Mohammed AlQuraishi, and Raul Rabadan. Recombination and lineage-specific mutations linked to the emergence of sars-cov-2. *Genome Medicine*, 13:1–14, 2021.
- [66] Sean R Eddy. Where did the blosum62 alignment score matrix come from? *Nature biotechnology*, 22(8):1035–1036, 2004.

- [67] Paul P Gardner, Andreas Wilm, and Stefan Washietl. A benchmark of multiple sequence alignment programs upon structural rnas. *Nucleic acids research*, 33(8):2433–2439, 2005.
- [68] Emidio Capriotti and Marc A Marti-Renom. Quantifying the relationship between sequence and three-dimensional structure conservation in rna. *BMC bioinformatics*, 11:1–10, 2010.
- [69] Andrzej Zielezinski, Susana Vinga, Jonas Almeida, and Wojciech M Karlowski. Alignment-free sequence comparison: benefits, applications, and tools. *Genome biology*, 18:1–17, 2017.
- [70] B Edwin Blaisdell. A measure of the similarity of sets of sequences not requiring sequence alignment. *Proceedings of the National Academy of Sciences*, 83(14):5155–5159, 1986.
- [71] Tiee-Jian Wu, John P Burke, and Daniel B Davison. A measure of dna sequence dissimilarity based on mahalanobis distance between frequencies of words. *Biometrics*, pages 1431–1439, 1997.
- [72] Tiee-Jian Wu, Ya-Ching Hsieh, and Lung-An Li. Statistical measures of dna sequence dissimilarity under markov chain models of base composition. *Biometrics*, 57(2):441–448, 2001.
- [73] Tiee-Jian Wu, Ying-Hsueh Huang, and Lung-An Li. Optimal word sizes for dissimilarity measures and estimation of the degree of dissimilarity between dna sequences. *Bioinformatics*, 21(22):4125–4132, 2005.
- [74] Ian F Korf and Alan B Rose. Applying word-based algorithms: the imeter. *Plant Systems Biology*, pages 287–301, 2009.
- [75] Se-Ran Jun, Gregory E Sims, Guohong A Wu, and Sung-Hou Kim. Whole-proteome phylogeny of prokaryotes by feature frequency profiles: An alignment-free method with optimal feature resolution. *Proceedings of the National Academy of Sciences*, 107(1):133–138, 2010.
- [76] Chenglong Yu, Troy Hernandez, Hui Zheng, Shek-Chung Yau, Hsin-Hsiung Huang, Rong Lucy He, Jie Yang, and Stephen S-T Yau. Real time classification of viruses in 12 dimensions. *PloS one*, 8(5):e64328, 2013.
- [77] Mo Deng, Chenglong Yu, Qian Liang, Rong L He, and Stephen S-T Yau. A novel method of characterizing genetic sequences: genome space with biological distance and applications. *PloS one*, 6(3):e17293, 2011.
- [78] Igor Ulitsky, David Burstein, Tamir Tuller, and Benny Chor. The average common substring approach to phylogenomic reconstruction. *Journal of Computational Biology*, 13(2):336–350, 2006.

- [79] Chris-Andre Leimeister and Burkhard Morgenstern. Kmacs: the k-mismatch average common substring approach to alignment-free sequence comparison. *Bioinformatics*, 30(14):2000–2008, 2014.
- [80] Lianping Yang, Xiangde Zhang, Haoyue Fu, and Chenhui Yang. An estimator for local analysis of genome based on the minimal absent word. *Journal of Theoretical Biology*, 395:23–30, 2016.
- [81] Lianping Yang, Xiangde Zhang, and Hegui Zhu. Alignment free comparison: similarity distribution between the dna primary sequences based on the shortest absent word. *Journal of theoretical biology*, 295:125–131, 2012.
- [82] Susana Vinga. Information theory applications for biological sequence analysis. *Briefings in bioinformatics*, 15(3):376–389, 2014.
- [83] H Joel Jeffrey. Chaos game representation of gene structure. *Nucleic acids research*, 18(8):2163–2170, 1990.
- [84] Milan Randić, Marjana Novič, and Dejan Plavšić. Milestones in graphical bioinformatics. *International Journal of Quantum Chemistry*, 113(22):2413–2446, 2013.
- [85] Pradeep Kumar Burma, Alok Raj, Jayant K Deb, and Samir K Brahmachari. Genome analysis: a new approach for visualization of sequence organization in genomes. *Journal of biosciences*, 17:395–411, 1992.
- [86] Jonas S Almeida, Joao A Carrico, Antonio Maretzek, Peter A Noble, and Madilyn Fletcher. Analysis of genomic sequences by chaos game representation. *Bioinformatics*, 17(5):429–437, 2001.
- [87] Patrick J Deschavanne, Alain Giron, Joseph Vilain, Guillaume Fagot, and Bernard Fertil. Genomic signature: characterization and classification of species assessed by chaos game representation of sequences. *Molecular biology and evolution*, 16(10):1391–1399, 1999.
- [88] Bai-Lin Hao. Fractals from genomes—exact solutions of a biology-inspired problem. *Physica A: Statistical Mechanics and its Applications*, 282(1-2):225–246, 2000.
- [89] Tung Hoang, Changchuan Yin, Hui Zheng, Chenglong Yu, Rong Lucy He, and Stephen S-T Yau. A new method to cluster dna sequences using fourier power spectrum. *Journal of theoretical biology*, 372:135–145, 2015.
- [90] Changchuan Yin, Ying Chen, and Stephen S-T Yau. A measure of dna sequence similarity by fourier transform with applications on hierarchical clustering. *Journal of theoretical biology*, 359:18–28, 2014.
- [91] Ajay Kumar Saw, Garima Raj, Manashi Das, Narayan Chandra Talukdar, Binod Chandra



- Tripathy, and Soumyadeep Nandi. Alignment-free method for dna sequence clustering using fuzzy integral similarity. *Scientific reports*, 9(1):3753, 2019.
- [92] Chenglong Yu, Qian Liang, Changchuan Yin, Rong L He, and Stephen S-T Yau. A novel construction of genome space with biological geometry. *DNA research*, 17(3):155–168, 2010.
  - [93] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. Single-cell rna sequencing technologies and bioinformatics pipelines. *Experimental & molecular medicine*, 50(8):1–14, 2018.
  - [94] Tallulah S Andrews, Vladimir Yu Kiselev, Davis McCarthy, and Martin Hemberg. Tutorial: guidelines for the computational analysis of single-cell rna sequencing data. *Nature protocols*, 16(1):1–9, 2021.
  - [95] Malte D Luecken and Fabian J Theis. Current best practices in single-cell rna-seq analysis: a tutorial. *Molecular systems biology*, 15(6):e8746, 2019.
  - [96] Geng Chen, Baitang Ning, and Tielu Shi. Single-cell rna-seq technologies and related computational data analysis. *Frontiers in genetics*, page 317, 2019.
  - [97] Raphael Petegrosso, Zhuliu Li, and Rui Kuang. Machine learning and statistical methods for clustering single-cell rna-sequencing data. *Briefings in bioinformatics*, 21(4):1209–1223, 2020.
  - [98] Wei Vivian Li and Jingyi Jessica Li. A statistical simulator scdesign for rational scrna-seq experimental design. *Bioinformatics*, 35(14):i41–i50, 2019.
  - [99] Ruiqing Zheng, Min Li, Zhenlan Liang, Fang-Xiang Wu, Yi Pan, and Jianxin Wang. Sinnlrr: a robust subspace clustering method for cell type detection by non-negative and low-rank representation. *Bioinformatics*, 35(19):3642–3650, 2019.
  - [100] Zhenqiu Shu, Qinghan Long, Luping Zhang, Zhengtao Yu, and Xiao-Jun Wu. Robust graph regularized nmf with dissimilarity and similarity constraints for scrna-seq data clustering. *Journal of Chemical Information and Modeling*, 62(23):6271–6286, 2022.
  - [101] Peng Wu, Mo An, Hai-Ren Zou, Cai-Ying Zhong, Wei Wang, and Chang-Peng Wu. A robust semi-supervised nmf model for single cell rna-seq data. *PeerJ*, 8:e10091, 2020.
  - [102] Jianwei Chen. Detecting cell type from single cell rna sequencing based on deep bi-stochastic graph regularized matrix factorization. *bioRxiv*, 2022.
  - [103] Qiu Xiao, Jiawei Luo, Cheng Liang, Jie Cai, and Pingjian Ding. A graph regularized non-negative matrix factorization method for identifying microrna-disease associations. *Bioinformatics*, 34(2):239–248, 2018.

- [104] Na Yu, Ying-Lian Gao, Jin-Xing Liu, Juan Wang, and Junliang Shang. Robust hypergraph regularized non-negative matrix factorization for sample clustering and feature selection in multi-view gene expression data. *Human genomics*, 13(1):1–10, 2019.
- [105] Jin-Xing Liu, Dong Wang, Ying-Lian Gao, Chun-Hou Zheng, Jun-Liang Shang, Feng Liu, and Yong Xu. A joint-l2, 1-norm-constraint-based semi-supervised feature extraction for rna-seq data analysis. *Neurocomputing*, 228:263–269, 2017.
- [106] Bo Wang, Junjie Zhu, Emma Pierson, Daniele Ramazzotti, and Serafim Batzoglou. Visualization and analysis of single-cell rna-seq data by kernel-based similarity learning. *Nature methods*, 14(4):414–416, 2017.
- [107] Shuangge Ma and Ying Dai. Principal component analysis based methods in bioinformatics studies. *Briefings in bioinformatics*, 12(6):714–722, 2011.
- [108] Seyoung Park and Hongyu Zhao. Sparse principal component analysis with missing observations. *The Annals of Applied Statistics*, 13(2):1016–1042, 2019.
- [109] F William Townes, Stephanie C Hicks, Martin J Aryee, and Rafael A Irizarry. Feature selection and dimension reduction for single-cell rna-seq based on a multinomial model. *Genome biology*, 20:1–16, 2019.
- [110] Dmitry Kobak and Philipp Berens. The art of using t-sne for single-cell transcriptomics. *Nature communications*, 10(1):1–14, 2019.
- [111] 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.
- [112] Carlos Torroja and Fatima Sanchez-Cabo. DigitalDlSorter: deep-learning on scrna-seq to deconvolute gene expression data. *Frontiers in Genetics*, 10:978, 2019.
- [113] Musu Yuan, Liang Chen, and Minghua Deng. scmra: a robust deep learning method to annotate scrna-seq data with multiple reference datasets. *Bioinformatics*, 38(3):738–745, 2022.
- [114] Zixiang Luo, Chenyu Xu, Zhen Zhang, and Wenfei Jin. A topology-preserving dimensionality reduction method for single-cell rna-seq data using graph autoencoder. *Scientific reports*, 11(1):20028, 2021.
- [115] Dongfang Wang and Jin Gu. Vasc: dimension reduction and visualization of single-cell rna-seq data by deep variational autoencoder. *Genomics, proteomics & bioinformatics*, 16(5):320–331, 2018.
- [116] Eugene Lin, Sudipto Mukherjee, and Sreeram Kannan. A deep adversarial variational

- autoencoder model for dimensionality reduction in single-cell rna sequencing analysis. *BMC bioinformatics*, 21(1):1–11, 2020.
- [117] Hao Wu, Haoru Zhou, Bing Zhou, and Meili Wang. Scmcluster: a high-precision cell clustering algorithm integrating marker gene set with single-cell rna sequencing data. *Briefings in Functional Genomics*, page elad004, 2023.
  - [118] Junlin Xu, Jielin Xu, Yajie Meng, Changcheng Lu, Lijun Cai, Xiangxiang Zeng, Ruth Nussinov, and Feixiong Cheng. Graph embedding and gaussian mixture variational autoencoder network for end-to-end analysis of single-cell rna sequencing data. *Cell Reports methods*, 3(1), 2023.
  - [119] Mehrshad Sadria and Anita Layton. The power of two: integrating deep diffusion models and variational autoencoders for single-cell transcriptomics analysis. *bioRxiv*, pages 2023–04, 2023.
  - [120] Alessandro Palma, Fabian J Theis, and Mohammad Lotfollahi. Predicting cell morphological responses to perturbations using generative modeling. *bioRxiv*, pages 2023–07, 2023.
  - [121] Valentina Giansanti, Francesca Giannese, Oronza A Botrugno, Giorgia Gandolfi, Chiara Balestrieri, Marco Antoniotti, Giovanni Tonon, and Davide Cittaro. Scalable integration of multiomic single cell data using generative adversarial networks. *bioRxiv*, pages 2023–06, 2023.
  - [122] Julius B Kirkegaard. Spontaneously breaking of symmetry in overlapping cell instance segmentation using diffusion models. *bioRxiv*, pages 2023–07, 2023.
  - [123] Hongru Shen, Jilei Liu, Jiani Hu, Xilin Shen, Chao Zhang, Dan Wu, Mengyao Feng, Meng Yang, Yang Li, Yichen Yang, et al. Generative pretraining from large-scale transcriptomes for single-cell deciphering. *Isience*, 26(5), 2023.
  - [124] William Connell, Umair Khan, and Michael J Keiser. A single-cell gene expression language model. *arXiv preprint arXiv:2210.14330*, 2022.
  - [125] Fan Yang, Wenchuan Wang, Fang Wang, Yuan Fang, Duyu Tang, Junzhou Huang, Hui Lu, and Jianhua Yao. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature Machine Intelligence*, 4(10):852–866, 2022.
  - [126] Jiawei Chen, Hao Xu, Wanyu Tao, Zhaoxiong Chen, Yuxuan Zhao, and Jing-Dong J Han. Transformer for one stop interpretable cell type annotation. *Nature Communications*, 14(1):223, 2023.
  - [127] Jing Xu, Aidi Zhang, Fang Liu, Liang Chen, and Xiujun Zhang. Ciform as a transformer-based model for cell-type annotation of large-scale single-cell rna-seq data. *Briefings in Bioinformatics*, page bbad195, 2023.

- [128] Linfang Jiao, Gan Wang, Huanhuan Dai, Xue Li, Shuang Wang, and Tao Song. setranssort: Transformers for intelligent annotation of cell types by gene embeddings. *Biomolecules*, 13(4):611, 2023.
- [129] Vladimir Yu Kiselev, Kristina Kirschner, Michael T Schaub, Tallulah Andrews, Andrew Yiu, Tamir Chandra, Kedar N Natarajan, Wolf Reik, Mauricio Barahona, Anthony R Green, et al. Sc3: consensus clustering of single-cell rna-seq data. *Nature methods*, 14(5):483–486, 2017.
- [130] Pengyu Zhang, Hongming Zhang, and Hao Wu. ipro-wael: a comprehensive and robust framework for identifying promoters in multiple species. *Nucleic Acids Research*, 50(18):10278–10289, 2022.
- [131] Pengyu Zhang, Yingfu Wu, Haoru Zhou, Bing Zhou, Hongming Zhang, and Hao Wu. Clnn-loop: a deep learning model to predict ctcf-mediated chromatin loops in the different cell lines and ctcf-binding sites (cbs) pair types. *Bioinformatics*, 38(19):4497–4504, 2022.
- [132] Pengyu Zhang and Hao Wu. Ichrom-deep: An attention-based deep learning model for identifying chromatin interactions. *IEEE Journal of Biomedical and Health Informatics*, 2023.
- [133] Heather J Zhou, Lei Li, Yumei Li, Wei Li, and Jingyi Jessica Li. Pca outperforms popular hidden variable inference methods for molecular qtl mapping. *Genome Biology*, 23(1):1–17, 2022.
- [134] Suoqin Jin, Christian F Guerrero-Juarez, Lihua Zhang, Ivan Chang, Raul Ramos, Chen-Hsiang Kuan, Peggy Myung, Maksim V Plikus, and Qing Nie. Inference and analysis of cell-cell communication using cellchat. *Nature communications*, 12(1):1088, 2021.
- [135] Floyd Maseda, Zixuan Cang, and Qing Nie. Deepsc: a deep learning-based map connecting single-cell transcriptomics and spatial imaging data. *Frontiers in Genetics*, 12:636743, 2021.
- [136] Yuhan Hao, Stephanie Hao, Erica Andersen-Nissen, William M Mauck, Shiwei Zheng, Andrew Butler, Maddie J Lee, Aaron J Wilk, Charlotte Darby, Michael Zager, et al. Integrated analysis of multimodal single-cell data. *Cell*, 184(13):3573–3587, 2021.
- [137] Hannah A Pliner, Jay Shendure, and Cole Trapnell. Supervised classification enables rapid annotation of cell atlases. *Nature methods*, 16(10):983–986, 2019.
- [138] Ze Zhang, Danni Luo, Xue Zhong, Jin Huk Choi, Yuanqing Ma, Stacy Wang, Elena Mahrt, Wei Guo, Eric W Stawiski, Zora Modrusan, et al. Scina: a semi-supervised subtyping algorithm of single cells and bulk samples. *Genes*, 10(7):531, 2019.

## CHAPTER 2

### BACKGROUND

#### 2.1 Overview of Dimensionality Reduction

##### 2.1.1 Principle Components Analysis

Principal component analysis (PCA) is one of the most commonly used dimensional reduction techniques for the exploratory analysis of high-dimensional data [1]. The first component, termed the principal component, is the direction that maximizes the variance, while the subsequent components are orthogonal to earlier ones.

Let  $X = \{\mathbf{x}_i\}_{i=1}^N$  be the input dataset, with  $N$  being the number of samples or data points. For each  $\mathbf{x}_i$ , let  $\mathbf{x}_i \in \mathbb{R}^M$ , where  $M$  is the number of features or data dimension. PCA seeks to find a linear combination of the columns of  $X$  with maximum variance.

$$\sum_{j=1}^n a_j \mathbf{x}_j = X\mathbf{a}, \quad (2.1)$$

where  $a_1, a_2, \dots, a_n$  are constants, and  $\mathbf{a}$  is the vectorized  $a_1, a_2, \dots, a_n$ . The variance of this linear combination is defined as

$$\text{var}(X\mathbf{a}) = \mathbf{a}^T S \mathbf{a}, \quad (2.2)$$

where  $S$  is the covariance matrix for the dataset. Note that we compute the eigenvalue of the covariance matrix. The maximum variance can be computed iteratively using Rayleigh's quotient

$$a_{(1)} = \arg \max_{\mathbf{a}} \frac{\mathbf{a}^T X^T X \mathbf{a}}{\mathbf{a}^T \mathbf{a}}. \quad (2.3)$$

The subsequent components can be computed by maximizing the variance of

$$\hat{X}_k = X - \sum_{j=1}^{k-1} X a_j a_j^T, \quad (2.4)$$

where  $k$  represents the  $k$ th principal component. Here,  $k - 1$  principal components are subtracted from the original matrix  $X$ . Therefore, the complexity of the method scales linearly with the number of components one seeks to find. In applications, we hope that the first few components give rise to a good PCA representation of the original data matrix  $X$ .

### 2.1.2 Nonnegative matrix factorization

Nonnegative matrix factorization (NMF) decomposes the original data matrix into 2 submatrix that are both nonnegative. Unlike PCA, NMF requires that the original matrix is nonnegative rather than orthogonal [2]. More formally, NMF solves the following optimization problem

$$\min_{W, H} \|X - WH\|_F^2, \quad \text{s.t. } W, H \geq 0 \quad (2.5)$$

where  $\|A\|_F^2 = \sum_{i,j} a_{ij}^2$  is the Frobenius norm. Lee et al.[3] proposed a multiplicative updating scheme, which preserves the nonnegativity. For the  $t + 1$ th iteration,

$$w^{t+1}_{ij} = w^t_{ij} \frac{(XH^T)_{ij}}{(WHH^T)_{ij}} \quad (2.6)$$

$$h^{t+1}_{ij} = h^t_{ij} \frac{(W^T X)_{ij}}{(W^T WH)_{ij}} \quad (2.7)$$

Note that NMF is convex in  $W$  or  $H$ , but is not convex in both  $W$  and  $H$ .

### 2.1.3 T-Distributed Stochastic Neighbor Embedding

The t-distributed stochastic neighbor embedding (t-SNE) is a nonlinear dimensional reduction algorithm that is well suited for reducing high dimensional data into the two- or three-dimensional space. There are two main stages in t-SNE. First, it constructs a probability distribution over pairs of data such that closer data points are assigned a high probability, while farther points are given a low probability. Second, t-SNE defines a probability distribution in the embedded space that is similar to that in the original high-dimensional space, and aims to minimize the Kullback-Leibler (KL) divergence between them [4].

Let  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N | \mathbf{x}_i \in \mathbb{R}^M\}$  be a high dimensional input dataset. Our goal is to find an optimal low dimensional representation  $\{\mathbf{y}_1, \dots, \mathbf{y}_N | \mathbf{y}_i \in \mathbb{R}^k\}$ , such that  $k \ll M$ . The first step in t-SNE is to compute the pairwise distribution between  $\mathbf{x}_i$  and  $\mathbf{x}_j$ , denoted as  $p_{ij}$ . First, we find the conditional probability of  $\mathbf{x}_j$ , given  $\mathbf{x}_i$ :

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2)}{\sum_{m \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_m\|^2 / 2\sigma_i^2)}, \quad i \neq j, \quad (2.8)$$

setting  $p_{i|i} = 0$ , and the denominator normalizes the probability. Here,  $\sigma_i$  is the predefined hyperparameter called perplexity. A smaller  $\sigma_i$  is used for a denser dataset. Notice that this

conditional probability is symmetric when the perplexity is fixed, i.e.  $p_{ij} = p_{ji}$ . Then, define the pairwise probability as

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}. \quad (2.9)$$

In the second step, we learn a  $k$ -dimensional embedding  $\{\mathbf{y}_1, \dots, \mathbf{y}_N | \mathbf{y}_i \in \mathbb{R}^k\}$ . To this end, t-SNE calculates a similar probability distribution  $q_{ij}$  defined as

$$q_{ij} = \frac{\frac{1}{1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2}}{\sum_m \sum_{l \neq m} \frac{1}{1 + \|\mathbf{y}_m - \mathbf{y}_l\|^2}}, \quad i \neq j \quad (2.10)$$

and setting  $q_{ii} = 0$ . Finally, the low dimensional embedding  $\{\mathbf{y}_1, \dots, \mathbf{y}_N | \mathbf{y}_i \in \mathbb{R}^k\}$  is found by minimizing the KL-divergence via a standard gradient descent method

$$\text{KL}(P|Q) = \sum_{i,j} p_{ij} \log \frac{p_{ij}}{q_{ij}}, \quad (2.11)$$

where  $P$  and  $Q$  are the distributions for  $p_{ij}$  and  $q_{ij}$ , respectively. Note that the probability distributions in Equation 2.8 and Equation 2.10 can be replaced by using many other delta sequence kernel of positive type [5].

#### 2.1.4 Uniform Manifold Approximation and Projection

Uniform manifold approximation and projection (UMAP) is a nonlinear dimensional reduction method, utilizing three assumptions: the data is uniformly distributed on Riemannian manifold, Riemannian metric is locally constant, and the manifold is locally connected. Unlike t-SNE which utilizes probabilistic model, UMAP is a graph-based algorithm. The essential idea is to create a predefined  $k$ -dimensional weighted UMAP graph representation for each of the original high-dimensional data points, with the aim of minimizing the edge-wise cross-entropy between the weighted graph and the original data. Finally, the  $k$ -dimensional eigenvectors of the UMAP graph are used to represent each of the original data point. In this section, a computational view of UMAP is presented. For a more theoretical account, the reader is referred to Ref. [6].

Similar to t-SNE, UMAP considers the input data  $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_i \in \mathbb{R}^M$  and look for an optimal low dimensional representation  $\{\mathbf{y}_1, \dots, \mathbf{y}_N | \mathbf{y}_i \in \mathbb{R}^k\}$ , such that  $k < M$ . The first stage is the

construction of weighted  $k$ -neighbor graphs. Define a metric  $d : X \times X \rightarrow \mathbb{R}^+$ . Let  $k \ll M$  be a hyperparameter, and compute the  $k$ -nearest neighbors of each  $x_i$  under a given metric  $d$ . For each  $x_i$ , let

$$\rho_i = \min\{d(\mathbf{x}_i, \mathbf{x}_j) | 1 \leq j \leq k, d(\mathbf{x}_i, \mathbf{x}_j) > 0\} \quad (2.12)$$

where  $\sigma_i$  is defined via

$$\sum_{j=1}^k \exp\left(\frac{-\max(0, d(\mathbf{x}_i, \mathbf{x}_j) - \rho_i)}{\sigma_i}\right) = \log_2 k. \quad (2.13)$$

Such choice of  $\rho_i$  ensure at least one data point is connected to  $\mathbf{x}_i$  and having edge weight of 1, and set  $\sigma_i$  as a scaling parameter. Then, define a weighted directed graph  $\bar{G} = (V, E, \omega)$ , where  $V$  is the set of vertices (in this case, the data  $X$ ),  $E$  is the set of edges  $E = \{(\mathbf{x}_i, \mathbf{x}_j) | 1 \leq j \leq k, 1 \leq i \leq N\}$ , and  $\omega$  is the weight for edges

$$\omega(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\max(0, d(\mathbf{x}_i, \mathbf{x}_j) - \rho_i)}{\sigma_i}\right). \quad (2.14)$$

UMAP tries to define an undirected weighted graph  $G$  from directed graph  $\bar{G}$  via symmetrization. Let  $A$  be the adjacency matrix of the graph  $\bar{G}$ . A symmetric matrix can be obtained

$$B = A + A^T - A \otimes A^T, \quad (2.15)$$

where  $T$  is the transpose and  $\otimes$  denotes the Hadamard product. Then, the undirected weighted Laplacian  $G$  (the UMAP graph) is defined by its adjacency matrix  $B$ .

In its realization, UMAP evolves an equivalent weighted graph  $H$  with a set of points  $\{\mathbf{y}_i\}_{i=1}^N$ , utilizing attractive and repulsive forces. The attractive and repulsive forces at coordinate  $\mathbf{y}_i$  and  $\mathbf{y}_j$  are given by

$$\frac{-2ab\|\mathbf{y}_i - \mathbf{y}_j\|_2^{2(b-1)}}{1 + \|\mathbf{y}_i - \mathbf{y}_j\|_2^2} \omega(\mathbf{x}_i, \mathbf{x}_j)(\mathbf{y}_i - \mathbf{y}_j), \text{ and} \quad (2.16)$$

$$\frac{2b}{(\epsilon + \|\mathbf{y}_i - \mathbf{y}_j\|_2^2)(1 + a\|\mathbf{y}_i - \mathbf{y}_j\|_2^{2b})} (1 - \omega(\mathbf{x}_i, \mathbf{x}_j))(\mathbf{y}_i - \mathbf{y}_j) \quad (2.17)$$

where  $a, b$  are hyperparameters, and  $\epsilon$  is taken to be a small value such that the denominator does not become 0. The goal is to find the optimal low-dimensional coordinates  $\{\mathbf{y}_i\}_{i=1}^N$ ,  $\mathbf{y}_i \in \mathbb{R}^k$ , that



minimizes the edge-wise cross entropy with the original data at each point. The evolution of the UMAP graph Laplacian  $G$  can be regarded as a discrete approximation of the Laplace-Beltrami operator on a manifold defined by the data [7]. Implementation and further detail of UMAP can be found in Ref. [6].

UMAP may not work well if the data points are non-uniform. If part of the data points have  $k$  important neighbors while other part of the data points have  $k' \gg k$  important neighbors, the  $k$ -dimensional UMAP will not work efficiently. Currently, there is no algorithm to automatically determine the critic minimal  $k_{\min}$  for a given dataset. Additionally, weights  $\omega(\mathbf{x}_i, \mathbf{x}_j)$  and force terms can be replaced by other functions that are easier to evaluate [5]. The metric  $d$  can be selected as Euclidean distance, Manhattan distance, Minkowski distance, and Chebyshev distance, depending on applications.

## 2.2 Clustering Algorithm

### 2.2.1 K-means Clustering

$K$ -means is the most commonly used unsupervised clustering method, where the aim is to partition  $\{\mathbf{x}_j | \mathbf{x}_j \in \mathbb{R}^m, 1 \leq j \leq N\}$  into  $K$  clusters  $\{C_1, \dots, C_k\}$ , where  $K \ll N$  [8].

$K$ -means begins by randomly selecting  $K$  points to be the cluster centers, called centroids. Here, centroids are denoted as  $\mu_1, \dots, \mu_k, \dots, \mu_K$ , and the centroid  $\mu_k$  is the center of cluster  $C_k$ . Then, each sample is assigned to the nearest centroid, forming the initial clusters. Centroids are then updated by minimizing the within-cluster sum of squares (WCSS), which is defined as:

$$\sum_{k=1}^K \sum_{\mathbf{x}_j \in C_k} \|\mathbf{x}_j - \mu_k\|^2, \quad (2.18)$$

which gives the updating scheme

$$\mu_k = \frac{1}{|C_k|} \sum_{\mathbf{x}_j \in C_k} \mathbf{x}_j.$$

This process is repeated until convergence or until the maximal number of iterations is reached.

### 2.2.2 $K$ -medoids Clustering

$K$ -medoids clustering is similar to  $k$ -means clustering. The main difference is that in  $k$ -means clustering, the centroid or the center of the cluster is not necessarily a sample in the data because the centroid is computed by taking the average of the samples in the cluster.  $K$ -medoids, on the other hand, will chose the medoids, or the ‘cluster center’ to be a sample of the data [9].

For a pre-selected  $K$ , we begin by randomly selecting  $K$  medoids  $\{\mathbf{m}^k\}_{k=1}^K$  and assign each vector to its nearest medoid, which gives rise to the initial partition  $\{C^k\}_{k=1}^K$ . Second, we denote the closest vector to the center of the  $k$ th partition  $C^k$  as the new medoid  $\{\mathbf{m}^k \in C^k\}_{k=1}^K$ . We reassign each vector into its nearest medoid, resulting in a new partition  $\{C^k\}_{k=1}^K$  to minimize the loss function or the accumulated distance. The process is repeated until  $\{C^k\}_{k=1}^K$  is optimized with respect to a specific distance definition,

$$\arg \min_{\{C^1, \dots, C^K\}} \sum_{k=1}^K \sum_{\mathbf{x}_i \in C^k} \|\mathbf{x}_i - \mathbf{m}^k\|, \quad (2.19)$$

where  $\|\cdot\|$  is some distance. Any distance can be used in  $k$ -medoids, such as Euclidean, Manhattan, covariance distance, correlation distance, etc.

## 2.3 Evaluation metrics for clustering and classification

### 2.3.1 Adjusted Rand Index (ARI)

The Adjusted Rand Index (ARI) measures the agreement between two clustering results by considering all pairs of data points and comparing their assignments in the two clustering results [10]. Let  $\{T_i\}_{i=1}^p$  and  $\{S_j\}_{j=1}^q$  be two partitioning of the data. Often times,  $T_i$  are the clustering result, and  $S_j$  are the true labels. Let  $n_{ij} = |T_i \cap S_j|$  be the number of samples that belong to true label  $j$  and cluster label  $i$ , and define  $a_i = \sum_j n_{ij}$  and  $b_j = \sum_i n_{ij}$ . Then, the ARI is defined as

$$\text{ARI} = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[ \sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right] / \binom{N}{2}}{\frac{1}{2} \left[ \sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2} \right] - \sum_i \left[ \binom{a_i}{2} \sum_j \binom{b_j}{2} \right] / \binom{N}{2}} \quad (2.20)$$

The ARI takes on a value between -0.5 and 1, where 1 is perfect match between two clustering methods, and 0 is completely random assignment of labels.

### 2.3.2 Normalized Mutual Information (NMI)

The normalized mutual information (NMI) measures the mutual information between two clustering results and normalizes it according to cluster size [11]. We fix the true labels  $Y$  as one of the clustering result, and use the predicted labels as the other to calculate NMI. The NMI is calculated as the following

$$\text{NMI} = \frac{2I(Y; C)}{H(Y)H(C)} \quad (2.21)$$

where  $H(\cdot)$  is the entropy and  $I(Y; C)$  is the mutual information between true labels  $Y$  and predicted labels  $C$ . NMI has a range of 0 and 1, where 1 is a perfect mutual information between the 2 sets of labels.

### 2.3.3 Accuracy (ACC)

Accuracy (ACC) calculates the percentage of correctly predicted class labels. The accuracy is given by

$$\text{ACC} = \frac{1}{N} \sum_{i=1}^N \delta(y^i, f(c^i)), \quad (2.22)$$

where  $\delta(a, b)$  is the indicator function. That is, if  $a = b$ ,  $\delta(a, b) = 1$ , and 0 otherwise.  $f : C \rightarrow Y$  maps the cluster labels to the true labels, where the mapping is the optimal permutation of the cluster labels and true labels obtained from the Hungarian algorithm [12].

### 2.3.4 Purity

Let  $\{C_i\}_{i=1}^p$  be a cluster label and  $\{Y_j\}_{j=1}^q$  be the true label. For purity calculation, each predicted label  $C_i$  is assigned to a true label  $Y_j$  such that the  $|C_i \cap Y_j|$  is maximized [13]. Taking the average over all the predicted label, we obtain the following

$$\text{Purity} = \frac{1}{N} \sum_{i=1}^p \max_j |C_i \cap Y_j| \quad (2.23)$$

Note that unlike accuracy, purity does not map the predicted labels to the true labels.

### 2.3.5 Silhouette Score

Silhouette score is a common metric used to determine the number of clusters for  $k$ -means clustering and assumes that the clusters are well-separated [14]. Let  $\mathbf{x}_i \in A \subset \mathcal{X}$  be a sample in cluster  $A$  from the clustering algorithm. Then, we define  $a(\mathbf{x}_i)$  as the dissimilarity of sample  $\mathbf{x}_i$  to all the other samples of  $A$ . That is,

$$a(\mathbf{x}_i) = \frac{1}{|A|} \sum_{\mathbf{x}_j \in A} \|\mathbf{x}_i - \mathbf{x}_j\| \quad (2.24)$$

where  $|A|$  is the number of samples in cluster  $A$ . Now, let  $C \neq A$  be another cluster from the clustering algorithm. Then,  $b(\mathbf{x}_i)$  is the minimum average distance from  $\mathbf{x}_i$  to the closest cluster that is not  $A$ . That is,

$$b(\mathbf{x}_i) = \min_{C \neq A} \frac{1}{|C|} \sum_{\mathbf{x}_j \in C} \|\mathbf{x}_i - \mathbf{x}_j\| \quad (2.25)$$

Then, the score for sample  $\mathbf{x}_i$ , denoted as  $s(\mathbf{x}_i)$  is obtained as the following

$$s(\mathbf{x}_i) = \begin{cases} 1 - \frac{a(\mathbf{x}_i)}{b(\mathbf{x}_i)} & a(\mathbf{x}_i) < b(\mathbf{x}_i) \\ 0 & a(\mathbf{x}_i) = b(\mathbf{x}_i) \\ \frac{b(\mathbf{x}_i)}{a(\mathbf{x}_i)} - 1 & a(\mathbf{x}_i) > b(\mathbf{x}_i). \end{cases} \quad (2.26)$$

One can then define the Silhouette score  $S$  as the average of  $s(\mathbf{x}_i)$  over all samples in the dataset. The Silhouette score is bounded by -1 and 1, where -1 indicates poor clustering, 0 indicates overlapping clusters, and 1 indicates well-separated clusters.

### 2.3.6 Balanced Accuracy (BA)

Accuracy is utilized to measure the quality of a classification algorithm. For data with unbalanced class size, standard accuracy does not give meaningful result; therefore, balanced accuracy (BA) is used[15], which computes the accuracy of each class separately, and the score is the average accuracy of each class. Let  $y_i$  and  $\hat{y}_i$  be the true and predicted labels of the  $i$ -th sample, respectively. Then, the balanced accuracy score is defined as:

$$\text{BA} = \frac{1}{\omega_l} \sum_i 1_{\{y_i = \hat{y}_i\}} \omega_i \quad (2.27)$$

where  $\omega_i$  is the sample weight, given as  $\omega_i = \frac{1}{\sum_j 1_{\{y_j=y_i\}}}$ , and  $1_{\{*\}}$  is the indicator function.

## 2.4 Topological Data Analysis

### 2.4.1 Simplex

Let  $\{v_0, v_1, \dots, v_q\}$  be a set of points in  $\mathbb{R}^n$ . An affine combination is defined as

$$v = \sum_{i=0}^q \lambda_i v_i, \quad \text{such that} \quad \sum_{i=0}^q \lambda_i = 1 \quad (2.28)$$

An affine hull is a set of affine combinations. The  $q + 1$  points  $\{v_i\}_{i=0}^q$  are affinely independent if  $v_1 - v_0, \dots, v_q - v_0$  are linearly independent. A  $q$ -plane is well defined if  $q + 1$  points are affinely independent. For example, in  $\mathbb{R}^n$ , there are at most  $n$  linearly independent vectors. Therefore, there are at most  $n + 1$  affinely independent points. Additionally, an affine combination, defined as Equation 2.28, is convex combination if we add an additional condition  $\lambda_i \geq 0$  for all  $i$ . Furthermore, the convex hull as the set of convex combination.

A  $q$ -simplex denoted as  $\sigma_q$  is a convex hull of  $q + 1$  points in  $\mathbb{R}^q$ , with the dimension  $\dim(\sigma_q) = q$ .  $\sigma_0$  is a vertex,  $\sigma_1$  is an edge,  $\sigma_2$  is a triangle,  $\sigma_3$  is a tetrahedron and so on, and an example is shown in Figure 2.1.

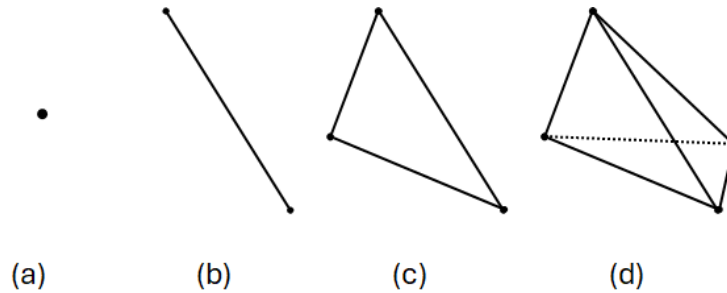


Figure 2.1 Illustration of (a) 0-simplex or vertex, (b) 1-simplex or edge, (c) 2-simplex or triangle, and (d) 3-simplex or tetrahedron.

### 2.4.2 Simplicial Complex

Let  $\sigma_q = [v_0, v_1, \dots, v_q]$  be the  $q$ -simplex, where  $v_i$  is a vertex. A simplicial complex  $K$  is a collection of simplices in  $\mathbb{R}^n$  satisfying the following conditions

1. If  $\sigma_q \in K$  and  $\sigma_p$  is a face of  $\sigma_q$ , then  $\sigma_p \in K$

2. The nonempty intersection of any 2 simplicies  $\sigma_q, \sigma_p \in K$  is a face of both  $\sigma_q$  and  $\sigma_p$

In essence, one can think of  $K$  as gluing together lower order simplicies together. Each element  $\sigma_q \in K$  is a  $q$ -simplex of  $K$ , and the dimension of  $K$  is defined as  $\dim(K) = \max\{\dim(\sigma_q) : \sigma_q \in K\}$ .

Alternatively, for  $q$  dimensional  $K$ , one can define simplicial complex as

$$K = \{\sigma_q | \sigma_q = \sum_{i=0}^n \lambda_i v_i, \lambda_i \geq 0, \sum_{i=0}^n \lambda_i = 1\} \quad (2.29)$$

### 2.4.3 Chain complex

A  $q$ -chain is a formal sum of  $q$ -simplicies in a simplicial complex  $K$  with coefficient  $\mathbb{Z}_2 \in \{0, 1\}$ . The set of all  $q$ -chains contains the basis for the set of  $q$ -simplicies in  $K$ . Such set forms a finitely generating free Abelian group  $C_q(K)$ . We can relate the chain groups by a boundary operator, which is a group homomorphism  $\partial_q : C_q(K) \rightarrow C_{q-1}(K)$ . This boundary operator is defined as

$$\partial_q \sigma_q : \sum_{i=0}^q (-1)^i \sigma_{q-1}^i \quad (2.30)$$

where  $\sigma_q^i = [v_0, v_1, \dots, v_i^*, \dots, v_q]$  be a  $(q-1)$ -simplex with the vertex  $v_i$  removed. We can then define a chain complex as a sequence of chain groups connected by the boundary operator.

$$\dots \xrightarrow{\partial_{q+2}} C_{q+1}(K) \xrightarrow{\partial_{q+1}} C_q(K) \xrightarrow{\partial_q} \dots \quad (2.31)$$

By utilizing the boundary operators, we can define the  $q$ th cycle group  $Z_q$  and the  $q$ th boundary group  $B_q$ . Both  $Z_q$  and  $B_q$  are subgroups of the  $q$ th chain group  $C_q$ , and are defined as

$$Z_q = \text{Ker} \partial_q = \{c \in C_q | \partial_q c = 0\} \quad (2.32)$$

$$B_q = \text{Im} \partial_{q+1} = \{c \in C_q | \exists d \in C_{q+1} : c = \partial_{q+1} d\} \quad (2.33)$$

Additionally,  $\partial_{q-1} \partial_q = 0$  implies that  $B_q \subseteq Z_q \subseteq C_q$ . Moreover, the  $q$ th cycle is the  $q$ -dimensional hole.

The, the  $q$ -homology group  $H_q$  is defined to be a quotient group of the  $q$ -cycle group modulo the  $q$ -boundary group, ie

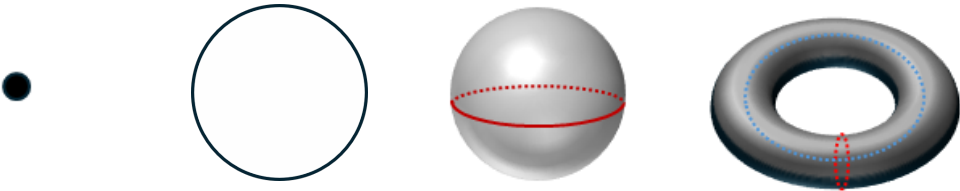
$$H_q = Z_q / B_q \quad (2.34)$$

The rank of the the  $H_q$  is the  $q$ th Betti number, denoted as  $\beta_q$ .

$$\beta_q = \text{rank}(H_q) = \text{rank}(Z_q) - \text{rank}(B_q) \quad (2.35)$$

The  $q$ th Betti number describes the  $q$ th dimensional hole. For example,  $\beta_0$  is the number of connected components,  $\beta_1$  is the number of loops,  $\beta_2$  is the number of cavity, and so on. Furthermore, the Betti numbers describe the topological property of the system. An example of Betti numbers of vertex, circle, sphere and torus can be found in Table 2.1

Table 2.1 Betti numbers of vertex, circle, sphere and torus.



$\beta_0$	1	1	1	1
$\beta_1$	0	1	0	2
$\beta_2$	0	0	1	1

## 2.4.4 Computational tools for constructing simplicial complex

In this section, we describe Vietoris-Rips (VR) complex and alpha complex.

### 2.4.4.1 Vietoris-Rips complex

VR complex is constructed from the 1-skeleton induced by pairwise distances of the point cloud. Let  $d(x, y)$  be dome distance, and  $\varepsilon$  be a threshold distance. VR complex of a finite point cloud  $X$  is given by

$$VR_\varepsilon(X) = \{\sigma \subseteq X | d(u, v) \leq \varepsilon, \forall u \neq v \in \sigma\} \quad (2.36)$$

The Euclidean distance is often used, but other distance can also be used depending of the application. Additionally, because VR complex does not rely on the exact geometry, other abstract distance that do not satisfy the triangle inequaility can be used, such as cosine distance, correlation

distance and kernel induced distance.

$$\begin{aligned}
d(x, y) &= 1 - \frac{x \cdot y}{\|x\|_2 \|y\|_2} \\
d(x, y) &= 1 - \frac{\text{cov}(x, y)}{\sigma_x \sigma_y} \\
d(x, y) &= 1 - e^{-\|x-y\|_2^2 / \tau}
\end{aligned} \tag{2.37}$$

Figure 2.2 shows a comparison of VR complex using the standard Euclidean distance and the kernel-induced distance. We can see that the kernel-induced distance has additional  $H_1$  barcode. Such kernel-induced distance has been utilized in molecular biology, where atom-specific properties, such as Van der Waal radius, electrostatic potential, etc. can be used to refine the distance. Such application has been shown to be effective at predicting B-factor and protein-ligand binding affinity [16, 17, 18].



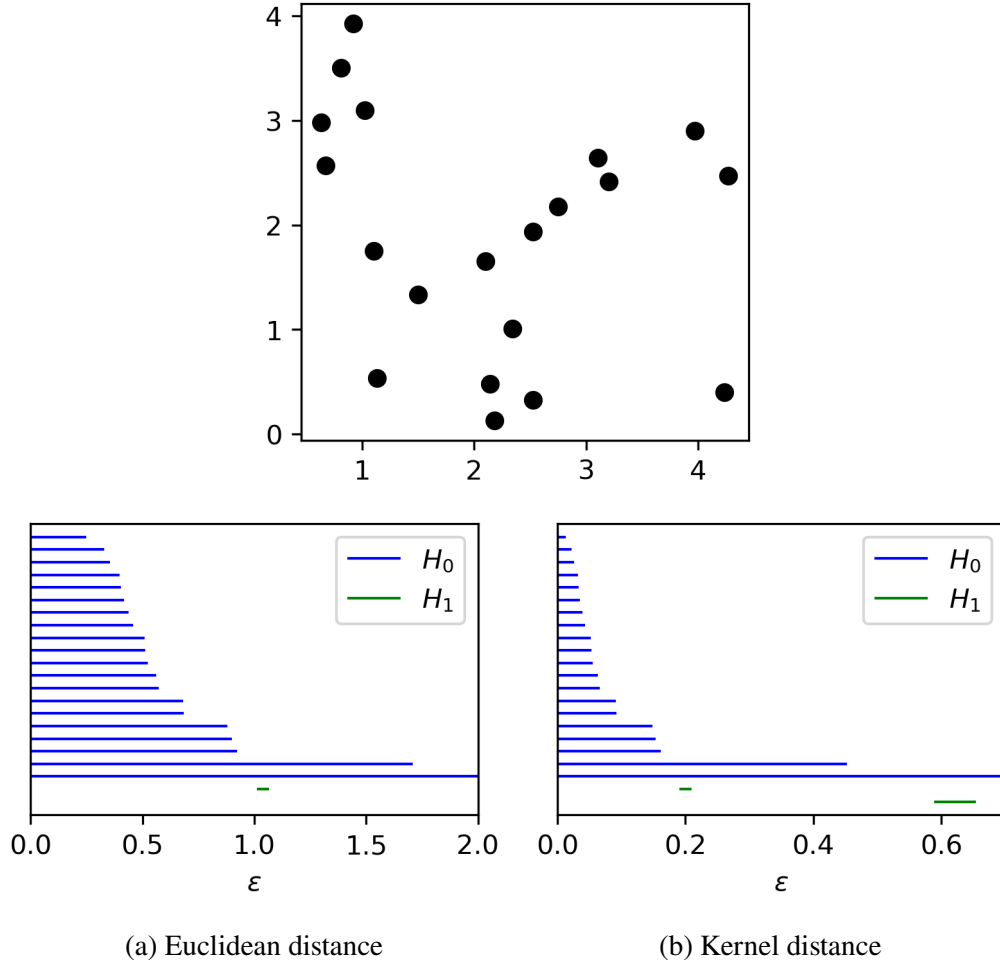


Figure 2.2 Comparison of VR complex using (a) standard Euclidean distance and (b) kernel-induced distance.

#### 2.4.4.2 Alpha complex

Unlike VR complex, alpha complex is related to geometric modeling and domain partitioning. Let  $X$  be a finite point cloud in a Euclidean space. We can build a Voroni diagram, and let  $V(x)$  be a Voronoi cell associated with  $x \in X$ . Then, for a given  $\epsilon$ , the alpha complex is given by

$$A_\epsilon(X) = \{\sigma \in X \mid \bigcap_{x \in \sigma} (V(x) \cap B_\epsilon(x)) \neq \emptyset\} \quad (2.38)$$

where  $B_\epsilon(x)$  is the ball of radius  $\epsilon$  around  $x \in X$ .

Figure 2.3 show a comparison between the alpha complex and VR complex. VR complex show a longer persistence of  $H_0$ , meaning that the points do not connect for longer period. Both complex show a loop, but the loop is located in a different interval.

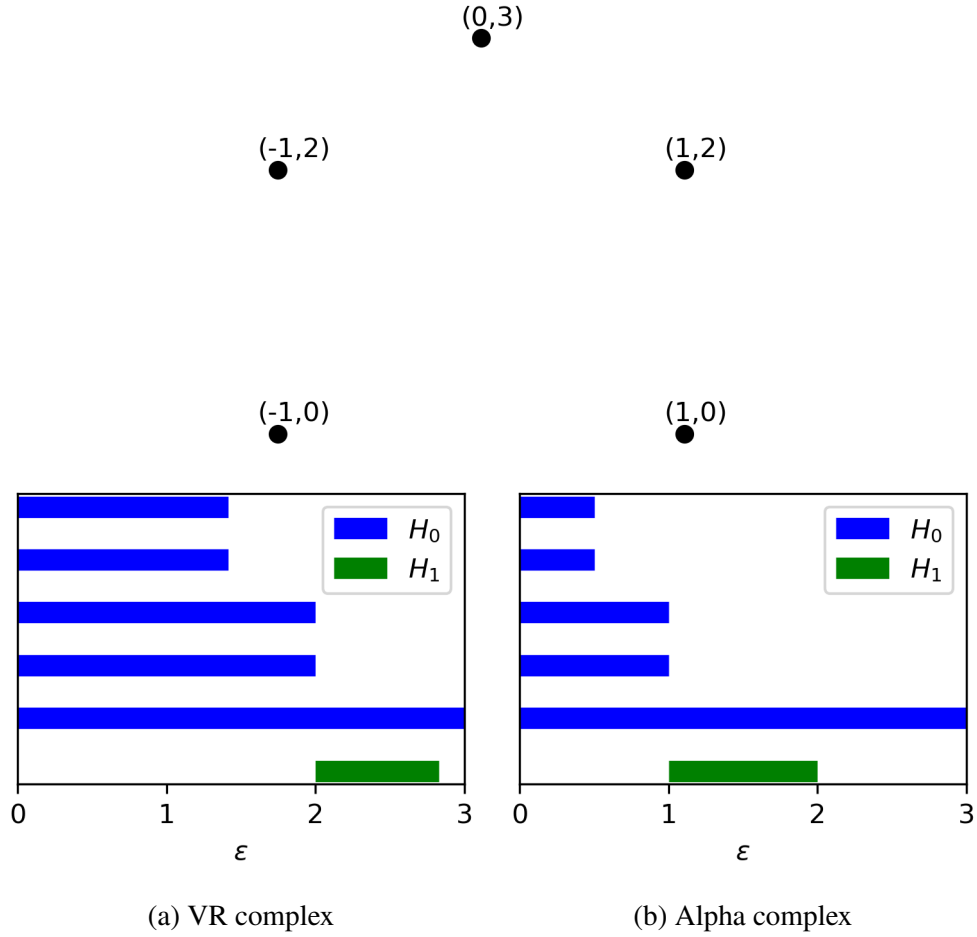


Figure 2.3 Comparison of VR complex and alpha complex on 5 points. (a) shows the barcode generated from VR complex, and (b) shows the barcode generated from alpha complex.

### 2.4.5 Persistent homology

One downside of utilizing the simplicial complex is that it does not provide sufficient information to understand the geometry of the data because we can only capture the data in a single scale. To this end, we utilize simplicial complex induced by filtration,

$$\emptyset = K_0 \subseteq K_1 \subseteq \dots \subseteq K_p = K \quad (2.39)$$

where  $P$  is the number of filtration. Using such filtration is the foundation of persistent homology, where the persistence is observed through long-lasting topological features. For each filtration  $p$ , we construct the simplicial complex, the chain group, the subgroups and the homology group. In

particular, the  $t$ -persistent  $q$ -th homology group  $K^i$  is

$$H_q^{i,t} = Z_q^i / \left( B_q^{i+t} \cap Z_q^i \right) \quad (2.40)$$

Computing the Betti numbers give the persistence of  $q$ -dimensional holes.

Figure 2.4 show an example of persistent barcode using benzene ( $C_6H_6$ ). We extracted the carbon atoms from benzene, and constructed the simplicial complex via vietoris-rips complex. Then, filtration was applied to obtain the Betti numbers. The top figures show the filtration process of benzene at  $r = 0.75, 1.50$  and  $3.00$ , and was visualized using ChimeraX [19]. Gudhi[20] was used to compute the simplicial complex and the barcodes.

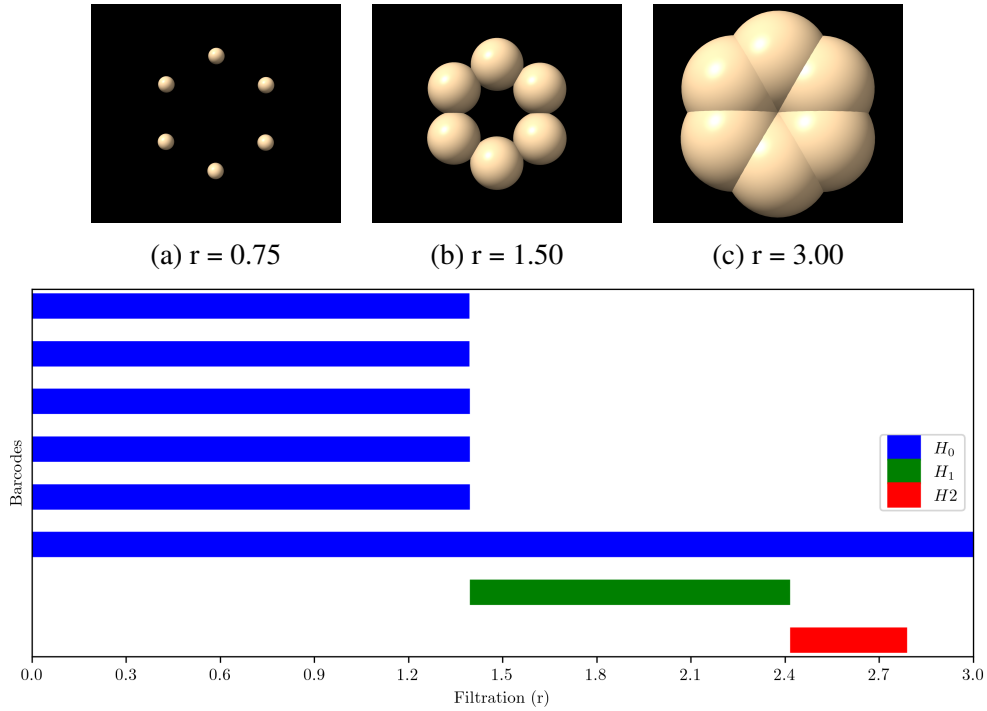


Figure 2.4 Illustration of vietoris-rips complex of benzene ( $C_6H_6$ ) as the radius increases to  $r = 3.0$ . (a)  $r = 0.75$ , (b)  $r = 1.50$  and (c)  $r = 3.00$  shows the filtration process, and the bottom figure shows the persistent barcode corresponding to the filtration.

#### 2.4.6 Combinatorial Laplacian Lapalcian

Combinatorial Laplacian gives insight from both spectral analysis and persistent homology [21, 22]. Recall that the chain complex of a simplicial complex is defined through a sequence of boundary operators, and that looking at the kernel and the image of the operators defined the cycle

group  $Z_q$  and the boundary  $B_q$ . Then, the  $q$ -th homology group  $H_q = Z_q/B_q$  (or  $H_q = \ker \partial_q / \text{Im} \partial_{q+1}$ ). Moreover, the dimensional of  $H_q$  gives the  $q$ -betti numbers.

We can now define the dual chain complex through the adjoint operator of  $\partial_q$ . The dual space is defined as  $C^q(K) \cong C_q^*(K)$ , and the coboundary operator  $\partial_q^*$  is defined as  $\partial_q^* : C^{q-1}(K) \rightarrow C^q(K)$ . For  $\omega^{q-1} \in C^{q-1}(K)$  and  $c_q \in C_q(K)$ , the coboundary operator is defined as

$$\partial^* \omega^{q-1}(c_q) \equiv \omega^{q-1}(\partial c_q). \quad (2.41)$$

Here  $\omega^{q-1}$  is a  $(q-1)$  cochain, or a homomorphic mapping from a chain to the coefficient group. The homology of the dual chain complex is called the cohomology.

We then define the  $q$ -combinatorial Laplacian operator  $\Delta_q : C^q(K) \rightarrow C^q(K)$

$$\Delta_q := \partial_{q+1} \partial_{q+1}^* + \partial_q^* \partial_q. \quad (2.42)$$

Let  $\mathcal{B}_q$  be the standard basis for the matrix representation of  $q$ -boundary operator from  $C_q(K)$  and  $C_{q-1}(K)$ , and  $\mathcal{B}_q^T$  be the  $q$ -coboundary operator. The matrix representation of the  $q$ -th order Laplacian operator  $\mathcal{L}_q$  is defined as

$$\mathcal{L}_q = \mathcal{B}_{q+1} \mathcal{B}_{q+1}^T + \mathcal{B}_q^T \mathcal{B}_q. \quad (2.43)$$

The multiplicity of zero eigenvalue of  $\mathcal{L}_q$  is the  $q$ -th Betti number of the simplicial complex. The nonzero eigenvalues (non-harmonic spectrum) contains other topological and geometrical features.

As stated before, simplicial complex does not provide sufficient information to understand the geometry of the data. To this end, we utilize simplicial complex induced by filtration

$$\{\emptyset\} = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_p = K, \quad (2.44)$$

where  $p$  is the number of filtration.

For each  $K_t$   $0 \leq t \leq p$ , denote  $C_q(K_t)$  as chain group induced by  $K_t$ , and the corresponding boundary operator  $\partial_q^t : C_q(K_t) \rightarrow C_{q-1}(K_t)$ , resulting in

$$\partial_q^t \sigma_q = \sum_{i=1}^q (-1)^i \sigma_{q-1}^i, \quad (2.45)$$

for  $\sigma_q \in K_t$ . The adjoint operator of  $\partial_q^t$  is similarity defined as  $\partial_q^{t*} : C^{q-1}(K_t) \rightarrow C^q(K_t)$ , which we regard as the mapping  $C_{q-1}(K_t) \rightarrow C_q(K_t)$  via the isomorphism between cochain and chain groups. Through these 2 operators, we can define the chain complexes induced by  $K_t$ .

#### 2.4.7 Persistent Laplacian

Utilizing filtration with simplicial complex, we can define persistence Laplacian spectra. Let  $C_q^{t+p}$  whose boundary is in  $C_{q-1}^t$  be  $\mathbb{C}_q^{t+p}$ , assuming an inclusion mapping  $C_{q-1}^t \rightarrow C_{q-1}^{t+p}$ . On this set, we can define the  $p$ -persistent  $q$ -boundary operator denoted  $\hat{\partial}_q^{t,p} : \mathbb{C}_q^{t,p} \rightarrow C_{q-1}^t$  and the corresponding adjoint operator  $(\hat{\partial}_q^{t,p})^* : C_{q-1}^t \rightarrow \mathbb{C}_q^{t,p}$ . Then, the  $q$ -order  $p$ -persistent Laplacian operator is computed as

$$\Delta_q^{t,p} = \hat{\partial}_{q+1}^{t,p} (\hat{\partial}_{q+1}^{t,p})^* + (\hat{\partial}_q^{t,p})^* \hat{\partial}_q^{t,p}, \quad (2.46)$$

and its matrix representation as

$$\mathcal{L}_q^{t,p} = \mathcal{B}_{q+1}^{t,p} (\mathcal{B}_{q+1}^{t,p})^T + (\mathcal{B}_q^t)^T \mathcal{B}_q^t. \quad (2.47)$$

Likewise as before, the multiplicity of the zero-eigenvalue is the  $q$ -th order  $p$ -persistent Betti number  $\beta_q^{t,p}$ , which is the  $q$ -dimensional hole in  $K_t$  that persists in  $K_{t+p}$ . Moreover, the  $q$ -th order Laplacian is just a particular case of  $\mathcal{L}_q^{t,p}$ , where  $p = 0$ , which is a snapshot of the topology at the filtration step  $t$  [23, 24].

Figure 2.5 show an example of persistent barcode feature, harmonic and nonharmonic spectra of persistent Laplacian of fullerene (C60) molecule. (a), (b) and (c) correspond to the radii  $r = 0.75$ ,  $r = 1.5$  and  $r = 2.5$ , respectively, and was visualized using ChimeraX[19]. (d) shows the persistent barcode of C60 molecule, which was generated using Gudhi. (e) shows the harmonic ( $\beta_k^{r,p}$ ) and the nonharmonic ( $\lambda_k^{r,p}$ ) spectra of persistent Laplacian, which was obtained via HERMES[23]. Vietoris-rips complex was used for both cases, and for the persistent Laplacian,  $p = 0$  was used. As indicated by the blue bars in (d), the connected components ( $H_0$ ) dies at about  $r = 1.5$ , which is consistent with  $\beta_0^{r,0}$ , which is to be expected. Persistent Laplacians captures the Betti information of persistent homology. However, persistent Laplacian also captures the non-harmonic spectra, which

is indicated by the red line. Such feature can capture the homotopic shape of evolution throughout the filtration process.

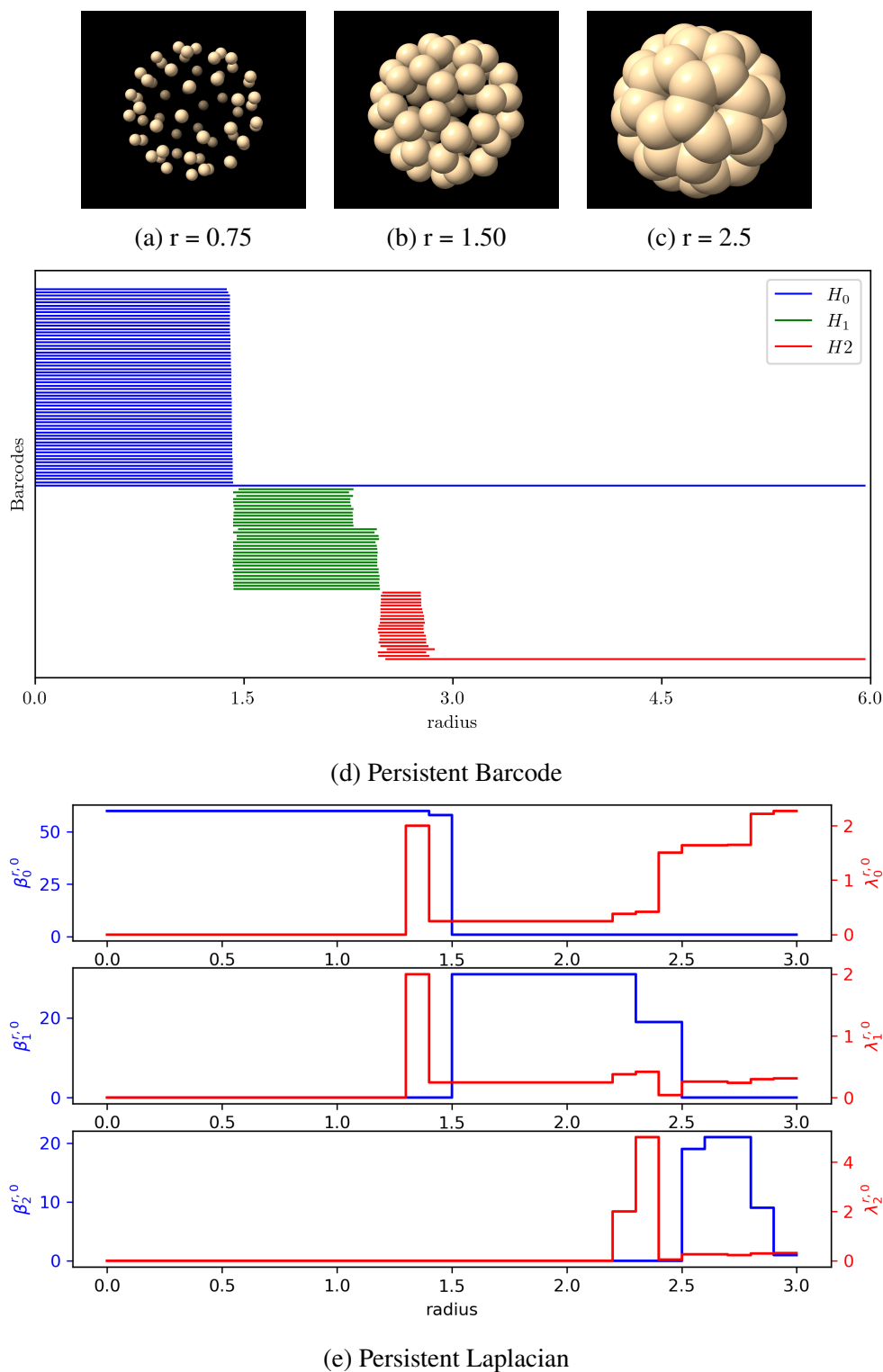


Figure 2.5 Comparison of persistent barcode and persistent Laplacian utilizing fullerene C60 molecule. HERMES[23] package was used to generate the Betti numbers and to non-harmonic spectra using the victoris-rips complex and  $p = 0$ .

## 2.5 Phylogenetic Analysis

### 2.5.1 $k$ -mers method

In DNA sequencing analysis,  $k$ -mers are words or motif of length  $k$ . For example, 1-mers consist of 4 types, adenine (A), cytosine (C), guanine (G) or thymine (T) (in the case of RNA sequencing, uracil (U)). 2-mers consists of 16 combinations, such as AA, GT, and GA. In general, for a given  $k$ , the number of subsequence of length  $k$  is  $4^k$ . In  $k$ -mers based methods, the frequency of each  $k$ -mers are computed. For example, the sequence GTAGAGCTGT contains 2A, 1C, 4G and 3T, which forms the frequency for 1-mers. Since the length of the sequence is 10, we can count up to  $k$ -mers of length 10.

We will now formulate that case for 1-mers. Let  $S = s_1 s_2 \dots s_N$  be a DNA sequence of length  $N$ , where  $s_i \in \{A, C, G, T\}$ . Define the indicator function for nucleotide  $l$

$$\delta_l(s_i) = \begin{cases} 1, & s_i = l \\ 0, & \text{otherwise} \end{cases} \quad (2.48)$$

where  $1 \leq i \leq N$  and  $l \in \{A, C, G, T\}$ . Then, the total number of nucleotide  $l$  is

$$N_l = \sum_{i=1}^N \delta_l(s_i). \quad (2.49)$$

It is easy to see that the length of the sequence, assuming that all nucleotides are valid, satisfies  $N = N_A + N_C + N_G$ .

Now, for the general  $k$ -mers case, we have a total of  $N-k+1$   $k$ -mers given by  $(s_1 \dots s_k)(s_2 \dots s_{k+1}) \dots (s_{N-k+1} \dots s_N)$ . Then, we label the  $4^k$  possible  $k$ -mers as  $l_1, l_2, \dots, l_j, \dots, l_{4^k}$ , and define the  $k$ -mers specific indicator function as

$$\delta_{l_j}(s_i \dots s_{i+k}) = \begin{cases} 1, & (s_i \dots s_{i+k}) = l_j \\ 0, & \text{otherwise} \end{cases} \quad (2.50)$$

Then, the total number of  $k$ -mers  $l_j$  is

$$N_{l_j} = \sum_{i=1}^{N-k+1} \delta_{l_j}(s_i \dots s_{i+k}) \quad (2.51)$$



Because  $k$ -mers has the computational efficiency of  $O(N)$ , it can handle long sequences, including mammalian chromosomes.

### 2.5.2 Natural Vector Method

The Natural Vector Method (NVM) was developed in 2011. It transforms the DNA sequence into a vector by capturing the moments of the position of the nucleotide [25]. This extends the traditional  $k$ -mers method by incorporating positional information of the  $k$ -mers.

Let  $S = s_1 s_2 \dots s_N$  be a DNA sequence of length  $N$ , where  $s_i \in \{A, C, G, T\}$ . Here, we consider uracil  $U$  as thymine  $T$ . Define a indicator function  $\delta_l(s_i)$  as

$$\delta_l(s_i) = \begin{cases} 1, & s_i = l \\ 0 & \text{otherwise} \end{cases}$$

where  $l = A, C, G, T$ . Then, the natural vector for the sequence is defined as

$$S = (N_A, N_C, N_G, N_T, \mu_A, \mu_C, \mu_G, \mu_T, D_2^A, D_2^C, D_2^G, D_2^T, \dots, D_M^A, D_M^C, D_M^G, D_M^T)$$

where

$$\begin{aligned} N_l &= \sum_{i=1}^N \delta_l(s_i) \\ \mu_l &= \sum_{i=1}^N \frac{i}{N_l} \delta_l(s_i) \\ D_m^l &= \sum_{i=1}^N \frac{(i - \mu_l)^m}{N_l^{m-1} N^{m-1}} \delta_l(s_i) \end{aligned} \tag{2.52}$$

Here,  $m = 0, 1, 2, \dots, M$  is the order of the moment,  $N = N_A + N_C + N_G + N_T$ .  $N_l$  and  $\mu_l$  are the 0-order and 1-order, respectively. In essence,  $N_l$  is the total number of nucleotide  $l$ , and  $\mu_l$  is the average position of nucleotide  $l$ . This results in a uniform feature vector of size  $4M$  for each sequence.

The natural vector was further improved via the implementation of  $k$ -mers specific natural vector [26]. Rather than looking at the nucleotide, the moments are calculated for each  $k$ -mers. For the sequence  $S = s_1 s_2 \dots s_N$ , there are a total of  $N - k + 1$   $k$ -mers  $(s_1 \dots s_k)(s_2 \dots s_{k+1}) \dots (s_{N-k+1} \dots s_N)$ .

Similarly, for a given  $k$ , the natural vector is defined as

$$S = (N_{l_1}, \dots, N_{l_{4^k}}, \mu_{l_1}, \dots, \mu_{l_{4^k}}, D_2^{l_1}, \dots, D_M^{l_{4^k}}, \dots, D_M^{l_1}, \dots, D_M^{l_{4^k}})$$

where

$$\begin{aligned} N_{l_j} &= \sum_{i=1}^{N-k+1} \delta_{l_j}(s_i \dots s_{i+k}) \\ \mu_{l_j} &= \sum_{i=1}^{N-k+1} \frac{i}{N_{l_j}} \delta_{l_j}(s_i \dots s_{i+k}) \\ D_m^{l_j} &= \sum_{i=1}^{N-k+1} \frac{(i - \mu_{l_j})^m}{N_{l_j}^{m-1} N^{m-1}} \delta_{l_j}(s_i \dots s_{i+k}) \end{aligned} \quad (2.53)$$

where  $\delta_{l_j}(s_i \dots s_{i+k})$  is an indicator function for the  $k$ -mer  $l_j$ . For a given  $k$ , there is a total of  $4^k$  combination of  $k$ -mers. Moreover, for a fixed  $k$  and  $M$ , the number of features will be  $4^k M$ . Then, combining all the  $k$ -mers together, we have a natural vector of length  $\sum_{k=1}^K 4^k M$

### 2.5.3 Markov $k$ -string model

In this section, we discuss the Markov  $k$ -string model, which is the foundation for all alignment-free probabilistic models. Markov model aims to characterize evolution by evaluating the difference between observed probability of an appearance of a  $k$ -mer and the predicted appearance of a  $k$ -mer [27].

Let  $\alpha_1 \alpha_2 \dots \alpha_k$  be the nucleotide of a  $k$ -mer, and let  $f(\alpha_1 \alpha_2 \dots \alpha_k)$  denote its frequency. We can normalize the frequency by dividing it by the total number of  $k$ -mers, which will give the probability of a particular  $k$ -mers appearing.

$$p(\alpha_1 \alpha_2 \dots \alpha_k) = \frac{f(\alpha_1 \alpha_2 \dots \alpha_k)}{N - k + 1} \quad (2.54)$$

Since the goal is to compare the observed and predicted probability, we compute the probability of  $(k - 1)$ -mers and  $(k - 2)$ -mer derived from the  $k$ -mer. The predicted probability of  $k$ -mers appearing can be computed using a Markov model:

$$p^0(\alpha_1 \alpha_2 \dots \alpha_k) = \frac{p(\alpha_1 \alpha_2 \dots \alpha_{k-1}) p(\alpha_2 \alpha_3 \dots \alpha_k)}{p(\alpha_2 \alpha_3 \dots \alpha_{k-1})} \quad (2.55)$$

where  $p^0$  denote the predicted probability, and such prediction model has been utilized for both DNA and protein sequences [28, 29]

Then, the evolution can be characterized by taking the normalized difference between the observe and predicted probabilities:

$$a(\alpha_1\alpha_2...\alpha_k) = \begin{cases} \frac{p(\alpha_1\alpha_2...\alpha_k)-p^0(\alpha_1\alpha_2...\alpha_k)}{p^0(\alpha_1\alpha_2...\alpha_k)} & p^0 \neq 0 \\ 0 & p^0 = 0 \end{cases} \quad (2.56)$$

Likewise with the original  $k$ -mers and NVM methods, Markov models are also computationally efficient, and has been applied to both proteins and DNA sequences.

## BIBLIOGRAPHY

- [1] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.
- [2] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. *IEEE Transactions on knowledge and data engineering*, 25(6):1336–1353, 2012.
- [3] Daniel Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13, 2000.
- [4] George C Linderman, Manas Rachh, Jeremy G Hoskins, Stefan Steinerberger, and Yuval Kluger. Fast interpolation-based t-sne for improved visualization of single-cell rna-seq data. *Nature methods*, 16(3):243–245, 2019.
- [5] GW Wei. Wavelets generated by using discrete singular convolution kernels. *Journal of Physics A: Mathematical and General*, 33(47):8577, 2000.
- [6] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
- [7] Jiahui Chen, Rundong Zhao, Yiyong Tong, and Guo-Wei Wei. Evolutionary de rham-hodge method. *arXiv preprint arXiv:1912.12388*, 2019.
- [8] J MacQueen. Some methods for classification and analysis of multivariate observations. *Proc. 5th Berkeley Symposium on Math., Stat., and Prob*, page 281, 1965.
- [9] Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, 36(2):3336–3341, 2009.
- [10] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2:193–218, 1985.
- [11] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th annual international conference on machine learning*, pages 1073–1080, 2009.
- [12] David F Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016.
- [13] KVSN Rama Rao and B Manjula Josephine. Exploring the impact of optimal clusters on cluster purity. In *2018 3rd International Conference on Communication and Electronics Systems (ICCES)*, pages 754–757. IEEE, 2018.

- [14] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.
- [15] Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th international conference on pattern recognition*, pages 3121–3124. IEEE, 2010.
- [16] Zixuan Cang and Guo-Wei Wei. Analysis and prediction of protein folding energy changes upon mutation by element specific persistent homology. *Bioinformatics*, 33(22):3549–3557, 2017.
- [17] Zixuan Cang and Guo-Wei Wei. Integration of element specific persistent homology and machine learning for protein-ligand binding affinity prediction. *International journal for numerical methods in biomedical engineering*, 34(2):e2914, 2018.
- [18] David Bramer and Guo-Wei Wei. Atom-specific persistent homology and its application to protein flexibility analysis. *Computational and mathematical biophysics*, 8(1):1–35, 2020.
- [19] Thomas D Goddard, Conrad C Huang, Elaine C Meng, Eric F Pettersen, Gregory S Couch, John H Morris, and Thomas E Ferrin. Ucsf chimeraX: Meeting modern challenges in visualization and analysis. *Protein science*, 27(1):14–25, 2018.
- [20] Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The gudhi library: Simplicial complexes and persistent homology. In *Mathematical Software–ICMS 2014: 4th International Congress, Seoul, South Korea, August 5-9, 2014. Proceedings 4*, pages 167–174. Springer, 2014.
- [21] Beno Eckmann. Harmonische funktionen und randwertaufgaben in einem komplex. *Commentarii Mathematici Helvetici*, 17(1):240–255, 1944.
- [22] Daniel Hernández Serrano, Juan Hernández-Serrano, and Darío Sánchez Gómez. Simplicial degree in complex networks. applications of topological data analysis to network science. *Chaos, Solitons and amp; Fractals*, 137:109839, August 2020.
- [23] Rui Wang, Rundong Zhao, Emily Ribando-Gros, Jiahui Chen, Yiying Tong, and Guo-Wei Wei. Hermes: Persistent spectral graph software. *Foundations of data science (Springfield, Mo.)*, 3(1):67, 2021.
- [24] Rui Wang, Duc Duy Nguyen, and Guo-Wei Wei. Persistent spectral graph. *International journal for numerical methods in biomedical engineering*, 36(9):e3376, 2020.
- [25] Mo Deng, Chenglong Yu, Qian Liang, Rong L He, and Stephen S-T Yau. A novel method of characterizing genetic sequences: genome space with biological distance and applications. *PloS one*, 6(3):e17293, 2011.

- [26] Nan Sun, Shaojun Pei, Lily He, Changchuan Yin, Rong Lucy He, and Stephen S-T Yau. Geometric construction of viral genome space and its applications. *Computational and Structural Biotechnology Journal*, 19:4226–4234, 2021.
- [27] Ji Qi, Bin Wang, and Bai-Iin Hao. Whole proteome prokaryote phylogeny without sequence alignment: Ak-string composition approach. *Journal of molecular evolution*, 58:1–11, 2004.
- [28] Bruce Alberts, Dennis Bray, Julian Lewis, Martin Raff, Keith Roberts, James D Watson, et al. *Molecular biology of the cell*, volume 3. Garland New York, 1994.
- [29] Rui Hu and Bin Wang. Statistically significant strings are related to regulatory elements in the promoter regions of *saccharomyces cerevisiae*. *Physica A: Statistical Mechanics and its Applications*, 290(3-4):464–474, 2001.

## CHAPTER 3

### PHYLOGENETIC ANALYSIS

#### 3.1 UMAP-assisted $k$ -means clustering of large-scale SARS-CoV-2 mutation datasets

##### 3.1.1 Introduction

Beginning in December 2019, coronavirus disease 2019 (COVID-19) caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) has become one of the most deadly global pandemics in history. The COVID-19 infections in the United States (US) and other nations are still spiking. As of January 20, 2021, the World Health Organization (WHO) has reported 93,217,287 confirmed cases of COVID-19 and 2,014,957 confirmed deaths. The virus has spread to Africa, Americas, Eastern Mediterranean, Europe, South-East Asia and Western Pacific [1]. To prevent further damage to our livelihood, we must control its spread through testing, social distancing, tracking the spread, and developing effective vaccines, drugs, diagnostics, and treatments.

SARS-CoV-2 is a positive-sense single-strand RNA virus that belongs to the Nidovirales order, coronaviridae family and betacoronavirus genus [2]. To effectively track the virus, testing patients with suspected exposure to COVID-19 and sequencing the strand via PCR (polymerase chain reaction) are important. From sequencing, we can analyze patterns in mutation and predict transmission pathways. Without understanding such pathways, current efforts to find effective medicines and vaccines could become futile because mutations may change viral genome or lead to resistance. As of January 20, 2021, there are 203,344 available sequences with 26,844 unique single nucleotide polymorphisms (SNPs) with respect to the first SARS-CoV-2 sequence collected in December 2019 [3] according to our mutation tracker [https://users.math.msu.edu/users/weig/SARS-CoV-2\\_Mutation\\_Tracker.html](https://users.math.msu.edu/users/weig/SARS-CoV-2_Mutation_Tracker.html).

A popular method for understanding mutational trends is to perform phylogenetic analysis, where one clusters mutations to find evolution patterns and transmission pathways. Phylogenetic analysis has been done on the Nidovirales family [4, 5, 6, 7, 4, 8] to understand genetic evolutionary pathways, protein level changes [9, 10, 11, 8], large scale variants [10, 9, 12, 13] and global trends

[14, 15, 16]. Commonly used techniques for phylogenetic analysis include tree based methods [17] and  $K$ -means clustering. Both methods belong to unsupervised machine learning techniques, where ground truth is unavailable. These approaches provide valuable information for exploratory research. A main issue with phylogenetic tree analysis is that as the number of samples increase, its computation becomes unpractical, making it unsuitable for large genome datasets. In contrast,  $K$ -means scales well with sample size increase, but does not perform well when the sample size is too small. Jaccard distance is commonly used to compare genome sequences [18] because it offers a phylogenetic or topological difference between samples. However, the tradeoff to the Jaccard distance is that its feature dimension is the same as its number of samples, suggesting that for a large sample size, the number of features is also large. Since  $K$ -means clustering relies on computing the distance between the center of the cluster and each sample, having a large feature space can result in expensive computation, large memory requirement, and poor clustering performance. This become a significant problem as the number of SARS-CoV-2 genome isolates from patients has reached 200,000 at this point. There is a pressing need for efficient clustering methods for SARS-CoV-2 genome sequences.

One technique to address this challenge is to perform dimensional reduction on the  $K$ -means input dataset so that the task becomes manageable. Commonly used dimension reduction algorithms focus on two aspects: 1) the pairwise distance structure of all the data samples and 2) preservation of the local distances over the global distance. Techniques such as principal component analysis (PCA) [19], Sammon mapping [20], and multidimensional scaling (MDS) [21] aim to preserve the pairwise distance structure of the dataset. In contrast, the t-distributed stochastic neighbor embedding (t-SNE) [22, 23], uniform manifold approximation and projection (UMAP) [24, 25], Laplacian eigenmaps [26], and LargeVis [27] focus on the preservation of local distances. Among them, PCA, t-SNE, and UMAP are the most frequently used algorithms in the applications of cell biology, bioinformatics, and visualization [25].

PCA is a popular method used in exploratory studies, aiming to find the directions of the maximum variance in high-dimensional data and projecting them onto a new subspace to obtain



low-dimensional feature spaces while preserving most of the variance. The principal components of the new subspace can be interpreted as the directions of the maximum variance, which makes the new feature axes orthogonal to each other. Although PCA is able to cover the maximum variance among features, it may lose some information if one chooses an inappropriate number of principal components. As a linear algorithm, PCA performs poorly on the features with nonlinear relationship. Therefore, in order to present high-dimensional data on low dimensional and non-linear manifold, some nonlinear dimensional reduction algorithms such as t-SNE and UMAP are employed. T-SNE is a nonlinear method that can preserve the local and global structures of data. There are two main steps in t-SNE. First, it finds a probability distribution of the high dimensional dataset, where similar data points are given higher probability. Second, it finds a similar probability distribution in the lower dimension space, and the difference between the two distributions is minimized. However, t-SNE computes pairwise conditional probabilities for each pair of samples and involves hyperparameters that are not always easy to tune, which makes it computationally complex. UMAP is a novel manifold learning technique that also captures a nonlinear structure, which is competitive with t-SNE for visualization quality and maintains more of the global structure with superior run-time performance [24]. UMAP is built upon the mathematical work of Belkin and Niyogi on Laplacian eigenmaps, aiming to address the importance of uniform data distributions on manifolds via Riemannian geometry and the metric realization of fuzzy simplicial sets by David Spivak [28]. Similar to t-SNE, UMAP can optimize the embedded low-dimensional representation with respect to fuzzy set cross-entropy loss function by using stochastic gradient descent. The embedding is found by finding a low-dimensional projection of the data that closely matches the fuzzy topological structure of the original space. The error between two topological spaces will be minimized by optimizing the spectral layout of data in a low dimensional space.

The objective of this work is to explore efficient computational methods for the SARS-CoV-2 phylogenetic analysis of large volume of SARS-CoV-2 genome sequences. Specifically, we are interested in developing a dimension-reduction assisted clustering method. With the increase in available sequencing data, the SNP dataset of SARS-CoV-2 has run into large-data problem. By

effectively analyzing clusters, we can find evolutionary trends, which will aid in finding effective medicines and vaccines. To this end, we compare the effectiveness and accuracy of PCA, t-SNE and UMAP for dimension reduction in association with the  $K$ -means clustering. To quantitatively evaluate the performance, we recast supervised classification problems with labels into a  $K$ -means clustering problems so that the accuracy of  $K$ -means clustering can be evaluated. As a result, the accuracy and performance of PCA, t-SNE and UMAP-assisted  $K$ -means clustering can be compared. By choosing the different dimensional reduction ratios, we examine the performance of these methods in  $K$ -means settings on four standard datasets. We found that UMAP is the most efficient, robust, reliable, and accurate algorithm. Based on this finding, we applied the UMAP-assisted  $K$ -means technique to large scale SARS-CoV-2 datasets generated from a Jaccard distance representation and a SNP position-based representation to further analyze its effectiveness, both in terms of speed and scalability. Our results are compared with those in the literature [9] to shed new light on SARS-CoV-2 phylogenetics.

### 3.1.2 Methods

#### 3.1.2.1 Sequence and alignment

The SARS-CoV-2 sequences were obtained from GISAID databank ([www.gisaid.com](http://www.gisaid.com)). Only complete genome sequences with collection date, high coverage, and without 'NNNNNN' in the sequences were considered. Each sequence was aligned to the reference sequence [3] using a multiple sequence alignment (MSA) package Clustal Omega [29]. A total of 203,344 complete SARS-CoV-2 sequences are analyzed in this work.

#### 3.1.2.2 SNP position based features

Let  $N$  be the number of SNP profiles with respect to the SARS-CoV-2 reference genome sequence, and let  $M$  be the number of unique mutation sites. Denote  $V_i$  as the position based feature of the  $i$ th SNP profile.

$$V_i = [v_i^1, v_i^2, \dots, v_i^M], \quad i = 1, 2, \dots, N \quad (3.1)$$

is a  $1 \times M$  vector. Here

$$v_i^j = \begin{cases} 1, & \text{mutation site} \\ 0, & \text{otherwise.} \end{cases} \quad (3.2)$$

We compile this into an  $N \times M$  position based feature,

$$S(i, j) = v_i^j \quad (3.3)$$

where each row represents a sample. Note that  $S(i, j)$  is a binary representation of the position and is sparse.

### 3.1.2.3 Jaccard based representation

The Jaccard distance measures the dissimilarity between two sets. It is widely used in the phylogenetic studies of SNP profiles. In this work, we utilize Jaccard distance to compare SNP profiles of SARS-CoV-2 genome isolates.

Let  $A$  and  $B$  be two sets. Consider the Jaccard index between  $A$  and  $B$ , denoted  $J(A, B)$ , as the cardinality of the intersection divided by the cardinality of the union

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}. \quad (3.4)$$

The Jaccard distance between the two sets is defined by subtracting the Jaccard index from 1:

$$d_J(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \quad (3.5)$$

We assume there are  $N$  SNP profiles or genome isolates that have been aligned to the reference SARS-CoV-2 genome. Let  $S_i$ ,  $i = 1, \dots, N$ , be the set with the position of the mutation of the  $i$ th sample. The Jaccard distance between two sets  $S_i$  and  $S_j$  is given by  $d_J(S_i, S_j)$ . Taking the pairwise distance between all the samples, we can construct the Jaccard based representation, resulting in an  $N \times N$  distance matrix  $D$

$$D(i, j) = d_J(S_i, S_j) \quad (3.6)$$

This distance defines a metric over the collections of all finite sets [30].

### 3.1.3 Validation

$K$ -means clustering is one of the unsupervised learning algorithms, suggesting that neither the accuracy nor the root-mean-square error can be calculated to evaluate the performance of the  $K$ -means clustering explicitly. Additionally,  $K$ -means clustering can be problematic for high-dimensional large datasets. Dimension-reduced  $K$ -means clustering is an efficient approach. To evaluate its accuracy and performance, we convert supervised classification problems with known solutions into dimension-reduced  $K$ -means clustering problems. In doing so, we apply the  $K$ -means clustering to the classification dataset by setting the number of clusters equals to the number of the real categories. Next, in each cluster, we will take the data with the dominant label as the test for all samples and then calculate the  $K$ -means clustering accuracy for the whole dataset.

#### 3.1.3.1 Validation data

In this work, we will consider the following classification datasets to test the performance of the clustering methods: Coil 20, Facebook large page-page network, MNIST, and Jaccard distanced-based MNIST. Previous work has been done on datasets using Euclidean and Minkowski distance for lower dimensions[24, 22, 23]. Here, we verify the result with higher reduction ratios, and tested the validity of using Jaccard distance as a metric.

- **Coil 20:** Coil 20 [31] is a dataset with 1440 gray scale images, consisting of 20 different objects, each with 72 orientation. Each image is of size  $128 \times 128$ , which was treated as a 16384 dimensional vector for dimensional reduction
- **Facebook Network:** Facebook large page-page network [32] is a page-page webgraph of verified Facebook sites. Each node represents a facebook page, and the links are the mutual links between sites. This is a binary dataset with 22,470 nodes; hence the sample size and feature size are both 22,470. Jaccard distance was computed between each nodes for the feature space.
- **MNIST:** MNIST [33] is a hand written digit dataset. Each image is a grey scale of size  $28 \times 28$ , which was treated as a 784 dimensional vector for the feature space, each with a

integer value in  $[0, 255]$ . Standard normalization was used before performing dimensional reduction. There are 70,000 sample, with 10 different labels.

- **Jaccard distanced-based MNIST:** The above dataset was converted to a Jaccard distance-based dataset. This is to simulate position based mutational dataset, where 1 indicates a mutation in a particular position. Jaccard distance was used to construct the feature space, hence for each sample, the feature size is 70,000. This dataset can be viewed as an additional validation on our Jaccard distance representation.

### 3.1.3.2 Validation results

In the present work, we implement three popular dimensional reduction methods, PCA, UMAP, and t-SNE, for the dimension reduction and compare their performance in  $K$ -means clustering. For a uniform comparison, we reduce the dimensions of the samples by a set of ratios. The minimum between the number of features and the number of samples was taken as base of the reduction. For the Coil 20 dataset, since the numbers of samples and features were 1440 and 16384, respectively, dimension-reductions were based on 1440. For the Facebook Network, since the numbers of samples and features were both 22,470, dimension-reductions were based on 22,470. For the MNIST dataset, since the numbers of samples and features were respectively 70,000 and 784, dimension-reductions were based on 784. Finally, for the Jaccard distanced-based MNIST dataset, since the numbers of samples and features were both 70,000, dimension-reductions were based on 70,000. Note that for the Jaccard distanced-based MNIST data, more aggressive ratios were used because the original feature size is huge, i.e., 70,000. The standard ratios of 2, 4, and 8, etc do not sufficiently reduce the dimension for effective  $K$ -means computation. For the purpose of visualization, two-dimensional reduction algorithms are applied to each reduction scheme. In order to validate PCA, UMAP, and t-SNE assisted  $K$ -means clustering, we observed their performance using labeled datasets.  $K$ -nearest neighbors ( $K$ -NN) was used to find the baseline of the reduction, which reveals how much information can be preserved in the feature after applying a dimensional reduction algorithm. For  $k$ -NN, 10 fold cross-validation was performed.

Notably,  $K$ -means clustering is an unsupervised learning algorithm, which does not have labels to evaluate the clustering performance explicitly. However, we can assess the  $K$ -means clustering accuracy via labeled datasets that has ground truth. In doing so, we choose the number of  $K$  as the original number of classes. The detail of computing accuracy of  $k$ -means clustering can be found in section 2.3.

### 3.1.3.3 Coil 20

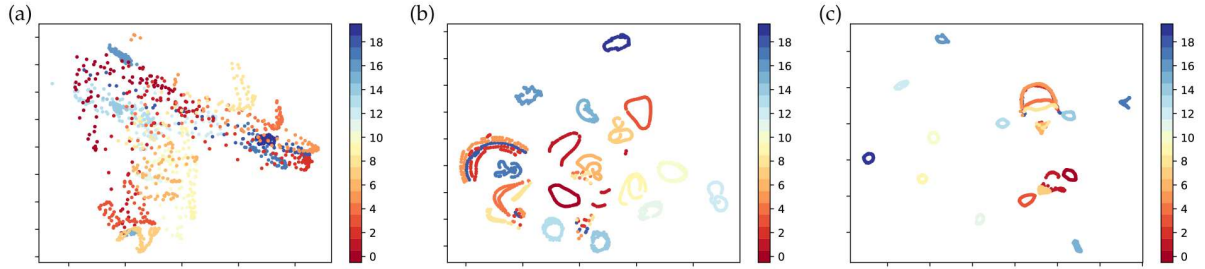


Figure 3.1 Comparison of different dimensional reduction algorithms on Coil 20 dataset. Total 20 different labels are in the Coil 20 dataset, and we use the ground truth label to color each data points. (a) Feature size is reduced to dimension 2 by PCA. (b) Feature size is reduced to dimension 2 by t-SNE. (c) Feature size is reduced to dimension 2 by UMAP.

Figure 3.1 shows the performance of PCA-assisted, UMAP-assisted and t-SNE-assisted clustering of the Coil 20 dataset. For each case, the dataset were reduced to dimension 2 using default parameters, and the plots were colored with the ground truth of the Coil 20 dataset. It can be seen that PCA does not present good clustering, whereas UMAP and t-SNE show very good clusters.

Table 3.1 Accuracy of  $k$ -NN of the Coil 20 dataset without applying any reduction algorithms, as well as the accuracy of  $k$ -NN assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the Coil 20 dataset are 1440, 16384, and 20, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
Coil 20 (1440,16384,20)	0.956	720 (1/2)	0.955	0.668	0.850
		360 (1/4)	0.957	0.861	0.889
		180 (1/8)	0.973	0.867	0.881
		90 (1/16)	0.977	0.860	0.885
		45 (1/32)	0.980	0.861	0.875
		22 (1/64)	0.985	0.868	0.743
		14 (1/100)	0.730	0.851	0.878
		7 (1/200)	0.985	0.870	0.845
		3	0.850	0.863	0.959
		2	0.730	0.853	0.948

Table 3.1 shows the accuracy of  $k$ -NN clustering of the Coil 20 dataset assisted by PCA, t-SNE, and UMAP with different dimensional reduction ratio. The Coil 20 dataset has 1,440 samples, 16,384 features, and 20 different labels. For PCA, the sklearn implementation on python was used with standard parameters. Note that for all methods, dimensions were reduced to 3 and 2 for a comparison. For t-SNE, Multicore-TSNE [34] was used because it offers up to 8 core processor, which is not available in the sklearn implementation, and it is the fastest performing t-SNE algorithm. For UMAP, we used standard parameters [24]. It can be seen that when we reduce the dimension to 3, t-SNE performs best. Moreover, when the dimensional reduction ratio is 1/100, PCA and UMAP also perform well. Notably, the  $k$ -NN accuracy for the data without applying any dimensional reduction algorithm is 0.956, indicating that UMAP does not provide the best clustering performance on the Coil 20 dataset. However, PCA and t-SNE will preserve the information of the original data with a dimensional reduction ratio larger than 1/100, and t-SNE even performs better for dimensional three on the Coil 20 dataset.

Table 3.2 Accuracy of  $K$ -means clustering of the Coil 20 dataset without applying any reduction algorithms, as well as the accuracy of  $K$ -means assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the Coil 20 dataset are 1440, 16384, and 20, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
Coil 20 (1440,16384,20)	0.626	720 (1/2)	0.64	0.301	0.798
		360 (1/4)	0.678	0.800	0.718
		180 (1/8)	0.633	0.822	0.648
		90 (1/16)	0.642	0.799	0.681
		45 (1/32)	0.666	0.800	0.615
		22 (1/64)	0.673	0.819	0.151
		14 (1/100)	0.631	0.817	0.154
		7 (1/200)	0.591	0.819	0.360
		3	0.561	0.800	0.780
		2	0.537	0.801	0.828

Table 3.2 describes the accuracy of  $K$ -means clustering of Coil 20 assisted by PCA, UMAP, and t-SNE with different dimensional reduction ratio. For consistency, we use the same set of standard parameters as  $k$ -NN. For the Coil 20 dataset, the accuracy of  $K$ -means clustering assisted by UMAP has the best performance. When the reduced dimension is 2048 (ratio 1/8), UMAP will result in a relatively high  $K$ -means accuracy (0.822). Moreover, although PCA performs best on  $k$ -NN accuracy, it performs poorly on the  $K$ -means accuracy, indicating that PCA is not a suitable dimensional reduction algorithm on the Coil 20 dataset. Furthermore, the highest accuracy of  $K$ -means clustering is 0.828, which is calculated from the t-SNE-assisted algorithm. However, the t-SNE-assisted accuracy under different reduction ratio changes dramatically. When the ratio is 1/64, the t-SNE-assisted accuracy is only 0.151, indicating that t-SNE is sensitive to the hyper-parameters settings. In contrast, the performance of UMAP is highly stable under all dimension-reduction ratios.

Note that dimension-reduced  $k$ -means clustering methods outperform the original  $k$ -means clustering. Therefore, the proposed dimension-reduced  $k$ -means clustering methods not only improve the  $k$ -means clustering efficiency, but also achieve better accuracy.



### 3.1.3.4 Facebook Network

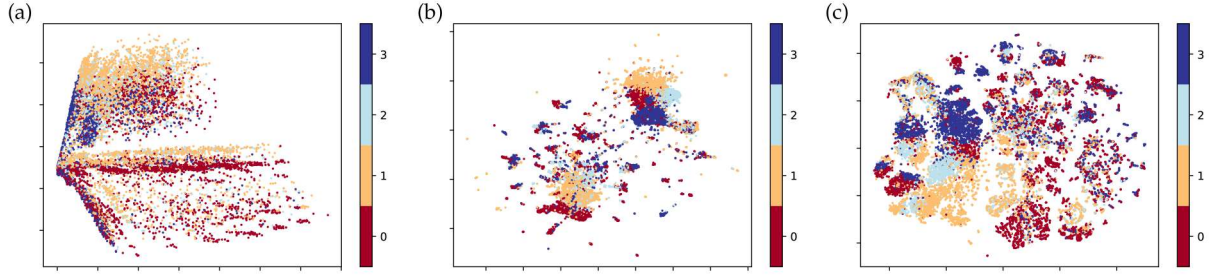


Figure 3.2 Comparison of different dimensional reduction algorithms on the Facebook Network dataset. Total 4 different labels are in the Facebook Network dataset, and we use the ground truth label to color each data points. (a) Feature size is reduced to dimension 2 by PCA. (b) Feature size is reduced to dimension 2 by t-SNE. (c) Feature size is reduced to dimension 2 by UMAP.

Figure 3.2 shows the visualization performance of PCA-assisted, UMAP-assisted, and t-SNE-assisted clustering of the Facebook Network. For each case, the dataset was reduced to dimension 2 using default parameters, and the plots were colored with the ground truth of the Facebook Network. Figure 3.2 shows that the PCA-based data is located distributively, while the t-SNE- and UMAP-based data show clusters.

Table 3.3 shows the accuracy of  $k$ -NN clustering of the Facebook Network assisted by PCA, t-SNE, and UMAP with different dimensional reduction ratio. The Facebook Network dataset has 22,470 samples with 4 different labels, and the feature size of the Facebook Network is also 22,470. For each algorithm, we use the same settings as the Coil 20 dataset. Without applying any dimensional reduction method, The Facebook Network has 0.755  $k$ -NN accuracy. The reduced feature from PCA has the best  $k$ -NN performance when the reduction ratio is 1/2. UMAP has a better performance compared to PCA and t-SNE when the reduction ratio is smaller than 1/16.

Table 3.3 Accuracy of  $k$ -NN of the Facebook Network without applying any reduction algorithms, as well as the accuracy of  $k$ -NN assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the Facebook Network are 22470, 22470, and 4, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
Facebook Network (22470, 22470, 4)	0.755	11235 (1/2)	0.756	0.360	0.307
		5617 (1/4)	0.755	0.669	0.316
		2808 (1/8)	0.754	0.754	0.355
		1404 (1/16)	0.751	0.816	0.707
		702 (1/32)	0.751	0.814	0.669
		351 (1/64)	0.746	0.815	0.690
		224 (1/100)	0.733	0.814	0.676
		112 (1/200)	0.721	0.819	0.633
		44 (1/500)	0.714	0.816	0.709
		22 (1/1000)	0.690	0.815	0.643
		3	0.552	0.801	0.741
		2	0.501	0.786	0.732

Table 3.4 describes the accuracy of  $K$ -means clustering of the Facebook Network assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. PCA, UMAP, and t-SNE all have very poor performance, which may be caused by the smaller number of labels. The highest accuracy 0.427 is observed in the t-SNE-assistant algorithm with dimension 2.

Table 3.4 Accuracy of  $K$ -means clustering of the Facebook Network without applying any reduction algorithms, as well as the accuracy of  $K$ -means assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the Facebook Network are 22470, 22470, and 4, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
Facebook Network (22470, 22470, 4)	0.374	11235 (1/2)	0.331	0.306	0.306
		5617 (1/4)	0.331	0.307	0.299
		2808 (1/8)	0.331	0.411	0.314
		1404 (1/16)	0.331	0.397	0.313
		702 (1/32)	0.331	0.401	0.306
		351 (1/64)	0.331	0.400	0.308
		224 (1/100)	0.331	0.400	0.327
		112 (1/200)	0.331	0.400	0.306
		44 (1/500)	0.331	0.400	0.313
		22 (1/1000)	0.331	0.401	0.306
		3	0.332	0.351	0.344
		2	0.358	0.345	0.427

Similar to the last case, UMAP-based and t-SNE-based dimension-reduced  $k$ -means clustering methods outperform the original  $k$ -means clustering with the full feature dimension. Therefore, it is useful to carry out dimension reduction before  $k$ -means clustering for large datasets.

### 3.1.3.5 MNIST

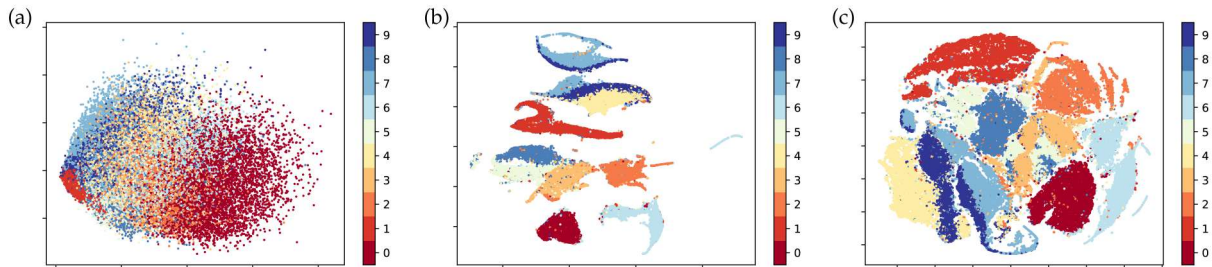


Figure 3.3 Comparison of different dimensional reduction algorithms on the MNIST dataset. Total 10 different labels are in the MNIST dataset, and we use the ground truth label to color each data points. (a) Feature size is reduced to dimension 2 by PCA. (b) Feature size is reduced to dimension 2 by t-SNE. (c) Feature size is reduced to dimension 2 by UMAP.

Table 3.5 Accuracy of  $k$ -NN of the MNIST dataset without applying any reduction algorithms, as well as the accuracy of  $k$ -NN assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the MNIST dataset are 70000, 784, and 10, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
MNIST (70000, 784, 10)	0.948	392 (1/2)	0.951	0.937	0.696
		196 (1/4)	0.956	0.938	0.846
		98 (1/8)	0.960	0.937	0.893
		49 (1/16)	0.961	0.937	0.886
		24 (1/32)	0.953	0.937	0.842
		12 (1/64)	0.926	0.937	0.676
		7 (1/100)	0.846	0.936	0.940
		3	0.513	0.929	0.938
		2	0.323	0.919	0.928

Figure 3.3 shows the performance of PCA-assisted, UMAP-assisted and t-SNE-assisted clustering of the MNIST dataset. The sample size of the MNIST dataset is 70000, which has 784 features with 10 different digit labels. For each case, the dataset was reduced to dimension 2 using default parameters, and the plots were colored with the ground truth of the MNIST dataset. In Figure 3.3, by applying the UMAP algorithm, the clear clusters can be detected for the MNIST dataset. The t-SNE offers a reasonable clustering at dimension 2 too. However, the PCA does not provide a good clustering.

Table 3.6 Accuracy of  $K$ -means clustering of the MNIST dataset without applying any reduction algorithms, as well as the accuracy of  $K$ -means assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the MNIST dataset are 70000, 784, and 10, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
MNIST (70000, 784, 10)	0.494	392 (1/2)	0.487	0.665	0.122
		196 (1/4)	0.492	0.667	0.113
		98 (1/8)	0.498	0.673	0.113
		49 (1/16)	0.496	0.718	0.113
		24 (1/32)	0.501	0.697	0.114
		12 (1/64)	0.489	0.682	0.138
		7 (1/100)	0.464	0.677	0.740
		3	0.365	0.727	0.537
		2	0.300	0.712	0.593

Table 3.5 shows the accuracy of  $k$ -NN clustering of the MNIST dataset assisted by PCA, t-SNE, and UMAP with different dimensional reduction ratios. For each algorithm, we use the same settings as the Coil 20 dataset. Without applying any dimensional reduction algorithms, the accuracy of  $k$ -NN is 0.948. By applying PCA/UMAP with the reduction ratio greater than 1/64, the accuracy of PCA/UMAP-assisted  $k$ -NN is at the same level without using any dimensional reduction algorithm. However, in contract with UMAP and t-SNE, when the reduced dimension is 2 or 3, PCA performs poorly. This indicates that the PCA may not be suitable for dimension-reduction for datasets with a large sample size.

Table 3.6 describes the accuracy of  $K$ -means clustering of the MNIST dataset assisted by PCA, UMAP, and t-SNE with different dimensional reduction ratios. By applying PCA, the accuracy of  $K$ -means is around 0.45. The t-SNE method performance is quite unstable, from very poor (0.113) to the best (0.740), and to a relatively low value of 0.593. In contrast, we can see a stable and improved accuracy from using UMAP at various reduction ratios, indicating that the reduced feature generated by UMAP can better represent the clustering properties of the MNIST dataset compared to the PCA and t-SNE.

As observed early, the present UMAP and t-SNE-assisted  $k$ -means clustering methods also significantly out-perform the original  $k$ -means clustering for this dataset.

### 3.1.3.6 Jaccard distanced-based MNIST

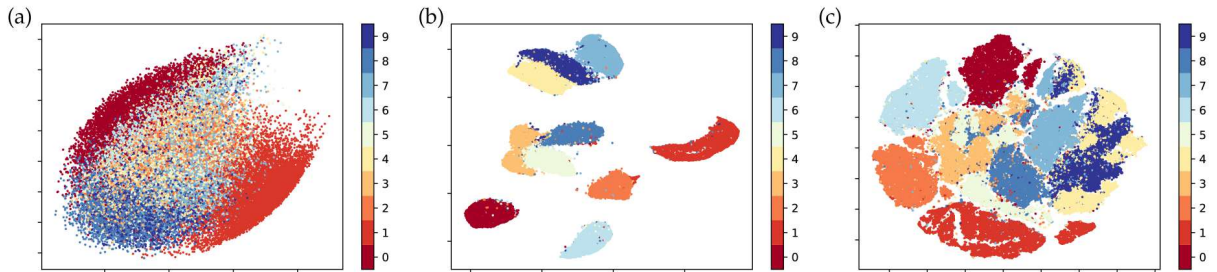


Figure 3.4 Comparison of different dimensional reduction algorithms on the Jaccard distanced-based MNIST dataset. Total 10 different labels are in the Jaccard distanced-based MNIST dataset, and we use the ground truth label to color each data points. (a) Feature size is reduced to dimension 2 by PCA. (b) Feature size is reduced to dimension 2 by t-SNE. (c) Feature size is reduced to dimension 2 by UMAP.

Our last validation dataset is Jaccard distanced-based MNIST. This dataset can be treated as a test on the Jaccard distance-based data representation. Figure 3.4 shows the performance of PCA-assisted, UMAP-assisted, and t-SNE-assisted clustering of the Jaccard distanced-based MNIST dataset. The dataset was reduced to dimension 2 using default parameters for visualization, and the plots were colored with the ground truth of the Jaccard distanced-based MNIST dataset. From Figure 3.4, we can see that UMAP provides the clearest clusters compared to PCA and t-SNE when the dimension is reduced to 2. The performance of t-SNE is reasonable while PCA does not give a good clustering.

Table 3.7 Accuracy of  $k$ -NN of the Jaccard distanced-based MNIST dataset without applying any reduction algorithms, as well as the accuracy of  $k$ -NN assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the Jaccard distanced-based MNIST dataset are 70000, 70000, and 10, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
Jaccard distanced based MNIST (70000, 70000, 10)	0.958	7000 (1/10)	0.958	0.958	0.588
		3500 (1/20)	0.958	0.966	0.601
		1750 (1/40)	0.958	0.967	0.725
		875 (1/80)	0.958	0.967	0.613
		437 (1/160)	0.958	0.968	0.718
		218 (1/320)	0.958	0.968	0.701
		109 (1/640)	0.958	0.968	0.873
		70 (1/1000)	0.958	0.968	0.915
		35 (1/2000)	0.956	0.968	0.872
		17 (1/5000)	0.938	0.968	0.916
		7 (1/10000)	0.867	0.967	0.942
		3	0.487	0.965	0.939
		2	0.313	0.960	0.924

Table 3.8 Accuracy of  $K$ -means clustering of the Jaccard distanced-based MNIST dataset without applying any reduction algorithms, as well as the accuracy of  $K$ -means assisted by PCA, UMAP and t-SNE with different dimensional reduction ratio. The sample size, feature size, and the number of labels of the Jaccard distanced-based MNIST dataset are 70000, 70000, and 10, respectively.

Dataset	$k$ -NN accuracy w/o reduction	Reduced dimension	PCA accuracy	UMAP accuracy	t-SNE accuracy
Jaccard distanced based MNIST (70000, 70000, 10)	0.555	7000 (1/10)	0.436	0.329	0.119
		3500 (1/20)	0.436	0.693	0.120
		1750 (1/40)	0.436	0.792	0.112
		875 (1/80)	0.435	0.793	0.112
		437 (1/160)	0.435	0.793	0.114
		218 (1/320)	0.435	0.793	0.156
		109 (1/640)	0.435	0.794	0.114
		70 (1/1000)	0.436	0.793	0.113
		35 (1/2000)	0.435	0.794	0.116
		17 (1/5000)	0.436	0.793	0.113
		7 (1/10000)	0.431	0.793	0.737
		3	0.364	0.798	0.635
		2	0.261	0.791	0.635

Table 3.7 shows the accuracy of  $k$ -NN clustering of Jaccard distanced-based MNIST assisted by PCA, t-SNE, and UMAP with different dimensional reduction ratios. For each algorithm, we use the same settings as the Coil 20 dataset. Notably, the  $k$ -NN accuracy for the data without applying any dimensional reduction algorithm is 0.958, which is at the same level as the PCA algorithm with a reduction ratio greater than 1/5000. Moreover, we can find that UMAP performs well compared to PCA and t-SNE, indicating that after applying UMAP, the reduced feature still preserves most of the valued information of the Jaccard distanced-based MNIST dataset. The stability and persistence of UMAP at various reduction ratios are the most important features.

Table 3.8 describes the accuracy of  $K$ -means clustering of the Jaccard distanced-based MNIST dataset assisted by PCA, UMAP, and t-SNE with different dimensional reduction ratio. For consistency, we will use the same standard parameters as  $k$ -NN. Similar to the MNIST dataset, the accuracy of  $K$ -means clustering assisted by UMAP still has the best performance. When the reduced dimension is 3, UMAP will result in the highest  $K$ -means accuracy 0.798. Noticeably, although PCA performs well on  $k$ -NN accuracy, it has the lowest  $K$ -mean accuracy, indicating that

PCA is not a suitable dimensional reduction algorithm, especially for those datasets with a large number of samples. To be noted, the t-SNE accuracy at four reduced dimensions are not available due to the extremely long running time.

In a nutshell, PCA, UMAP, and t-SNE can all perform well for  $k$ -NN. However, for the Coil 20 dataset, UMAP performs slightly poorly, whereas the t-SNE performs well, which may be caused by a lack of data size. In order to train UMAP, it needs a suitable data size. The Coil 20 dataset has 20 labels, each with only 72 samples. This may not be enough to train UMAP properly. However, even in this case, UMAP performance is still very stable at various reduction ratios and is the best method in terms of reliability, which become the major advantages of UMAP. Another strength of UMAP comes from its dimension-reduction for  $K$ -means clustering. In most cases, UMAP can improve  $K$ -means clustering accuracy, especially for the Jaccard distanced-based MNIST dataset. Furthermore, UMAP can generate a very clear and elegant visualization of clusters with low dimensional reduction value such as 2. Additionally, UMAP performed better than PCA and t-SNE for a larger dataset (MNIST and Jaccard distanced-based MNIST). Especially for the Jaccard distanced-based MNIST data, where Jaccard distance was used as the metric, UMAP performed best, which indicates the merit of using UMAP for Jaccard distanced-based datasets, such as COVID-19 SNP datasets. Furthermore, the accuracies for  $k$ -NN classification and  $K$ -means clustering are both improved on the Jaccard distance-based MNIST dataset compared to the original MNIST dataset, which provides convincing evidence that the Jaccard distance representation will help improve the performance of the clustering on the SARS-CoV-2 mutation dataset in the following sections.

### **3.1.3.7 Efficiency comparison**

It is important to understand the computational time behaviors of various methods. To this end, we compare computational time for three dimension-reduction techniques. Figure 3.5 depicts the computational time of three methods for the four datasets under various reduction ratios. The green, orange, and blue lines represent the computational time of t-SNE, UMAP, and PCA, respectively. Some points in green line of Figure 3.5 (d) are not available, which due to the extremely long



running time. PCA performed best in most cases, except for the Coil 20 dataset, where UMAP had comparable computational time. This behavior is expected because PCA is a linear transformation, and its time should scale linearly with the number of components in the lower dimensional space. UMAP and t-SNE were slower than PCA, but it is evident from MNIST and Jaccard distanced-based MNIST datasets that UMAP scales better with the increase in the number of samples. Note that for Jaccard distanced-based MNIST, a higher dimension was not computed because the computational time was too long. For Facebook Network, UMAP is outperforming t-SNE; however, for higher dimensions, t-SNE computed faster. Nonetheless, from our baseline test Table 3.3, t-SNE does not perform well, indicating instability. Faster computation time may indicate too fast of a convergence, which leads to poor embedding.

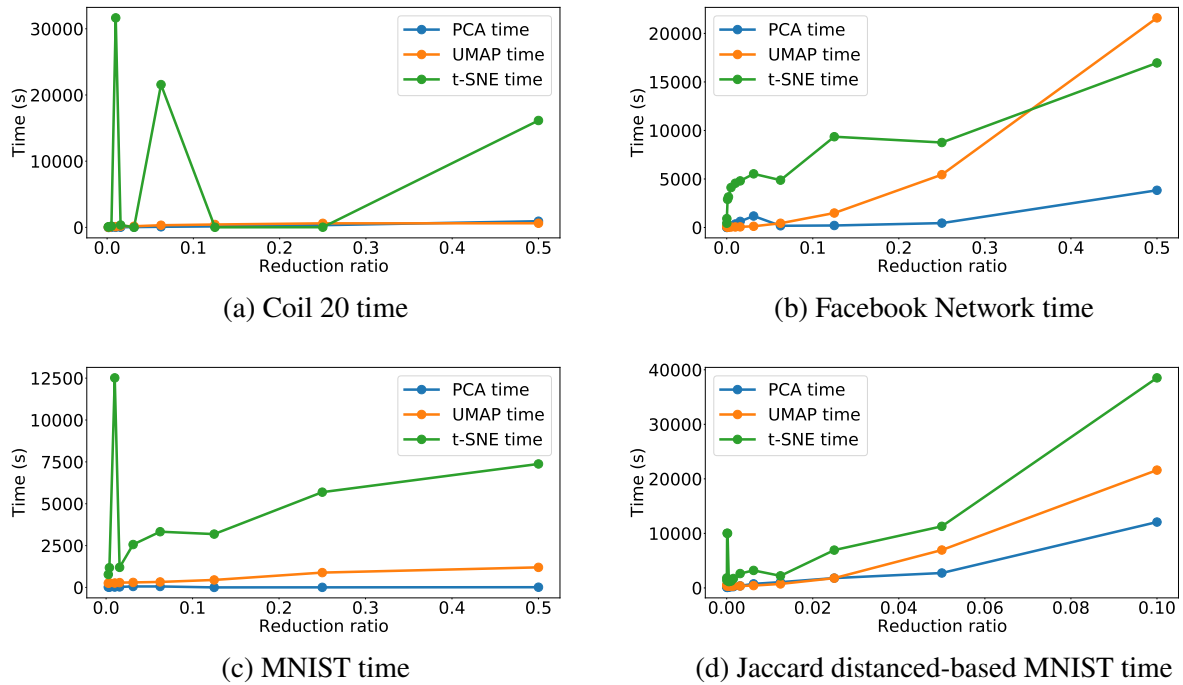


Figure 3.5 Computational time of each reduction ratio. The green, orange and blue lines represent the computational time of t-SNE, UMAP, and PCA, respectively. Not surprisingly, PCA performs the best in the majority of cases, except for the Coil 20 dataset. UMAP and t-SNE perform worse than PCA, but UMAP scales better when there are more samples, as evident from MNIST and Jaccard distanced-based MNIST datasets. Note that for Jaccard distanced-based MNIST, the higher dimension was not computed because the computational time was too long.

### 3.1.4 SARS-CoV-2 mutation clustering

#### 3.1.4.1 World SARS-CoV-2 mutation clustering

We gather data submitted to GISAID up to January 20, 2021, and the total number of samples is 203,344. We first get the SNP information by applying the multiple sequence alignment, which leads to 26,844 unique SNPs. Next, we calculate the pairwise Jaccard distance of our dataset in order to generate the Jaccard distance-based features. Here, the number of rows is the number of samples (203,344), and the number of columns is the feature size (203,344). As we mentioned in Section 3.1.2.3, the Jaccard distance-based feature is a square matrix. However, due to the large size of samples and features, applying *K*-means clustering directly on the feature of the size of  $203,344 \times 203,344$  is a very time-consuming process. Considering that UMAP outperforms the other two dimensional reduction algorithms (PCA and t-SNE) on the Jaccard distance-based MNIST dataset, we employ UMAP to reduce our original feature with the size of  $203,344 \times 203,344$  to  $203,344 \times 203$ . To be noted, UMAP is a reliable and stable algorithm, which performs consistently in clustering at various reduction ratios. Therefore, there is no need to use the same reduction dimension of 203 and one can also choose a different reduction dimension value to generate similar results.

With the reduced dimension feature that has the size of  $203,344 \times 203$ , we split our SARS-CoV-2 dataset into different clusters by applying the *K*-means clustering methods. After comparing the WCSS under a different number of clusters, we find that there are 6 clusters forming within the SARS-CoV-2 population based on the elbow method, which can be determined from Figure 3A.1 in the Supporting Information. Table 3A.1 in the Supporting Information shows the top 25 single mutations of each cluster. In order to understand the relationship, we also analyzed the co-mutation occurring in each cluster (Table 3.9). Here, we define a co-mutation as mutations that occur simultaneously in one SNP profile. For example, mutations occurring at position 241 and 3037 in a single SNP sample is a co-mutation [241, 3037]. From Table 3A.1 and Table 3.9 we see the following:

Table 3.9 The frequency and occurrence percentage of SARS-CoV-2 co-mutations from each clusters in the world.

Cluster	Co-mutations	Frequency	Percentage
Cluster 1	[241, 3037, 14408, 23403, 28881, 28882, 28883]	21802	0.926
Cluster 2	[241, 1059, 3037, 14408, 23403, 25563]	15008	0.660
Cluster 3	[241, 1163, 3037, 7540, 14408, 16647, 18555, 22992, 23401, 23403, 28881, 28882, 28883]	2089	0.606
Cluster 4	[241, 3037, 14408, 23403]	13387	0.936
Cluster 5	[241, 3037, 14408, 23403]	124290	0.915
Cluster 6	[241, 3037, 4543, 5629, 9526, 11497, 13993, 14408, 15766, 16889, 17019, 18877, 22992, 23403, 25563, 25710, 26735, 26876, 28975, 29399]	3279	0.940

- Though Clusters 1 and 6 seem similar from the top 25 single mutations, the co-mutations tells a different story.
- Clusters 2 and 5 have high frequency of [241, 3037, 14408, 23403] mutations, but Cluster 5 has a clear co-mutation descendant with high frequency.
- Cluster 3 has a unique combination of mutation that is only popular in Cluster 3.
- Cluster 6 have high frequency of multiple co-mutations. Since it shares similarity with Clusters 4 and 5, it may be that Cluster 6 branched from Clusters 4 and 5.
- Cluster 6 has many co-mutations when compared to other clusters. As seen in Table 3A.2, the majority of the cases is found in Europe, including the United Kingdom (UK), Denmark (DK), Netherlands (NL), Switzerland (CH) and Luxemborg (LU).

Table 3A.2 in the Supporting Information shows the cluster distributions of samples from 25 countries. Here, we use the ISO 3166-1 alpha-2 codes as the country code. The listed countries are the United Kingdom (UK), the United States (US), Australia (AU), India (IN), Switzerland (CH), Netherlands (NL), Canada (CA), France (FR), Belgium (BE), Singapore (SG), Spain (ES), Russia (RU), Portugal (PT), Denmark (DK), Sweden (SE), Austria (AT), Japan (JP), South Africa (ZA), Iceland (IS), Brazil (BR), Saudi Arabia (SA), Norway (NO), China (CN), Italy (IT), and Korea (KR). We can visualize the clusters on the world map from Figure 3.7, which was visualized using

Highcharts. The underlying color indicates the dominant cluster for each country. Furthermore, from Table 3A.2, we can see the following:

- SNP profiles from UK and DK are dominated in Clusters 5.
- Clusters 3's SNP profiles are predominantly found in AU. This may indicate that SARS-CoV-2 are mutating differently in AU.
- SNP profiles from the US are found mostly in Clusters 2 and 5.
- Most country's SNP profiles are found in Clusters 1,2,4,5 and 6, with some having slightly higher numbers.

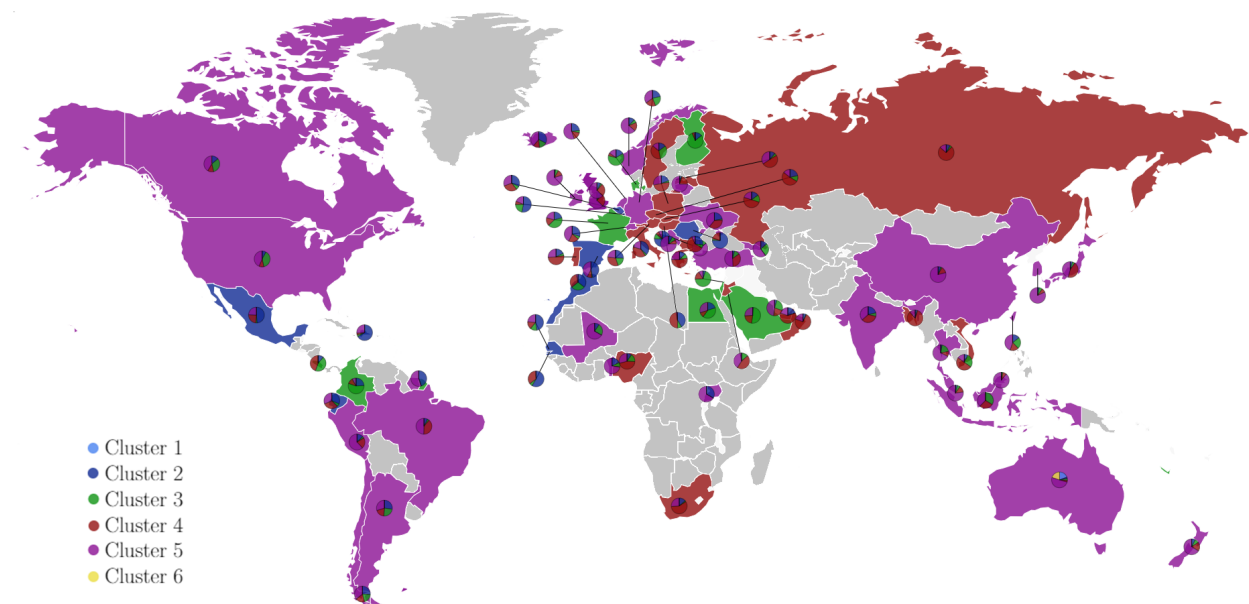


Figure 3.6 Cluster distribution of the global SARS-CoV-2 mutation dataset. Using Highchart, the world map was colored, according to the dominant cluster. For example, United States have SNP profiles from all clusters, but Cluster 5 (purple) is the dominant type in the US. Only countries with more than 25 sequenced data available on GISAID were considered. Countries with fewer than 25 samples are labeled grayed.

Notably, in Table 3.9, Cluster 4 and 5 have the same co-mutations with relatively high frequencies, which indicates the Clusters 4 and 5 share the same "root". Clusters 1, 2, 3, and 6 shares the co-mutation as Clusters 4 and 5, indicating that Clusters 1, 2,3, and 6 may have branched from

Cluster 4 and 5 in the 203-dimensional (203D) space. However, we cannot visualize the distribution of our reduced dataset in the 203D space. Therefore, benefit from the stable and reliable performance of UMAP at various reduction ratios, we reduce the dimension of our original dataset to 2, which enables us to observe the distribution of the dataset in the two-dimensional (2D) space. Figure 3.7 visualizes the distribution of our dataset with 6 distinct clusters with 2D UMAP. It can be seen that Clusters 2, 3 and 4 share a same “root” in the middle. Clusters 3 and 6 are farther away from the center, indicating that they are a descendants of the middle root. In addition, we looked specifically at the spike (S) protein because of its significance in viral infectivity. In all the clusters, 23403A>G (D614G) is present. Studies have shown that D614G increases the infectivity of SARS-CoV-2 [35], hence the high frequency in our data reflect such infectivity. In Clusters 1, 2 and 4, there are no significant co-mutations in the S protein. In Cluster 3, 100% of the variants contain the co-mutation [22992, 23401, 23403], which further supports its geographical isolation, where it is predominantly found in AU. Cluster 5 does not have a significant co-mutation, but the co-mutations [21614, 22227, 23403, 24334] occurred in 11290 SNP profiles (0.083). Cluster 6 has a pair of co-mutations [22992, 23403], which occurs in 99.7% of samples.

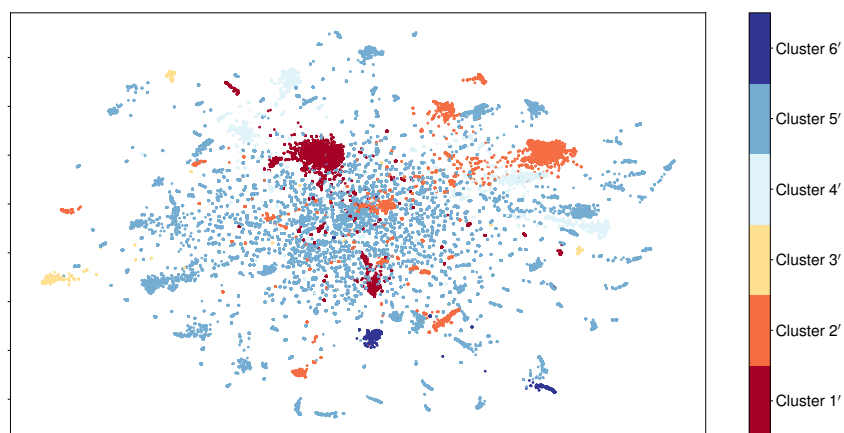


Figure 3.7 2D UMAP visualization of the world SARS-CoV-2 mutation dataset with 6 distinct clusters.

### 3.1.4.2 United States SARS-CoV-2 mutation clustering

In addition to analyzing the clustering in the world, SNP profiles of SARS-CoV-2 from the US were considered. In this section, the US dataset has 17164 unique single mutations and 43395 samples. Therefore, the dimension of the Jaccard distance-based dataset is  $43395 \times 43395$ . After applying the UMAP, we reduce the dimension of the original dataset to be  $43395 \times 216$ . Following the similar *K*-means clustering processes as we did for the world dataset, we find that using the elbow method, we can see from Figure 3A.2 that there are 6 predominant clusters forming in the United States. Figure 3.8 show the US map with the cluster statistic. Here, Highchart was used to generate the plot with the pie chart. Each states were colored based on the dominant cluster.

Table 3A.3 in the Supporting Information shows the top 25 mutations from each clusters in the United States. The cluster distribution of each states is listed in Table 3A.4. Table 3.10 shows the common occurring co-mutations, and we can observe the following:

- Cluster A has a high frequency of co-mutations [241, 1059, 3037, 14408, 23403, 25563], which is a descendant of common co-mutations of Cluster 2 [241, 1059, 3037, 14408, 23403, 25563] from Table 3A.3.
- Cluster B has a high frequency of co-mutations [241, 3037, 14408, 23403], which is a descendant of common co-mutations of Cluster 4 and 5 [241, 3037, 14408, 23403].
- Cluster C have high frequency of co-mutations [241, 3037, 14408, 23403, 28881, 28882, 28883], which is a descendant of common co-mutations of Cluster 1 [241, 3037, 14408, 23403, 28881, 28882, 28883] from Table 3.10.
- Clusters D has high frequency of co-mutations [241, 3037, 14408, 20268, 23403, 28854], which is descendant of Clusters 4 and 5 [241, 3037, 14408, 23403]. US accounts for more than one third of mutations at site 23403 and half of mutations at site 28854
- Cluster E and F have a high frequency of co-mutations [8782, 17747, 17858, 18060, 28144] and [241, 1059, 3037, 11916, 14408, 18998, 23403, 25563, 29540], respectively, which are

descendants of Cluster 4 and 5 [241, 3037, 14408, 23403].

- Cluster F has a high frequency of co-mutations [241, 1059, 3037, 11916, 14408, 18998, 23403, 25563, 29540], which is a descendant of Cluster 2's co-mutation [241, 1059, 3037, 14408, 23403, 25563]

Table 3.10 The frequency and occurrence percentage of SARS-CoV-2 co-mutations from each clusters in US clusters.

Cluster	Co-mutations	Frequency	Percentage
Cluster A	[241, 1059, 3037, 14408, 23403, 25563]	6646	0.702
Cluster B	[241, 3037, 14408, 23403]	20442	0.932
Cluster C	[241, 3037, 14408, 23403, 28881, 28882, 28883]	4429	0.945
Cluster D	[241, 3037, 14408, 20268, 23403, 28854]	3276	0.643
Cluster E	[8782, 17747, 17858, 18060, 28144]	1183	0.744
Cluster F	[241, 1059, 3037, 11916, 14408, 18998, 23403, 25563, 29540]	501	0.789

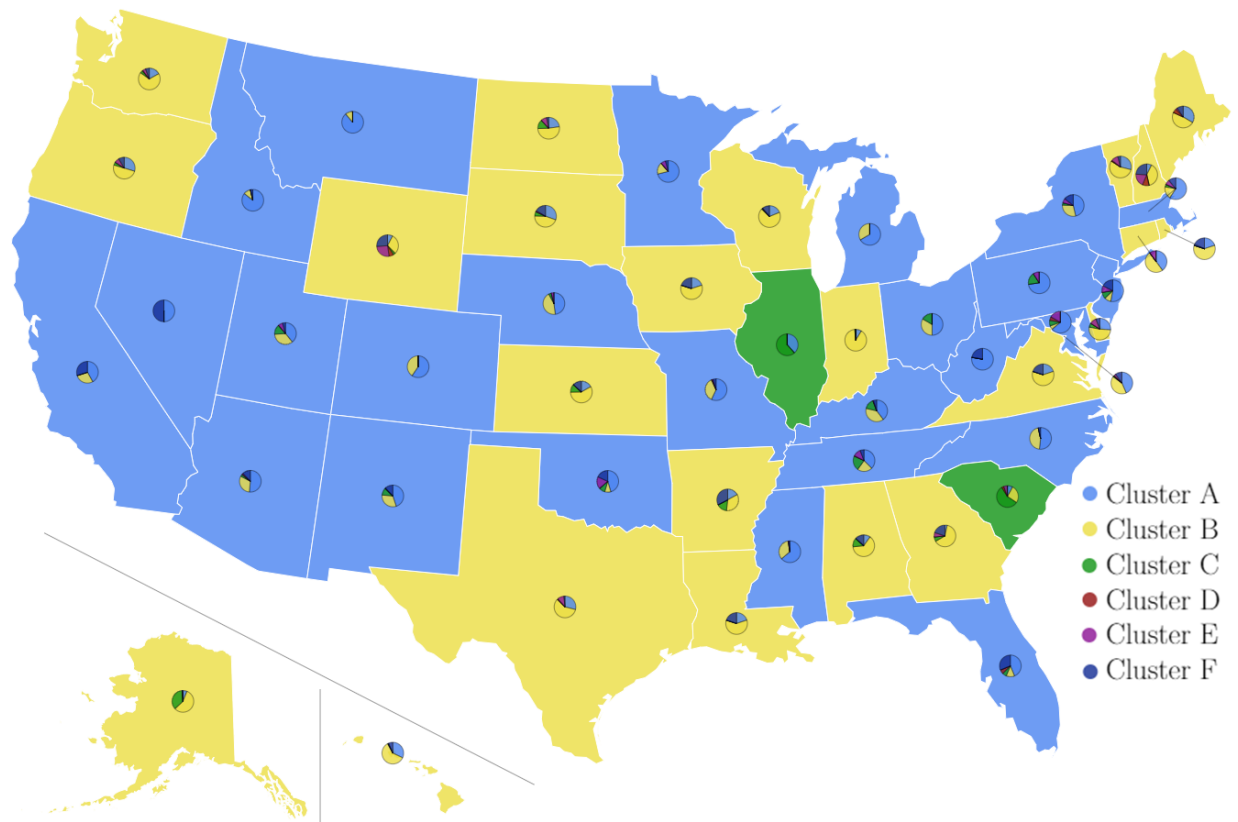


Figure 3.8 Cluster distribution of United States SARS-CoV-2 mutation dataset. Using Highchart, the US map was colored, according to the dominant cluster. For example, United States have SNP profiles from all clusters, but Cluster E (purple) is the dominant type in the US. Only those countries that have more than 25 sequenced data available on GISAID were considered in the plot.

Notably, in Table 3.10, Cluster B has a co-mutation that is present in Clusters A, C, D and F, indicating that Clusters A, C, D and E are descendants of Cluster B. Interestingly, Cluster E has a completely different set of co-mutations as the other clusters, indicating that they are a different strands of mutation. Considering the stability and reliability of UMAP at various reduction ratios, we employ UMAP to the original US dataset with reduced dimension 2, aiming to observe the distribution of the dataset in the 2D space. Figure 3.9 illustrates the 2D visualization of the US dataset with 6 distinct clusters. We can see that there are 3 clusters (Clusters A', B', and C') share the same "root" located in the middle of the figure, while the other 3 clusters (Clusters D', E', and F') are not. Cluster E' is quite distinct from other clusters. This confirms our deduction about why Clusters E' has a high frequency of different co-mutations in Table 3.10. In addition Cluster D' is located close to Cluster A', which may indicate that they have similar root that diverted.



In addition, we looked at co-mutations on the S protein. Every cluster, except for Cluster E, contains mutation 23403, which is expected due to its ability to increase the infectivity of SARS-CoV-2. Clusters A, C and F does not have any significant co-mutation occurring in the S protein, aside from 23403. Cluster E does not have a significant co-mutation nor a significant mutation in the S protein. Cluster B has co-mutations [22255, 23403], which occur in 780 samples. Cluster D has co-mutations [23403, 23604, 24076] that occur in 892 samples.

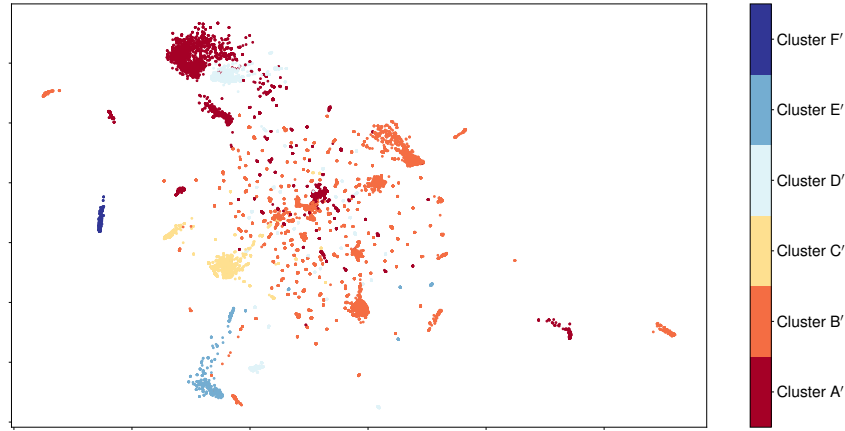


Figure 3.9 The 2D UMAP visualization of the US SARS-CoV-2 mutation dataset with 6 distinct clusters.

### 3.1.5 Discussion

In this section, we compared our past results [9] with our new method to gain a different perspective in clustering with the SNP profiles of COVID-19. In our previous work, a total of 8,309 unique single mutations are detected in 15,140 SARS-CoV-2 isolates. Here, we also calculate the pairwise distance among 15140 SNP profiles and set the number of clusters to be six. Table 3A.5 shows the cluster distribution of samples from the 15 countries [9]. The listed countries are the United States (US), Canada (CA), Australia (AU), United Kingdom (UK), Germany (DE), France (FR), Italy (IT), Russia (RU), China (CN), Japan (JP), Korean (KR), India (IN), Spain (ES), Saudi Arabia (SA), and Turkey (TR), and we use Cluster I, II, III, IV, V, and VI to represent six clusters without applying any dimensional reduction algorithm. Table 3A.6 lists the cluster distribution

of samples from the same 15 countries, where we use  $I_p$ ,  $II_p$ ,  $III_p$ ,  $IV_p$ ,  $V_p$ , and  $VI_p$  to represent six clusters performed by PCA with the reduction ratio to be 1/160. Table 3A.7 lists the cluster distribution of samples from the same 15 countries, where we use  $I_u$ ,  $II_u$ ,  $III_u$ ,  $IV_u$ ,  $V_u$ , and  $VI_u$  to represent six clusters performed by UMAP with the reduction ratio setting to be 1/160. Noticeably, the SNP profile is focused in Cluster  $I_u$ , whereas in the non-reduced version, the samples are more spread out. This may be caused by the large number of features, making computed distance between the centroid and each data too similar, and leading to samples being placed in incorrect clusters.

Not surprisingly, PCA and the original method for [9] has nearly identical result. It has been shown in [9] that PCA is the continuous solution of the cluster indicators in the  $K$ -means clustering method. On the other hand, UMAP shows a slightly different result. In the PCA method, the distribution is more spread out. In addition, the top occurrence for each country is higher for UMAP. On the other hand, we see that there are more samples in Cluster  $I_u$  for UMAP, which may indicate that mutations in Cluster  $I_u$  are the main strand.

Moreover, Figure 3.10 illustrates the 2D visualizations of the US dataset up to June 01, 2020, with 6 distinct clusters by applying two different dimensional reduction algorithms. We can see that the data distribute disorderly under both PCA- and UMAP-assisted  $K$ -means clustering algorithms. Specifically, the PCA-assisted algorithm has a really poor clustering performance, while the UMAP-assisted algorithm forms more clear and better clusters than the PCA-assisted algorithm, which is consistent with our previous analysis in Section 3.1.3.1.

Table 3.11 shows co-mutations occurred in each cluster from the UMAP-assisted  $K$ -means from data collected up to June 01, 2020. Cluster  $III_u$  has 2 dominant co-mutations. Note that the dataset had 15140 SARS-CoV-2 isolates, whereas our current dataset has over 200,000 isolates. Nonetheless, we can compare the clusters to see which clusters persists. Cluster 1's co-mutations are the same as those of Cluster  $V_u$ , indicating that Cluster 1 may have been derived from Cluster  $V_u$ . Cluster 2 shares the same co-mutations as those of Cluster  $II_u$ . Cluster 3's co-mutations are the descendants of Cluster  $V_u$ . Clusters 4 and 5 have the same co-mutations as those of Clusters  $III_u$  and  $VI_u$ , indicating Clusters 4 and 5 are derived from Cluster  $III_u$  and  $VI_u$ . Cluster 6's co-

mutations are descendants of Clusters  $III_u$  and  $VI_u$ . Note that co-mutations of Cluster  $I_u$  and the second set of co-mutations of Cluster  $II_u$  ([8782, 28144]) are not predominant co-mutations in our dataset, which may indicate a weaker infectivity. For example, every co-mutation in Table 3.9 has mutation 23403A>G (D614G) in the spike protein, which has been shown to increase infectivity of COVID-19 [35]. It is not surprising to see a co-mutation group not being dominant in our current dataset. By comparing these co-mutations, we can see that co-mutations that are dominant in both datasets (up to June 01 and January 20) will most likely persist in the future.

Table 3.11 The frequency and occurrence percentage of SARS-CoV-2 co-mutations from each clusters collected from June 01, 2020.

Cluster	Co-mutations	Frequency	Percentage
Cluster $I_u$	[11083, 14805, 26144]	948	0.730
Cluster $II_u$	[241, 3037, 14408, 23403, 25563]	2800	0.893
Cluster $III_u$	[241, 3037, 14408, 23403]	1468	0.412
	[8782, 28144]	1475	0.414
Cluster $IV_u$	[241, 1059, 3037, 14408, 23403, 25563]	1318	0.621
Cluster $V_u$	[241, 3037, 14408, 23403, 28881, 28882, 28883]	1872	0.817
Cluster $VI_u$	[241, 3037, 14408, 23403]	2222	0.969

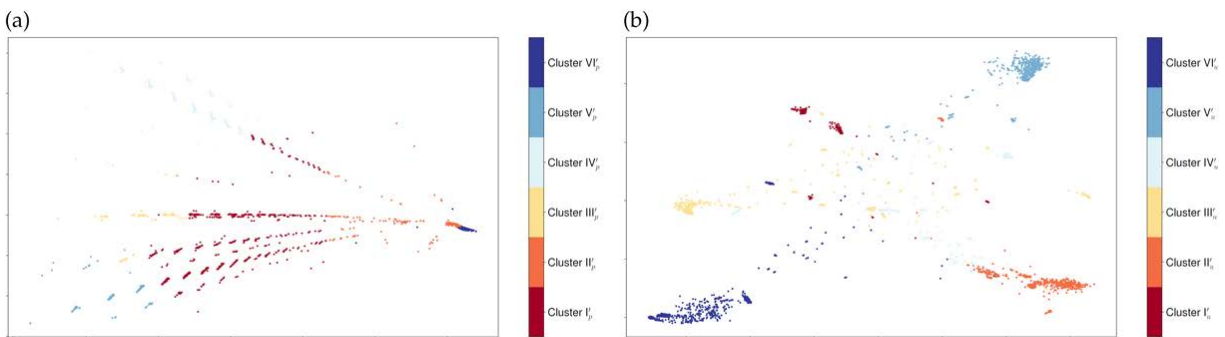


Figure 3.10 2D visualizations of the US SARS-CoV-2 mutation dataset up to June 01, 2020 with 6 distinct clusters by applying two different dimensional reduction algorithms. (a) 2D PCA visualization. (b) 2D UMAP visualization.

### 3.1.6 Conclusion

The rapid global spread of coronavirus disease 2019 (COVID-19) caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) has led to genetic mutation stimulated by genetic evolution and adaptation. Up to January 20, 2021, 203,344 complete SARS-CoV-2

sequences, and a total of 26,844 unique SNPs have been detected. Our previous work traced the COVID-19 transmission pathways and analyzed the distribution of the subtypes of SARS-CoV-2 across the world based on 15,140 complete SARS-CoV-2 sequences. The  $K$ -means clustering separated the sequences into six distinguished clusters. However, considering the tremendous increase in the number of available SARS-CoV-2 sequences, an efficient and reliable dimensional reduction method is urgently required. Therefore, the objective of the present work is to explore the best suited dimension reduction algorithm based on their performance and effectiveness. Here, a linear algorithm PCA and two non-linear algorithms, t-distributed stochastic neighbor embedding (t-SNE) and uniform manifold approximation and projection (UMAP), have been discussed. To evaluate the performance of dimension reduction techniques in clustering, which is an unsupervised problem, we first cast classification problems into clustering problems with labels. Next, by setting different reduction ratios, we test the effectiveness and accuracy of PCA, t-SNE, and UMAP for  $k$ -NN and  $K$ -means using four benchmark datasets. The results show that overall, UMAP outperforms other two algorithms. The major strengths of UMAP is that UMAP-assisted  $k$ -NN classification and UMAP-assisted  $K$ -means clustering at various dimension reduction ratios have a consistent performance in terms of accuracy, which proves that UMAP is a stable and reliable dimension reduction algorithm. Moreover, compared to the  $K$ -means clustering accuracy that does not involve any dimensional reduction, UMAP-assisted  $K$ -means clustering can improve the accuracy for most cases. Furthermore, when the dimension is reduced to two, the UMAP clustering visualization is clear and elegant. Additionally, UMAP is a relatively efficient algorithm compared to t-SNE. Although PCA is a faster algorithm, its major limitation is its poor performance in accuracy. To be noted, UMAP performs better than PCA and t-SNE for the dataset with a large number of samples, indicating it is the best suited dimensional reduction algorithm for our SARS-CoV-2 mutation dataset. Moreover, we apply the UMAP-assisted  $K$ -means clustering to the world SARS-CoV-2 mutation dataset (up to January 20, 2021), which displays six distinct clusters. Correspondingly, the same approaches are also applied to the United States SARS-CoV-2 mutation dataset (up to January 20, 2021), resulting in six different clusters as well. Furthermore, we provide a new

perspective by utilizing UMAP-assisted  $K$ -means clustering to analyze our previous SARS-CoV-2 mutation datasets, and the 2D visualization of UMAP-assisted  $K$ -means clustering of our previous world SARS-CoV-2 mutation dataset (up to June 01, 2020) forms more clear clusters than the PCA-assisted  $K$ -means clustering. Finally, one of our four datasets was generated by the Jaccard distance representation, which improves both  $k$ NN classification and  $k$ -means clustering accuracies on the original dataset.

## 3.2 $K$ -mer Topology for Whole Genome Analysis

### 3.2.1 Introduction

Topological data analysis (TDA) is an emerging field in data science and has also been utilized in DNA sequence alignment. TDA, or more specifically, persistent homology, begins by first representing the point cloud data with vertices, edges, triangles, tetrahedra, etc., or more generally, a simplicial complex. Then, concepts from algebraic topology, such as connected components, holes, and voids, are utilized to extract topological invariants, and filtration is applied to capture the persistence of such topological invariants across different scales. Chan et al. [36] utilized persistent homology to model both vertical and horizontal evolution in viruses, including clonal evolution, reassortment, and recombination. They computed sequence dissimilarity via sequence alignment and used this information to calculate persistent homology based on genetic dissimilarity. The 0th order homology represents vertical evolution, while the 1st order homology corresponds to horizontal evolution. This method was applied to SARS-CoV-2 to analyze mutations and utilized a novel metric called the topological recurrence index (tRI), which calculated the number of cycles in a tree and was used to measure convergent evolution [37]. In Nguyen et al. [38], persistent homology was applied to the CGR representation of viral sequences, and the resulting persistent diagram was computed. Subsequently, the Wasserstein distance was calculated between these diagrams to construct the phylogenetic tree. These methods represent some of the first adopters of persistent homology for DNA sequencing analysis. However, due to the high computational cost associated with computing persistent homology, their approach may not be suitable for longer sequences, general whole viral sequences, and large-scale comparisons.

In this work, we introduce the  $k$ -mer topology, a novel persistent homology approach based on  $k$ -mer position. For each  $k$ -mer, we construct position-based distances. Using the standard persistent homology methodology, we obtain Betti curves for each  $k$ -mer. We benchmarked our method using the NCBI virus reference genome and conducted phylogenetic tree analysis on six datasets. Our method demonstrates the highest accuracy in NCBI reference virus classification and proves to be an effective tool for phylogenetic analysis. Furthermore, we demonstrate that our method can handle large whole bacterial genomes, a capability not achievable with other persistent homology tools.

### 3.2.2 Method

In this section, we describe the  $k$ -mer specific topology. For an overview of persistent homology, refer to section 2.4

#### 3.2.2.1 Persistent Homology on Nucleotide Sequence

In this section, we describe the persistent homology for nucleotide sequence, and the construction of the features. Then, we define a metric on the features, which will be used for the phylogenetic analysis.

**Position based distance** Let  $S = s_1s_2\dots s_N$  be a DNA sequence of length  $N$ , where  $s_i \in \{A, C, G, T\}$  is a nucleotide. Define the nucleotide-specific indicator function

$$\delta_l(s_i) = \begin{cases} 1, & s_i = l \\ 0, & \text{otherwise} \end{cases} \quad (3.7)$$

where  $l \in \{A, C, G, T\}$ . Then, we can define a nucleotide specific vector  $S^l$  as

$$S^l = \{i | \delta_l(s_i) = 1, 1 \leq i \leq N\} \quad (3.8)$$

For example, for a sequence  $S = CGGATAACGTCCAGCAGTCAGTGATCGCATATCTTGAC$ , we

have

$$S^A = [4, 6, 7, 13, 16] \quad (3.9)$$

$$S^C = [1, 8, 11, 12, 15] \quad (3.10)$$

$$S^G = [2, 3, 9, 14, 17] \quad (3.11)$$

$$S^T = [5, 10] \quad (3.12)$$

The distance based in the nucleotide position, denoted  $D^l$ , is computed as the following:

$$D^l(i, j) = |S^l(i) - S^l(j)|. \quad (3.13)$$

For example, the position based distance for  $S^A$  is

$$D^A = \begin{pmatrix} 0 & 2 & 3 & 9 & 12 \\ 2 & 0 & 1 & 7 & 10 \\ 3 & 1 & 0 & 6 & 9 \\ 9 & 7 & 6 & 0 & 3 \\ 12 & 10 & 9 & 3 & 0 \end{pmatrix}. \quad (3.14)$$

Additionally, we can define a  $k$ -mer's specific position, where instead of focusing solely on individual nucleotides, we examine strings of nucleotides of length  $k$ . Moreover, for a given  $k$ , there are  $4^k$  different combinations of  $k$ -mers, denoted as  $l_1, l_2, \dots, l_{4^k}$ . Similarly, we can define the  $k$ -mer specific indicator function as

$$\delta_{l_p}(s_i s_{i+1} \dots s_{i+k}) = \begin{cases} 1, & s_i s_{i+1} \dots s_{i+k} = l_p \\ 0, & \text{otherwise} \end{cases} \quad (3.15)$$

Then, the  $k$ -mer specific position is given by

$$S^{l_p} = \{i | \delta_{l_p}(s_i s_{i+1} \dots s_{i+k}) = 1, 1 \leq i \leq N - k + 1\} \quad (3.16)$$

Then the  $k$ -mer specific position based distance can be defined as  $D^{l_p}(i, j) = |S^{l_p}(i) - S^{l_p}(j)|$ .

Using the same sequence as an example, AG specific position and distance can be defined as the

following

$$S^{AG} = [13, 16], \quad D^{AG} = \begin{pmatrix} 0 & 3 \\ 3 & 0 \end{pmatrix} \quad (3.17)$$

**Persistent homology features** Using the position-based distance, Ripser [39] was used to construct the simplicial complex and the persistent barcodes. Then, Gudhi [40] was used to obtain the betti-0 curves. Denote  $\beta_{0,r}^{l_p}$  as the Betti number at filtration  $r$  for a particular  $k$ -mer  $l_p$ . For example,  $\beta_{0,3}^{AT}$  would be the betti-0 for 2-mers  $AT$  at filtration 3. We can then collect the Betti values across increasing filtration values to get the Betti curve  $\beta_0^{l_p}$ , which can be defined as the following vector

$$\beta_0^{l_p} = (\beta_{0,0}^{l_p}, \dots, \beta_{0,r}^{l_p}, \dots, \beta_{0,R}^{l_p}) \quad (3.18)$$

where  $R$  is the maximum number of filtrations.

Figure 3.11 shows an example of a persistent barcode and the Betti curve. We generated a random sequence of length  $N = 100$ , and allowed the sequence to be circular. For such circular sequence, we can define the distance between  $S^{l_p}(i)$  and  $S^{l_p}(j)$  to be

$$D^{l_p}(i, j) = \min(S^{l_p}(j) - S^{l_p}(i), N - S^{l_p}(j) + S^{l_p}(i)) \quad (3.19)$$

for  $i < j$ . Then, the persistent homology will be based on a loop. The left figure shows the persistent barcode of  $l = A$ , where the barcode shows the birth and death of a connected component ( $H_0$  in blue), loop ( $H_1$  in green) and cavity ( $H_2$  in red). The right figure shows  $\beta_0^l$ . The 4 colors correspond to the nucleotides  $l = A, C, G, T$ .



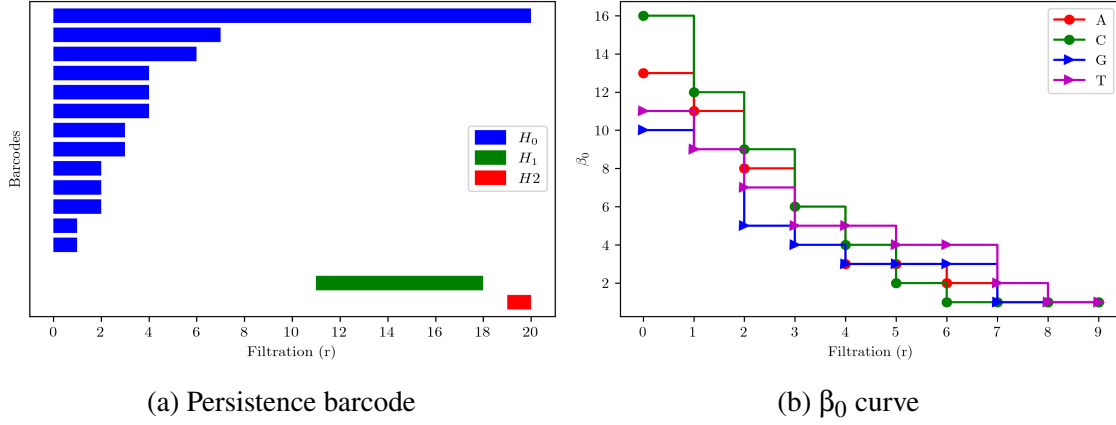


Figure 3.11 Persistent barcode and Betti curve of nucleotide  $l = A$  of a randomly generated circular sequence of length 100 base pairs. (a) Blue, red and green correspond to the birth and death of  $H_0$ ,  $H_1$  and  $H_2$  features. (b) The Betti curves of  $A$  (red),  $C$  (green),  $G$  (blue),  $T$  (purple). The y axis corresponds to the  $\beta_0$  number at a particular filtration value.

Additionally, we provide a real virus example using NC\_001330, a reference sequence from the *Microviridae* family. The *Microviridae* family comprises bacteriophages with single-stranded DNA (ssDNA) genomes. NC\_001330 consists of 6087 base pairs, including 1431  $A$ , 1325  $C$ , 1425  $G$ , and 1906  $T$ . Figure 3.12 shows the Betti curves for individual nucleotides, as well as for 2-mers, 3-mers, and 4-mers.

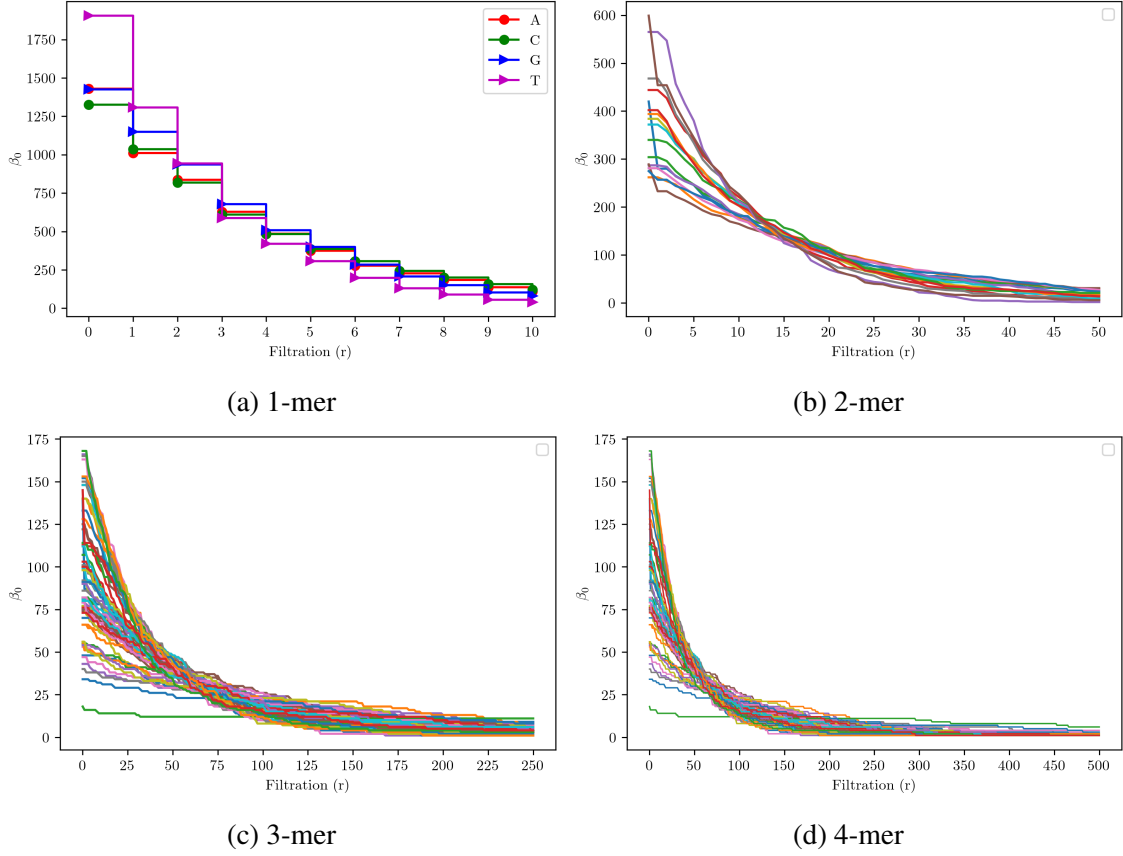


Figure 3.12 Betti curves of 1-mer, 2-mer, 3-mer and 4-mer of NC\_001330 reference genome of the *Microviridae* family.

For a given  $k$ , there are a total of  $4^k$  combinations of  $k$ -mers. Therefore, if the sequence is truly random, we would expect to see the same  $k$ -mer roughly every  $4^k$  position. Not surprisingly, as we increase the  $k$ , we see that the connected components persists for longer time, which indicate that the filtration number must increase. Moreover, in order to reduce the number of features we obtain from the Betti curves, we compute the Betti numbers at a step size of  $4^{k-1}$  for a total of 50 steps.

**Distance on the  $k$ -mers persistent homology features** We now define the metric used to perform phylogenetic analysis. For each  $k$ , we have  $4^k$  different  $k$ -mer specific Betti curves. We concatenate these Betti curves to obtain the vector

$$\beta^k = (\beta_0^{l_1}, \dots, \beta_0^{l_{4^k}}). \quad (3.20)$$

Then, for viruses  $i$  and  $j$ , we can define the metric between the  $k$ -mer specific Betti curves as the

following

$$\text{dist}^k(i, j) = \|\beta_i^k - \beta_j^k\|_2 \quad (3.21)$$

where  $\|\cdot\|_2$  denote the euclidean distance.

Then, we can take a weighted sum over different  $k$  to obtain the distance between viruses  $i$  and  $j$

$$\text{Dist}(i, j) = \sum_{k=1}^K a_k \text{dist}^k(i, j). \quad (3.22)$$

### 3.2.3 Results

#### 3.2.3.1 Classification of the reference viral genome

In order to verify the effectiveness of our method, we conducted classification on the viral reference genome from the National Center for Biotechnology Information (NCBI) virus database. NCBI adopts the virus taxonomy from the International Committee on Taxonomy of Viruses (ICTV), which updates the nomenclature of viruses periodically according to new findings. We considered NCBI dataset collected in 2020, 2022 and 2024, and the details of the preprocessing process can be found in Table 3.12. For NCBI 2020 and NCBI 2022, we have removed sequences that were no longer considered reference sequences as of January 20, 2024. For NCBI 2024, we removed families without a proper taxonomy rank, i.e., any sequences with a family name not ending in '-viridae'. For example, the viral family Tolecusatellitidae was removed because its rank is undetermined. We included these undetermined families in NCBI 2024 full. Additionally, we included sequences with invalid nucleotides to simulate real-world scenarios. We benchmarked our method on these data because viral taxonomy is regularly updated. Therefore, our method not only needs to identify the correct family but also find the most similar sequence within that family. For example, viral families such as Herpesviridae and Reoviridae from the NCBI 2020 and NCBI 2022 datasets were abolished in late 2022, with some sequences being divided into smaller families or remaining unclassified. Therefore, it is crucial for the alignment method to be robust to these changes and accurately identify the correct viral family.

Table 3.12 Dataset, NCBI collection date, proprocessing procedure, number of families and number of sequences.

Name	NCBI date	Removed sequence	# families	# sequence
NCBI 2020 [41]	03/2020	Unknown Baltimore class Unknown family families <3 sequence	83	6,993
NCBI 2022	03/2022	Partial sequence Unknown family families <2 sequence Invalid nucleotides	123	11,428
NCBI 2024	01/20/2024	Partial sequence Unknown family Only '-viridae' families <3 sequence Invalid nucleotide	199	12,154
NCBI 2024 full	01/20/2024	Partial sequence Unknown family families <3 sequence	209	13,645

For benchmarking, we adopt the procedure outlined in [41]. After constructing the  $k$ -mer specific Betti curves and the distance matrix, we performed a 1-nearest neighbor (1-NN) classification. The identification is considered correct if the most similar sequence has the same family label. This simulates a real-world scenario, where identifying the most similar sequence is critical for further analysis. We compare our method with generalize natural vector (GNV) and Markov  $K$ -string (Markov) model [42], and their details can be found in section 2.5

Table 3.13 show the 1-NN classification accuracy of individual  $k$ -mer specific Betti curves as well as the weighted sum. 3-mers and 4-mers have the highest individual accuracy, and 1-mers have the lowest accuracy. The weighted sum has the higher accuracy on all the data except for NCBI 2022.

Table 3.13 1-NN classification of the 4 dataset. The accuracy of individual  $k$ -mer specific Betti curves were obtained, as well as the weighted sums. The bolded number correspond to the highest accuracy.

Data	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	Sum
NCBI 2020	0.817	0.902	0.929	<b>0.933</b>	0.927	<b>0.933</b>
NCBI 2022	0.814	0.894	0.917	<b>0.921</b>	0.912	0.920
NCBI 2024	0.750	0.857	0.887	0.895	0.828	<b>0.898</b>
NCBI 2024 full	0.748	0.857	0.889	0.897	0.887	<b>0.901</b>

Figure 3.13 shows the comparison of our method to  $k$ -mer specific GNV and Markov model. For GNV, we utilized order 2 for each  $k$  and, used Euclidean distance to construct the distance matrix. Notice that our model's accuracy increases as  $k$  increases, plateauing at  $k = 4$ . On the other hand, GNV and Markov model decreases in accuracy at  $k = 2$  and  $k = 3$ , respectively. The decrease in accuracy for the Markov model is not surprising. This is because the Markov  $k$ -string method assesses sequence differences by comparing the observed appearance of  $k$ -strings with the predicted appearance of  $k$ -strings computed through the Markov model. The predicted model evaluates the left and right  $k-1$  substrings, along with the middle  $k-2$  substring. Therefore, 3-mers exhibit the most reliable computation of predicted appearance. The decrease in GNV performance is consistent with the report given by the original authors [41]; however, it is important to note the GNV's strength comes from the metric defined on all the  $k$ -mer specific GNV.

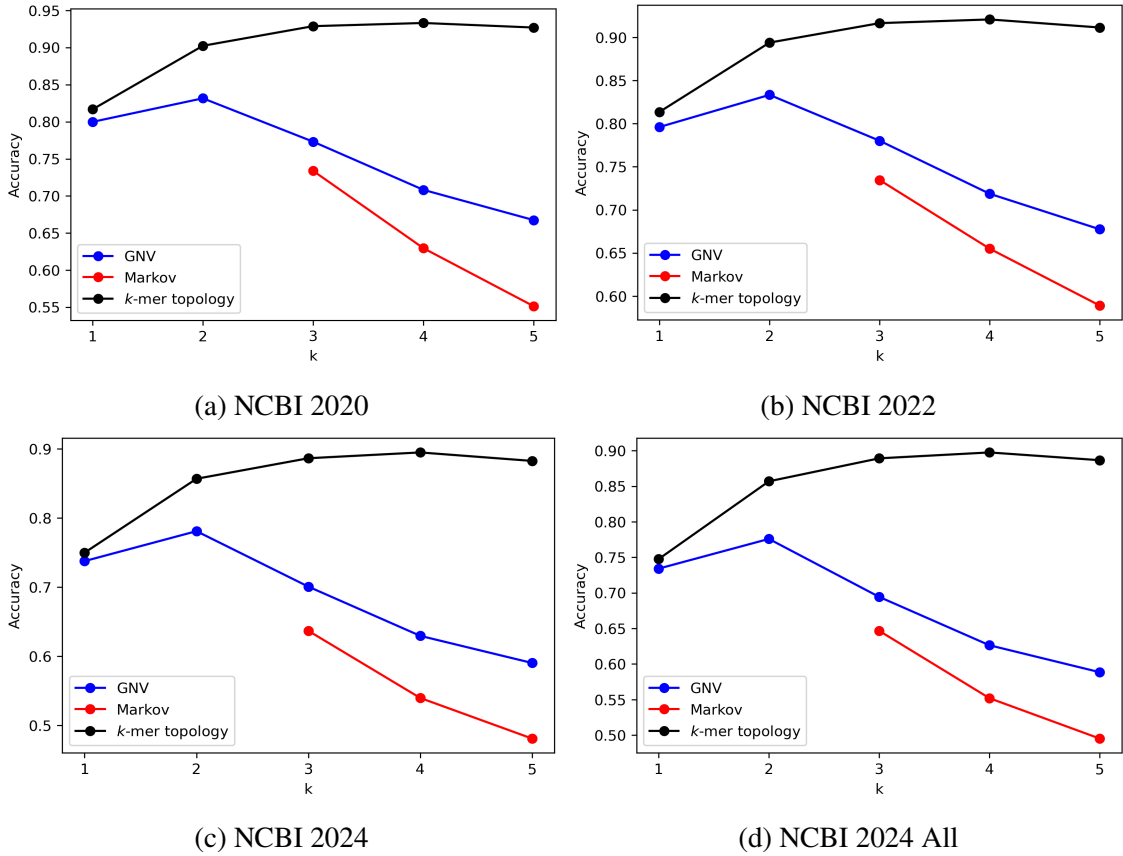


Figure 3.13 Comparison of  $k$ -mer topology with GNV and Markov models. For GNV, order 2 for each  $k$  is presented.

Figure 3.14 shows the comparison of our method with GNV and the Markov  $k$ -string model. For GNV, we utilized order 2 with  $K = 9$ , and for the Markov model, we employed  $k = 3$ . In all models, there is a decrease in accuracy from 2020 to 2024, which is not surprising given the increase in the number of families and the division of large families into smaller ones. Notably, the Markov model proves to be less robust to these changes. Interestingly, our method demonstrates slightly better performance on NCBI 2024 All data compared to NCBI 2024 data. This is promising for real-world applications, as most sequences contain some invalid nucleotides due to experimental errors. Moreover, our method outperforms GNV by 6.13%, 5.12%, 8.33%, and 9.17% for NCBI 2020, NCBI 2022, NCBI 2024, and NCBI 2024 All data, respectively.

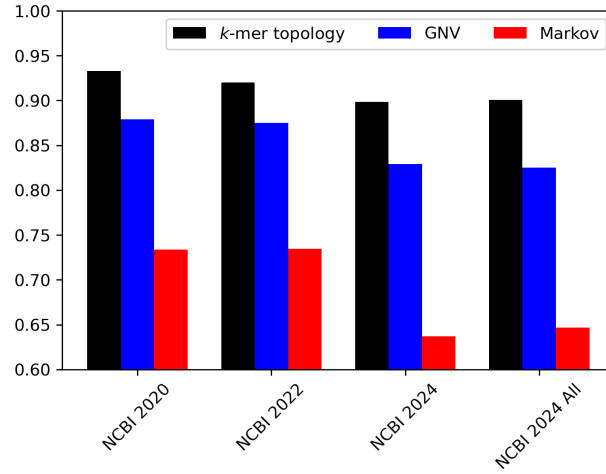


Figure 3.14 Accuracy of  $k$ -mer topology, GNV and Markov models on the 4 dataset. For  $k$ -mer topology and GNV, the weighted sum over the different  $k$  was used. For GNV,  $k = 9$  and order 2 was used, and for Markov model,  $k = 3$  was used.

### 3.2.3.2 Phylogenetic analysis using $k$ -mer topology

We constructed a phylogenetic tree utilizing the  $k$ -mer topology method. After computing the distance, we utilized unweighted pair group method with arithmetic mean (UPGMA) method to construct the tree.

**Ebolavirus** We obtained 59 complete genomes of ebolavirus, representing 5 species: Bundibugyo virus (BDBV), Reston virus (RESTV), Ebola virus (EBOV, formerly known as Zaire virus), Sudan virus (SUDV), and Tai Forest virus (TAFV). Details of the data can be found in Table 3B.1.

The EBOV species was further divided into 6 subtypes based on the location and year of the outbreak: Zaire (now known as DRC) pandemic of 1976-1977, DRC pandemic of 2007, Guinea outbreak of 2014, Gabon outbreak of 1994, and DRC outbreak of 1995.

Figure 3.15 displays the phylogenetic tree generated using our method. We employed  $K = 5$  to compute the  $k$ -mer topology features and utilized the UPGMA algorithm to generate the phylogenetic tree. The colors of the clades and labels correspond to the ebolavirus types and the EBOV subtypes. Notably, BDBV, RESTV, EBOV, SUDV, and TAFV are separated into distinct clades. Additionally, the EBOV subtypes are placed in separate clades, representing an improvement over [43], where the Fuzzy Integral Similarity method incorrectly placed one of the DRC outbreak of 2007 strains into the original Zaire pandemic clade.

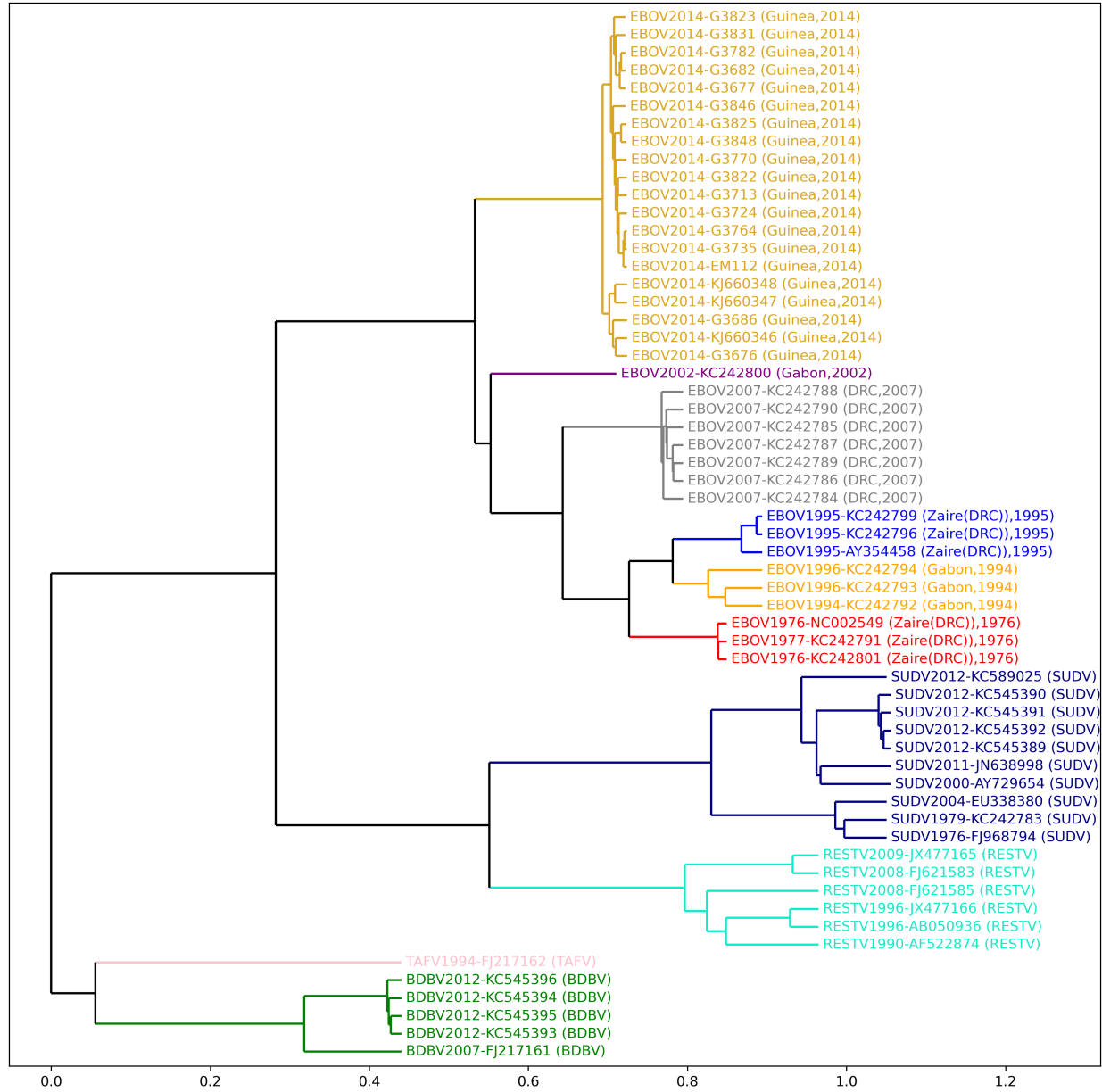


Figure 3.15 Phylogenetic tree of ebolavirus complete genomes, including those from Bundibugyo virus (BDBV), Reston virus (RESTV), Ebola virus (EBOV, formally known as Zaire virus), Sudan virus (SUDV) and Tai Forest virus (TAFV). EBOV was divided into 6 subtypes corresponding to the pandemic or outbreak. The labels and clades were colored according to their types and pandemic.

**Mammalian Mitochondria RNA** We obtained 41 full mitochondria genome of mammals, consisting of 8 types Artiodactyla, Carnivore, Cetacea, Erinaceomorpha, Lagomorpha, Perissodactyla, Primates, Rodentia. The details of the data can be found in Table 3B.2. We utilized  $K = 4$  to compute the  $k$ -mer topology features and the metric, and the UPGMA algorithm was employed



to generate the phylogenetic tree. Figure 3.16 illustrates the phylogenetic tree of the 8 classes of species. The labels and the clade are colored according to their group labels. Our method correctly clusters each type into individual clades, which is an improvement from other methods, which often placed Rabbit (Lagomorpha) into a clade with Carnivores [43].

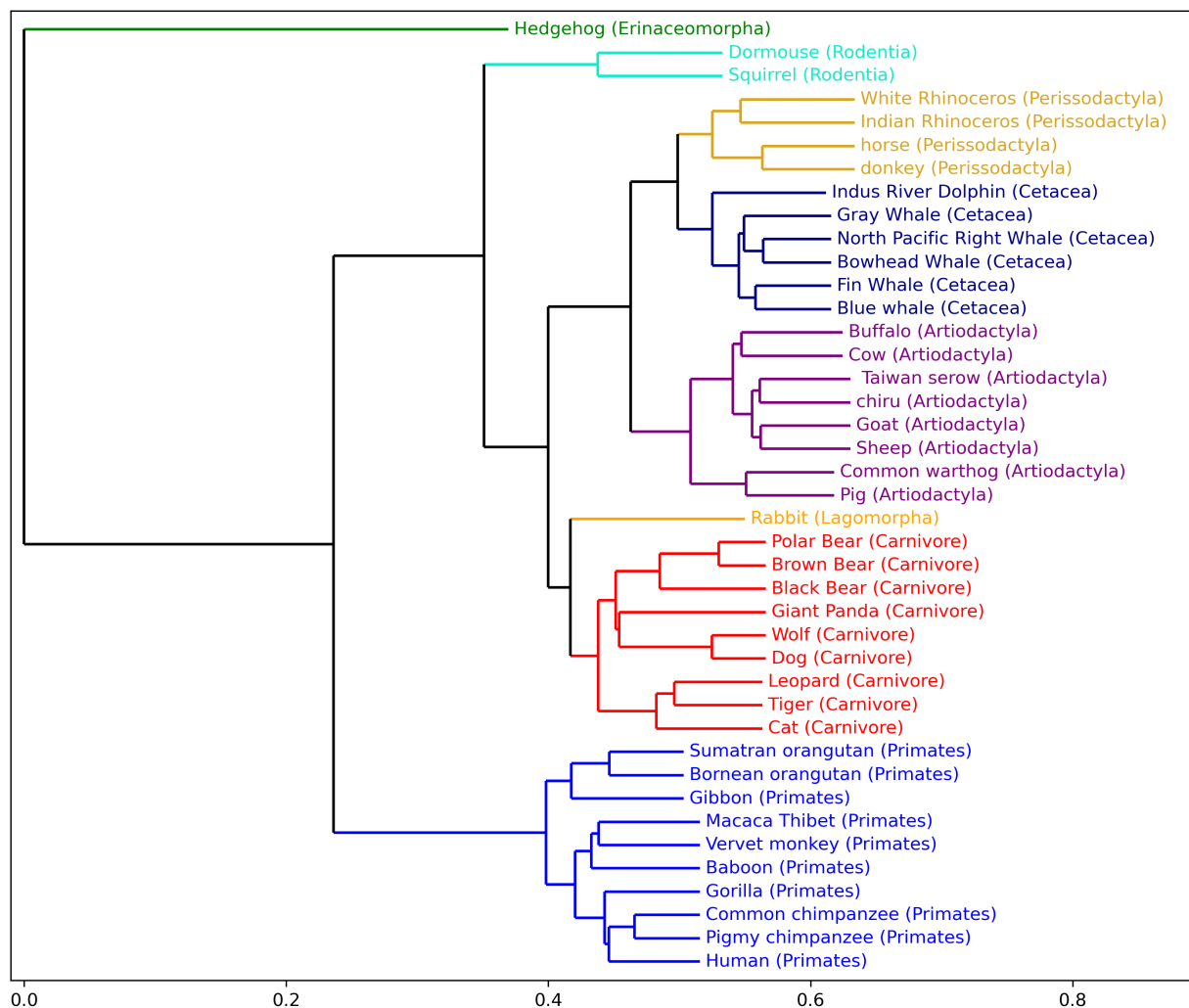


Figure 3.16 Phylogenetic tree of mammalian mitochondrial genomes, including those from Artiodactyla, Carnivora, Cetacea, Erinaceomorpha, Lagomorpha, Perissodactyla, Primates, and Rodentia. Details of the data can be found in Table 3B.2. The species and clades are colored according to their groups.  $K = 4$  was used to compute the  $k$ -mer topology features, and the UPGMA algorithm was used to construct the tree.

**Rhinovirus** We obtained 113 rhinovirus (HRV) and 3 non-rhinovirus outgroup. HRV is a positive-sense, single-stranded RNA virus (ssRNA(+)) belonging to the viral family *Picornaviridae* and the genus Enterovirus, and is often the predominant cause of common cold [44, 45]. The details

of the data can be found in Table 3B.3. There are 3 distinct groups within *Picornaviridae*, HRV-A, HRV-B and HRV-C. Figure 3.17 show the phylogenetic tree of 113 whole HRV genome and 3 outgroup sequence (HEV-C). We utilized  $K = 3$  to compute the homology features and the distance. The labels and clades are colored according to their groups.

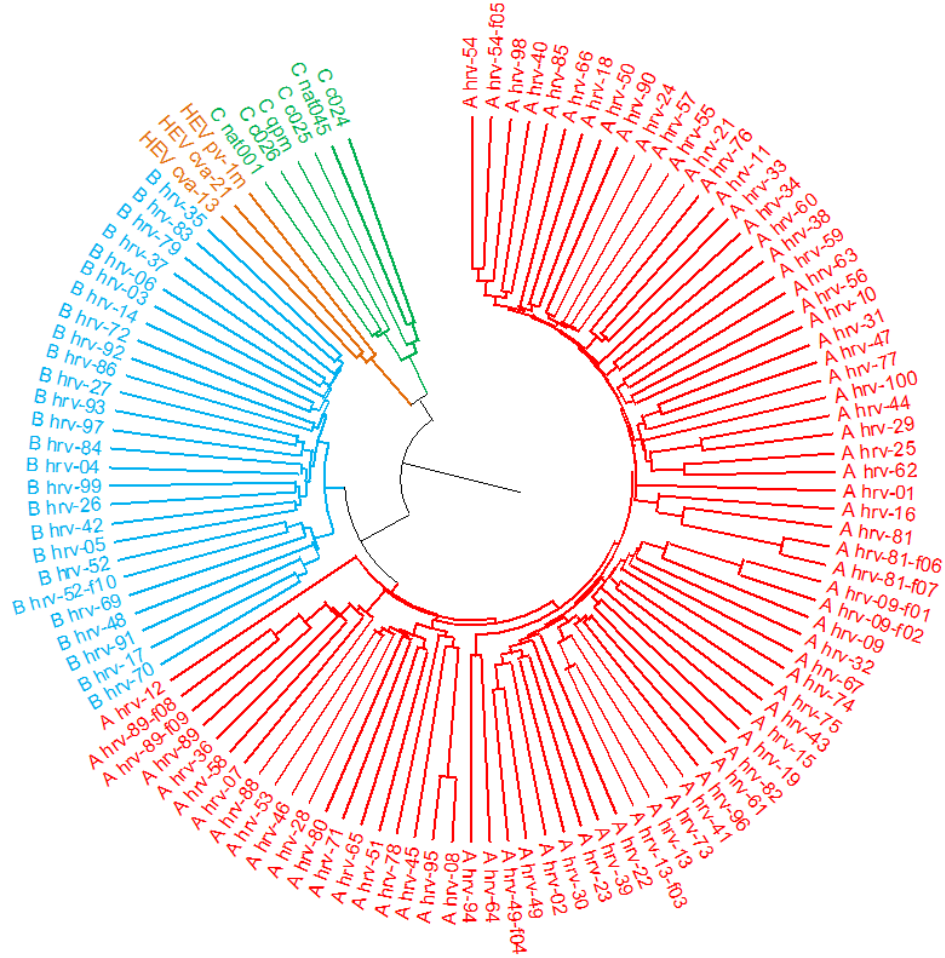


Figure 3.17 Phylogenetic tree of 113 rhinovirus (HRV) and 3 non-rhinovirus outgroup. The colors correspond to different host animals.  $K = 5$  was used to compute the  $k$ -mer topology features and distance, and the UPGMA algorithm was used to construct the tree.

Our method correctly separates all the groups into individual clades. However, we noticed that HRV-C shares a clade with the HEV-C outgroup. This result is similar to the findings in [46], where HRV-A and HRV-B shared a clade, and HRV-C shared a clade with the HRV-A/HRV-B parent clade.

**Coronavirus** We obtained 30 coronavirus full genome sequences, which range from about 25,000 to 30,000 nucleotides. This dataset has been studied in various other studies [47, 48, 49, 46, 43]." Coronaviruses are enveloped, single-stranded, positive-sense RNA (ssRNA(+)) viruses belonging to the *Coronaviridae* family. They are known to infect many species, including bats, birds, humans, and other mammals, and can spread across different species [50, 51]. With the SARS-CoV-2 pandemic (COVID-19), there has been interest in uncovering the evolutionary relationship between other coronaviruses. The details of the data can be found in Table 3B.4. Figure 3.18 show the phylogenetic tree generated by our method.  $K = 7$  was used to compute the  $k$ -mer topology features and the metric, in UPGMA algorithm was used to construct the tree. The samples were colored according to the 5 coronavirus groups.

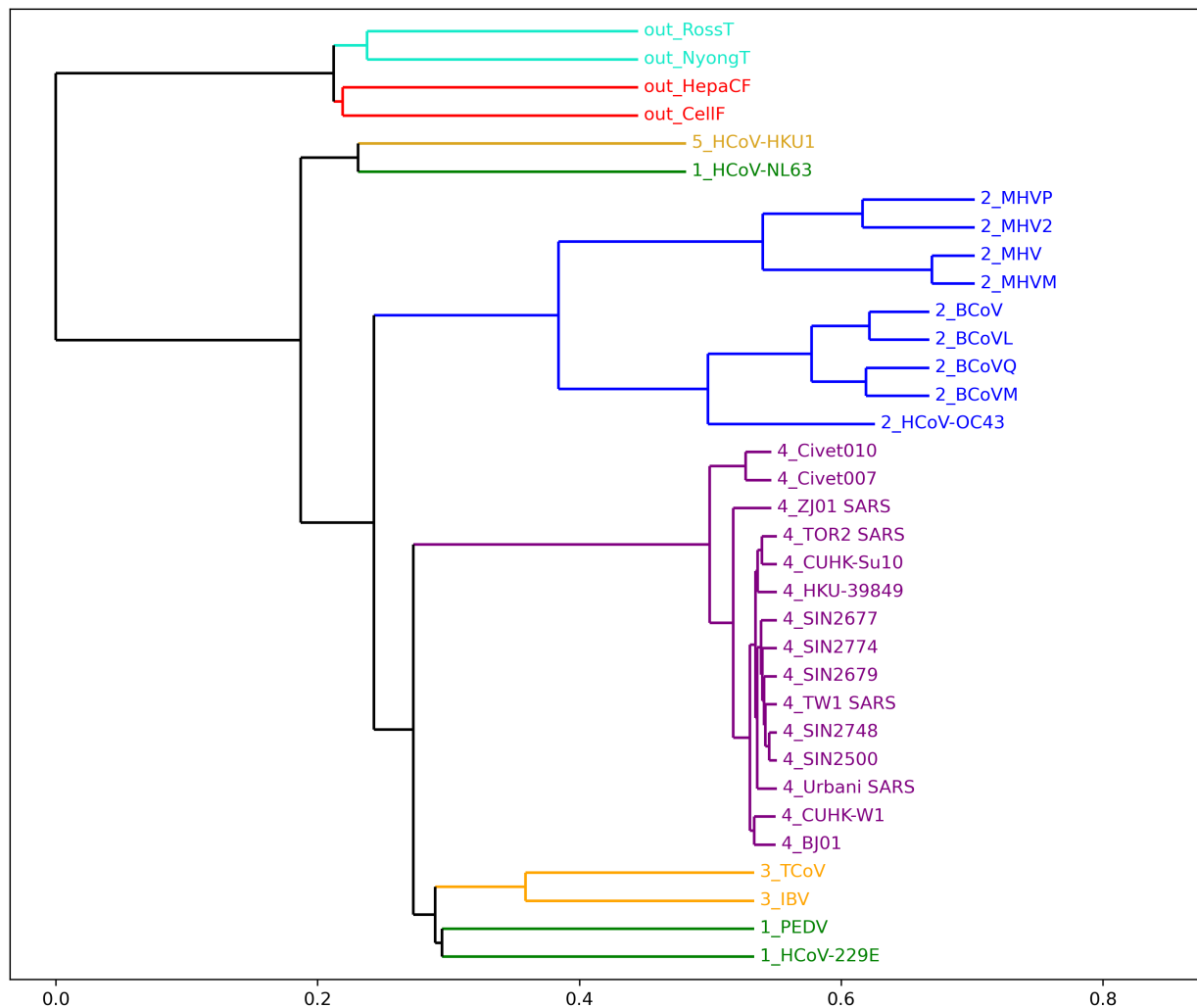


Figure 3.18 Phylogenetic tree of 30 coronavirus and 4 non-coronavirus outgroup whole genome. The colors correspond to different host animals.  $K = 8$  was used to compute the  $k$ -mer topology features and distance, and the UPGMA algorithm was used to construct the tree.

Our method correctly clusters the non-coronavirus outgroups and groups 2, 3, 4, and 5. The NL63 strain from group 1 was not clustered with the other group 1 human coronaviruses, consistent with many alignment-free studies [43]. Additionally, our method separates group 2 coronaviruses into their hosts, namely the mouse hepatitis virus (MHV) and the bat coronavirus (BCoV).

**Influenza Type A** We obtained 36 full genomes of influenza A viruses. Influenza A viruses are single-stranded, segmented RNA viruses classified based on the surface proteins hemagglutinin (H) and neuraminidase (N) [52]. Additionally, they pose a major health threat to humans and have resulted in numerous outbreaks [53]. We obtained 5 subtypes H7N9, H7N3, H2N2, H5N1, and

H1N1 for the analysis, and the details of the data can be found in Table 3B.5. Figure 3.19 show the phylogenetic tree generated using our methods. The left and right figures correspond to homology features and distance computed using  $K = 5$  and  $K = 1$ . Notice that in  $K = 5$ , we have multiple clades of H1N1 that share the same clade as H5N1 strands; however the  $K = 1$  was separated the H5N1 and H1N1 strands into different clades. This indicate the potential overfitting in  $K = 5$ .

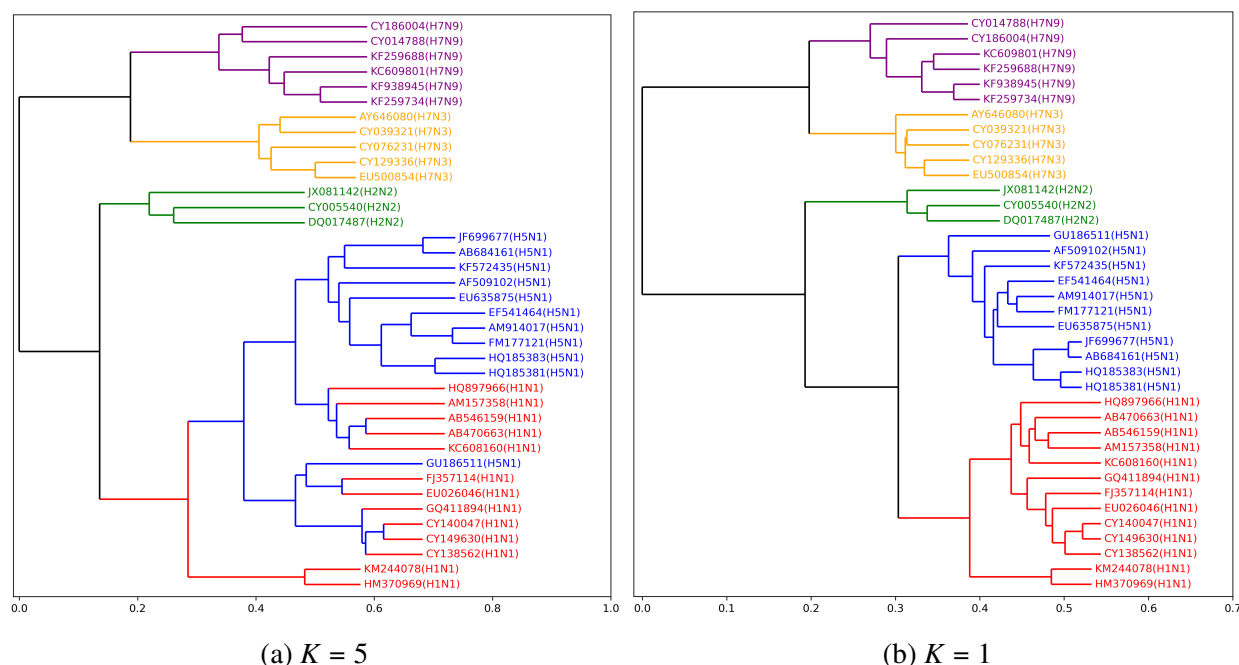


Figure 3.19 Phylogenetic tree of 6 influenza A virus. The colors correspond to different subtypes, including H7N9, H7N3, H2N2, H5N1, and H1N1. (a)  $K = 5$  (b)  $K = 1$ . After computing the  $k$ -mer topology features and distances, UPGMA algorithm was used to construct the tree. The label and the clades are colored according to the subtypes.

**16S rDNA from Bacteria Isolates** We obtained 40 bacterial sequences that was cultured from endophytic bacteria isolated from the surface sterilized mature endosperms of 6 rices varieties. The 40 sequences have length  $< 1500$ bp. The details of the data can be found in Table 3B.6. Figure 3.20 show the phylogenetic tree of the 16S rDNA from bacteria isolates using our method. We utilized  $K = 5$  to compute the homology features and the distance. Then, UPGMA algorithm was used to obtain the tree. The clades and labels were colored according to the bacterial families.

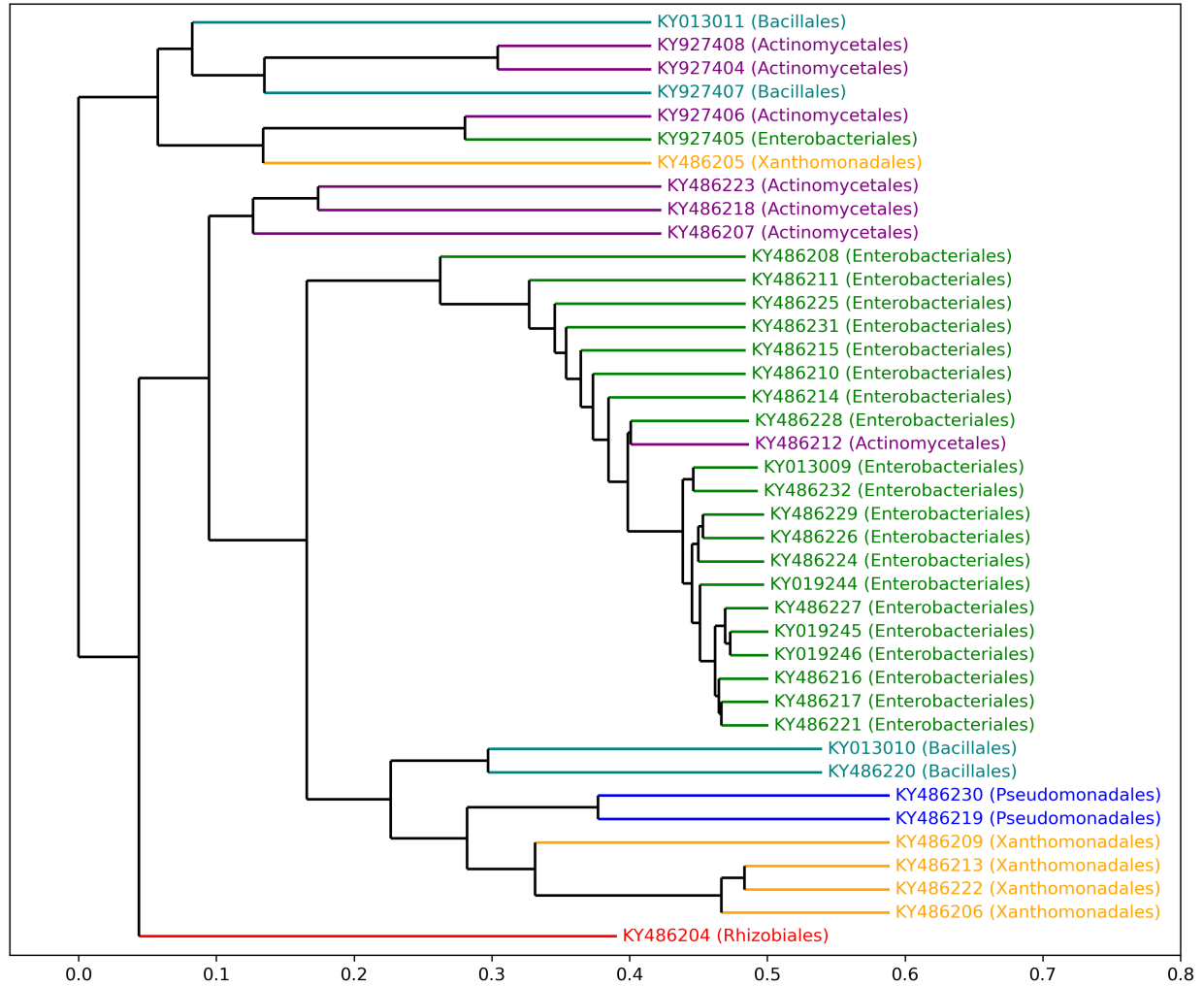


Figure 3.20 Phylogenetic tree of 40 16S rDNA from bacteria isolates. The colors correspond to different host animals.  $K = 5$  was used to compute the  $k$ -mer topology features and distance, and the UPGMA algorithm was used to construct the tree.

Our method shows a main clades for Actinomycetales, Xanthomandales, Enterobacteriales, Pseudomandales, and Rhizobiales. However, the top clade shows a mix of misclustered sequences. This is consistent with [43], but our method show dominant clades for each bacteria family. This result suggest the limitation of our method for shorter sequence data, where more computationally expensive methods, such as multiple sequencing alignment can benefit greatly.

**Large bacteria genome** We obtained 30 bacteria whole genomes from families *Bacillaceae*, *Borreliaceae*, *Burkholderiaceae*, *Clostridiaceae*, *Desulfovibrionaceae*, *Enterobacteriaceae*, *Rhodobacteraceae*, *Staphylococcaceae*, and *Yersiniaceae*. The details of the data can be found in Table 3B.7.

All the sequences have over 1 million base pair, and we tested our method to see the computational limits of our method. Because the sequences are long, we computed the homology of 3-mers, 4-mers and 5-mers, and computed the distance using these 3  $k$ -mers. Figure 3.21 show the phylogenetic tree generated using our method. After the distance was computed, we utilized UPGMA algorithm to construct the tree. The labels and clades were colored according to the family label.

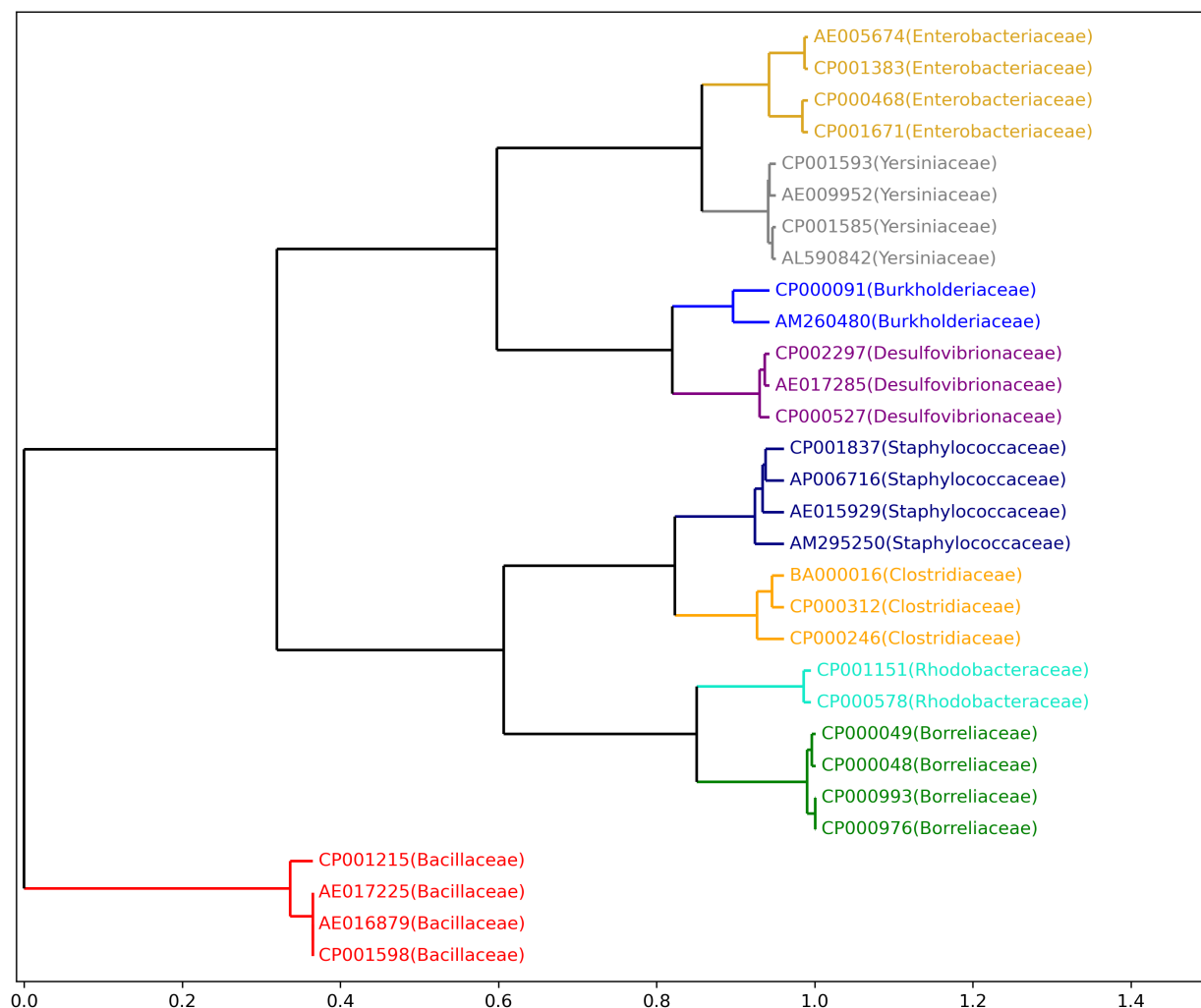


Figure 3.21 Phylogenetic tree of 30 bacterial whole genomes. Betti curves for 3-mers, 4-mers and 5-mers were computed, and the distance was computed using these 3  $k$ -mers. The labels and clades were colored according to their family labels.

Our method correctly separates individual families into a clade. However, our method fails to place the samples into the appropriate phylum. *Enterobacteriales*, *Yersiniaceae*, *Burkholderiaceae* and *Rhodobacteraceae* belong to the phylum Pseudomonadota, and our method only clusters *Enter-*

*obacterales*, *Yersiniaceae* into the same clade. Additionally, *Staphylococcaceae*, *Clostridiaceae*, *Bacillaceae* belongs to the phylum Bacillota, but only *Staphylococcaceae* and *Clostridiaceae* share a clade. This is most likely caused by our method not being able to compute the 1-mer and 2-mers topology because lower-order  $k$ -mers should provide insight to higher order taxonomy.

### 3.2.4 Discussion and conclusion

$K$ -mer topology is a novel method that computes the persistent homology based on the position of the  $k$ -mer. Our method outperformed GNV method in viral classification tasks on different versions of the data, indicating the robustness to changes in the taxonomy. We have also validated our method on standard phylogenetic analysis, including bacteria sequence, indicating the scalability of our method. However, our method does come with limitation. First, the accuracy for identifying the correct family in bacteria and virus is high; however, our method do not perform well when clustering high order taxonomy, most notably in the case of bacteria. This is due to the high computational cost of 1-mers and 2-mers topology for long sequences. Additionally, analyzing genomes longer than bacteria is not possible at this moment. In the future, we want to look at more algebraic topological tools, such as topological Laplacians, hypergraphs, and more. Moreover, we would like to evaluate the nonharmonic portion of our spectra to obtain more geometric information about the DNA sequence. Additionally, to combat the computational limitation for long sequence, we would like to explore potential of taking segments to increase the efficiency of our algorithm. Lastly, we would like to explore the application in protein sequences.



## BIBLIOGRAPHY

- [1] Covid-19 weekly epidemiological update, 19 january 2021, 2021.
- [2] The species severe acute respiratory syndrome-related coronavirus: classifying 2019-ncov and naming it sars-cov-2. *Nature microbiology*, 5(4):536–544, 2020.
- [3] Fan Wu, Su Zhao, Bin Yu, Yan-Mei Chen, Wen Wang, Zhi-Gang Song, Yi Hu, Zhao-Wu Tao, Jun-Hua Tian, Yuan-Yuan Pei, et al. A new coronavirus associated with human respiratory disease in china. *Nature*, 579(7798):265–269, 2020.
- [4] Intikhab Alam, Allan A Kamau, Maxat Kulmanov, Łukasz Jaremko, Stefan T Arold, Arnab Pain, Takashi Gojobori, and Carlos M Duarte. Functional pangenome analysis shows key features of e protein are preserved in sars and sars-cov-2. *Frontiers in cellular and infection microbiology*, 10:405, 2020.
- [5] Yu-Nong Gong, Kuo-Chien Tsao, Mei-Jen Hsiao, Chung-Guei Huang, Peng-Nien Huang, Po-Wei Huang, Kuo-Ming Lee, Yi-Chun Liu, Shu-Li Yang, Rei-Lin Kuo, et al. Sars-cov-2 genomic surveillance in taiwan revealed novel orf8-deletion mutant and clade possibly associated with infections in middle east. *Emerging microbes & infections*, 9(1):1457–1466, 2020.
- [6] Peter Forster, Lucy Forster, Colin Renfrew, and Michael Forster. Phylogenetic network analysis of sars-cov-2 genomes. *Proceedings of the National Academy of Sciences*, 117(17):9241–9243, 2020.
- [7] Xingguang Li, Junjie Zai, Qiang Zhao, Qing Nie, Yi Li, Brian T Foley, and Antoine Chaillon. Evolutionary history, potential intermediate animal host, and cross-species analyses of sars-cov-2. *Journal of medical virology*, 92(6):602–611, 2020.
- [8] Sunitha M Kasibhatla, Meenal Kinikar, Sanket Limaye, Mohan M Kale, and Urmila Kulkarni-Kale. Understanding evolution of sars-cov-2: a perspective from analysis of genetic diversity of rdrp gene. *Journal of medical virology*, 92(10):1932–1937, 2020.
- [9] Rui Wang, Yuta Hozumi, Changchuan Yin, and Guo-Wei Wei. Decoding sars-cov-2 transmission and evolution and ramifications for covid-19 diagnosis, vaccine, and medicine. *Journal of chemical information and modeling*, 60(12):5853–5865, 2020.
- [10] Rui Wang, Jiahui Chen, Kaifu Gao, Yuta Hozumi, Changchuan Yin, and Guo-Wei Wei. Characterizing sars-cov-2 mutations in the united states. *Research square*, 2020.
- [11] Jiahui Chen, Rui Wang, Menglun Wang, and Guo-Wei Wei. Mutations strengthened sars-cov-2 infectivity. *Journal of molecular biology*, 432(19):5212–5226, 2020.
- [12] Rui Wang, Jiahui Chen, Yuta Hozumi, Changchuan Yin, and Guo-Wei Wei. Decoding

- asymptomatic covid-19 infection and transmission. *The journal of physical chemistry letters*, 11(23):10007–10015, 2020.
- [13] Michael Worobey, Jonathan Pekar, Brendan B Larsen, Martha I Nelson, Verity Hill, Jeffrey B Joy, Andrew Rambaut, Marc A Suchard, Joel O Wertheim, and Philippe Lemey. The emergence of sars-cov-2 in europe and north america. *Science*, 370(6516):564–570, 2020.
  - [14] Yunmeng Bai, Dawei Jiang, Jerome R Lon, Xiaoshi Chen, Meiling Hu, Shudai Lin, Zixi Chen, Xiaoning Wang, Yuhuan Meng, and Hongli Du. Comprehensive evolution and molecular characteristics of a large number of sars-cov-2 genomes reveal its epidemic trends. *International Journal of Infectious Diseases*, 100:164–173, 2020.
  - [15] Yujiro Toyoshima, Kensaku Nemoto, Saki Matsumoto, Yusuke Nakamura, and Kazuma Kiyotani. Sars-cov-2 genomic variations associated with mortality rate of covid-19. *Journal of human genetics*, 65(12):1075–1082, 2020.
  - [16] Lucy Van Dorp, Mislav Acman, Damien Richard, Liam P Shaw, Charlotte E Ford, Louise Ormond, Christopher J Owen, Juanita Pang, Cedric CS Tan, Florencia AT Boshier, et al. Emergence of genomic diversity and recurrent mutations in sars-cov-2. *Infection, Genetics and Evolution*, 83:104351, 2020.
  - [17] Roderic DM Page. Space, time, form: viewing the tree of life. *Trends in ecology & evolution*, 27(2):113–120, 2012.
  - [18] Tingting Zhou, Keith CC Chan, Yi Pan, and Zhenghua Wang. An approach for determining evolutionary distance in network-based phylogenetic analysis. In *Bioinformatics Research and Applications: Fourth International Symposium, ISBRA 2008, Atlanta, GA, USA, May 6-9, 2008. Proceedings 4*, pages 38–49. Springer, 2008.
  - [19] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.
  - [20] John W Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 100(5):401–409, 1969.
  - [21] Chun-houh Chen, Wolfgang Härdle, Antony Unwin, Michael AA Cox, and Trevor F Cox. *Multidimensional scaling*. Springer, 2008.
  - [22] George C Linderman, Manas Rachh, Jeremy G Hoskins, Stefan Steinerberger, and Yuval Kluger. Fast interpolation-based t-sne for improved visualization of single-cell rna-seq data. *Nature methods*, 16(3):243–245, 2019.
  - [23] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

- [24] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
- [25] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel WH Kwok, Lai Guan Ng, Florent Gehlhof, and Evan W Newell. Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology*, 37(1):38–44, 2019.
- [26] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14, 2001.
- [27] Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualizing large-scale and high-dimensional data. In *Proceedings of the 25th international conference on world wide web*, pages 287–297, 2016.
- [28] David I Spivak. Metric realization of fuzzy simplicial sets. *Preprint*, page 4, 2009.
- [29] Fabian Sievers, Andreas Wilm, David Dineen, Toby J Gibson, Kevin Karplus, Weizhong Li, Rodrigo Lopez, Hamish McWilliam, Michael Remmert, Johannes Söding, et al. Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. *Molecular systems biology*, 7(1):539, 2011.
- [30] Michael Levandowsky and David Winter. Distance between sets. *Nature*, 234(5323):34–35, 1971.
- [31] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-20). 1996.
- [32] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.
- [33] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [34] D Ulyanov. Multicore-tsne <https://github.com/dmitryulyanov>. *Multicore-TSNE2016*, 2016.
- [35] Bette Korber, Will M Fischer, Sandrasegaram Gnanakaran, Hyejin Yoon, James Theiler, Werner Abfalterer, Nick Hengartner, Elena E Giorgi, Tanmoy Bhattacharya, Brian Foley, et al. Tracking changes in sars-cov-2 spike: evidence that d614g increases infectivity of the covid-19 virus. *Cell*, 182(4):812–827, 2020.
- [36] Joseph Minhow Chan, Gunnar Carlsson, and Raul Rabadan. Topology of viral evolution. *Proceedings of the National Academy of Sciences*, 110(46):18566–18571, 2013.
- [37] Michael Bleher, Lukas Hahn, Maximilian Neumann, Juan Angel Patino-Galindo, Mathieu Carriere, Ulrich Bauer, Raul Rabadan, and Andreas Ott. Topological data analysis identifies

emerging adaptive mutations in sars-cov-2. *arXiv preprint arXiv:2106.07292*, 2021.

- [38] Dong Quan Ngoc Nguyen, Phuong Dong Tan Le, Lin Xing, and Lizhen Lin. A topological characterization of dna sequences based on chaos geometry and persistent homology. In *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1591–1597. IEEE, 2022.
- [39] Christopher Tralie, Nathaniel Saul, and Rann Bar-On. Ripser.py: A lean persistent homology library for python. *The Journal of Open Source Software*, 3(29):925, Sep 2018.
- [40] Clément Maria, Jean-Daniel Boissonnat, Marc Glisse, and Mariette Yvinec. The gudhi library: Simplicial complexes and persistent homology. In *Mathematical Software–ICMS 2014: 4th International Congress, Seoul, South Korea, August 5-9, 2014. Proceedings 4*, pages 167–174. Springer, 2014.
- [41] Nan Sun, Shaojun Pei, Lily He, Changchuan Yin, Rong Lucy He, and Stephen S-T Yau. Geometric construction of viral genome space and its applications. *Computational and Structural Biotechnology Journal*, 19:4226–4234, 2021.
- [42] Ji Qi, Bin Wang, and Bai-Iin Hao. Whole proteome prokaryote phylogeny without sequence alignment: Ak-string composition approach. *Journal of molecular evolution*, 58:1–11, 2004.
- [43] Ajay Kumar Saw, Garima Raj, Manashi Das, Narayan Chandra Talukdar, Binod Chandra Tripathy, and Soumyadeep Nandi. Alignment-free method for dna sequence clustering using fuzzy integral similarity. *Scientific reports*, 9(1):3753, 2019.
- [44] Ann C Palmenberg, David Spiro, Ryan Kuzmickas, Shiliang Wang, Appolinaire Djikeng, Jennifer A Rathe, Claire M Fraser-Liggett, and Stephen B Liggett. Sequencing and analyses of all known human rhinovirus genomes reveal structure and evolution. *Science*, 324(5923):55–59, 2009.
- [45] Mo Deng, Chenglong Yu, Qian Liang, Rong L He, and Stephen S-T Yau. A novel method of characterizing genetic sequences: genome space with biological distance and applications. *PloS one*, 6(3):e17293, 2011.
- [46] Tung Hoang, Changchuan Yin, Hui Zheng, Chenglong Yu, Rong Lucy He, and Stephen S-T Yau. A new method to cluster dna sequences using fourier power spectrum. *Journal of theoretical biology*, 372:135–145, 2015.
- [47] Patrick CY Woo, Susanna KP Lau, Chung-ming Chu, Kwok-hung Chan, Hoi-wah Tsoi, Yi Huang, Beatrice HL Wong, Rosana WS Poon, James J Cai, Wei-kwang Luk, et al. Characterization and complete genome sequence of a novel coronavirus, coronavirus hku1, from patients with pneumonia. *Journal of virology*, 79(2):884–895, 2005.
- [48] Chenglong Yu, Qian Liang, Changchuan Yin, Rong L He, and Stephen S-T Yau. A novel

- construction of genome space with biological geometry. *DNA research*, 17(3):155–168, 2010.
- [49] Lia Van Der Hoek, Krzysztof Pyrc, Maarten F Jebbink, Wilma Vermeulen-Oost, Ron JM Berkhout, Katja C Wolthers, Pauline ME Wertheim-van Dillen, Jos Kaandorp, Joke Spaargaren, and Ben Berkhout. Identification of a new human coronavirus. *Nature medicine*, 10(4):368–373, 2004.
- [50] Paul S Masters. The molecular biology of coronaviruses. *Advances in virus research*, 66:193–292, 2006.
- [51] Kristian G Andersen, Andrew Rambaut, W Ian Lipkin, Edward C Holmes, and Robert F Garry. The proximal origin of sars-cov-2. *Nature medicine*, 26(4):450–452, 2020.
- [52] Robert G Webster, William J Bean, Owen T Gorman, Thomas M Chambers, and Yoshihiro Kawaoka. Evolution and ecology of influenza a viruses. *Microbiological reviews*, 56(1):152–179, 1992.
- [53] Dennis J Alexander. A review of avian influenza in different bird species. *Veterinary microbiology*, 74(1-2):3–13, 2000.

## APPENDIX 3A

### ADDITIONAL MATERIALS FOR UMAP-ASSISTED *K*-MEANS CLUSTERING OF LARGE-SCALE SARS-COV-2 MUTATION DATASETS

#### 3A.1 Cluster Stability

The *K*-means clustering is used to classify the SARS-CoV-2 the SNP variants. The Elbow method is used to determine the optimal number of clusters. Our results demonstrate four main clusters as shown in Figure S1, which plots the within-cluster sum of squares according to the number of clusters  $k$  for the SNP variants based on Jaccard distance metric. The optimal values of *K*-mean clusters is shown as the turning point in the in the elbow plots.

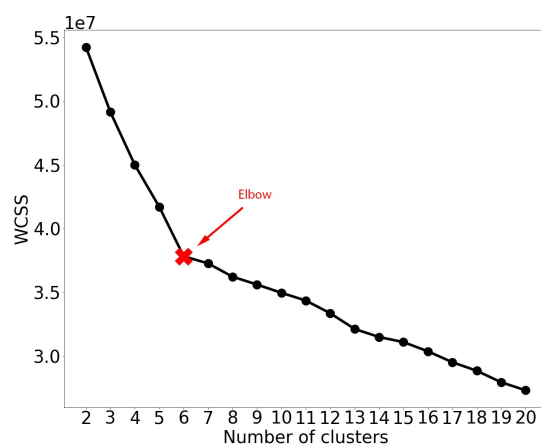


Figure 3A.1 Within cluster sum of squares (WCSS) of the UMAP-assisted *k*-means result of SARS-CoV-2 collected up to January 20, 2021. Using elbow method, the number of cluster was determined to be 6.

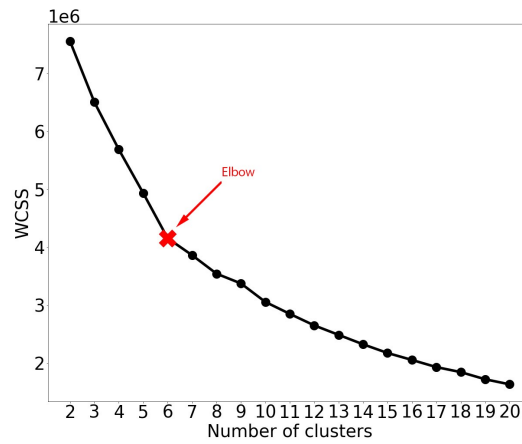


Figure 3A.2 Within cluster sum of squares (WCSS) of the UMAP-assisted  $k$ -means result of US SARS-CoV-2 collected up to January 20, 2021. Using elbow method, the number of cluster was determined to be 6.

### 3A.2 Clusters distribution

Table 3A.1 shows the top 25 mutations of each cluster from the dataset collected up to January 20, 2021.

Table 3A.1 Clusters distribution of the top 25 single mutations of SARS-CoV-2 in the world, collected up to January 20, 2021.

Top	Position	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
Top 1	23403	23395	22649	3450	14230	125155	3488
Top 2	14408	23381	22687	3450	14181	125032	3488
Top 3	3037	23358	22631	3450	14174	125052	3481
Top 4	241	23132	22395	3064	13955	124492	3477
Top 5	26801	62	76	24	33	57139	22
Top 6	22227	33	32	0	27	57097	20
Top 7	6286	17	15	1	24	56961	16
Top 8	21255	82	57	1	11	56844	13
Top 9	29645	26	13	1	11	56792	15
Top 10	28932	7	18	1	26	56731	10
Top 11	445	9	1	0	2	56754	12
Top 12	28881	23418	896	3450	92	28396	12
Top 13	28882	23413	838	3450	21	28268	15
Top 14	28883	23435	816	3450	16	28274	13
Top 15	25563	112	21663	0	34	22757	3475
Top 16	27944	19	18	0	3	38443	3
Top 17	204	37	27	6	8	32527	3
Top 18	1059	60	17219	0	33	13045	2
Top 19	21614	106	26	0	150	25209	20
Top 20	20268	19	131	0	6487	9379	1
Top 21	22992	87	34	3449	37	7546	3477
Top 22	18877	42	2499	0	11	7393	3480
Top 23	28854	72	386	0	3772	9147	2
Top 24	11083	611	738	40	414	10985	103
Top 25	26735	35	977	0	52	7317	3468

Table 3A.2 shows cluster distribution among the top 25 available sequence from each country.



Table 3A.2 Cluster distributions of SARS-CoV-2 sequences from top 25 countries with the highest number of sequences as of October 30, 2020. The top 25 countries are the United Kingdom (UK), the United States (US), Australia (AU), India (IN), Switzerland (CH), Netherlands (NL), Canada (CA), France (FR), Belgium (BE), Singapore (SG), Spain (ES), Russia (RU), Portugal (PT), Denmark (DK), Sweden (SE), Austria (AT), Japan (JP), South Africa (ZA), Iceland (IS), Brazil (BR), Saudi Arabia (SA), Norway (NO), China (CN), Italy (IT), and Korea (KR).

Top	Country	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
Top 1	UK	8200	1291	0	3385	65107	1030
Top 2	US	3691	15042	0	3760	20897	5
Top 3	DK	954	1256	0	215	20177	317
Top 4	AU	466	397	3449	280	5001	9
Top 5	NL	368	123	0	356	2973	238
Top 6	CA	258	728	0	496	1993	3
Top 7	CH	357	90	0	656	1669	425
Top 8	IS	112	79	0	156	2753	19
Top 9	IN	1168	154	0	568	815	0
Top 10	FR	175	592	0	505	627	439
Top 11	BE	473	113	0	361	1099	164
Top 12	ES	139	20	0	598	1161	15
Top 13	DE	421	361	0	329	757	40
Top 14	LU	97	43	0	110	1130	512
Top 15	IT	307	25	0	279	889	27
Top 16	SG	152	72	0	44	972	1
Top 17	SE	304	243	0	113	394	18
Top 18	AE	718	27	0	82	240	0
Top 19	BR	456	12	0	36	508	0
Top 20	RU	845	27	0	61	21	0
Top 21	NO	82	65	0	50	683	62
Top 22	CL	153	144	0	131	461	0
Top 23	ZA	567	2	0	170	96	0
Top 24	FI	49	198	0	47	414	61
Top 25	PT	388	31	0	118	225	1

Table 3A.3 shows the top 25 mutation of each cluster in the US dataset, collected up to January 20, 2021.

Table 3A.3 Clusters distribution of the top 25 single mutations of SARS-CoV-2 in the United States, collected up to January 20, 2021.

Top	Position	Cluster A	Cluster B	Cluster C	Cluster D	Cluster E	Cluster F
Top 1	23403	9438	20621	4677	4890	3	635
Top 2	14408	9445	20609	4661	4889	3	634
Top 3	3037	9416	20577	4666	4888	2	630
Top 4	241	9345	20498	4594	4762	3	630
Top 5	25563	9432	15710	33	44	0	635
Top 6	1059	8030	15160	21	36	1	634
Top 7	27964	3	10205	8	9	1	0
Top 8	10319	7	7790	13	11	0	0
Top 9	21304	13	6333	19	13	0	0
Top 10	18424	3	6321	5	7	0	0
Top 11	28869	7	6251	6	22	0	0
Top 12	20268	4	2369	3	3785	0	0
Top 13	25907	10	6133	8	7	0	0
Top 14	28472	8	6086	10	16	0	1
Top 15	28854	62	2585	12	3440	0	6
Top 16	28881	20	1207	4644	35	1	2
Top 17	28882	10	1207	4641	5	0	1
Top 18	28883	2	1193	4656	2	0	1
Top 19	14805	4	2857	32	4	16	0
Top 20	8083	1	2527	0	1	1	3
Top 21	8782	2	768	12	0	1575	0
Top 22	28144	5	749	3	3	1572	0
Top 23	24076	0	320	2	1759	0	0
Top 24	18060	7	271	10	40	1551	0
Top 25	17747	12	311	13	8	1498	0

Table 3A.3 shows the cluster statistics for each states with more than 50 SARS-CoV-2 genome isolates.

Table 3A.4 Cluster statistics for states with more than 50 SARS-CoV-2 genome samples.

State	Cluster A	Cluster B	Cluster C	Cluster D	Cluster E	Cluster F
Michigan	709	5884	968	527	3	25
Washington	1275	1961	943	1962	216	100
Virginia	874	1853	1084	169	921	8
Maryland	1790	1525	275	433	37	341
Alaska	477	2034	293	279	12	5
Idaho	389	1053	50	152	121	3
New Jersey	45	1564	45	36	1	0
Kentucky	783	518	66	64	7	3
Wyoming	546	544	146	39	16	59
Arizona	231	678	50	205	47	3
Arkansas	343	569	168	46	16	5
Tennessee	120	432	82	204	15	0
Pennsylvania	339	346	54	68	10	9
Missouri	181	377	63	49	1	2
California	36	210	18	391	4	0
Indiana	232	157	35	59	30	1
Florida	80	288	26	54	7	0
New Mexico	105	201	44	40	13	9
Maine	131	181	26	52	2	0
Nevada	53	211	8	29	11	0
Iowa	63	158	24	47	10	2
Minnesota	25	213	11	4	5	0
Colorado	36	76	48	2	43	0
Kansas	82	75	18	5	4	18
Nebraska	110	22	11	17	17	2
New York	22	70	47	10	0	27
Oregon	72	77	13	5	2	0
Vermont	13	129	0	4	7	0
South Carolina	67	16	5	5	1	8
Montana	17	43	7	34	0	0
South Dakota	32	39	0	9	0	2
Alabama	5	72	0	3	0	0
Georgia	19	44	9	2	2	3
District of Columbia	9	23	13	21	5	0
Utah	22	29	15	3	0	0
Massachusetts	33	31	0	3	0	0
Oklahoma	31	26	3	1	1	0
North Carolina	8	37	2	12	0	0
Connecticut	10	30	1	1	1	0

Table 3A.5 shows the world wide cluster statistics from SARS-CoV-2 genome collected up to June 01, 2020, which is our previous result from [9]. The listed countries are the United States

(US), Canada (CA), Australia (AU), United Kingdom (UK), Germany (DE), France (FR), Italy (IT), Russia (RU), China (CN), Japan (JP), Korean (KR), India (IN), Spain (ES), Saudi Arabia (SA), and Turkey (TR).

Table 3A.5 The world wide clusters from SARS-CoV-2 genome data available up to June 01, 2020. The listed countries are the United States (US), Canada (CA), Australia (AU), United Kingdom (UK), Germany (DE), France (FR), Italy (IT), Russia (RU), China (CN), Japan (JP), Korean (KR), India (IN), Spain (ES), Saudi Arabia (SA), and Turkey (TR) [9].

Country	Cluster I	Cluster II	Cluster III	Cluster IV	Cluster V	Cluster VI
US	844	311	488	156	1813	975
CA	12	29	17	16	19	41
AU	163	149	410	135	146	77
UK	539	875	908	1532	119	3
DE	10	20	21	38	42	0
FR	41	85	14	12	82	0
IT	26	24	9	17	0	0
RU	10	27	1	109	3	0
CN	8	3	215	1	1	25
JP	0	3	68	20	3	0
KR	0	0	28 0	0	0	0
IN	93	69	141	10	3	0
ES	27	100	74	25	3	2
SA	14	31	9	1	2	0
TR	25	3	24	9	0	0

PCA was used to reduce the same dataset as Table 3A.5 by reduction ratio of 1/160. Table 3A.6 shows the world-wide cluster statistics of the PCA-assisted  $K$ -means.  $K = 6$  was used to compare with the original result from [9].

Table 3A.6 The world wide clusters from SARS-CoV-2 genome data available up to June 01, 2020 using PCA embedding with reduction ratio of 1/160.

Country	Cluster $I_p$	Cluster $II_p$	Cluster $III_p$	Cluster $IV_p$	Cluster $V_p$	Cluster $VI_p$
US	915	489	239	156	1813	975
CA	14	17	27	16	19	41
AU	164	414	143	136	146	77
UK	543	908	857	1546	119	3
DE	10	21	20	38	42	0
FR	46	14	80	12	82	0
IT	26	9	24	17	0	0
RU	10	1	27	109	3	0
CN	8	213	3	1	1	24
JP	0	68	3	20	3	0
KR	0	28	0	0	0	0
IN	95	141	67	10	3	0
ES	27	74	100	25	3	2
SA	30	9	15	1	2	0
TR	27	24	1	9	0	0

UMAP was used to reduce the same dataset as Table 3A.5 by reduction ratio of 1/160. Table 3A.7 shows the world-wide cluster statistics of the UMAP-assisted  $K$ -means.  $K = 6$  was used to compare with the original result from [9].

Table 3A.7 The world wide clusters from SARS-CoV-2 genome data available up to June 01, 2020 using UMAP embedding with reduction ratio of 1/160.

Country	Cluster $I_u$	Cluster $II_u$	Cluster $III_u$	Cluster $IV_u$	Cluster $V_u$	Cluster $VI_u$
US	2446	1096	90	751	110	94
CA	71	15	9	35	1	3
AU	784	94	64	18	83	37
UK	2171	115	828	2	534	326
DE	57	40	14	0	5	15
FR	163	45	10	0	11	5
IT	13	1	35	0	5	22
RU	92	2	49	0	0	7
CN	178	28	6	10	22	6
JP	36	0	11	0	47	0
KR	18	0	0	1	9	0
IN	232	3	7	0	2	72
ES	205	2	12	0	7	5
SA	56	0	1	0	0	0
TR	56	1	4	0	0	0

## APPENDIX 3B

### ADDITIONAL MATERIALS FOR K-MERS TOPOLOGY FOR ALIGNMENT-FREE SEQUENCE ANALYSIS

The statistics of each dataset used for the phylogenetic is listed below.

- Table 3B.1: Ebolavirus data
- Table 3B.2: Mammalian mitochondria data
- Table 3B.3: Rhinovirus data
- Table 3B.4: Coronavirus data
- Table 3B.5: Influenza type A data
- Table 3B.6: Bacterial 16S rDNA isolates data
- Table 3B.7: Bacteria whole genome data.

Table 3B.1 Accession, group, sequence length of ebolavirus data.

Accession	Group	$N_A$	$N_C$	$N_G$	$N_T$	Accession	Group	$N_A$	$N_C$	$N_G$	$N_T$
FJ217161.1	BDBV	5964	4324	3632	5020	KC545393.1	BDBV	5974	4293	3636	5036
KC545395.1	BDBV	5975	4290	3635	5039	KC545394.1	BDBV	5974	4292	3637	5036
KC545396.1	BDBV	5975	4293	3635	5036	FJ217162.1	TAFV	6020	4371	3630	4914
AF522874.1	RESTV	5937	3929	3746	5279	AB050936.1	RESTV	5927	3924	3762	5277
JX477166.1	RESTV	5935	3920	3755	5281	FJ621585.1	RESTV	5900	3898	3747	5251
FJ621583.1	RESTV	5928	3929	3767	5263	JX477165.1	RESTV	5924	3929	3774	5260
FJ968794.1	SUDV	5905	4034	3756	5180	KC242783.2	SUDV	5911	4028	3750	5186
EU338380.1	SUDV	5914	4032	3750	5179	AY729654.1	SUDV	5920	4071	3732	5152
JN638998.1	SUDV	5931	4059	3729	5156	KC545389.1	SUDV	5924	4080	3731	5139
KC545390.1	SUDV	5925	4080	3730	5139	KC545391.1	SUDV	5924	4080	3731	5139
KC545392.1	SUDV	5923	4081	3732	5138	KC589025.1	SUDV	5921	4047	3734	5173
KC242801.1	EBOV	6061	4037	3752	5109	NC_002549.1	EBOV	6061	4035	3752	5111
KC242791.1	EBOV	6061	4037	3752	5109	KC242792.1	EBOV	6047	4052	3756	5104
KC242793.1	EBOV	6043	4052	3761	5102	KC242794.1	EBOV	6039	4063	3762	5095
AY354458.1	EBOV	6054	4051	3747	5109	KC242796.1	EBOV	6055	4049	3748	5107
KC242799.1	EBOV	6054	4050	3748	5107	KC242784.1	EBOV	6061	4028	3750	5119
KC242786.1	EBOV	6063	4025	3749	5121	KC242787.1	EBOV	6062	4025	3750	5121
KC242789.1	EBOV	6062	4023	3750	5123	KC242785.1	EBOV	6060	4026	3752	5120
KC242790.1	EBOV	6060	4025	3751	5122	KC242788.1	EBOV	6063	4032	3749	5114
KC242800.1	EBOV	6042	4052	3762	5102	KM034555.1	EBOV	6049	4049	3753	5099
KM034562.1	EBOV	6051	4050	3756	5100	KM233039.1	EBOV	6052	4051	3751	5099
KM034557.1	EBOV	6052	4052	3753	5099	KM034560.1	EBOV	6050	4051	3752	5099
KM233050.1	EBOV	6053	4050	3753	5100	KM233053.1	EBOV	6053	4051	3753	5100
KM233057.1	EBOV	6052	4052	3751	5099	KM233063.1	EBOV	6053	4052	3751	5099
KM233072.1	EBOV	6049	4049	3752	5099	KM233110.1	EBOV	6053	4053	3752	5098
KM233070.1	EBOV	6055	4052	3753	5099	KM233099.1	EBOV	6052	4052	3751	5098
KM233097.1	EBOV	6051	4052	3752	5098	KM233109.1	EBOV	6053	4055	3753	5097
KM233096.1	EBOV	6054	4049	3751	5099	KM233103.1	EBOV	6051	4050	3751	5098
KJ660346.2	EBOV	6053	4052	3755	5099	KJ660347.2	EBOV	6054	4052	3754	5099
KJ660348.2	EBOV	6055	4051	3754	5099						

Table 3B.2 Accession, group, sequence length of mammalian mitochondria data.

Accession	Groups	Length	$N_A$	$N_C$	$N_G$	$N_T$
V00662.1	Primates	16569	5123	5176	2176	4094
D38116.1	Primates	16563	5189	5084	2104	4186
D38113.1	Primates	16554	5154	5099	2133	4168
D38114.1	Primates	16364	5059	5022	2160	4123
X99256.1	Primates	16472	5039	5231	2256	3946
Y18001.1	Primates	16521	5195	5047	2169	4110
AY863426.1	Primates	16389	5243	4953	2049	4137
D38115.1	Primates	16389	5007	5317	2168	3897
NC_002083.1	Primates	16499	5031	5403	2176	3889
NC_002764.1	Primates	16586	5306	5027	2116	4137
U20753.1	Carnivore	17009	5543	4454	2406	4606
U96639.2	Carnivore	16727	5290	4267	2366	4804
EU442884.2	Carnivore	16774	5293	4265	2398	4812
EF551003.1	Carnivore	16990	5418	4513	2478	4581
EF551002.1	Carnivore	16964	5397	4508	2467	4592
DQ402478.1	Carnivore	16868	5270	4285	2601	4712
AF303110.1	Carnivore	17020	5258	4355	2676	4731
AF303111.1	Carnivore	17017	5253	4346	2692	4726
EF212882.1	Carnivore	16805	5338	4000	2518	4949
AJ002189.1	Artiodactyla	16680	5790	4384	2210	4296
AF010406.1	Artiodactyla	16616	5594	4289	2181	4552
AF533441.1	Artiodactyla	16640	5569	4313	2189	4569
V00654.1	Artiodactyla	16338	5460	4237	2198	4443
AY488491.1	Artiodactyla	16355	5421	4298	2261	4375
NC_007441.1	Artiodactyla	16498	5542	4358	2164	4434
NC_008830.1	Artiodactyla	16719	5786	4340	2222	4371
NC_010640.1	Artiodactyla	16524	5519	4404	2205	4396
X72204.1	Cetacea	16402	5374	4527	2140	4361
NC_005268.1	Cetacea	16390	5354	4609	2162	4265
NC_001321.1	Cetacea	16398	5359	4474	2182	4383
NC_005270.1	Cetacea	16412	5374	4626	2153	4259
NC_005275.1	Cetacea	16324	5377	4525	2040	4382
NC_006931.1	Cetacea	16386	5357	4573	2164	4292
NC_001788.1	Perissodactyla	16670	5394	4819	2198	4259
X97336.1	Perissodactyla	16829	5663	4630	2131	4405
Y07726.1	Perissodactyla	16832	5623	4707	2169	4333
NC_001640.1	Perissodactyla	16660	5358	4754	2236	4312
AJ238588.1	Rodentia	16507	5301	4041	2071	5094
AJ001562.1	Rodentia	16602	5386	3913	2096	5207
AJ001588.1	Lagomorpha	17245	5429	4584	2350	4882
X88898.2	Erinaceomorpha	17447	5937	3503	2185	5822

Table 3B.3 Accession, group, sequence length of rhinovirus data.

Accession	Group	$N_A$	$N_C$	$N_G$	$N_T$	Accession	Group	$N_A$	$N_C$	$N_G$	$N_T$
AF499637.1	HEV	2243	1677	1662	1876	AF546702.1	HEV	2209	1649	1681	1867
AY751783.1	A	2348	1362	1428	1999	DQ473485.1	B	2338	1436	1443	1991
DQ473486.1	B	2381	1428	1431	1976	DQ473488.1	B	2382	1456	1447	1929
DQ473489.1	B	2365	1476	1463	1919	DQ473490.1	B	2362	1417	1420	2013
DQ473491.1	A	2301	1383	1454	2007	DQ473492.1	A	2297	1371	1443	2029
DQ473493.1	A	2370	1287	1442	2035	DQ473494.1	A	2389	1314	1437	1980
DQ473496.1	A	2363	1342	1389	2012	DQ473497.1	A	2309	1322	1396	1998
DQ473499.1	A	2333	1301	1427	2062	DQ473500.1	A	2336	1344	1409	2046
DQ473504.1	A	2253	1380	1402	2108	DQ473505.1	A	2247	1404	1409	2081
DQ473506.1	A	2415	1349	1412	1972	DQ473507.1	A	2407	1350	1426	1960
DQ473508.1	A	2371	1389	1390	1998	DQ473510.1	A	2406	1313	1424	1994
DQ473511.1	A	2367	1288	1386	1995	EF077279.1	C	2176	1503	1509	1756
EF077280.1	C	2153	1550	1500	1812	EF173414.1	A	2369	1310	1403	2043
EF173415.1	A	2299	1396	1416	2013	EF173420.1	B	2356	1500	1477	1886
EF173423.1	B	2384	1456	1407	1969	EF173425.1	B	2426	1412	1432	1944
EF186077.2	C	2296	1549	1520	1769	EF582385.1	C	2195	1565	1473	1866
EF582386.1	C	2304	1492	1480	1838	EF582387.1	C	2261	1533	1513	1779
FJ445111.1	A	2388	1284	1388	2074	FJ445112.1	B	2402	1391	1417	2002
FJ445113.1	A	2376	1387	1407	1938	FJ445114.1	A	2375	1329	1428	2002
FJ445115.1	A	2377	1321	1426	2009	FJ445116.1	A	2298	1356	1437	2049
FJ445117.1	A	2321	1352	1421	2049	FJ445118.1	A	2386	1340	1423	1970
FJ445119.1	A	2354	1310	1421	2048	FJ445121.1	A	2376	1328	1403	2026
FJ445122.1	A	2327	1369	1445	1971	FJ445123.1	A	2358	1277	1399	2091
FJ445124.1	B	2403	1397	1413	1996	FJ445125.1	A	2332	1299	1416	2075
FJ445126.1	A	2354	1292	1414	2069	FJ445127.1	A	2375	1287	1408	2062
FJ445128.1	A	2336	1347	1429	2019	FJ445129.1	A	2380	1311	1391	2056
FJ445130.1	B	2405	1423	1405	1989	FJ445131.1	A	2397	1275	1436	2019
FJ445132.1	A	2352	1406	1424	1932	FJ445133.1	A	2342	1275	1439	2074
FJ445134.1	A	2360	1364	1404	1981	FJ445135.1	A	2371	1328	1395	2022
FJ445136.1	A	2349	1350	1425	2025	FJ445137.1	B	2334	1517	1473	1891
FJ445138.1	A	2353	1326	1418	2036	FJ445139.1	A	2364	1327	1413	2029
FJ445140.1	A	2342	1323	1381	2088	FJ445141.1	A	2414	1287	1401	2031
FJ445142.1	A	2233	1363	1415	2128	FJ445143.1	A	2390	1315	1391	2042
FJ445144.1	A	2340	1361	1416	2022	FJ445145.1	A	2371	1270	1391	2095
FJ445146.1	A	2320	1352	1411	2058	FJ445147.1	A	2353	1383	1429	1997
FJ445148.1	A	2390	1339	1393	2016	FJ445149.1	A	2377	1325	1435	1998
FJ445151.1	B	2316	1499	1504	1890	FJ445152.1	A	2375	1362	1427	1997
FJ445153.1	B	2372	1461	1452	1931	FJ445154.1	A	2368	1281	1406	2077
FJ445155.1	B	2369	1421	1433	2001	FJ445156.1	A	2389	1341	1421	1984
FJ445157.1	A	2357	1339	1406	2011	FJ445158.1	A	2336	1333	1434	2013
FJ445159.1	A	2331	1334	1437	2014	FJ445160.1	A	2325	1325	1454	2019
FJ445161.1	B	2382	1376	1460	2011	FJ445162.1	B	2392	1420	1410	1977
FJ445163.1	A	2355	1334	1412	2039	FJ445164.1	B	2388	1374	1420	2028
FJ445165.1	A	2235	1388	1416	2111	FJ445166.1	A	2227	1387	1425	2113
FJ445167.1	A	2394	1295	1409	2025	FJ445168.1	B	2376	1515	1450	1878
FJ445169.1	B	2338	1430	1463	2002	FJ445170.1	A	2371	1382	1413	1944
FJ445171.1	A	2316	1344	1446	2026	FJ445172.1	B	2403	1410	1418	1975
FJ445173.1	A	2387	1299	1381	2066	FJ445174.1	B	2404	1400	1384	2018
FJ445175.1	A	2319	1317	1433	2071	FJ445176.1	A	2284	1339	1382	2140
FJ445177.1	A	2347	1309	1449	2021	FJ445178.1	A	2331	1369	1420	2017
FJ445179.1	A	2318	1336	1417	2022	FJ445180.1	A	2387	1314	1425	2010
FJ445181.1	A	2333	1379	1441	1972	FJ445182.1	A	2334	1347	1420	2027
FJ445183.1	A	2356	1379	1429	1981	FJ445184.1	A	2241	1381	1406	2121
FJ445185.1	A	2344	1384	1444	1957	FJ445186.1	B	2377	1401	1447	1992
FJ445187.1	B	2372	1411	1466	1971	FJ445188.1	B	2322	1497	1490	1907
FJ445189.1	A	2370	1328	1408	2011	FJ445190.1	A	2392	1307	1389	2044
L05355.1	B	2313	1460	1475	1964	L24917.1	A	2383	1331	1412	1998
V01149.1	HEV	2206	1737	1711	1786	X02316.1	A	2324	1347	1418	2013



Table 3B.4 Accession, group, sequence length of coronavirus data.

Accession	Group	Length	$N_A$	$N_C$	$N_G$	$N_T$
AF304460.1	Group 1	27317	7420	4549	5903	9445
AF353511.1	Group 1	28033	6937	5382	6397	9317
NC_005831.2	Group 1	27553	7253	3979	5516	10805
AY391777.1	Group 2	30738	8485	4658	6655	10940
U00735.2	Group 2	31032	8490	4713	6774	11055
AF391542.1	Group 2	31028	8486	4743	6772	11027
AF220295.1	Group 2	31100	8544	4711	6790	11055
NC_003045.1	Group 2	31028	8487	4752	6767	11022
AF208067.1	Group 2	31233	8087	5591	7466	10089
AF201929.1	Group 2	31276	8117	5548	7422	10189
AF208066.1	Group 2	31112	8030	5534	7416	10132
NC_001846.1	Group 2	31357	8138	5614	7487	10118
NC_001451.1	Group 3	27608	7967	4479	5993	9169
EU095850.1	Group 3	27657	7969	4513	6066	9108
AY278488.2	Group 4	29725	8465	5941	6185	9134
AY278741.1	Group 4	29727	8455	5940	6188	9144
AY278491.2	Group 4	29742	8475	5942	6183	9142
AY278554.2	Group 4	29736	8476	5942	6185	9133
AY282752.2	Group 4	29736	8476	5939	6185	9136
AY283794.1	Group 4	29711	8453	5937	6184	9137
AY283795.1	Group 4	29705	8447	5936	6187	9135
AY283796.1	Group 4	29711	8453	5936	6185	9137
AY283797.1	Group 4	29706	8451	5935	6184	9135
AY283798.2	Group 4	29711	8453	5935	6185	9138
AY291451.1	Group 4	29729	8457	5940	6188	9144
NC_004718.3	Group 4	29751	8481	5940	6187	9143
AY297028.1	Group 4	29715	8458	5934	6187	9135
AY572034.1	Group 4	29540	8402	5911	6154	9073
AY572035.1	Group 4	29518	8395	5907	6151	9065
NC_006577.2	Group 5	29926	8331	3895	5699	12001
NC_001564.2	Flaviviridae outgroup	10682	2618	2531	2919	2614
NC_004102.1	Flaviviridae outgroup	9646	1889	2893	2724	2140
NC_001512.1	Togaviridae outgroup	11835	3676	2860	2859	2440
NC_001544.1	Togaviridae outgroup	11657	3220	2901	3065	2416

Table 3B.5 Accession, group, sequence length of influenza A virus data.

Accession	Group	Length	$N_A$	$N_C$	$N_G$	$N_T$
HM370969.1	H1N1	1419	453	263	330	373
CY138562.1	H1N1	1422	437	259	343	383
CY149630.1	H1N1	1433	441	261	346	385
KC608160.1	H1N1	1398	409	251	357	381
AM157358.1	H1N1	1413	418	259	355	381
AB470663.1	H1N1	1422	418	252	359	393
AB546159.1	H1N1	1410	421	260	351	378
HQ897966.1	H1N1	1410	422	246	353	389
EU026046.2	H1N1	1433	439	263	347	384
FJ357114.1	H1N1	1433	438	253	350	392
GQ411894.1	H1N1	1413	430	260	346	376
CY140047.1	H1N1	1433	440	261	347	385
KM244078.1	H1N1	1410	447	261	335	367
HQ185381.1	H5N1	1350	406	240	339	365
HQ185383.1	H5N1	1350	408	240	336	366
EU635875.1	H5N1	1350	397	248	347	358
FM177121.1	H5N1	1370	407	245	350	368
AM914017.1	H5N1	1350	398	243	344	365
KF572435.1	H5N1	1350	403	247	345	355
AF509102.2	H5N1	1366	401	257	344	364
AB684161.1	H5N1	1350	404	235	348	363
EF541464.1	H5N1	1350	396	246	349	359
JF699677.1	H5N1	1350	404	236	348	362
GU186511.1	H5N1	1370	407	244	345	374
EU500854.1	H7N3	1453	475	284	339	355
CY129336.1	H7N3	1428	470	278	332	348
CY076231.1	H7N3	1420	467	286	327	340
CY039321.1	H7N3	1434	470	288	333	343
AY646080.1	H7N3	1453	485	284	329	355
KF259734.1	H7N9	1398	478	290	321	309
KF938945.1	H7N9	1404	483	287	322	312
KF259688.1	H7N9	1413	490	291	320	312
KC609801.1	H7N9	1426	488	292	332	314
CY014788.1	H7N9	1460	500	306	337	317
CY186004.1	H7N9	1422	494	303	317	308
DQ017487.1	H2N2	1467	445	281	355	386
CY005540.1	H2N2	1467	455	284	344	384
JX081142.1	H2N2	1457	446	265	349	397

Table 3B.6 Accession, group, sequence length of bacteria 16S rDNA data.

Accession	Family	Length	$N_A$	$N_C$	$N_G$	$N_T$
KY486204.1	Methylobacteriaceae	1104	263	265	345	230
KY486205.1	Xanthomonadaceae	761	194	165	256	146
KY486206.1	Xanthomonadaceae	1452	361	340	464	287
KY486207.1	Intrasporangiaceae	1253	301	295	395	262
KY486218.1	Microbacteriaceae	1195	296	286	372	241
KY486219.1	Pseudomonadaceae	1099	282	250	339	228
KY927407.1	Bacillaceae	718	179	179	206	154
KY486220.1	Paenibacillaceae	1335	327	326	419	263
KY486221.1	Enterobacteriaceae	1339	335	314	430	260
KY486222.1	Xanthomonadaceae	1337	335	313	422	267
KY486223.1	Microbacteriaceae	1334	317	314	435	268
KY486209.1	Rhodanobacteraceae	1366	332	319	447	268
KY486210.1	Enterobacteriaceae	1356	338	324	431	262
KY486232.1	Enterobacteriaceae	1350	337	318	432	262
KY019246.1	Enterobacteriaceae	1346	335	318	431	262
KY013009.1	Enterobacteriaceae	1351	337	318	434	262
KY927404.1	Microbacteriaceae	742	184	177	232	149
KY486211.1	Enterobacteriaceae	1365	337	325	441	262
KY013011.1	Staphylococcaceae	1035	278	226	299	231
KY019245.1	Enterobacteriaceae	1344	334	317	430	262
KY013010.1	Bacillaceae	1343	336	318	420	269
KY486208.1	Enterobacteriaceae	1269	318	298	400	253
KY486212.1	Microbacteriaceae	1364	341	324	436	263
KY486213.1	Xanthomonadaceae	1387	345	322	442	278
KY486228.1	Enterobacteriaceae	1356	337	325	429	265
KY486224.1	Enterobacteriaceae	1346	335	317	431	262
KY486225.1	Enterobacteriaceae	1294	329	312	401	251
KY486226.1	Enterobacteriaceae	1347	338	316	432	261
KY927405.1	Enterobacteriaceae	753	189	169	252	143
KY486227.1	Enterobacteriaceae	1345	336	317	431	261
KY927408.1	Microbacteriaceae	785	193	188	243	161
KY486214.1	Enterobacteriaceae	1411	352	330	455	274
KY927406.1	Microbacteriaceae	796	201	183	266	146
KY486215.1	Enterobacteriaceae	1319	338	309	417	255
KY486216.1	Enterobacteriaceae	1345	335	317	430	263
KY486217.1	Enterobacteriaceae	1341	335	315	431	260
KY486229.1	Enterobacteriaceae	1345	335	318	432	260
KY486230.1	Pseudomonadaceae	1346	345	306	416	279
KY486231.1	Enterobacteriaceae	1322	325	321	419	255
KY019244.1	Enterobacteriaceae	1337	331	315	428	260

Table 3B.7 Accession, group, sequence length of bacteria data.

Accession	Family	Length	$N_A$	$N_C$	$N_G$	$N_T$
CP001598.1	Bacillaceae	5227419	1685408	930043	919269	1692699
AE016879.1	Bacillaceae	5227293	1685374	930007	919244	1692668
CP001215.1	Bacillaceae	5230115	1667671	974191	876267	1711986
AE017225.1	Bacillaceae	5228663	1685622	930391	919481	1693169
CP000976.1	Borreliaceae	931674	335148	129225	127787	339514
CP000048.1	Borreliaceae	922307	321202	137556	137547	326002
CP000993.1	Borreliaceae	930981	335785	129350	126839	339007
CP000049.1	Borreliaceae	917330	322493	133424	133693	327720
CP000246.1	Clostridiaceae	3256683	1148078	470943	453276	1184386
CP000312.1	Clostridiaceae	2897393	1017083	423439	395046	1061825
BA000016.3	Clostridiaceae	3031430	1060154	446732	419228	1105316
CP000527.1	Desulfovibrionaceae	3462887	639427	1092219	1089813	641428
AE017285.1	Desulfovibrionaceae	3570858	659017	1127624	1127109	657108
CP002297.1	Desulfovibrionaceae	3532052	652227	1113805	1116142	649878
AM260480.1	Burkholderiaceae	2912490	486037	972789	972298	481366
CP000091.1	Burkholderiaceae	2726152	480326	883953	887595	474278
CP000578.1	Rhodobacteraceae	1219053	192966	416292	420323	189472
CP001151.1	Rhodobacteraceae	1297647	204455	445520	446045	201627
AM295250.1	Staphylococcaceae	2566424	833636	449856	438966	843965
AE015929.1	Staphylococcaceae	2499279	837991	405441	396707	859140
AP006716.1	Staphylococcaceae	2685015	907537	437414	443072	896992
CP001837.1	Staphylococcaceae	2658366	878689	445972	454367	879338
AL590842.1	Yersiniaceae	4653728	1219520	1102670	1114185	1217353
CP001585.1	Yersiniaceae	4640720	1216182	1097565	1112625	1214348
AE009952.1	Yersiniaceae	4600755	1200303	1090469	1101384	1208599
CP001593.1	Yersiniaceae	4553586	1187961	1077463	1093635	1194527
CP001671.1	Enterobacteriaceae	5131397	1271011	1298314	1297349	1264723
CP000468.1	Enterobacteriaceae	5082025	1256126	1285309	1283517	1256945
CP001383.1	Enterobacteriaceae	4650856	1145625	1187110	1177854	1140266
AE005674.2	Enterobacteriaceae	4607202	1133784	1176618	1167963	1128831

## CHAPTER 4

### RESIDUE-SIMILARITY SCORES AND INDEXES

Traditionally, data visualization involves reducing data into 2 or 3 feature components. However, aggressive reduction often leads to poor representations for data with high intrinsic dimensions, despite producing visually appealing results. For classification problems with 2 classes, the Receiver Operating Characteristic (ROC) curve and Area Under the ROC Curve (AUC) curve can effectively show performance. However, not all classification problems are binary. In this section, we introduce a new visualization tool called Residue-Similarity (R-S) scores or R-S plots. Unlike traditional methods, R-S plots can be applied to an arbitrary number of classes, providing a more comprehensive visualization solution.

#### 4.1 Methods

##### 4.1.1 Residue-Similarity score and indexes

An R-S plot consists of two components, residue and similarity scores. Assume that the data is  $\{(\mathbf{x}_m, y_m) | \mathbf{x}_m \in \mathbb{R}^N, y_m \in \mathbb{Z}_L\}_{m=1}^M$ , where  $\mathbf{x}_m$  is the  $m$ th data. For classification problems,  $y_m$  is the ground truth, and for clustering problems,  $y_m$  is the cluster label. Here,  $N$  is the number of features and  $M$  is the number of samples.  $L$  is the number of classes, that is  $y_m \in [0, 1, \dots, L-1]$ . We can partition  $\mathcal{X} = \{\mathbf{x}_m\}_{m=1}^M$  into  $L$  classes by taking  $C_l = \{\mathbf{x}_m \in \mathcal{X} | y_m = l\}$ . Note that  $\cup_{l=0}^{L-1} C_l = \mathcal{X}$ .

The residue score is defined as the inter-class sum of distances. Suppose  $y_m = l$ . Then, the residue score for  $\mathbf{x}_m$  is given by

$$R_m := R(\mathbf{x}_m) = \frac{1}{R_{\max}} \sum_{\mathbf{x}_j \notin C_l} \|\mathbf{x}_m - \mathbf{x}_j\|, \quad (4.1)$$

where  $\|\cdot\|$  is the distance between a pair of vectors and  $R_{\max} = \max_{\mathbf{x}_m \in \mathcal{X}} R(\mathbf{x}_m)$  is the maximal residue score. The similarity score is given by taking the average intra-class score. That is, for  $y_m = l$ ,

$$S_m := S(\mathbf{x}_m) = \frac{1}{|C_l|} \sum_{\mathbf{x}_j \in C_l} \left(1 - \frac{\|\mathbf{x}_m - \mathbf{x}_j\|}{d_{\max}}\right), \quad (4.2)$$

where  $d_{\max} = \max_{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}} \|\mathbf{x}_i - \mathbf{x}_j\|$  is the maximal pairwise distance of the dataset. Note that by scaling,  $0 \leq R(\mathbf{x}_m) \leq 1$  and  $0 \leq S(\mathbf{x}_m) \leq 1$  for all  $\mathbf{x}_m$ . In this work, we employ the Euclidean distance in

our R-S scores. However, other distance metrics can be similarly used as well. In general, a large  $R(\mathbf{x}_m)$  indicates that the data is far from other classes, and a large  $S(\mathbf{x}_m)$  indicates that the data is well clustered. Since  $R_{\max}$  and  $d_{\max}$  are for the whole dataset, residue and similarity scores in different classes can be compared.

The residue score and similarity score can be used to visualize each class separately, where  $R(\mathbf{x})$  is the  $x$ -axis, and  $S(\mathbf{x})$  is the  $y$ -axis. In the case of classification, define  $\{(\mathbf{x}_m, y_m, \hat{y}_m) | \mathbf{x}_m \in \mathbb{R}^N, y_m \in \mathbb{Z}_L, \hat{y}_m \in \mathbb{Z}_L\}_{m=1}^M$ , where  $\hat{y}_m$  is the predicted label for the  $m$ th sample. Then, we can repeat the above process by using the ground truth and visualize each class separately. By coloring the data point with the predicted label  $\hat{y}_m$ , we get the R-S score visualization of the classification.

Class residue index (CRI) and class similarity index (CSI) can be easily defined for the  $l$ th class as  $\text{CRI}_l = \frac{1}{|C_l|} \sum_m R_m$  and  $\text{CSI}_l = \frac{1}{|C_l|} \sum_m S_m$ , respectively. Such indexes can be used to compare the distributions in different classes obtained by different methods.

The above indices depend on clusters or classes. It is more useful to construct class-independent global indices. To this end, we first define residue index (RI) and similarity index (SI) as  $\text{RI} = \frac{1}{L} \sum_l \text{CRI}_l$  and  $\text{SI} = \frac{1}{L} \sum_l \text{CSI}_l$ , respectively. All of these indexes have the range of  $[0,1]$  and the larger the better for a given dataset. Additionally, we define R-S disparity (RSD) as  $\text{RSD} = \text{RI} - \text{SI}$ . RSD ranges  $[-1,1]$ . Finally, we define R-S index (RSI) as  $\text{RSI} = 1 - |\text{RI} - \text{SI}|$ . R-S index has the range of  $[0,1]$ .

The Rand index is known to correlate with accuracy [1]. We speculate that the R-S disparity may correlate with the convergence of clustering and the R-S index may correlate with the accuracy of classification. R-S disparity and R-S index can be used to measure the performance of different methods.

#### 4.1.2 Persistent Spectral Graph (PSG)

Further analysis of point cloud data or the points in the R-S plot can be carried out with Topology Data Analysis (TDA). Persistent homology [2, 3, 4, 5, 6, 7, 8] is an algebraic topology technique and the main workhorse of TDA. It introduces a filtration process to generate a family of topological spaces so that the original data can be analyzed in multiscales. However, it cannot

detect the homotopic shape evolution of data during filtration. Topological Laplacians, such as persistent spectral graph (aka persistent Laplacian) [9, 10] and evolutionary Hodge Laplacian [11] are designed to preserve full topological persistence and capture homotopic shape evolution of data during a filtration. The persistent spectral graph returns the same multiscale topological invariants in its kernels of various dimensions and scales but offers additional homotopic shape information in its non-harmonic spectra.

Considering two boundary operators  $\partial_q^t : C_q(K_t) \mapsto C_{q-1}(K_t)$  and  $\partial_{q+1}^{t+p} : C_{q+1}(K_{t+p}) \mapsto C_q(K_{t+p})$ , where  $C_q(K_{t+p})$  is a chain group and  $K_t \subset K_{t+p}$  are simplicial complexes generated by a filtration. Denote  $\partial_{q+1}^{t+p}|_{\mathbb{C}_{q+1}^{t,p}}$  as  $\delta_{q+1}^{t,p}$  such as

$$\mathbb{C}_{q+1}^{t,p} = \{\alpha \in C_{q+1}^{t+p} \mid \partial_{q+1}^{t+p}(\alpha) \in C_q^t\}. \quad (4.3)$$

Namely,  $\mathbb{C}_{q+1}^{t,p}$  consists of elements whose images under  $\partial_{q+1}^{t+p}$  are in  $C_q^t$ . The  $p$ -persistent  $q$ -combinatorial Laplacian operator [12] is given by

$$\Delta_q^{t,p} = \delta_{q+1}^{t,p}(\delta_{q+1}^{t,p})^* + (\partial_q^t)^* \partial_q^t. \quad (4.4)$$

The topological invariants of the corresponding persistent homology defined by the same filtration are recovered from the kernel of the persistent Laplacian Eq. (4.4) [12],

$$\beta_q^{t,p} = \dim \ker \partial_q^t - \dim \text{im } \delta_{q+1}^{t,p} = \dim \ker \Delta_q^{t,p}. \quad (4.5)$$

State differently, the zero eigenvalues of the persistent Laplacian operator Eq. (4.4) give rise to the entire topological variants of the persistent homology. Then, the non-harmonic part of the spectra (i.e., the non-zero eigenvalues of the persistent Laplacian) and associated eigenvectors offer additional shape information of the underlying data.

Note that for small-sized high-dimensional datasets, PSG can be directly employed to reduce the dimensionality in terms of the statistical quantities of the data spectra. The resulting spectra or their statistics can be directly used to represent the original datasets.

### 4.1.3 The shape of data

Continuous FRI was defined to offer the shape of  $M$  data entries in  $\mathbb{R}^3$  [13]. A similar idea was used to define interactive differentiable Riemannian manifolds [14]. Here, we extend these ideas to construct Grassmann manifolds  $\text{Gr}(N - 1, I)$ .

Let  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m, \dots, \mathbf{x}_M\}$  be a finite set of  $M$  data entries. Denote  $\mathbf{x}_m \in \mathbb{R}^N$  be the feature vector for the  $m$ th sample, and  $\|\mathbf{x} - \mathbf{x}_m\|$  be the Euclidean distance between a point  $\mathbf{x} \in \mathbb{R}^N$  to the  $j$ th sample. Let  $\eta = \frac{1}{M} \sum_{m=1}^M \min_{\mathbf{x}_j} \|\mathbf{x}_m - \mathbf{x}_j\|$  be the average minimum pairwise distance of the input data. Then, the unnormalized rigidity density at point  $\mathbf{x} \in \mathbb{R}^N$  is given by

$$\mu(\mathbf{x}) = \sum_{m=1}^M \omega_m \Phi(\|\mathbf{x} - \mathbf{x}_m\|; \tau, \eta, \kappa), \quad (4.6)$$

where  $\omega_m = 1$ , and  $\tau$  and  $\kappa$  are the hyperparameters of the correlation kernel  $\Phi$ . Notice that we can choose an isosurface  $\mu(\mathbf{x}) = c\mu_{\max}$ , which defines an  $(N - 1)$ -dimensional Riemannian manifold by the collection of points

$$\{\mathbf{x} | \mathbf{x} \in \mathbb{R}^N, \mu(\mathbf{x}) = c\mu_{\max}\}, \quad (4.7)$$

where  $c \in (0, 1)$  and  $\mu_{\max} = \max_{\mathbf{x}} \mu(\mathbf{x})$ . The shape of data can be directly visualized for  $2 \leq N \leq 3$  as shown in Ref. [14].

One can restrict  $\mathbf{x}_m$  to a given subset in Eq. (4.6) to compare the shape of data in different classes when the class labels are known.

For further analysis, one can obtain  $(N - 1)$  independent curvatures via fundamental forms [14]. Additionally, Hodge decomposition can be applied to analyze topological connectivity (i.e., Betti numbers associated with the harmonic spectra) and non-harmonic spectra of the Hodge Laplacians of the data [15]. For evolving manifolds, the evolutionary de Rham-Hodge theory can be used to analyze the geometry and topology of data [11].



## 4.2 Results

### 4.2.1 Geometric shape, Residue-Similarity (R-S), and topological analysis

In this section, we compare the 3D shape, R-S plot, and topological persistence of the TCGA-PANCAN data. For the comparison, CCP was used to reduce the data to  $N = 3$  components. The data were divided according to their true labels into 5 classes. The 5-fold cross-validation was used to obtain the predicted labels for visualization (coloring).

For the 3D shape visualization, after the dimensionality reduction, the Gaussian surface was used to generate the volumetric representation. The Chimera [16] was used to visualize the shape of data at the isovalue of 0.1. The surface was colored according to the predicted labels.

For the persistence plot, after the dimensionality reduction, the data was divided according to their true labels. The HERMES package [9] with the  $\alpha$  complex was used to generate topological dimensions 0 (Betti-0), 1 (Betti-1), and 2 (Betti-2) curves and the corresponding smallest non-zero eigenvalue curves. Note that persistent Laplacian itself offers low-dimensional geometric and topological representations of the original high-dimensional data [17].

Figure 4.1 shows the 3 different visualizations of class 1. Notice that the shape analysis shows predominately red regions or dots mixed with misclassified labels. We can see this mixing in the R-S plot as well. The yellow points have lower R scores, indicating that these samples are more likely to be mislabeled in machine learning. We can see that blue points with low S scores are isolated in the shape visualization.

Note that  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$  and  $\beta_2^{\alpha,0}$  offer the same information as persistent homology does for the data. The  $\beta_0^{\alpha,0}$  shows there are about 290 samples in this class that become fully connected at radius 6 ( $\beta_0^{\alpha,0} = 1$ ). The  $\beta_1^{\alpha,0}$  shows there are many cycles in the sample. The  $\beta_1^{\alpha,0}$  indicates there are at most 7 cavities. There are no topological changes in the data after radius=11. However, the smallest non-zero eigenvalue ( $\lambda_0^{\alpha,0}$ ) keeps changing as the filtration radius increases, indicating that persistent Laplacian reveals more information about the data than persistent homology does.

Finally, it is noticed that most misclassified samples have relatively low R-S scores. This observation indicates the effectiveness of our R-S scores and indexes.

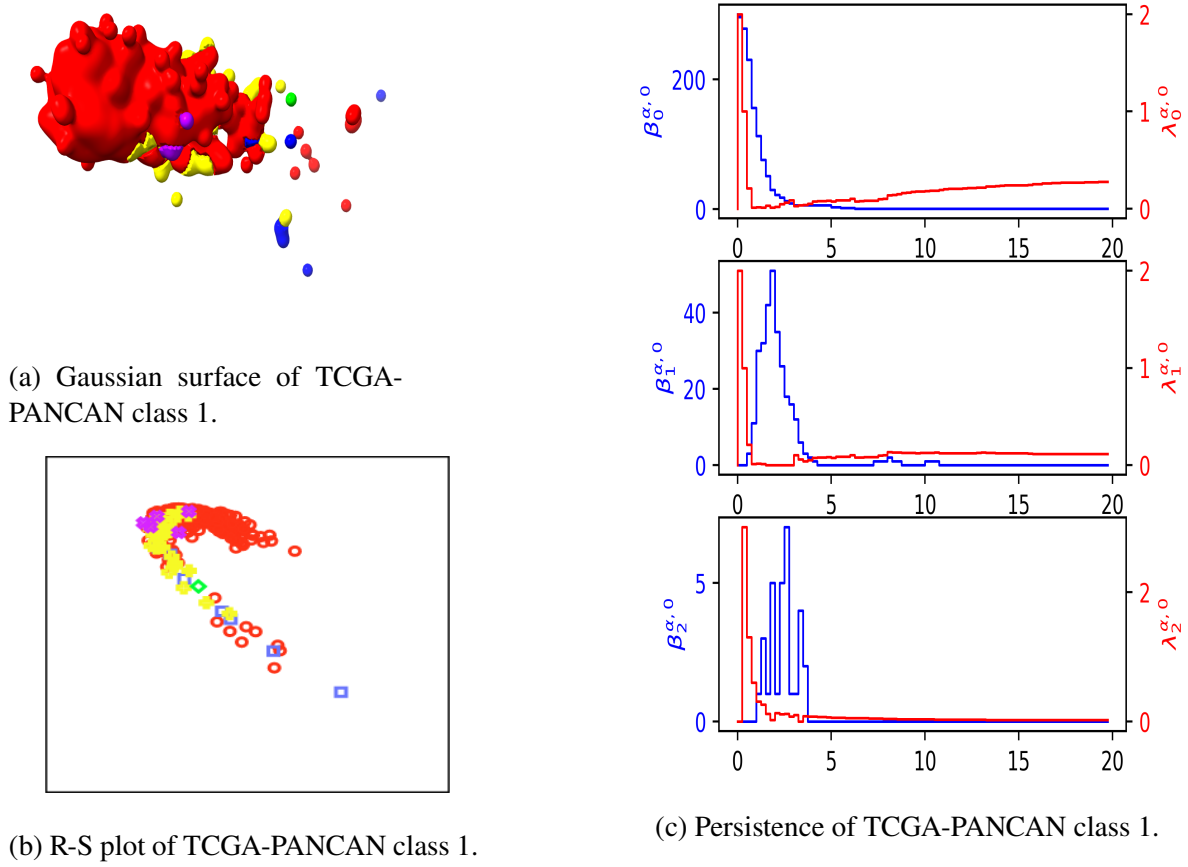


Figure 4.1 Shape of data, R-S and persistence visualization of TCGA-PANCAN class 1 data. CCP was used to reduce the data to  $N = 3$ . (a) Shape of data was visualized with isovalue 0.1 in ChimeraX [16]. Red color indicates the correctly classified data. (b) R-S plot of class 1. Red circle is the correct label. The  $x$  and  $y$ -axes correspond to the residue and similarity scores, respectively. (c) Visualization of the smallest non-zero eigenvalue curves along the filtration (indicated by red color)  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$ , and the harmonic spectral curves (indicated by blue color)  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  for class 1. HERMES package [9] with the  $\alpha$  complex was used to calculate the harmonic and non-harmonic spectra. The  $x$ -axis is the filtration radius. The left  $y$ -axis corresponds to the  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  from top to bottom, and the right  $y$ -axis corresponds to  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$  from top to bottom.

The shape, R-S, and topological analysis of class 2 are given in Figure 4.2. The  $\beta_0^{\alpha,0}$  indicates class 2 has about 130 samples, which become fully connected near radius=10. The  $\lambda_0^{\alpha,0}$  curve shows a significant discontinuity at radius near radius=10. The  $\beta_1^{\alpha,0}$  shows about 28 cycles at its peak value. At most two cavities in data have been found in  $\beta_2^{\alpha,0}$  at a given filtration. The shape shows four major pieces and only a few samples were mislabeled by machine learning. R-S plots should have four different labels in this class. Most of the samples were correctly predicted, which

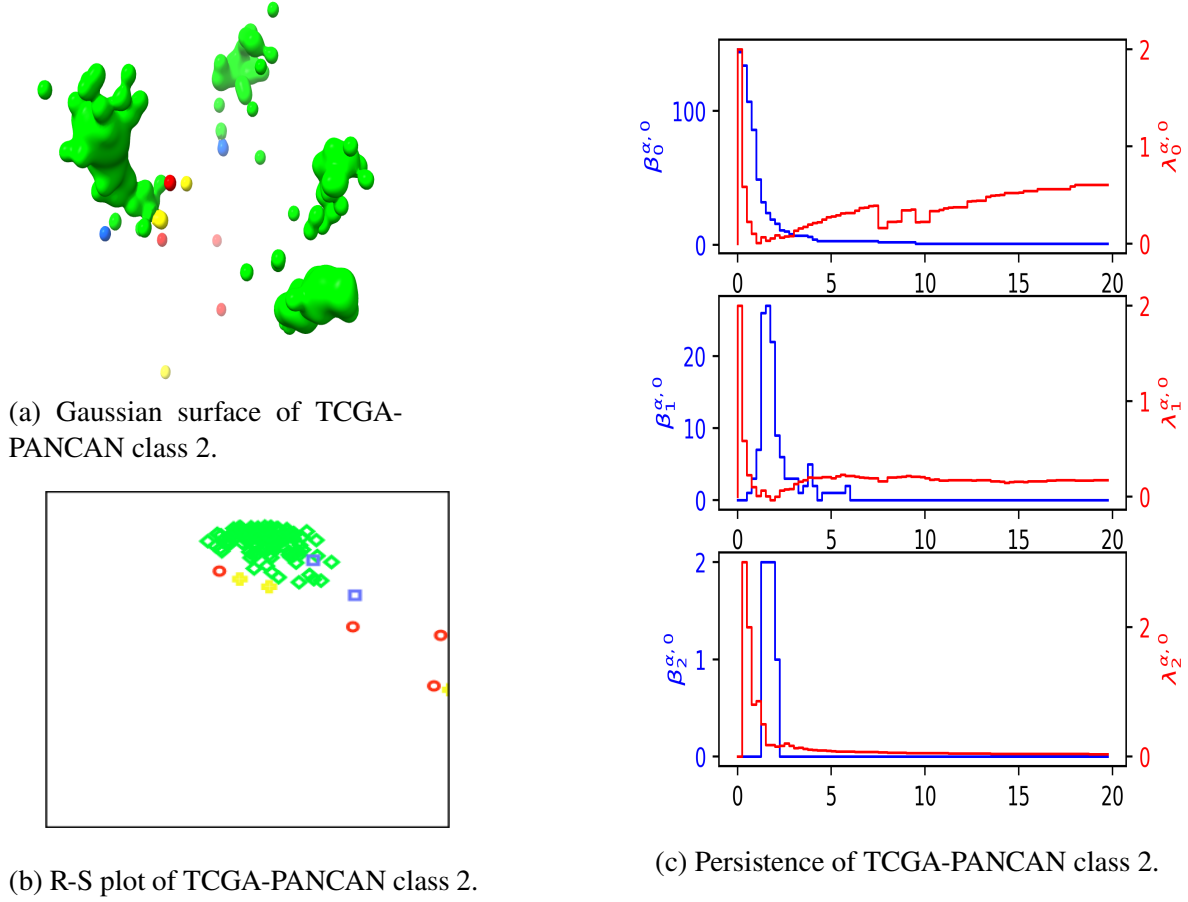


Figure 4.2 Shape of data, R-S and persistence visualization of TCGA-PANCAN class 2 data. CCP was used to reduce the data to  $N = 3$ . (a) Shape of data was visualized with isovalue 0.1 in ChimeraX [16]. Green color indicates the correctly classified data. (b) R-S plot of class 2. Green circle is the correct label. The  $x$  and  $y$ -axes correspond to the residue and similarity scores, respectively. (c) Visualization of the smallest non-zero eigenvalue curves along the filtration (indicated by red color)  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$ , and the harmonic spectral curves (indicated by blue color)  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  for class 2. HERMES package [9] with the  $\alpha$  complex was used to calculate the harmonic and non-harmonic spectra. The  $x$ -axis is the filtration radius. The left  $y$ -axis corresponds to the  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  from top to bottom, and the right  $y$ -axis corresponds to  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$  from top to bottom.

is consistent with the shape analysis. Most class 1 labels (red ones) have lower S scores, indicating that they do not belong.

Most misclassified samples have low R-S scores and are disconnected from other samples in the class.

Figure 4.3 gives three types of analyses for class 3. This class has only about 78 samples, as shown by the  $\beta_0^{\alpha,0}$  curve. At filtration radius=2, there were 11 one-dimensional holes in the data.

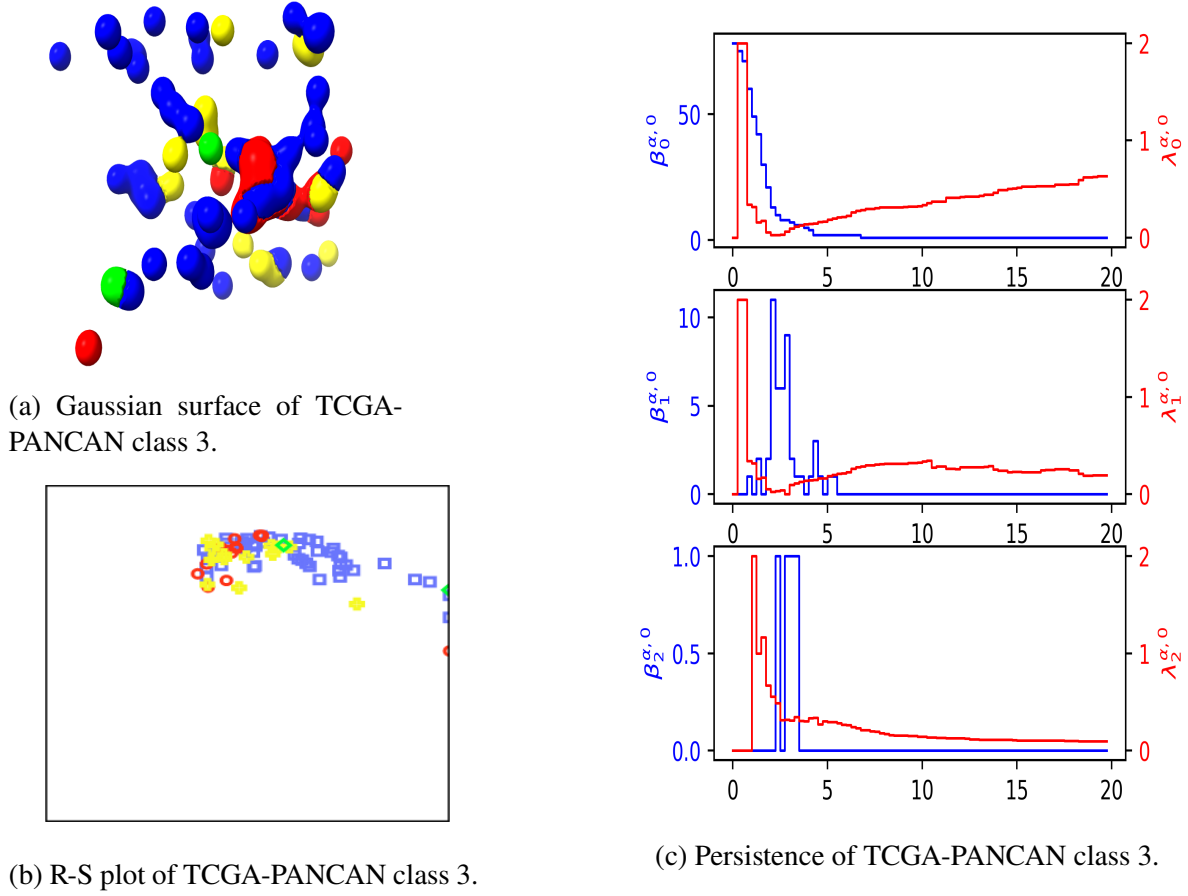


Figure 4.3 Shape of data, R-S and persistence visualization of TCGA-PANCAN class 3 data. CCP was used to reduce the data to  $N = 3$ . (a) Shape of data was visualized with isovalue 0.1 in ChimeraX [16]. Blue color indicates the correctly classified data. (b) R-S plot of class 3. Blue circle is the correct label. The  $x$  and  $y$ -axes correspond to the residue and similarity scores, respectively. (c) Visualization of the smallest non-zero eigenvalue curves along the filtration (indicated by red color)  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$ , and the harmonic spectral curves (indicated by blue color)  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  for class 3. HERMES package [9] with the  $\alpha$  complex was used to calculate the harmonic and non-harmonic spectra. The  $x$ -axis is the filtration radius. The left  $y$ -axis corresponds to the  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  from top to bottom, and the right  $y$ -axis corresponds to  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$  from top to bottom.

There were only two cavities found by  $\beta_2^{\alpha,0}$ . The shape plot indicates most samples are disconnected at isovalue 0.1 but merge at radius 7 as detected by  $\beta_0^{\alpha,0}$ . The yellow labels are close to the red ones, as shown by the shape and R-S plots. The  $\lambda_1^{\alpha,0}$  curve demonstrates a few discontinuities after topological persistence ( $\beta_1^{\alpha,0}$ ) becomes flat, indicating important homotopic events in the data. As in other classes, most misclassified samples have relatively low R-S scores.

In Figure 4.4, the  $\beta_0^{\alpha,0}$  curve suggests 150 samples in class 4 (yellow). Some samples are

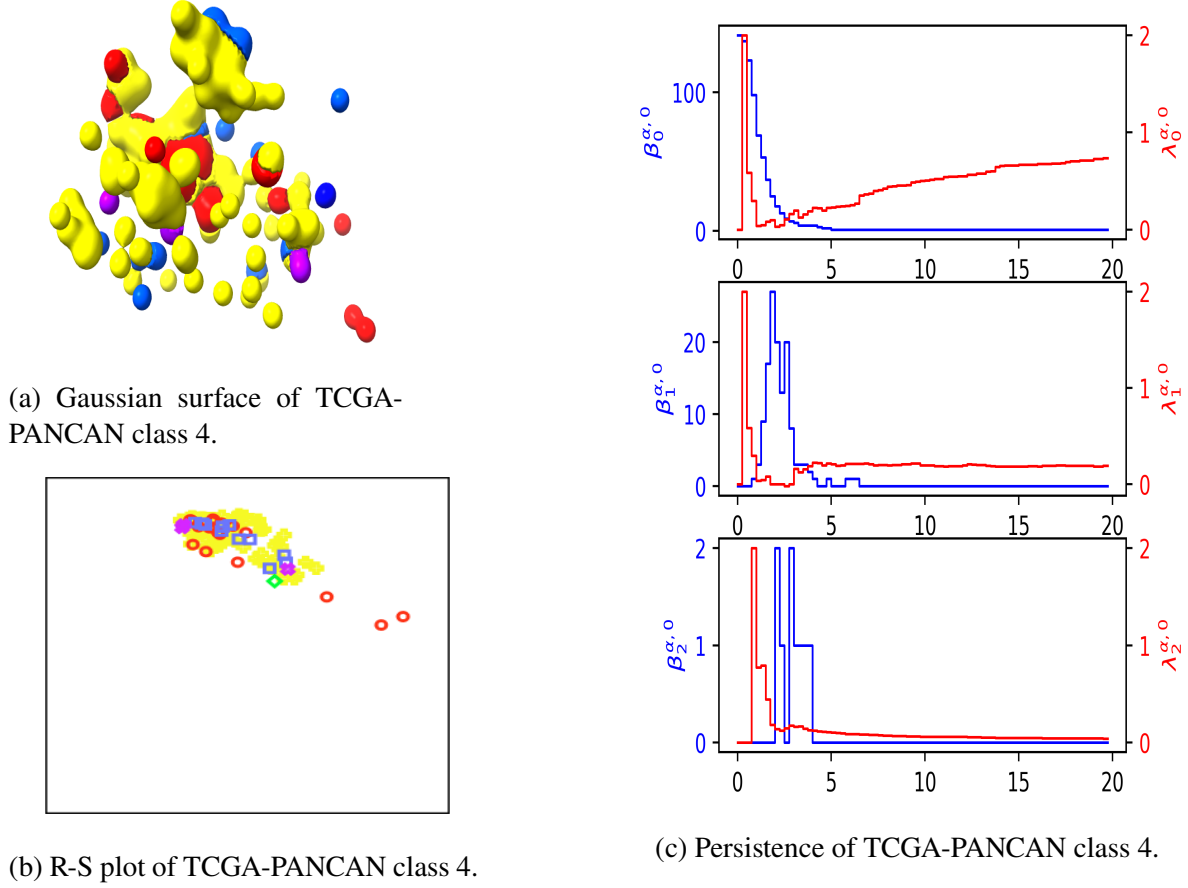


Figure 4.4 Shape of data, R-S and persistence visualization of TCGA-PANCAN class 4 data. CCP was used to reduce the data to  $N = 3$ . (a) Shape of data was visualized with isovalue 0.1 in ChimeraX [16]. Yellow color indicates the correctly classified data. (b) R-S plot of class 4. Yellow circle is the correct label. The  $x$  and  $y$ -axes correspond to the residue and similarity scores, respectively. (c) Visualization of the smallest non-zero eigenvalue curves along the filtration (indicated by red color)  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$ , and the harmonic spectral curves (indicated by blue color)  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  for class 4. HERMES package [9] with the  $\alpha$  complex was used to calculate the harmonic and non-harmonic spectra. The  $x$ -axis is the filtration radius. The left  $y$ -axis corresponds to the  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  from top to bottom, and the right  $y$ -axis corresponds to  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$  from top to bottom.

misclassified as class 1 (red), class 3 (blue), and class 5 (purple) as shown in shape and R-S plots. The topological persistence indicates many topological invariants along the filtration axis, which can be a faithful representation of the data [17]. Specifically, all data points overlap at radius 5, as shown by  $\beta_0^{\alpha,0}$ . However,  $\lambda_0^{\alpha,0}$  still indicates a discontinuity at radius 7. All cycles disappear after radius 7 as revealed by  $\beta_1^{\alpha,0}$ . The last cycle persists from radius 6 to 7. The  $\beta_2^{\alpha,0}$  curve becomes flat at radius 4. The misclassified red samples show low S scores.

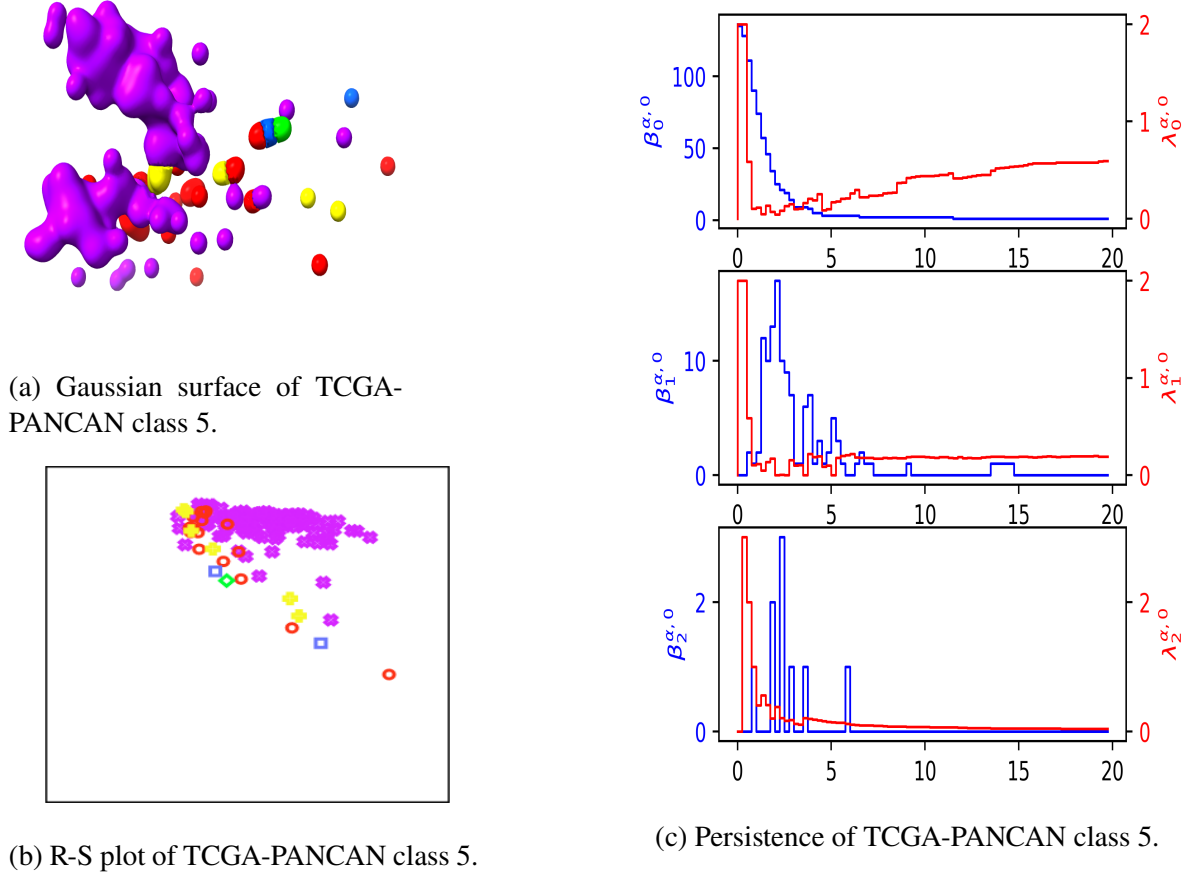


Figure 4.5 Shape of data, R-S and persistence visualization of TCGA-PANCAN class 5 data. CCP was used to reduce the data to  $N = 3$ . (a) Shape of data was visualized with isovalue 0.1 in ChimeraX [16]. Purple color indicates the correctly classified data. (b) R-S plot of class 5. Purple circle is the correct label. The  $x$  and  $y$ -axes correspond to the residue and similarity scores, respectively. (c) Visualization of the smallest non-zero eigenvalue curves along the filtration (indicated by red color)  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$ , and the harmonic spectral curves (indicated by blue color)  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  for class 5. HERMES package [9] with the  $\alpha$  complex was used to calculate the harmonic and non-harmonic spectra. The  $x$ -axis is the filtration radius. The left  $y$ -axis corresponds to the  $\beta_0^{\alpha,0}$ ,  $\beta_1^{\alpha,0}$ , and  $\beta_2^{\alpha,0}$  from top to bottom, and the right  $y$ -axis corresponds to  $\lambda_0^{\alpha,0}$ ,  $\lambda_1^{\alpha,0}$ , and  $\lambda_2^{\alpha,0}$  from top to bottom.

Figure 4.5 illustrates our shape, R-S, and topological analyses of class 5. Although  $\beta_0^{\alpha,0}$  indicates there are only about 140 samples, class 5 is very rich in its topological persistence. The  $\beta_1^{\alpha,0}$  shows the data points did not connect before the filtration radius reached 12. The  $\beta_1^{\alpha,0}$  curve indicates a large one-dimensional hole from radius 13.5 to 15. The  $\beta_2^{\alpha,0}$  curve shows 9 short-living cavities in the data. It is clear from the R-S plot that samples having low R-S scores are more likely to be mislabeled.

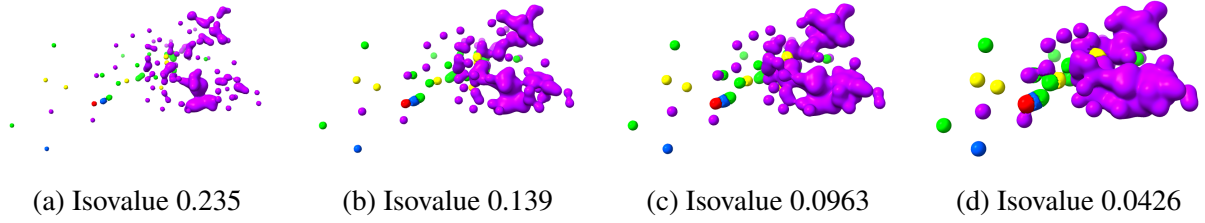


Figure 4.6 Shape of class 5 of TCGA-PANCAN dataset visualized in multiscale using ChimeraX [16] when isovalues were varied. The different colors indicated the predicted labels, and purple is the true label of class 5.

Because Figure 4.5's persistence plot indicates interesting topological features, and class 5 was further visualized in Figure 4.6 with varying isovalues or scales. We can see that from isovalues 0.235 to 0.139, 2 holes form in the bottom right corner. We can also see another hole beginning to form in the bottom center. From isovalues 0.139 to 0.0963, the two holes are no longer visible, but the hole that was forming is now completed. This corresponds to  $\beta_1^{\alpha,0}$ . The voids are short-lived, as shown by  $\beta_2^{\alpha,0}$  in Figure 4.5 and cannot be visible in the isosurface. Decreasing the isovalue further to 0.0426 shows the combination of two main parts of the data, which would stabilize the structure. This corresponds to the decrease of  $\beta_0^{\alpha,0}$  because the number of components is decreasing, but the increase in  $\lambda_0^{\alpha,0}$  indicates that the structure is more stable.

### 4.3 Discussion

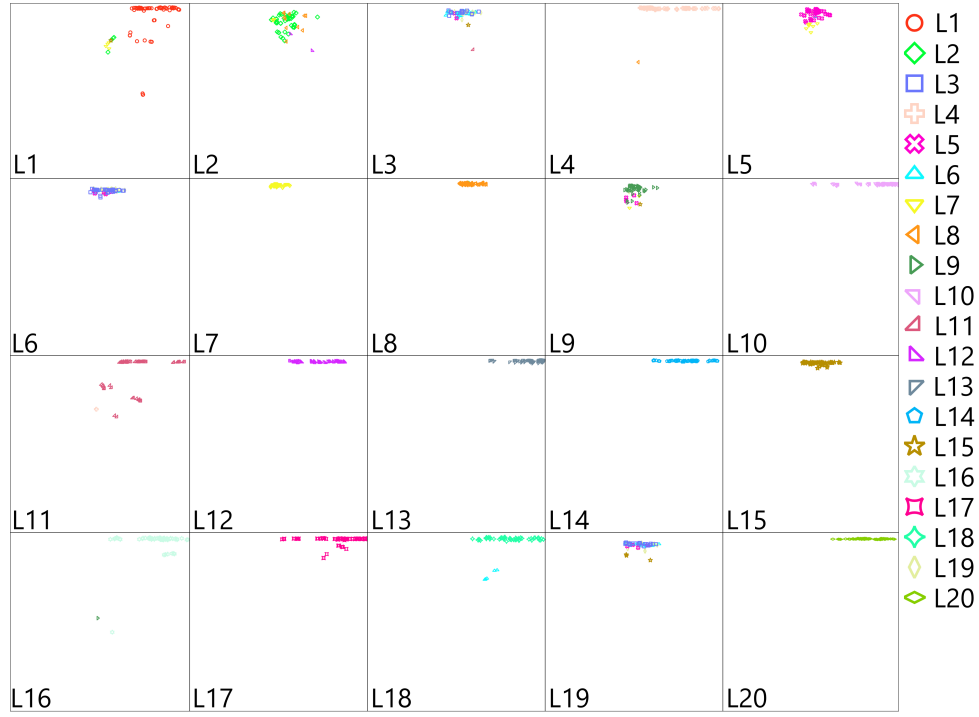
#### 4.3.1 R-S plot vs 2D plot

R-S plot is an effective tool to visualize the performance of classification in general. Figure 4.7 shows the comparison of the R-S plot and the traditional 2D visualization of the Coil-20 dataset when the dimension is reduced to 2 by UMAP. For the traditional 2D plot, each data point was colored by the ground truth, and for the R-S plot, each section represents one of the 20 different classes, and data points were colored by the predicted labels from the  $k$ -NN classification. We can see that in the traditional 2D visualization, labels 3, 9, and 19 are located in the same region. It is interesting to see that this situation is reflected in our R-S plot as three labels mixed up. In the R-S plot, Labels 3, 9, and 19 have a high similarity score but a low residue score, meaning that the data points are not separated well among different classes and show the limitation of preserving the

local structure of a high dimensional data represented in the 2D space. Essentially, some data lay in an intrinsically high-dimensional space that cannot be well-described in the 2D representation.

Note that 2D plots work best when the data dimension is reduced to 2, whereas the R-S plots can be applied to arbitrarily high dimensions.





(a) R-S plot of Coil-20 with 2 dimension.



(b) Traditional 2d plot of Coil-20 using UMAP.

Figure 4.7 (a) shows the R-S plot of the Coil-20 dataset when reduced by UMAP to  $N = 2$ . Each section represents a different class, where the data points were colored according to their predicted labels from  $k$ -NN classification via 10-fold cross-validation. (b) Coil-20 dataset reduced to  $N = 2$  by UMAP. The data points were colored according to the ground truth.

## BIBLIOGRAPHY

- [1] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.
- [2] Patrizio Frosini. Measuring shapes by size functions. In *Intelligent Robots and Computer Vision X: Algorithms and Techniques*, volume 1607, pages 122–133. International Society for Optics and Photonics, 1992.
- [3] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *Proceedings 41st annual symposium on foundations of computer science*, pages 454–463. IEEE, 2000.
- [4] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33(2):249–274, 2005.
- [5] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.
- [6] Konstantin Mischaikow and Vidit Nanda. Morse theory for filtrations and efficient computation of persistent homology. *Discrete & Computational Geometry*, 50(2):330–353, 2013.
- [7] K. L. Xia and G. W. Wei. Persistent homology analysis of protein structure, flexibility and folding. *International Journal for Numerical Methods in Biomedical Engineering*, 30:814–844, 2014.
- [8] Jacob Townsend, Cassie Putman Micucci, John H Hymel, Vasileios Maroulas, and Konstantinos D Vogiatzis. Representation of molecular structures with persistent homology for machine learning applications in chemistry. *Nature communications*, 11(1):1–9, 2020.
- [9] Rui Wang, Rundong Zhao, Emily Ribando-Gros, Jiahui Chen, Yiying Tong, and Guo-Wei Wei. Hermes: Persistent spectral graph software. *Foundations of data science (Springfield, Mo.)*, 3(1):67, 2021.
- [10] Facundo Mémoli, Zhengchao Wan, and Yusu Wang. Persistent laplacians: Properties, algorithms and implications. *SIAM Journal on Mathematics of Data Science*, 2022.
- [11] Jiahui Chen, Rundong Zhao, Yiying Tong, and Guo-Wei Wei. Evolutionary de rham-hodge method. *Discrete and continuous dynamical systems. Series B*, 26(7):3785, 2021.
- [12] Rui Wang, Duc Duy Nguyen, and Guo-Wei Wei. Persistent spectral graph. *International journal for numerical methods in biomedical engineering*, 36(9):e3376, 2020.
- [13] Kelin Xia, Kristopher Opron, and Guo-Wei Wei. Multiscale multiphysics and multidomain models—flexibility and rigidity. *The Journal of chemical physics*, 139(19):11B614\_1, 2013.

- [14] Duc Duy Nguyen and Guo-Wei Wei. Dg-gl: Differential geometry-based geometric learning of molecular datasets. *International journal for numerical methods in biomedical engineering*, 35(3):e3179, 2019.
- [15] Rundong Zhao, Menglun Wang, Jiahui Chen, Yiyong Tong, and Guo-Wei Wei. The de rham–hodge analysis and modeling of biomolecules. *Bulletin of mathematical biology*, 82(8):1–38, 2020.
- [16] Eric F Pettersen, Thomas D Goddard, Conrad C Huang, Elaine C Meng, Gregory S Couch, Tristan I Croll, John H Morris, and Thomas E Ferrin. Ucsf chimeraX: Structure visualization for researchers, educators, and developers. *Protein Science*, 30(1):70–82, 2021.
- [17] Jiahui Chen, Yuchi Qiu, Rui Wang, and Guo-Wei Wei. Persistent laplacian projected omicron ba. 4 and ba. 5 to become new dominating variants. *Computers in biology and medicine*, 151:106262, 2022.

## CHAPTER 5

### CORRELATED CLUSTERING AND PROJECTION

#### 5.1 Introduction

In this work, we propose a two-step data-domain method that seeks an optimal clustering in terms of a distance describing intrinsic feature correlations among samples to divide  $I$  feature vectors into  $N$  correlated clusters and then, non-linearly project correlated features in each cluster into a single descriptor by using Flexibility Rigidity Index (FRI) [1], which results in a low-dimensional representation of the original data. Additionally, the complex global correlations among samples are embedded into samples' local representations during the FRI-based nonlinear projection  $\mathbb{R}^I \rightarrow \mathbb{R}^N$ . To gain computational efficiency, one may further compute the pairwise correlation matrix of samples and impose a cutoff distance to avoid the global summation during the projection. The resulting method, called Correlated Clustering and Projection (CCP), precedes the other DR algorithms in the following aspects. (1) Instead of solving a matrix to reduce the dimensionality, CCP does not involve matrix diagonalization and thus can handle the dimensionality reduction of large sample sizes. (2) CCP exploits statistical measures, such as (distance) covariance, to quantify the high-level dependence between random feature vectors, rendering a stable algorithm when dealing with high dimensional data. (3) CCP is flexible with respect to targeted dimension  $N$  because the partition of features is based on  $N$ , whereas other methods may rely on  $\min(M, N)$ . The performance of CCP is stable with respect to the increase of  $N$ , which is important for datasets with high or moderately high intrinsic dimensions. In contrast, many existing methods stop working when the intrinsic data dimension is moderately high. (4) CCP is stable with respect to subsampling, which allows continuously adding new samples into a pre-existing dataset without the need to restart the calculation from the very beginning and thus, is advantageous for continuous data acquisition, collection, and analysis. This capability is valuable when the transient data are too expensive to be kept permanently, e.g., molecular dynamics simulations. Additionally, this subsampling property enables parameter optimization using a small amount of data in case of large data size. (5) As a data-domain method, CCP can be combined with a frequency-domain method, such as UMAP or

t-SNE, for a secondary dimensionality reduction to better preserve global structures of data and achieve higher accuracy. (6) Finally, the performance of CCP is validated on several benchmark classification datasets: Leukemia, Carcinoma, ALL-AML, TCGA-PANCAN, Coil-20, Coil-100, and Smallnorb based on various traditional algorithms such as  $k$ -NN, support vector machine, random forest and gradient boost decision tree. In all cases, CCP is very competitive with the state-of-art algorithms.

Additionally, we have also proposed a new method, called Residue-Similarity (R-S) scores or R-S plot, for the performance visualization of unsupervised clustering and supervised classification algorithms. Although Receiving Operation Characteristic curve (ROC) and Area Under the ROC Curve (AUC) are typically used for the performance visualization of binary classes, they are not convenient for multiple classes. The proposed R-S scores can be used for visualizing the performance in an arbitrary number of classes. Finally, R index, S index, R-S disparity, and total R-S index are proposed to characterize clustering and classification results.

Recent years have witnessed the growth of Topological Data Analysis (TDA) via persistent homology [2, 3, 4, 5, 6, 7, 8] in data sciences. It can be used to analyze the topological invariant of the R-S scores. However, persistent homology is insensitive to the homotopic shape evolution of data during filtration. We introduce a topological Laplacian, Persistent Spectral Graph (PSG) [9], to capture the homotopic shape of data, in addition to topological invariants. Note that TDA and PSG are dimensionality reduction algorithms that can generate low dimensional representations of the original high-dimensional data [10, 11].

To further analyze the shape of data, we transform point cloud data into a Grassmann manifold representation by using FRI [1]. When  $N = 3$ , the 2-manifold shape of data can be directly visualized. Such shape of data can be further analyzed by differential geometry apparatuses, including curvatures [12], Hodge decomposition [13] and evolutionary de Rham-Hodge theory [14].

## 5.2 Methods and Algorithms

Let  $\mathcal{Z} := \{z_m^i\}_{m=1, i=1}^{M, I}$  with  $M$  and  $I$  being the number of input data entries (i.e., samples) and the number of features for each data entry, respectively. Our goal is to find an  $N$ -dimensional representation of the original data, denoted as  $\mathcal{X} := \{x_m^i\}_{m=1, i=1}^{M, N}$ , such that  $1 \leq N \ll I$ , by using a data-domain two-step clustering-projection strategy.

### 5.2.1 Feature clustering

Let  $\mathcal{Z} = \{\mathbf{z}^1, \dots, \mathbf{z}^i, \dots, \mathbf{z}^I\}$  be the set of data, where  $\mathbf{z}^i \in \mathbb{R}^M$  represents the  $i$ th feature vector for the data. The objective is to partition the feature vector into  $N$  parts, where  $1 \leq N \ll I$  is a preselected reduced feature dimension. To this end, we find an optimal disjoint partition of the data  $\mathcal{Z} := \uplus_{n=1}^N \mathcal{Z}^n$ , for a given  $N$ , where  $\mathcal{Z}^n$  is the  $n$ th partition (cluster) of the features.

To seek the optimal partition, we first analyze the correlation among feature vectors  $\mathbf{z}^i$ . A variety of correlation measures can be used for this purpose. We discuss two standard approaches.

**Covariance distance** First, we consider an  $I \times I$  normalized covariance matrix with component

$$\rho(\mathbf{z}^i, \mathbf{z}^j) = \frac{\text{Cov}(\mathbf{z}^i, \mathbf{z}^j)}{\sigma(\mathbf{z}^i)\sigma(\mathbf{z}^j)}, \quad 1 \leq i, j \leq I, \quad (5.1)$$

where  $\text{Cov}(\mathbf{z}^i, \mathbf{z}^j)$  is the covariance of  $\mathbf{z}^i$  and  $\mathbf{z}^j$ , and  $\sigma(\mathbf{z}^i)$  and  $\sigma(\mathbf{z}^j)$  are the variances of  $\mathbf{z}^i$  and  $\mathbf{z}^j$ , respectively.

We set negative covariances to zero and subtract from 1 to obtain a *covariance distance* between feature vectors

$$\|\mathbf{z}^i - \mathbf{z}^j\|_{\text{dCov}} = \begin{cases} 1 - \rho(\mathbf{z}^i, \mathbf{z}^j), & \rho(\mathbf{z}^i, \mathbf{z}^j) > 0 \\ 0, & \text{otherwise.} \end{cases} \quad (5.2)$$

Note that covariance distances have the range of  $0 < \|\mathbf{z}^i - \mathbf{z}^j\|_{\text{dCov}} < 1$ , for all pairs of vectors,  $\mathbf{z}^i$  and  $\mathbf{z}^j$ . Highly correlated feature vectors will have their covariance distances close to 0, while the uncorrelated feature vectors will have their covariance distances close to 1.

**Correlation distance** Alternatively, one may also consider the correlation distance defined via the distance correlation [15]. First, one computes a distance matrix for each vector  $\mathbf{z}^i, i = 1, 2, \dots, I$

$$a_{jk}^i = \|\mathbf{z}_m^i - \mathbf{z}_k^i\|, m, k = 1, 2, \dots, M, \quad (5.3)$$

where  $\|\cdot\|$  denotes the Euclidean norm. Define doubly centered distance for vector  $\mathbf{z}^i$ ,

$$A_{jk}^i := a_{jk}^i - \bar{a}_{j\cdot} - \bar{a}_{\cdot k} + \bar{a}_{\cdot\cdot}, \quad (5.4)$$

where  $\bar{a}_{j\cdot}$  is the  $j$ th row mean,  $\bar{a}_{\cdot k}$  is the  $k$ th column mean, and  $\bar{a}_{\cdot\cdot}$  is the grand mean of the distance matrix for vector  $\mathbf{z}^i$ .

For a pair of vectors  $(\mathbf{z}^i, \mathbf{z}^j)$ , the squared distance covariance is given by

$$\text{dCov}^2(\mathbf{z}^i, \mathbf{z}^j) := \frac{1}{M^2} \sum_j \sum_k A_{jk}^i A_{jk}^j. \quad (5.5)$$

The distance correlation between vectors  $(\mathbf{z}^i, \mathbf{z}^j)$  is given by

$$\text{dCor}(\mathbf{z}^i, \mathbf{z}^j) := \frac{\text{dCov}^2(\mathbf{z}^i, \mathbf{z}^j)}{\text{dCov}(\mathbf{z}^i, \mathbf{z}^i) \text{dCov}(\mathbf{z}^j, \mathbf{z}^j)}. \quad (5.6)$$

We define a *correlation distance* between vectors  $\mathbf{z}^i$  and  $\mathbf{z}^j$  as

$$\|\mathbf{z}^i - \mathbf{z}^j\|_{\text{dCor}} = 1 - \text{dCor}(\mathbf{z}^i, \mathbf{z}^j). \quad (5.7)$$

The correlation distance has values in range  $0 \leq \|\mathbf{z}^i - \mathbf{z}^j\|_{\text{dCor}} \leq 1$ . It gives  $\|\mathbf{z}^i - \mathbf{z}^j\|_{\text{dCor}} = 1$  if  $\mathbf{z}^i$  and  $\mathbf{z}^j$  are uncorrelated or independent. When  $\mathbf{z}^i$  and  $\mathbf{z}^j$  are linearly depends on each other, one has  $\|\mathbf{z}^i - \mathbf{z}^j\|_{\text{dCor}} = 0$ .

**Correlated clustering** Feature partition can be achieved with a variety of clustering methods. Here, as an example, we utilize a modified  $K$ -medoids method to perform the partition in a minimization process. Certainly, other  $K$ -means type of algorithms, including BFR algorithm, centroidal Voronoi tessellation,  $k$   $q$ -flats,  $K$ -means++, etc., can be utilized for our feature partition as well.

For a pre-selected  $N$ , we begin by randomly selecting  $N$  medoids  $\{\mathbf{m}^n\}_{n=1}^N$  and assign each vector to its nearest medoid, which gives rise to the initial partition  $\{\mathcal{Z}^n\}_{n=1}^N$ . Second, we denote the closest vector to the center of the  $n$ th partition  $\mathcal{Z}^n$  as the new medoid  $\{\mathbf{m}^n \in \mathcal{Z}^n\}_{n=1}^N$ . We reassign each vector into its nearest medoid, resulting in a new partition  $\{\mathcal{Z}^n\}_{n=1}^N$  to minimize the loss function or the accumulated distance. The process is repeated until  $\{\mathcal{Z}^n\}_{n=1}^N$  is optimized with respect to a specific distance definition,

$$\arg \min_{\{\mathcal{Z}^1, \dots, \mathcal{Z}^n, \dots, \mathcal{Z}^N\}} \sum_{n=1}^N \sum_{\mathbf{z}^i \in \mathcal{Z}^n} \|\mathbf{z}^i - \mathbf{m}^n\|, \quad (5.8)$$

where  $\|\cdot\|$  is either the covariance distance or the correlation distance. In comparison, covariance distance is easy to compute, while correlation distance can deal with complex nonlinear high-level correlations among feature vectors and samples. Note that many other metrics can be used too. For a given  $N$ , the minimization partitions similar feature vectors into  $N$  clusters, which provides the foundation for further projections.

Our next goal is to project the original  $I$ -dimensional dataset  $\mathcal{Z}$  into an  $N$ -dimensional representation  $\mathcal{X}$  according to the partition result.

### 5.2.2 Feature projection

**Flexibility Rigidity Index (FRI)** In this section, we review Flexibility Rigidity Index (FRI) [1].

Let  $\{\mathbf{z}_1, \dots, \mathbf{z}_m, \dots, \mathbf{z}_M\}$  be the input dataset, where  $\mathbf{z}_m \in \mathbb{R}^I$ . Denote  $\|\mathbf{z}_i - \mathbf{z}_j\|$  some metric between  $i$ th and  $j$ th data entries, and the correlations between data entries are computed as

$$C_{ij} = \Phi(\|\mathbf{z}_i - \mathbf{z}_j\|; \eta, \tau, \kappa), \quad 1 \leq i, j \leq M \quad (5.9)$$

where  $\Phi$  is the correlation kernel, and  $\tau, \eta, \kappa > 0$  are the parameters for the kernel. Commonly used metrics include the Euclidean distance, the Manhattan distance, the Wasserstein distance, etc.

The correlation kernel is a real-valued smooth monotonically decreasing function, satisfying the two properties

$$\Phi(\|\mathbf{z}_i - \mathbf{z}_i\|; \eta, \tau, \kappa) = 1, \quad \text{as } |\mathbf{z}_i - \mathbf{z}_j| \rightarrow 0$$

$$\Phi(\|\mathbf{z}_i - \mathbf{z}_j\|; \eta, \tau, \kappa) = 0, \quad \text{as } |\mathbf{z}_i - \mathbf{z}_j| \rightarrow \infty$$



A popular choice for such functions is a radial basis function. For example, one may use the generalized exponential function

$$\Phi(\|\mathbf{z}_i - \mathbf{z}_j\|; \eta, \tau, \kappa) = \begin{cases} e^{-(\|\mathbf{z}_i - \mathbf{z}_j\|/\tau\eta)^\kappa}, & \|\mathbf{z}_i - \mathbf{z}_j\| < r_c \\ 0, & \text{otherwise} \end{cases} \quad (5.10)$$

or the generalized Lorentz kernel

$$\Phi(\|\mathbf{z}_i - \mathbf{z}_j\|; \eta, \tau, \kappa) = \begin{cases} \frac{1}{1 + (\|\mathbf{z}_i - \mathbf{z}_j\|/\tau\eta)^\kappa} & \|\mathbf{z}_i - \mathbf{z}_j\| < r_c, \\ 0 & \text{otherwise} \end{cases} \quad (5.11)$$

where  $\kappa$  is the power,  $\tau$  is the multiscale parameter,  $\eta$  is the scale to be computed from the given data, and  $r_c$  is the cutoff distance, which is useful in a certain data structure to reduce the computational complexity [16]. In the context of t-SNE,  $\eta$  would be the perplexity, and in UMAP,  $\eta$  would define the geodesic of the Riemannian metric.

We can construct a correlation matrix  $C = \{C_{ij}\}$ , which reveals the topological connectivity between samples [1]. We can also view such correlation map as a weighted graph [9, 17], where  $r_c$  is the cutoff function. In order to understand the connectivity, we choose  $\eta$  to be the average minimum distance between the data entries

$$\eta = \frac{\sum_{m=1}^M \min_{\mathbf{z}_j} \|\mathbf{z}_m - \mathbf{z}_j\|}{M}.$$

Using the correlation function, we can define the rigidity of  $\mathbf{x}_i$  as

$$\mu_i = \sum_{m=1}^M \omega_{im} \Phi(\|\mathbf{z}_i - \mathbf{z}_m\|; \eta, \tau, \kappa),$$

where  $\omega_{im}$  are the weights. Here, we set  $\omega_{im} = 1$  for all  $i$  and  $m$ . From the graph perspective, one can also view  $\mu_i$  as the degree matrix of node  $\mathbf{x}_i$ .

**Correlated projection** In this subsection, we employ FRI for the correlative dimensional reduction of input dataset  $\{\mathbf{z}_1, \dots, \mathbf{z}_m, \dots, \mathbf{z}_M\}$ , where  $\mathbf{z}_m \in \mathbb{R}^I$ , leading to a low-dimensional representation  $\{\mathbf{x}_1, \dots, \mathbf{x}_m, \dots, \mathbf{x}_M\}$ , with  $\mathbf{x}_m \in \mathbb{R}^N$ . The FRI reduction captures the intrinsic correlation among samples.

Recall that  $\{\mathcal{Z}^n\}_{n=1}^N$  are optimal partition of feature vectors from the  $K$ -medoids or another clustering method. Let  $\mathcal{S} = \{1, \dots, I\}$  be the whole set of indices of the feature vectors, and  $\mathcal{S} = \uplus_{n=1}^N \mathcal{S}^n$ , where  $\mathcal{S}^n = \{i | \mathbf{z}^i \in \mathcal{Z}^n\}$ . We can define  $\mathbf{z}_m^{S^n}$  as  $m$ th input data with the  $n$ th collection of feature indices  $\mathcal{S}^n$ , i.e.,  $\mathbf{z}_m^{S^n} := \{z_m^i | i \in \mathcal{S}^n\}$ .

We can now define the  $n$ th correlation matrix  $\{C_{ij}^n\}_{i,j=1,\dots,M}$  associated with subset  $\mathcal{S}^n$  of features

$$C_{ij}^n = \Phi^n(\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|; \eta^n, \tau, \kappa), \quad 1 \leq i, j \leq M, \quad 1 \leq n \leq N, \quad (5.12)$$

where  $\Phi^n$  is the radial basis kernel for the  $k$ th grouping. For example, one may choose

$$\Phi^n(\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|; \eta^n, \tau, \kappa) = \begin{cases} e^{-(\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|/\tau\eta^n)^\kappa}, & \|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\| < r_c^n \\ 0, & \text{otherwise,} \end{cases} \quad \text{or} \quad (5.13)$$

$$\Phi^n(\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|; \eta^n, \tau, \kappa) = \begin{cases} \frac{1}{1 + (\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|/\tau\eta^n)^\kappa}, & \|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\| < r_c^n \\ 0, & \text{otherwise,} \end{cases} \quad (5.14)$$

where truncation distance ( $r_c^n$ ) can be set to 2 or 3-standard deviations, and  $\eta^n$  is set to

$$\eta^n = \frac{\sum_{m=1}^M \min_{\mathbf{z}_j^{S^n}} \|\mathbf{z}_m^{S^n} - \mathbf{z}_j^{S^n}\|}{M}.$$

Then, we can project the data to an  $N$ -dimensional space representation by taking the rigidity function defined by correlation kernels,

$$x_i^n = \sum_{m=1}^M \omega_{im}^n \Phi^n(\|\mathbf{z}_i^{S^n} - \mathbf{z}_m^{S^n}\|; \eta^n, \tau, \kappa), \quad n = 1, 2, \dots, N; i = 1, 2, \dots, M \quad (5.15)$$

where  $\omega_{im}^n$  are the weights associated with  $\Phi^n$  for the  $n$ th cluster and can be set to 1. Moreover, the  $m$ th data in the reduced  $n$ -dimension representation is a vector  $\mathbf{x}_m = (x_m^1, \dots, x_m^n, \dots, x_m^N)^T$ .

We also propose to improve the computational efficiency in Eq. Equation 5.15 by avoiding the global summation. This can be easily done as follows. First, we construct an  $M \times M$  global distance matrix of the samples to obtain the nearest neighbors of each sample. Then, we use the cell lists

algorithm with the cutoff value to replace the global summations Eq. Equation 5.15 by considering only a few nearest neighbors [16]. This approach significantly reduces the memory requirement. Since the projections of  $x_i^n$  for different  $i$  and  $n$  are independent of each other, massively parallel computations can be used for large datasets.

### 5.3 Results

In this section, we numerically explore CCP’s performance on a variety of high dimensional benchmark test datasets. For each dataset, we use 10 random seeds for 5-fold or 10-fold cross-validations, depending on the number of samples of the data.

In order to validate the effectiveness of CCP, we compare it with Uniform Manifold Approximation and Projection (UMAP), Principle Components Analysis (PCA), Locally Linear Embedding (LLE), and Isomap.

For metric-based embedding, the Euclidean distance was used. All parameters were set to default, according to the packages outlined in Table 5.1. In order to test the effectiveness of the dimensionality reduction, a  $k$ -nearest neighbor classifier ( $k$ NN) was used. Table 5.1 shows the versions of the packages used in our comparison.

In order to visualize the accuracy, R-S scores were utilized. In R-S plots, the residue and similarity scores were represented as the  $x$  and  $y$  axes, respectively, and the data points were colored according to the predicted labels from classification results.

Table 5.1 Python packages used for the dimensionality reduction and benchmark tests.

Package
Python v3.8.5
Numpy v1.19.2
Scikit-learn v0.23.2
Scikit-learn-extra v0.2.0
Sklearn v0.0
umap-learn v0.5.1

#### 5.3.1 Datasets

We test CCP and several other commonly used algorithms on benchmark datasets, including Leukemia ALL-AML, Carcinoma, Arcene, TCGA-PANCAN, Coil-20 and Coil-100, and Small-

norb. Table 7.2 summarizes the datasets used in the present work.

Table 5.2 Datasets used in the benchmark tests.

Dataset [ref]	$(M, I, L)$	Description
Leukemia [18]	(72, 7070, 2)	Microarray dataset of Leukemia. The data contains 72 samples, each with 7070 gene expressions.
Carcinoma [19, 20]	(174, 9182, 11)	Microarray dataset of human carcionmas. Original data [19] contains 12533 genes, which were processed to 9182 dimensions in Ref. [20].
ALL-AML [21]	(72, 7129, 2)	Cancer classification dataset based on gene expressions by DNA microarrays of acute myeloid Leukemia (AML) and acute lymphoblastic Leukemia (ALL).
TCGA-PANCAN [22]	(801, 20531, 5)	Gene expression dataset. Part of the RNA-seq (HiSeq) PANCAN data, where expressions of 5 different types of tumors were extracted.
Coil-20 [23]	(1440, 16384, 20)	Image classification dataset with 1440 images. Each image has size $128 \times 128 = 16384$ , where 20 objects are captured at 72 angles. Each image was treated as a vector of length 16384.
Coil-100 [24]	(7200, 49152, 100)	Image classification dataset of 7200 images. Each image has size $128 \times 128 = 16384$ with 3 channels, where 100 objects are captured at 72 angles. Each image was treated as a vector of length 49152.
Smallnorb [25]	(24300, 18432, 5)	Image classification dataset with 5 generic categories: four-legged animals, human figures, airplanes, trucks, and cars. Each object was taken from a variety of radial and azimuthal angles. Each sample consists of 2 images, the left and the right views, both of size $96 \times 96$ . Both images for flattened to create a vector of length $2 \times 96 \times 96$ .

### 5.3.2 Validation

#### 5.3.2.1 Clustering analysis

Since CCP uses clustering to partition features based on correlations, it is vital to analyze clustering effectiveness.

One of the best methods to evaluate the effectiveness of the clustering from  $k$ -medoids and/or  $k$ -

means is to compute the loss function with respect to the varying number  $k$  of clusters. Then, using the elbow method, one finds the inflection point of the loss function and sets the corresponding  $k$  to be the optimal number of clusters.

In this work, we present another method for visualizing the feature clusters using our R-S scores. Figure 5.1 shows the loss function of the  $k$ -medoids feature clustering given in Eq. (5.8) for Coil-20 dataset for  $N = 2$  to 200. In addition, for  $N = 4, 16, 36$ , and 64, the clustering results were visualized using R-S scores. The red circles on the loss function correspond to  $N = 4, 16, 36$ , and 64, plotted in four charts. Each section in a given chart represents one cluster, and the  $x$  and the  $y$ -axes represent the residue and similarity scores, respectively for the cluster. For  $N = 4$ , the cluster colored in blue has a low similarity score, indicating that the data is spread out within the cluster. It indicates that a larger  $N$  is needed. From  $N = 36$  to  $N = 64$ , it is seen that the loss function has an elbow, indicating that these numbers of clusters are appropriate. The R-S plots in these two charts are more compact, indicating that the clustering performs well.

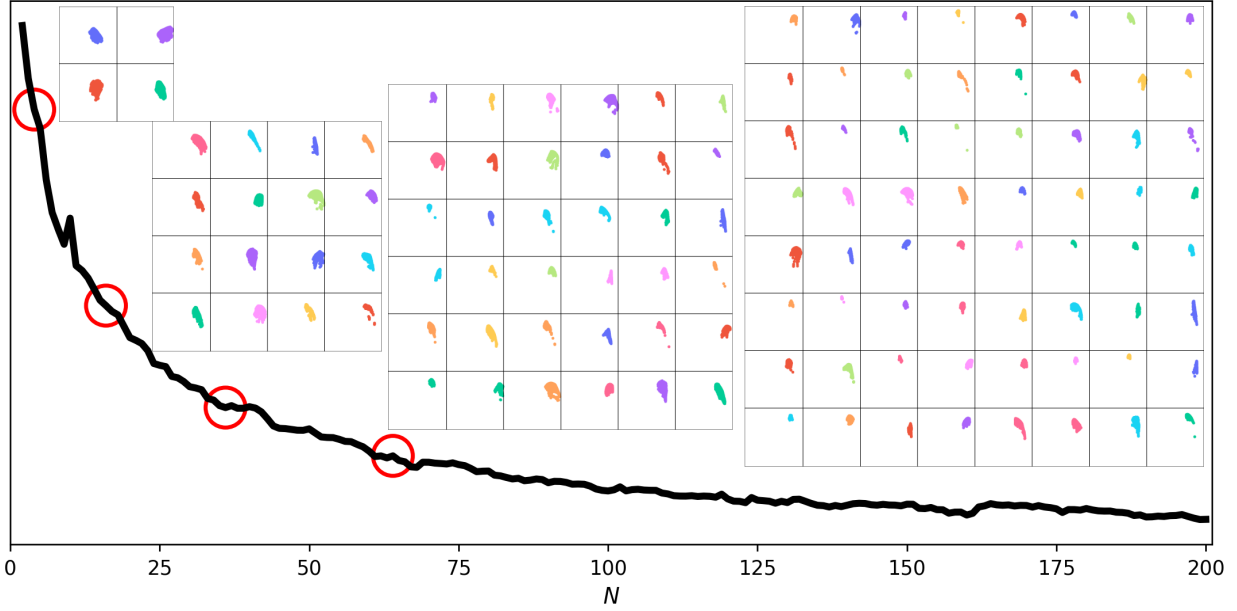


Figure 5.1 Illustration of the partition and R-S visualization of 16384 features of the Coil-20 dataset into different numbers ( $N$ ) of clusters. The line is the loss function defined in Eq. (5.8) with respect to different  $N$  values ranging from 2 to 200. Four red circles indicate  $N = 4, 16, 36$ , and  $64$ , for which the corresponding R-S charts are given in charts from the left to the right. Each section of the charts represents an individual feature cluster with the  $x$  and  $y$  axes are the residue and similarity scores, respectively. The R-S indices for  $N = 4, 16, 36$  and  $64$  are  $0.842, 0.887, 0.952$  and  $0.992$ , respectively.

As speculated earlier, the R-S disparity may correlate with the convergence of clustering. To verify this speculation, we have computed R-S disparity for the feature clustering results at  $N = 4, 16, 36$ , and  $64$ . We found that  $RSD_{N=4} = 0.158$ ,  $RSD_{N=16} = 0.113$ ,  $RSD_{N=36} = 0.048$ , and  $RSD_{N=64} = -0.008$ . Therefore, R-S disparity correlates strongly with the loss function of the  $k$ -medoids clustering shown in Figure 5.1.

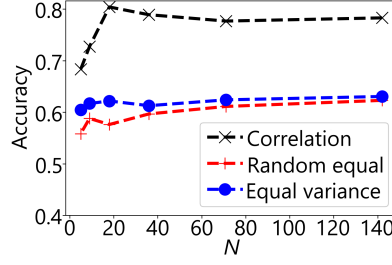
One of the advantages of the  $N$ -medoids method is that the cluster centers will always be one of the feature vectors. In addition, since each medoid is always one of feature vectors, the method is much more efficient when dealing with high dimensional data. Other clustering methods, such as  $k$ -means, spectral clustering, DBSCAN, and hierarchical clustering, can be utilized for the clustering as well.

### 5.3.2.2 Partition scheme evaluation

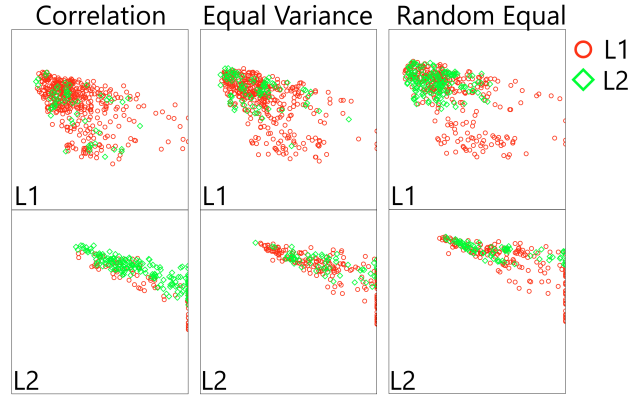
In order to explore the effectiveness of different partitions, we compare results obtained with three feature partitions: correlation partition, random equal partition, and equal variance partition.

In the random equal partition, the features are randomly shuffled and split into  $N$  equal-sized clusters (i.e.,  $N$  dimensions in the CCP). Therefore, each cluster has the same number of features, which will be projected into a one-dimensional representation in CCP. In equal variance partition, the features are normalized with respect to the largest one and ordered, and then, are split into  $N$  clusters such that all clusters have a similar amount of variance. In this partition, the first cluster contains the largest number of low-variance features, whereas the last cluster, cluster  $N$ , contains the least number of high-variance features. Notice that in the correlation partition, the numbers of features in clusters also vary and are determined by minimization according to Eq. (5.8).

The Leukemia and Carcinoma datasets were used to compare the 3 feature partition schemes. For both tests, 5-fold cross-validations with 10 random seeds were used for the dimensionality reduction, and  $k$ -NN was used to obtain classification accuracy. For each fold of partition, all results attained from 10 seeds were included to evaluate partition schemes.



(a) Comparison of the correlation partition, random equal partition, and equal variance partition-based dimensionality reduction and classification for the Leukemia dataset of 7070 features. The accuracies are computed from  $k$ NN classifications using the reduced  $N$  features generated from CCP with three partitions.



(b) R-S plots of clusters generated from CCP-based classification using correlation partition, equal variance partition, and random equal partition of the Leukemia dataset at  $N = 18$ .

Figure 5.2 Comparing the CCP-based classification effectiveness using correlation partition, equal random partition, and equal variance partition of the features of the leukemia dataset. For FRI, exponential kernel with  $\kappa = 2$  and  $\tau = 1.0$  was used. For each test, results from all 10 seeds were plotted. From left to right: R-S plots of correlation partition, equal variance partition, and random equal partition. The  $x$ -axis is the residual score, and the  $y$ -axis is the similarity score. Each section corresponds to one cluster and the data was colored according to the predicted labels from the  $k$ -NN classification.

Figure 5.2(a) shows the accuracy of CCP-based classification of the Leukemia dataset under various CCP reduced dimensions  $N$  equipped with 3 feature partition schemes. The correlation partition outperforms both equal random and variance partitions over all  $N$  values. In addition, as shown in Figure 5.2(b), for R-S plots, correlation partition outperforms other two partitions in each cluster at  $N = 18$ . In particular, equal random partition and equal variance partition do not work well in classifying label 2.

Figure 5.3 shows the accuracies of CCP-based classifications of the Carcinoma dataset under



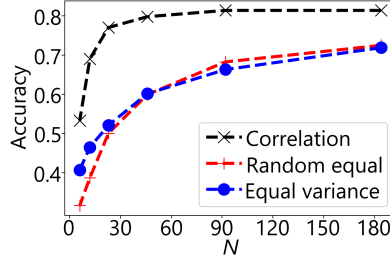


Figure 5.3 Comparison of the correlation partition, random equal partition, and equal variance partition-based dimensionality reduction and classification for the Carcinoma dataset of 7070 features. The accuracies are computed from  $k$ NN classifications using the reduced  $N$  features generated from CCP with three partitions.

various CCP reduced dimensions  $N$  equipped with 3 feature partition schemes. The correlation partition outperforms both equal random and equal variance partitions over all  $N$  values. In addition, as shown in Figure 5.4, for the R-S plots, the correlation partition outperforms the other two partitions in each cluster at  $N = 46$ .

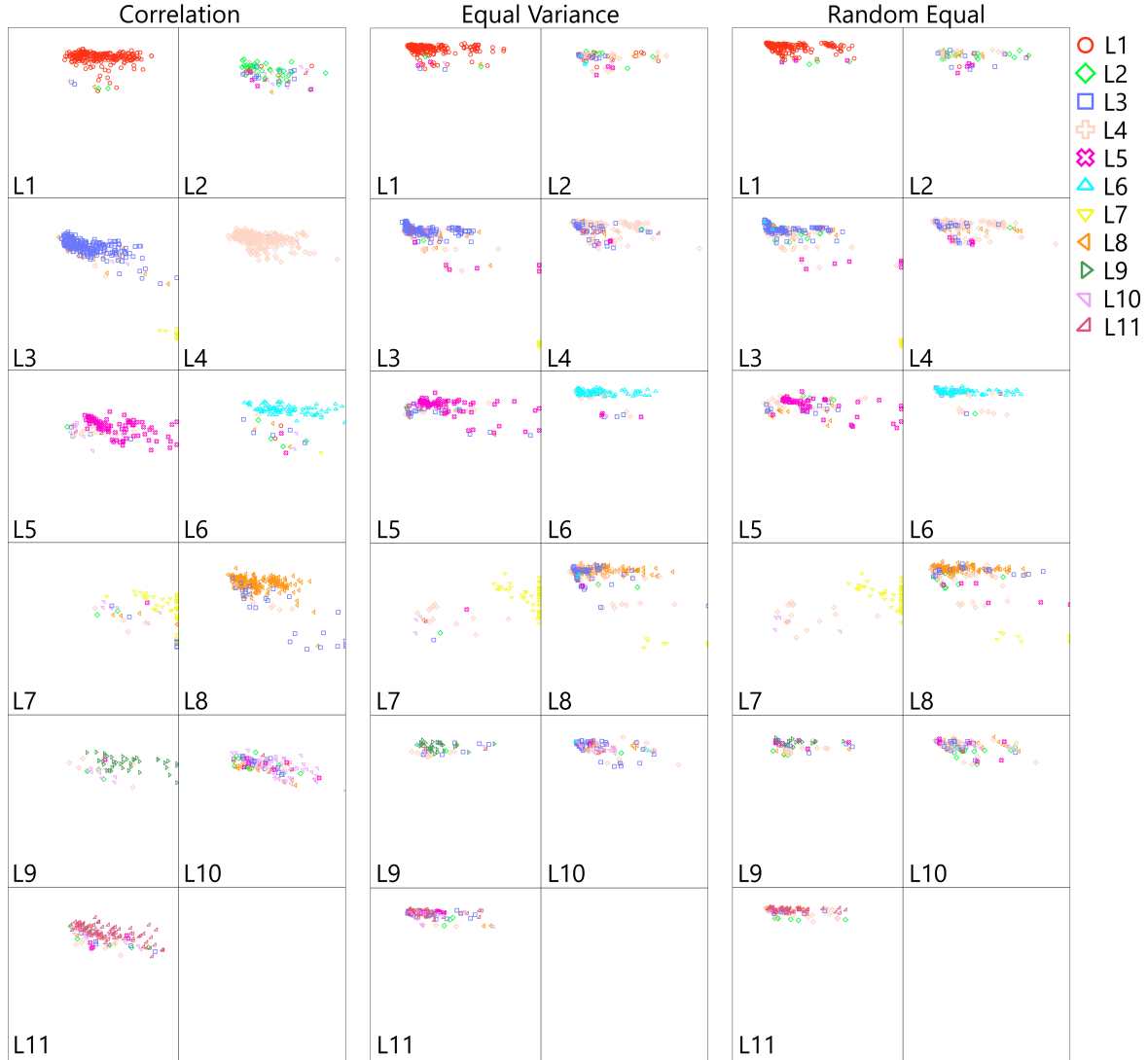


Figure 5.4 Comparing the CCP-based classification effectiveness using correlation partition, equal random partition, and equal variance partition of the features of the Carcinoma dataset. For FRI, exponential kernel with  $\kappa = 2$  and  $\tau = 2.0$  was used. For each test, the results of all 10 seeds were plotted. From left to right: R-S plots of correlation partition, equal variance partition, and random equal partition. The  $x$ -axis is the residual score, and the  $y$ -axis is the similarity score. Each section corresponds to one cluster and the data were colored according to the predicted labels from  $k$ -NN.

### 5.3.3 Comparison with other dimensionality reduction methods

In this section, we compare CCP's performance with UMAP, PCA, LLE, and Isomap on ALL-AML, TCGA-PANCAN, Coil-20, and Coil-100 datasets. For each dataset, we performed 5-fold or 10-fold cross-validation depending on the size of the dataset to test the accuracy using  $k$ -nearest neighbors ( $k$ -NN). Results of all 10 random seeds were used in performance evaluation.

### 5.3.3.1 ALL-AML

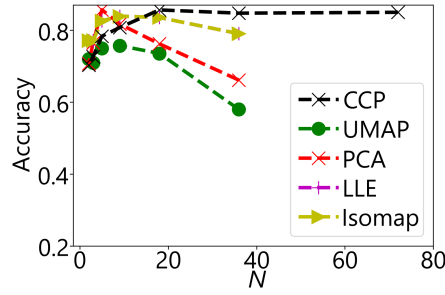


Figure 5.5 Accuracy of  $k$ -NN classification of the ALL-AML dataset when the dimension is reduced to  $N$  by using CCP, UMAP, PCA, LLE, and Isomap. Here, a 5-fold cross-validation with 10 random seedings was used. Test-train split was done prior to the dimensionality reduction. For CCP, exponential kernel with  $\kappa = 1$  and  $\tau = 2.0$  was used. The sample size, feature size, and the number of classes of the ALLAML dataset are 72, 7129, and 2, respectively.

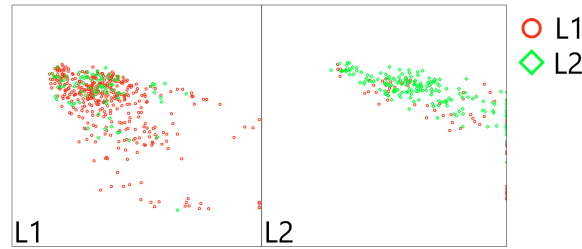


Figure 5.6 Visualization of the ALL-AML dataset when dimension reduced by CCP with exponential kernel,  $\kappa = 1$  and  $\tau = 2.0$  to  $N = 36$ . Each section represents a class and the samples were colored according to their predicted labels from the  $k$ -NN classification via 5-fold cross-validation. Results of all 10 seeds were used for the visualization. The  $x$  and  $y$  axes are the residue and similarity scores, respectively. The sample size, feature size, and the number of classes of the ALL-AML dataset are 72, 7129, and 2, respectively.

The dimension of the ALL-AML dataset was reduced using an exponential kernel with  $\kappa = 1$  and  $\tau = 2.0$ . Figure 5.5 shows the performance of CCP, UMAP, PCA, Isomap, and LLE. Here, a 5-fold cross-validation with 10 random seeds was used. CCP performs better than the other algorithms do and is stable with a wide range of  $N$  values. All other methods show a drop in their accuracy beyond dimension  $N = 36$ . Since the ALL-AML dataset only has 72 samples, UMAP, PCA, LLE, and Isomap cannot reduce the ALL-AML dimension to  $N > 72$  because their dimension is limited by the size of the matrix diagonalization.

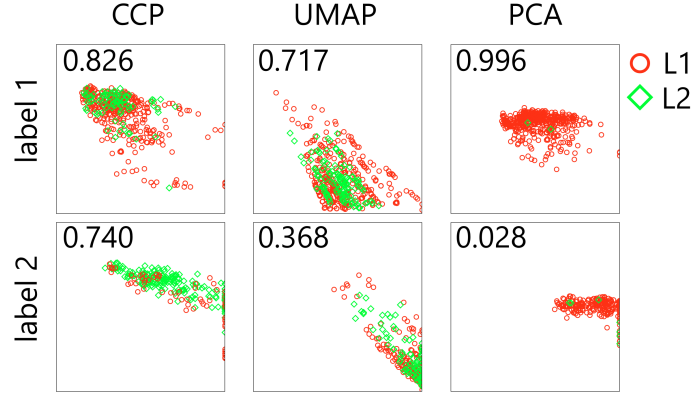


Figure 5.7 Visualization of the ALL-AML dataset when the dimension was reduced to  $N = 36$  by using CCP, UMAP, and PCA. Since there are only 72 samples in the ALL-AML dataset, results from all 10 seeds were plotted, leading to 720 sample points in the plot. For CCP, exponential kernel with  $\kappa = 1$  and  $\tau = 2.0$  was used. The  $x$  and  $y$  axes are the residue and similarity scores, respectively. Each row is for one class, and the data points are colored based on the predicted labels from the  $k$ -NN classifier, using 5-fold cross-validation. The sample size, feature size, and the number of classes of the ALL-AML are 72, 7129, and 2, respectively.

Figure 5.6 shows the R-S plot of the ALL-AML dataset when the dimension is reduced by CCP to  $N = 36$ . The left and right sections correspond to the 2 classes. Samples were plotted according to their 36 features and colored with the predicted labels from  $k$ -NN. Results of all 10 seeds were plotted into one chart (i.e., 720 samples), and the residue and the similarity scores were calculated separately for each random seed. The  $x$  and the  $y$ -axes are the residue and similarity scores, respectively. Class 2 has a better R-S distribution.

Figure 5.7 shows the R-S plot of ALL-AML when the feature dimension is reduced to  $N = 36$  by using CCP, UMAP, and PCA. Results of all 10 random seeds were used in the visualization to obtain a better understanding of the performance, and the residue and similarity scores were computed separately for each seed. In each class, the samples were colored according to their predicted labels obtained from  $k$ -NN. The  $x$ -axis and  $y$ -axis of each R-S plot are the residue and similarity scores, respectively. The top row is class 1 (ALL), and the bottom row is class 2 (AML). The numerical number inside the plot is the accuracy for the class. Notice that UMAP's R-S plot indicates that UMAP's reduction has a low similarity score, which explains its low accuracy. On the other hand, PCA has higher accuracy than that UMAP, but most AML samples are mislabeled. This indicates that PCA is unable to distinguish the difference between ALL and AML when  $N = 36$ .

### 5.3.3.2 TCGA-PANCAN

For CCP, the dimension of the TCGA-PANCAN dataset was reduced using Lorentz kernel with  $\kappa = 1$  and  $\tau = 1.0$ . Figure 5.8 shows the performance comparison of CCP, UMAP, PCA, Isomap, and LLE. Here, a 5-fold cross-validation with 10 random seeds was used. Notice that CCP is comparable to Isomap and LLE in accuracy, whereas UMAP and PCA are unstable at higher dimensions.

Figure 5.9 shows the R-S plot of the TCGA-PANCAN dataset when the dimension was reduced by CCP to  $N = 103$ . Each section corresponds to the 5 classes of TCGA-PANCAN. Samples were plotted according to 103 features and colored with the predicted labels from  $k$ -NN. The  $x$  and the  $y$ -axes are the residue and similarity scores, respectively.

Figure 5.10 shows the R-S plot of TCGA-PANCAN when the dimension is reduced to  $N = 103$  by using CCP, UMAP, and PCA, respectively. The samples were plotted based on 103 features and colored with their predicted labels from  $k$ -NN. The  $x$ -axis and  $y$ -axis of each plot are the residue and similarity scores, respectively. Each row corresponds to one of the 5 classes, and the number inside the plot is the accuracy for each class. Notice that UMAP has a cluster in each plot, but the cluster has a low similarity score. This means that in UMAP's embedding, the sample within each class is not near each other, which results in low accuracy. PCA has comparable accuracy to CCP, but CCP has a notable improvement for class 1 and class 4.

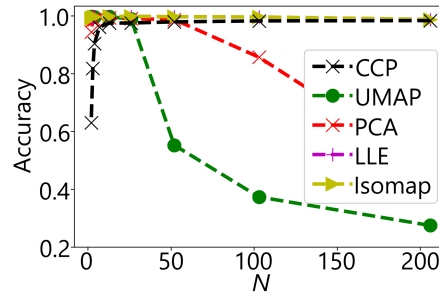


Figure 5.8 Accuracy of  $k$ -NN classification of the TCGA-PANCAN dataset when the dimension is reduced to  $N$  by using CCP, UMAP, PCA, LLE, and Isomap. Here, 5-fold cross-validation with 10 random seedings was used, and the test-train split was done prior to the reduction. For CCP, Lorentz kernel with  $\kappa = 1$  and  $\tau = 1.0$  was used. The sample size, feature size, and the number of classes of the TCGA-PANCAN are 801, 20531, and 5, respectively.

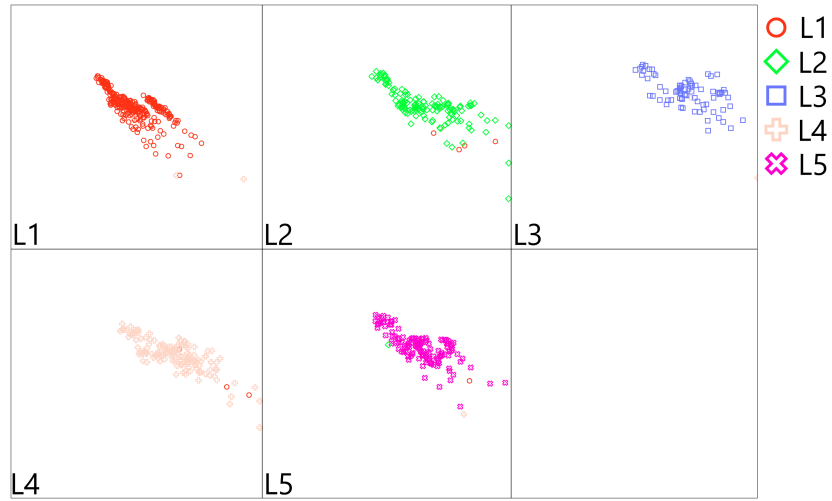


Figure 5.9 Visualization of the TCGA-PANCAN dataset when the dimension is reduced to  $N = 103$  by using CCP with Lorentz kernel,  $\kappa = 1$  and  $\tau = 1.0$ . Each section represents a different class. The samples were plotted based on 103 features and colored with their predicted labels from  $k$ -NN classification via 5-fold cross-validation. The  $x$  and  $y$  axes are the residue and similarity scores, respectively. The sample size, feature size, and the number of classes of the TCGA-PANCAN are 801, 20531, and 5, respectively.

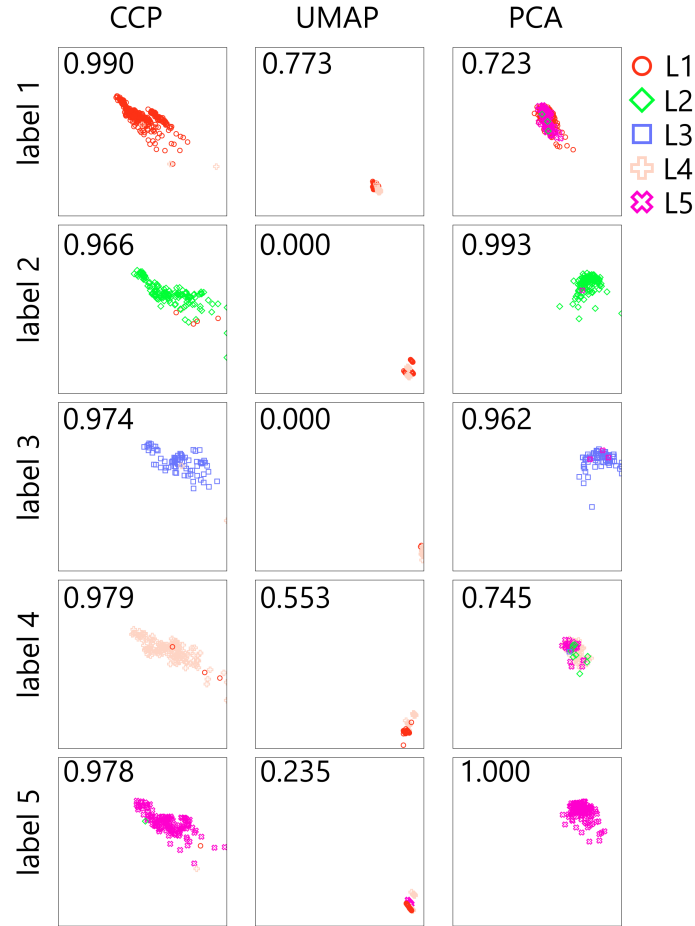


Figure 5.10 Visualization of TCGA-PANCAN dataset when the dimension is reduced to  $N = 103$  by using CCP, UMAP, and PCA. For CCP, Lorentz kernel with  $\kappa = 1$  and  $\tau = 1.0$  was used. The  $x$  and  $y$  axes are the residue and similarity scores, respectively. Each row contains a class. The data is plotted based on 103 features and colored with the predicted labels of the  $k$ -NN classifier, using 5-fold cross-validation. The sample size, feature size, and the number of classes of the TCGA-PANCAN are 801, 20531, and 5, respectively.

### 5.3.3.3 Coil-20

The dimension of the Coil-20 dataset was reduced using Lorentz kernel with  $\kappa = 1$  and  $\tau = 6.0$ . Figure 5.11 shows the performance of CCP, UMAP, PCA, Isomap, and LLE. The 10-fold cross-validation with 10 random seeds was used. CCP has the best performance out of the 5 algorithms and maintains its accuracy in higher dimensions. PCA also has high accuracy but loses its accuracy in higher dimensions.

Figure 5.12 shows the R-S plot of the Coil-20 dataset when the dimension is reduced by CCP to  $N = 82$ . Each section corresponds to the 20 classes of Coil-20. Samples were plotted based on 82 features and colored with the predicted labels from  $k$ -NN. The  $x$  and the  $y$ -axes are the residue and similarity scores, respectively.

Figure 5.13 shows the R-S plot of Coil-20 when its dimension is reduced to  $N = 82$  by using CCP, UMAP, and PCA. Samples in each class are plotted according to their 82 features and colored according to their predicted labels from  $k$ -NN. The  $x$ -axis and  $y$ -axes of each plot are the residue and similarity scores, respectively. Each row corresponds to one of the 20 classes, and the number inside the plot is the classification accuracy for each class. Notice that all of UMAP's visualizations show a poor distribution in the bottom right, indicating that the residual score is high and the similarity score is low, which gives rise to poor performance in the classification. In order to further investigate the performance, labels 1, 2, and 3 were visualized in Figure 5.15. We can see that in the zoomed-in view, there are small subclusters within each plot, which come from different folds of the cross-validation.



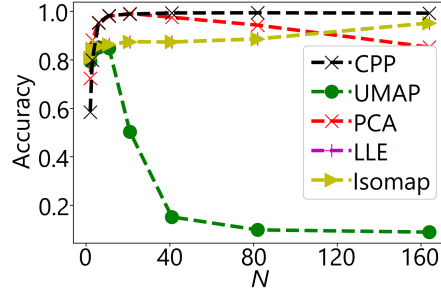


Figure 5.11 Accuracy of  $k$ -NN classification of Coil-20 dataset when its dimension is reduced to different dimensions  $N$  by using CCP, UMAP, PCA, LLE, and Isomap. The 10-fold cross-validation with 10 random seedings was used, and the test-train split was done prior to the dimensionality reduction. For CCP, Lorentz kernel with  $\kappa = 1$  and  $\tau = 6.0$  was used. The sample size, feature size, and the number of classes of the Coil-20 are 1440, 16384, and 20, respectively.

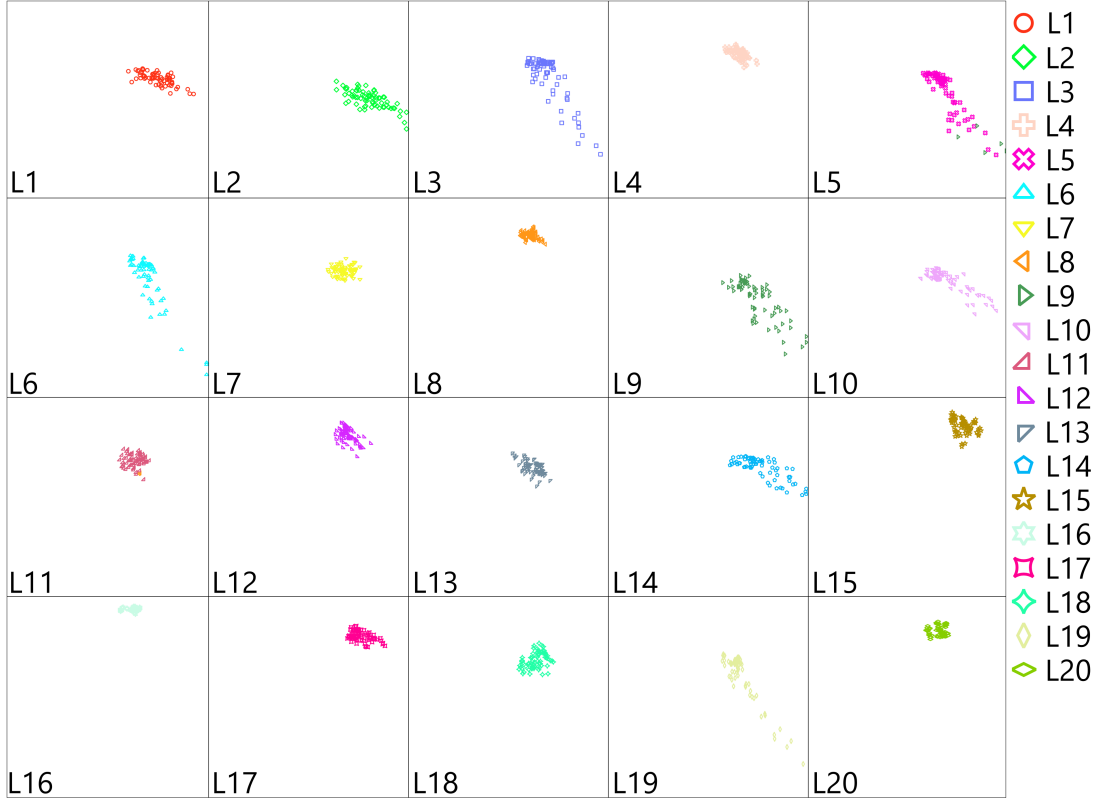


Figure 5.12 Visualization of Coil-20 dataset when the dimension is reduced to  $N = 82$  by using CCP with Lorentz kernel,  $\kappa = 1$  and  $\tau = 6.0$ . Each section represents a different class, and the samples were plotted based on 82 features and colored with their predicted labels from the  $k$ -NN classification via 10-fold cross-validation. The  $x$  and  $y$  axis are the residue and similarity scores, respectively. The sample size, feature size, and the number of classes of the Coil-20 are 1440, 16384, and 20, respectively.

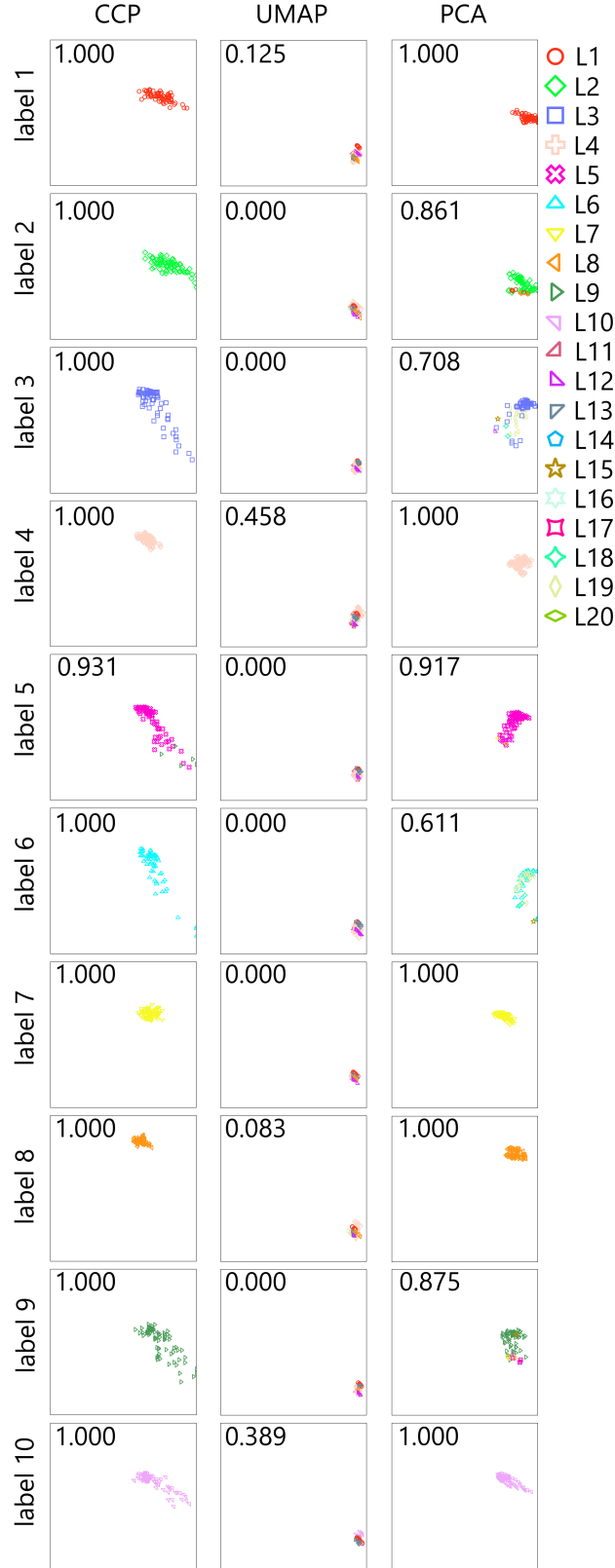


Figure 5.13 Visualization of Coil-20 dataset for classes 1 to 10 when the dimension is reduced to  $N = 82$  by using CCP, UMAP, and PCA. For CCP, Lorentz kernel with  $\kappa = 1$  and  $\tau = 6.0$  was used. The  $x$  and  $y$  axis are the residue and similarity scores, respectively.

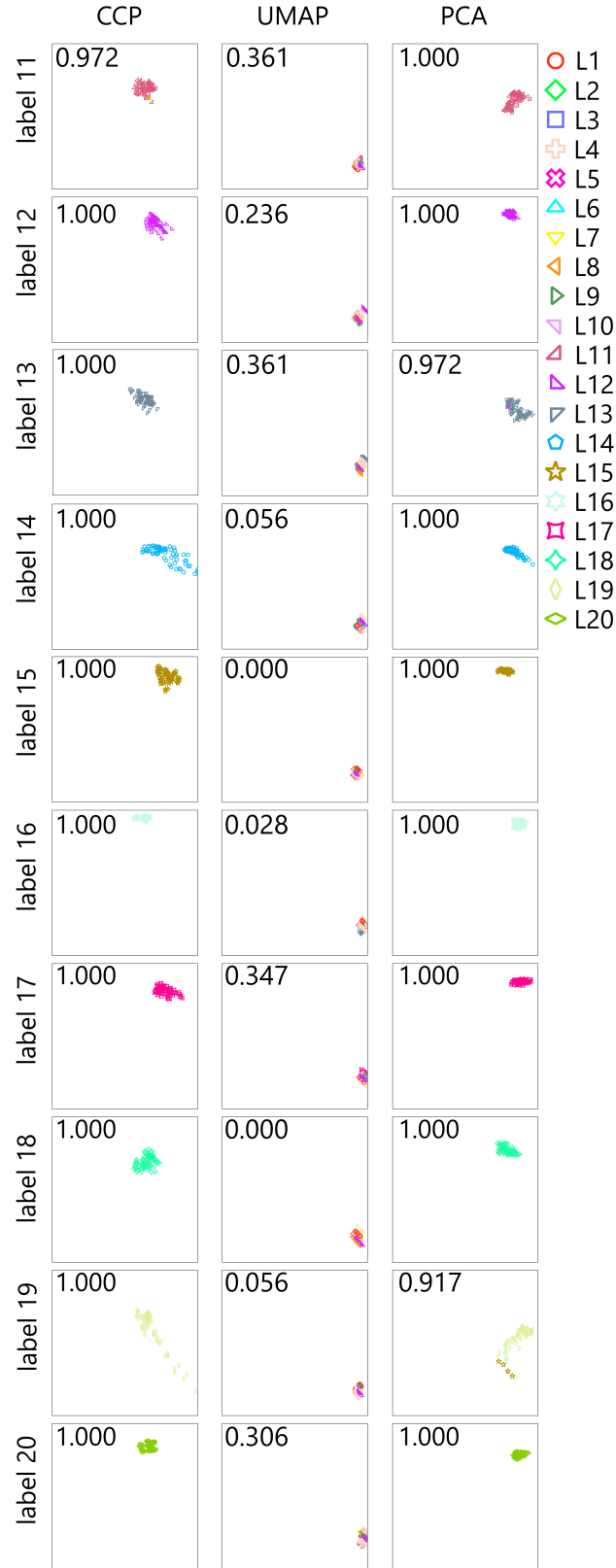


Figure 5.14 Visualization of Coil-20 dataset for classes 11 to 20 when the dimension is reduced to  $N = 82$  by using CCP, UMAP, and PCA. For CCP, Lorentz kernel with  $\kappa = 1$  and  $\tau = 6.0$  was used. The  $x$  and  $y$  axis are the residue and similarity scores, respectively.

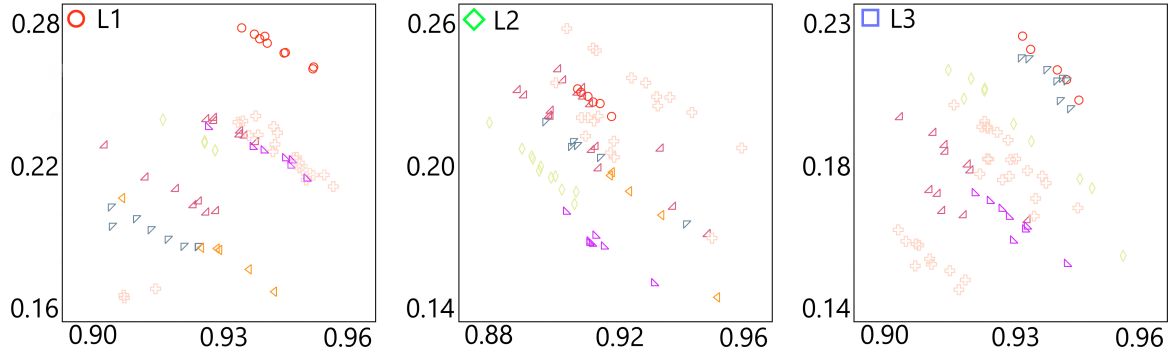


Figure 5.15 Visualization of Coil-20 dataset class 1, 2, and 3, when the data dimension is reduced to  $N = 82$  by UMAP. The figures are zoomed-in view. The data were plotted based on 82 features and colored according to their predicted labels from the  $k$ -NN classifier using 10-fold cross-validation. Label 1 has an accuracy of 0.125, whereas labels 2 and 3 have accuracy 0.000.

#### 5.3.3.4 Coil-100

The dimension of the Coil-100 dataset was reduced using exponential kernel with  $\kappa = 1$  and  $\tau = 6.0$ . Figure 5.16 shows the performance of CCP, UMAP, PCA, Isomap, and LLE. Here, 10-fold cross-validation with 10 random seeds was used. CCP, PCA, LLE, and Isomap have comparable results, whereas UMAP is unstable at a higher dimension  $N$ . The best performance of UMAP was not as good as those of CCP and PCA. This indicates that Coil-100 has a high intrinsic dimension, for which UMAP has poor performance.

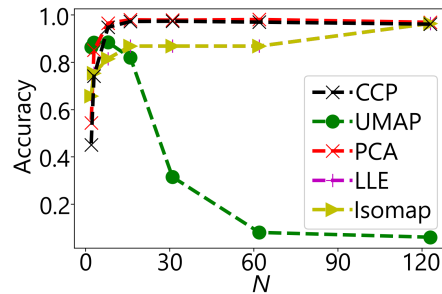


Figure 5.16 Accuracy of  $k$ -NN classification of Coil-100 dataset when the dimension is reduced to  $N$  by using CCP, UMAP, PCA, LLE, and Isomap. The 10-fold cross-validation with 10 random seedings was used, and a test-train split was done prior to the reduction. For CCP, Lorentz kernel with  $\kappa = 1$  and  $\tau = 6.0$  was used. The sample size, feature size, and the number of classes of the Coil-20 are 1440, 16384 and 20, respectively.

## 5.4 Discussion

### 5.4.1 Centrality based CCP

CCP used FRI to project a group of correlated features into a 1D representation. If we observe the projection in a graph setting, the FRI projection can be viewed as computing the degree centrality of the graph. That is, let  $Z \in \mathbb{R}^{M \times I}$  be the data, with  $M$  samples and  $I$  features. For each partition, we can define a graph  $G^n = (V^n, E^n, W^n)$ ,  $n = 1, 2, \dots, N$ , where  $V^n$ ,  $E^n$  and  $W^n$  are the vertex, edge and weight sets of the graph of the  $n$ th component, respectively. The weights are precisely the kernels defined in Eq. (5.12). Then, the FRI projection for  $\mathbf{x}_i^n$  can be viewed as the degree centrality ( $C_d$ ) of a weighted graph

$$C_d(\mathbf{z}_i^{S^n}) = \sum_{\mathbf{z}_j^{S^n}} \Phi^n(\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|; \eta^n, \tau, \kappa), \quad (5.16)$$

where  $C_d(\mathbf{z}_i^{S^n})$  is the degree centrality of vertex  $\mathbf{z}_i^{S^n}$ . In this case, we treat each data  $\mathbf{z}_i^{S^n}$  as a vertex.

Instead of using the FRI projection, we can impose a traditional graph-based approach, setting the edge weight  $\omega_{ij}^n = 1$  for all  $1 \leq i, j \leq M$  and  $1 \leq n \leq N$ , when the node-node distance satisfies a cutoff. That is, instead of applying Eq. (5.12), we take

$$A^n = \{A_{ij}^n\}, \quad A_{ij}^n = \begin{cases} 1, & \text{if } \|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\| < r_c^n \\ 0, & \text{otherwise} \end{cases}, \quad 1 \leq i, j \leq M. \quad (5.17)$$

Here, instead of writing  $C_{ij}^n$  as in Eq. (5.12), we use  $A_{ij}^n$  to denote the adjacency matrix of the graph, and  $r_c^n$  is the cutoff distance. Then, the reduced new variables  $x_i^n$  can be computed by replacing  $\Phi^n(\|\mathbf{z}_i^{S^n} - \mathbf{z}_j^{S^n}\|; \eta^n, \tau, \kappa)$  in Equation 5.15 with  $A_{ij}^n$ . In such a manner, we can implement other centrality formulations, such as the degree centrality, closeness centrality [26], betweenness centrality [27], and eigenvector centrality [28] in CCP.

Figure 5.17 shows the accuracy of using different centrality formulations instead of the FRI projection. Using the adjacency matrix, degree centrality, closeness centrality, betweenness centrality, and eigenvector centrality were computed with  $r_c = 0.7d_{\max}$ , where  $d_{\max}$  is the maximum pairwise distance between the input data. The performance of all methods is quite similar. However, the

stability of computing the centrality is heavily reliant on  $r_c$ . Moreover, if the data is well clustered within each class, the graph may not be connected, which may affect the stability of centrality computations.

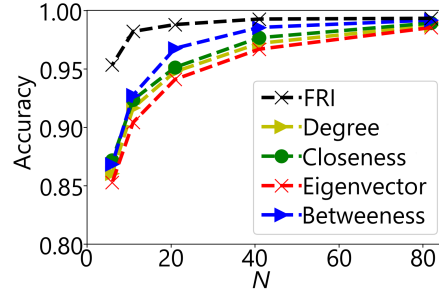


Figure 5.17 The coil-20 dataset was reduced using centrality formulations, instead of the FRI projection. Degree, closeness, eigenvector centrality, and betweenness centrality were tested, with  $r_c = 0.7d_{\max}$ . The accuracy was calculated from 10-fold cross-validation with 10 random seeds. The sample size, feature size, and the number of classes of the Coil-20 are 1440, 16384, and 20, respectively.

#### 5.4.2 Correlation distance based CCP

CCP utilizes covariance distance in clustering to partition features. However, other distance metrics can be used in clustering as well, depending on the size of the data and the relationship between the features. In particular, correlation distance can be used instead of covariance distance, when the relationship between features is highly nonlinear. Figure 5.18 shows the effectiveness of correlation distance-based CCP when compared to covariance distance-based CCP and other DR algorithms. Notice that the correlation distance-based CCP significantly outperforms covariance-based CCP and other DR algorithms. Therefore, correlation distance-based CCP can be employed if high accuracy is desirable. However, it is noted that correlation distance-based CCP is very time-consuming and memory-demanding. This limitation may constrain the use of correlation distance-based CCP in high-dimensional data with large data sizes.

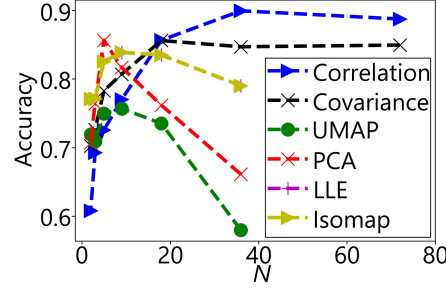


Figure 5.18 Comparison between correlation distance-based partitioning and covariance distance-based partitioning of ALL-AML dataset.  $k$ -NN with 10-fold cross-validation was used to compute the accuracy. The sample size, feature size, and the number of classes of ALL-AML are 72, 7129, and 2, respectively.

### 5.4.3 Parameter-free CCP

The performance of the proposed two-step CCP depends on a few parameters, such as the dimension  $N$ , kernel type (i.e., generalized exponential and generalized Lorentz), power ( $\kappa$ ), and scale ( $\tau$ ). Among them, the dimension may be chosen by the user. Although a set of default parameters is prescribed, it may not be optimal for different datasets. It will be a burden for users to select parameters. Fortunately, CCP is very stable under subsampling. Therefore, we can use subsampling to search the optimal parameter range for a given dataset automatically.

In this subsection, we show that CCP is stable under subsampling. To verify this claim, we test CCP on the Smallnorb dataset, which has 46,800 samples and 5 classes. Each sample consists of a binocular picture of an object of size 96×96 pixels, taken from different radial and azimuthal angles. We flattened each image and combined the images to make an 18,432-dimensional feature vector. We subsample 1%, 5%, 10% and 20% samples to optimize CCP kernel parameters, respectively. Then, based on these CCP parameter sets, we carry out the CCP dimensionality reduction of the whole dataset for classification. The resulting 10-fold cross-validation accuracies of classification for the Smallnorb dataset are shown in Figure 5.19 for subsampled at 1%, 5%, 10% and 20%. It is clear that the accuracy increases as the subsampling size is increased from 1% to 20%. However, the accuracy difference between 1% subsampling and 20% subsampling is under 2% for all classes. It is seen that under different subsampling ratios, CCP can capture the structure of the data. Even at 1% subsampling, CCP is still very accurate.

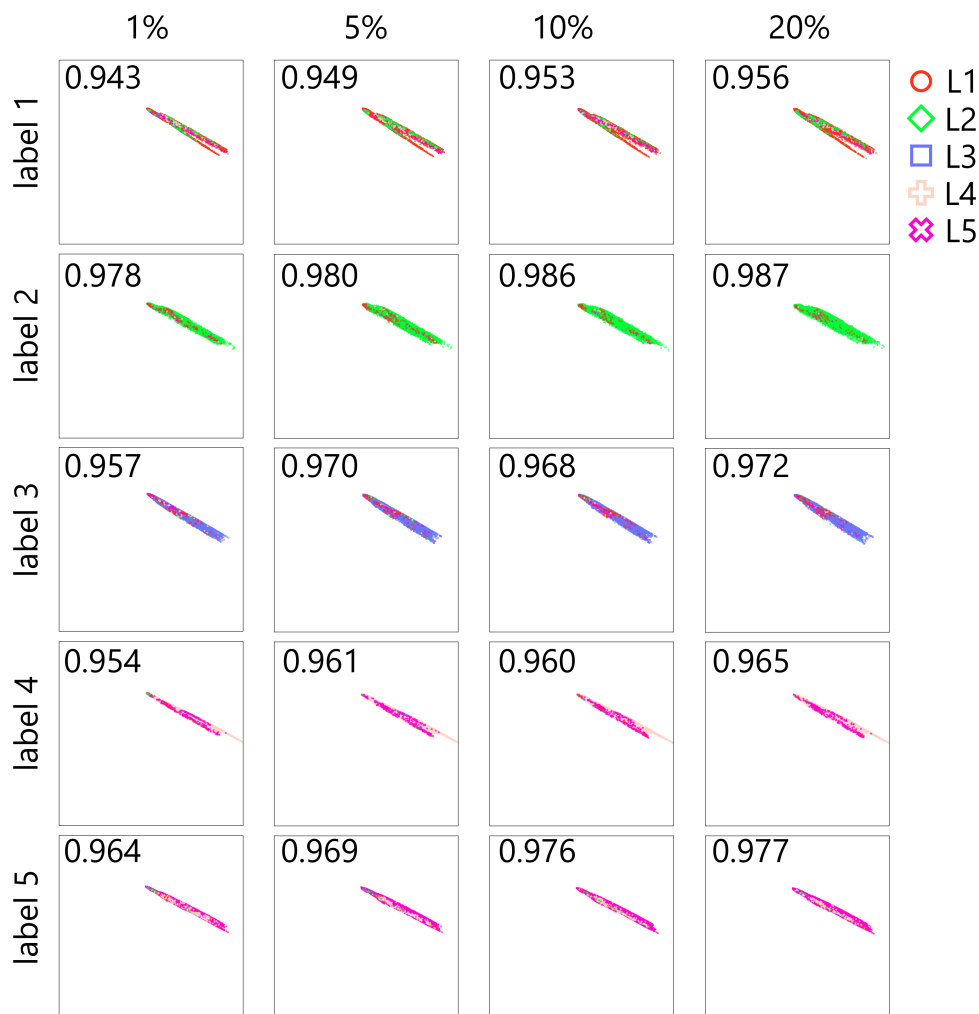


Figure 5.19 R-S plot visualization of Smallnorb classification using CCP with the reduction ratio of 400 (47 dimensions) at 1%, 5%, 10% and 20% subsampling. Each row represents the data plotted based on 47 features and colored with the predicted labels from the  $k$ -NN classifier, using 10-fold cross-validation. The number in each plot shows the accuracy within each label obtained with subsampling-generated kernel parameters. The  $x$  and  $y$  axis are the residue score and the similarity scores, respectively.

Since CCP is very stable under subsampling, one can make CCP a parameter-free method by using a relatively small amount of dataset to determine CCP parameters automatically.

CCP's stability under subsampling implies that CCP can be used in the dynamic data acquisition of excessively large datasets. Newly collected data can be added to the existing data without the need to restart the CCP calculation from the very beginning.



#### 5.4.4 Accuracy comparison using four classifiers

We have shown the effectiveness of CCP on various datasets. However, all the aforementioned analysis was based on the  $k$ -NN classifier. It is important to know whether the same pattern returns if other classification algorithms are employed. To this end, we compare CCP with other dimensionality reduction methods using  $k$ -NN, support vector machine (SVM), random forest (RF), and gradient boost decision tree (GBDT).

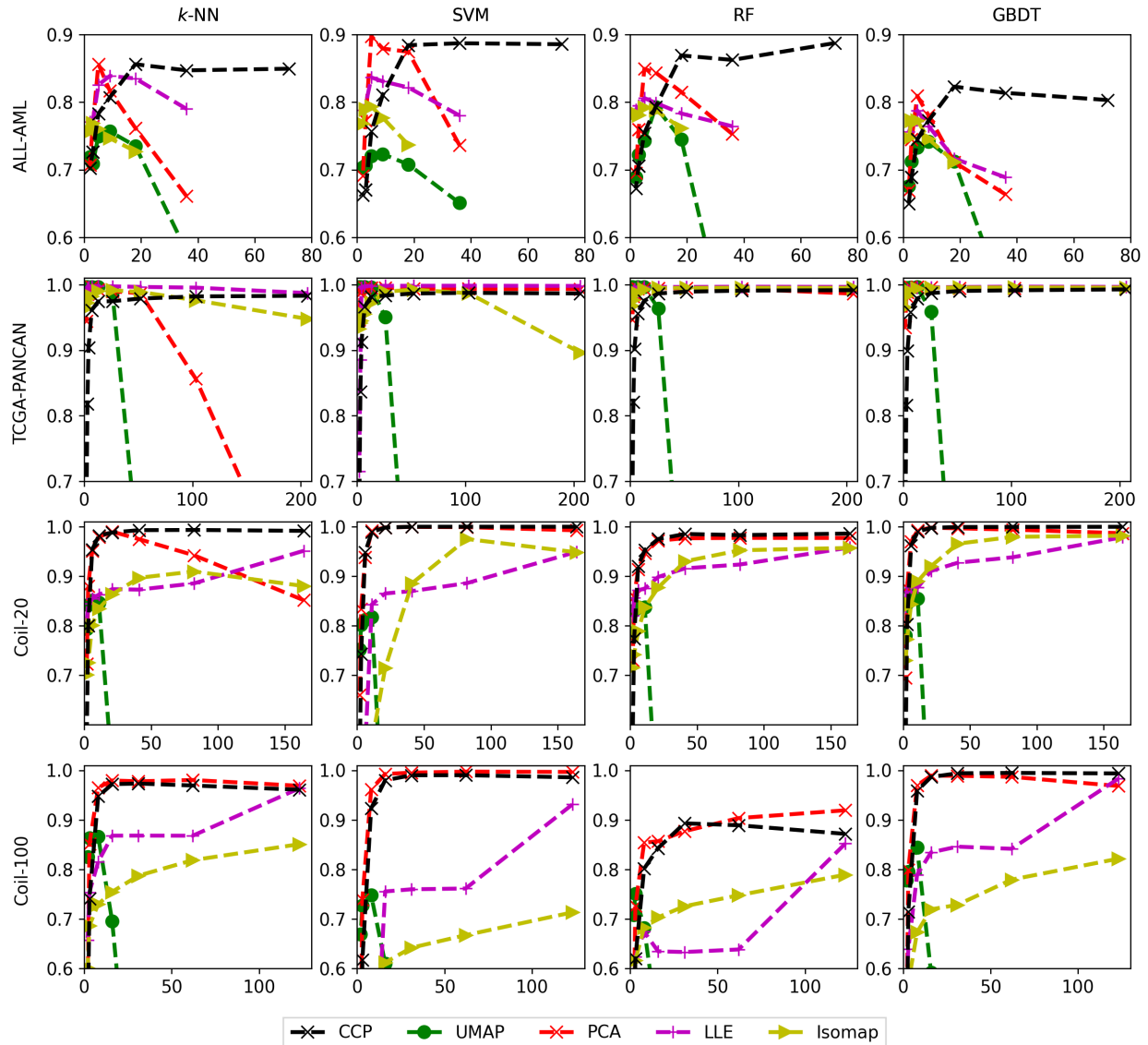


Figure 5.20 Comparison of the accuracy of CCP on a variety of datasets and classification algorithms. The rows represent four datasets, from the top to the bottom: ALL-AML, TCGA-PANCAN, Coil-20, and Coil-100. The columns are for four classification algorithms, namely,  $k$ -NN, SVM, RF, and GBDT. The  $x$ -axes are the reduced dimension  $N$  and the  $y$ -axes are accuracy.

Figure 5.20 shows the comparison of CCP when utilizing  $k$ -NN, SVM, RF, and GBDT on ALL-AML, TCGA-PANCAN, Coil-20, and Coil-100 datasets. The rows are the 4 datasets, and the columns are 4 classification methods. For all the tests, sklearn's classification package was utilized. For  $k$ -NN and SVC, default parameters were used. For RF and GBDT, `{n_estimators=1000, max_depth = 7, min_samples_split = 3, max_features = 'sqrt', n_jobs = -1 }` were used. For all tests, standard scaling was used after the reduction.

First, CCP remains very competitive against all other dimensionality reduction methods over all datasets when other classifiers are employed. The relative behaviors of all dimensionality reduction methods did not change much under different classifiers. Therefore, our earlier comparison is fair and our findings remain correct.

Second, SVM appears to slightly improve the performance of CCP and PCA. However, LLE and Isomap do not work well with SVM.

Third, UMAP did not perform well on ALL-AML, Coil-20, and Coil-100 when the  $k$ -NN method was used. However, its performance does not improve much with SVM, RF, and GBDT. Its instability with relatively large reduced dimension  $N$  persists over different classifiers. In fact, its best results have never reached those of other methods for these three datasets. A possible reason is that UMAP does not work well for data having moderately large intrinsic dimensions.

Fourth, LLE had some instability in TCGA-PANCAN and Coil-100 datasets. Because the input data led the computed matrix to become singular, some of the tests from the cross-validation were not computed. For these cases, the average was taken over the working tests.

Finally, we noticed that all dimensionality reduction methods underperformed with RF for the Coil-100 dataset and with GBDT for the ALL-AML dataset. This behavior might be due to the fact that for a given classifier, a uniform set of parameters was used for all datasets and RF does not work well for large datasets.

#### 5.4.5 Efficiency comparison

Although accuracy is very important, computational cost can be a crucial factor for huge datasets. In this section, we assess the computational times of various methods with elementary computer

resource allocations. Specifically, 4 central processing units (CPUs) with 64GB of memory from the High-Performance Computing Center (HPCC) of Michigan State University were used for all methods and all datasets.

Figure 5.21 shows the computational time of the three-dimensionality reduction methods on ALL-AML, TCGA-PANCAN, Coil-20, and Coil-100. For ALL-AML and TCGA-PANCAN datasets, the average time from the 5-fold cross-validation over 10 random seeds was computed. For Coil-20 and Coil-100, the average time from the 10-fold cross-validation over 10 random seeds was recorded.

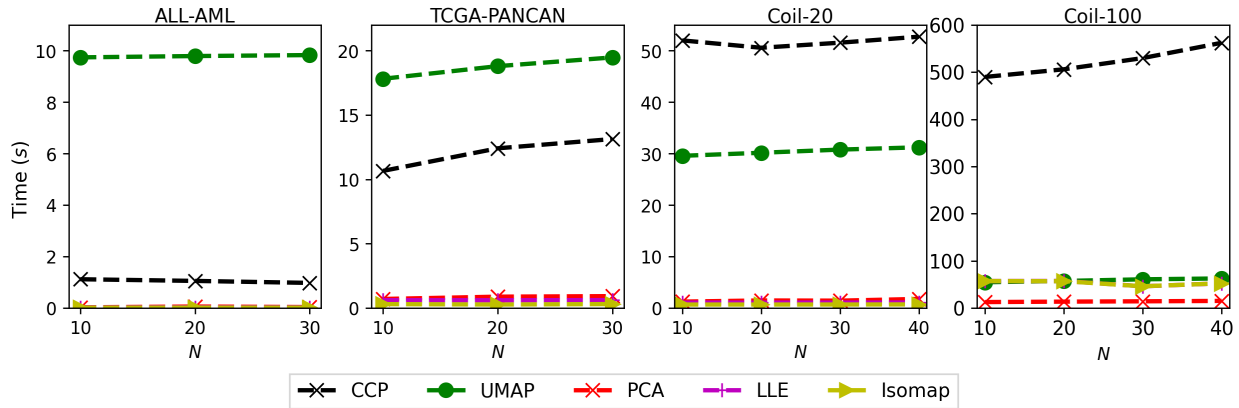


Figure 5.21 CPU run time comparison among CCP, UMAP, and PCA on ALL-AML, TCGA-PANCAN, Coil-20, and Coil-100 datasets. For ALL-AML and TCGA-PANCAN, computational times for  $N = 10, 20$ , and  $30$  were calculated by taking the average of the 5-fold cross-validation over 10 random seeds. For Coil-20 and Coil-100 computational times for  $N = 10, 20, 30$ , and  $40$  were calculated by taking the average of the 10-fold cross-validation over 10 random seeds. In each chart, the  $x$  axis corresponds to the reduced dimension  $N$ , and the  $y$  axis is the average time (s).

PCA shows essentially the fastest computation for all datasets. Isomap and LLE have very similar behaviors for all datasets. Their time efficiencies are quite similar to that of PCA.

UMAP is faster than CCP for Coil-20 and Coil-100. For ALL-AML and TCGA-PANCAN, CCP is faster because of the small data size. Note that Equation 5.15 indicates the summation over all samples that satisfies cutoff of within 3 standard deviations of the average pairwise distance. This cutoff can be reduced for faster computation. However, it reduces the overall accuracy.

For CCP, because clustered features are projected independently, each reduced dimension can be computed independently using the parallel architecture. Similar parallel computations can be

applied to different samples. Therefore, CCP can be further accelerated by using parallel and graphics processing unit (GPU) algorithms in practical applications.

## 5.5 Concluding remarks

Like other dimensionality reduction algorithms, CCP has its advantage and disadvantages. First, CCP is a unique data-domain method and its features are highly interpretable. Because CCP partitions feature into clusters according to some metric, such as covariance distance or correlation distance, features with high correlation will perform better. One limitation for many methods relying on matrix diagonalization is that pairwise distance computation can encounter the “curse of dimensionality”, where distance computation of high dimensional data could become unreliable. By clustering features, CCP can more reliably compute distances because the dimension in each cluster will be much lower. Moreover, CCP performs better for data with a large number of features, such as TCGA-PANCAN, Coil-20, and Coil-100 datasets. Therefore, CCP is suitable for the dimensionality reduction of data with relatively large intrinsic dimensions, for which many other popular methods may not work well.

However, for datasets with a smaller number of features, CCP may not be as good as other methods. In this case, dimensionality reduction is unnecessary anyway. Also, we noticed that CCP might not be as good as UMAP and some other frequency-domain methods for extremely low final dimensions, say  $N = 2$  or  $3$ .

In addition to doing well for data having moderately large intrinsic dimensions, CCP allows embedding for streamlined datasets, such as molecular dynamics generated transient data. We have shown that CCP is stable under subsampling, which enables users to optimize the CCP model with a small portion of initial data, and allows subsequent data to be embedded with the initial set. We noticed that dimensionality reduction algorithms that rely on matrix diagonalization have instability when dealing with streamlined data.

Because CCP does not compute the nearest neighbors graph and does not diagonalize, a traditional 2D plot does not give a meaningful visualization. However, each dimension of CCP is computed by projecting the partitioned features. Hence we can easily interpret each dimension

of CCP. In tree-based classification algorithms, such as random forest and gradient boost decision trees, feature importance can be computed for each feature component, which gives a rank on how much impact each component has on the classification. For CCP, feature importance may be interpreted as how meaningful a set of highly correlated features is in the classification.

CCP can be further optimized in various ways. It allows a wide variety of alternative data-domain embedding strategies in each of its two steps: clustering and projection. For example, in the clustering step, one might select alternative distance metrics, clustering algorithms, and loss functions to optimize feature vector partition for a given dataset. In the project step, one might choose alternative distance metrics based on Riemannian geometry or statistical theories and select alternative projections based on linear/nonlinear, orthogonal/non-orthogonal, and Grassmannian considerations.

A wide variety of multistep dimensionality reduction methods can be developed. Unlike frequency-domain dimensionality reduction techniques, CCP renders a data-domain representation of the original high-dimensional data. Therefore, the resulting low-dimensional data can be reused as an input for a Secondary Dimensionality Reduction (SDR) with a frequency-domain technique to achieve specific goals. For example, one can use CCP as an initializer for local methods to capture global patterns [29]. The combination of CCP with UMAP and t-SNE, called CCP-UMAP and CCP-t-SNE, respectively, may generate better 2D visualizations for datasets with global structures. Additionally, for real-world problems, better accuracy is always desirable. New hybrid methods, such as three-step CCP-UMAP and CCP-Isomap, may achieve better dimensionality reduction performance for clustering, classification, and regression.

Finally, the R-S scores, R index, S index, R-S disparity, and R-S index introduced in this work can be used for general-purpose data visualization and analysis. The shape of data and persistent Laplacian discussed in this work offer new geometric, topological, and spectral tools for data analysis and visualization.

### **Server availability**

The CCP online server is available at <https://weilab.math.msu.edu/CCP/>.

## BIBLIOGRAPHY

- [1] Kelin Xia, Kristopher Opron, and Guo-Wei Wei. Multiscale multiphysics and multidomain models—flexibility and rigidity. *The Journal of chemical physics*, 139(19):11B614\_1, 2013.
- [2] Patrizio Frosini. Measuring shapes by size functions. In *Intelligent Robots and Computer Vision X: Algorithms and Techniques*, volume 1607, pages 122–133. International Society for Optics and Photonics, 1992.
- [3] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *Proceedings 41st annual symposium on foundations of computer science*, pages 454–463. IEEE, 2000.
- [4] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33(2):249–274, 2005.
- [5] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.
- [6] Konstantin Mischaikow and Vidit Nanda. Morse theory for filtrations and efficient computation of persistent homology. *Discrete & Computational Geometry*, 50(2):330–353, 2013.
- [7] K. L. Xia and G. W. Wei. Persistent homology analysis of protein structure, flexibility and folding. *International Journal for Numerical Methods in Biomedical Engineering*, 30:814–844, 2014.
- [8] Jacob Townsend, Cassie Putman Micucci, John H Hymel, Vasileios Maroulas, and Konstantinos D Vogiatzis. Representation of molecular structures with persistent homology for machine learning applications in chemistry. *Nature communications*, 11(1):1–9, 2020.
- [9] Rui Wang, Duc Duy Nguyen, and Guo-Wei Wei. Persistent spectral graph. *International journal for numerical methods in biomedical engineering*, 36(9):e3376, 2020.
- [10] Jiahui Chen, Yuchi Qiu, Rui Wang, and Guo-Wei Wei. Persistent laplacian projected omicron ba. 4 and ba. 5 to become new dominating variants. *Computers in biology and medicine*, 151:106262, 2022.
- [11] Duc Duy Nguyen, Zixuan Cang, and Guo-Wei Wei. A review of mathematical representations of biomolecular data. *Physical Chemistry Chemical Physics*, 22(8):4343–4367, 2020.
- [12] Duc Duy Nguyen and Guo-Wei Wei. Dg-gl: Differential geometry-based geometric learning of molecular datasets. *International journal for numerical methods in biomedical engineering*, 35(3):e3179, 2019.
- [13] Rundong Zhao, Menglun Wang, Jiahui Chen, Yiying Tong, and Guo-Wei Wei. The de rham–

- hodge analysis and modeling of biomolecules. *Bulletin of mathematical biology*, 82(8):1–38, 2020.
- [14] Jiahui Chen, Rundong Zhao, Yiyong Tong, and Guo-Wei Wei. Evolutionary de rham-hodge method. *Discrete and continuous dynamical systems. Series B*, 26(7):3785, 2021.
  - [15] 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.
  - [16] Kristopher Opron, Keli Xia, and Guo-Wei Wei. Fast and anisotropic flexibility-rigidity index for protein flexibility and fluctuation analysis. *The Journal of chemical physics*, 140(23):06B617\_1, 2014.
  - [17] Duc Duy Nguyen and Guo-Wei Wei. Agl-score: algebraic graph learning score for protein–ligand binding scoring, ranking, docking, and screening. *Journal of chemical information and modeling*, 59(7):3291–3304, 2019.
  - [18] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.
  - [19] Andrew I Su, John B Welsh, Lisa M Sapinoso, Suzanne G Kern, Petre Dimitrov, Hilmar Lapp, Peter G Schultz, Steven M Powell, Christopher A Moskaluk, Henry F Frierson, et al. Molecular classification of human carcinomas by use of gene expression signatures. *Cancer research*, 61(20):7388–7393, 2001.
  - [20] Kun Yang, Zhipeng Cai, Jianzhong Li, and Guohui Lin. A stable gene selection in microarray data analysis. *BMC bioinformatics*, 7(1):1–16, 2006.
  - [21] Todd R Golub, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Coller, Mignon L Loh, James R Downing, Mark A Caligiuri, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537, 1999.
  - [22] Kyle Chang, Chad J Creighton, Caleb Davis, Lawrence Donehower, Jennifer Drummond, David Wheeler, Adrian Ally, Miruna Balasundaram, Inanc Birol, Yaron SN Butterfield, et al. The cancer genome atlas pan-cancer analysis project. *Nat Genet*, 45(10):1113–1120, 2013.
  - [23] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-20). 1996.
  - [24] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-100). 1996.
  - [25] Yann LeCun, Fu Jie Huang, and Léon Bottou. Learning methods for generic object recognition

with invariance to pose and lighting. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:II–104 Vol.2, 2004.

- [26] Linton C Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978.
- [27] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.
- [28] Phillip Bonacich. Power and centrality: A family of measures. *American journal of sociology*, 92(5):1170–1182, 1987.
- [29] Dmitry Kobak and George C Linderman. Initialization is critical for preserving global data structure in both t-sne and umap. *Nature biotechnology*, 39(2):156–157, 2021.



## CHAPTER 6

### TOPOLOGICAL NONNEGATIVE MATRIX FACTORIZATION

#### 6.1 Introduction

Nonnegative matrix factorization (NMF) is a dimensionality reduction method in which the objective is to decompose the original count matrix into two nonnegative factor matrices [1, 2]. The resulting basis matrices are often referred to as meta-genes and represent nonnegative linear combinations of the original genes. Consequently, NMF results are highly interpretable. However, the original formulation employs a least-squares optimization scheme, making the method susceptible to outlier errors [3].

To address this issue, Kong et al. [4] introduced robust NMF (rNMF), or  $l_{2,1}$ -NMF, which utilizes the  $l_{2,1}$ -norm and can better handle outliers while maintaining comparable computational efficiency to standard NMF. Manifold regularization has also been employed to incorporate geometric structures into dimensionality reduction, utilizing a graph Laplacian, leading to Graph Regularized NMF (GNMF) [5]. Semi-supervised methods, such as those incorporating marker genes [6], similarity and dissimilarity constraints [7], have been proposed to enhance NMF's robustness. Additionally, various other NMF derivatives have been introduced [8, 9, 10].

Despite these advancements in NMF, manifold regularization remains an essential component to ensure that the lower-dimensional representation of the data can form meaningful clusters. However, using graph Laplacians can only capture a single scale of the data, specifically the scaling factor in the heat kernel. Therefore, single-scale graph Laplacians lack multiscale information.

In this work, we introduce persistent Laplacian (PL)-regularized NMF, namely the topological NMF (TNMF) and robust topological NMF (rTNMF). Both TNMF and rTNMF can better capture multiscale geometric information than the standard GNMF and rGNMF. To achieve improved performance, PL is constructed by observing sample-sample interactions at multiple scales through filtration, creating a sequence of simplicial complexes. We can then view the spectra at each complex associated with a filtration to capture both topological and geometric information. Additionally, we introduce  $k$ -NN based PL to TNMF and rTNMF, referred to as  $k$ -TNMF and  $k$ -rTNMF, respectively.

The  $k$ -NN based PL reduces the number of hyperparameters compared to the standard PL algorithm.

The outline of this work is as follows. First, we provide a brief overview of NMF, rNMF, GNMF, and rGNMF. Next, we present a concise theoretical formulation of PL and derive the multiplicative updating scheme for TNMF and rTNMF. Additionally, we introduce an alternative construction of PL, termed  $k$ -NN PL.

## 6.2 Prior Work

In this section, we provide an overview of NMF methods, including the standard NMF,  $l_{21}$ -NMF or rNMF, graph regularized NMF (GNMF), and the robust graph regularized NMF (rGNMF), and we utilize single cell RNA sequencing (scRNA-seq) data as an example for the interpretation of NMF. The notations and their descriptions are summarized in Table 6.1.

Table 6.1 Abbreviations and notations used in the methods.

Notation	Description
NMF	Nonnegative matrix factorization
rNMF	Robust Nonnegative Matrix Factorization
GNMF	Graph Regularized Nonnegative Matrix Factorization
rGNMF	Robust Graph Regularized Nonnegative Matrix Factorization
TNMF	Topological Nonnegative Matrix Factorization
rTNMF	Robust Topological Nonnegative Matrix Factorization
$k$ -TNMF	$k$ -NN induced Topological Nonnegative Matrix Factorization
$k$ -rTNMF	$k$ -NN induced Robust Topological Nonnegative Matrix Factorization
$X \in \mathbb{R}^{M \times N}$	Nonnegative data matrix with $M$ genes and $N$ cells
$W \in \mathbb{R}^{M \times p}$	The basis or the meta-genes, and $p$ is the rank
$H \in \mathbb{R}^{p \times N}$	Lower dimensional representation of the data, and $p$ is the rank
$A \in \mathbb{R}^{N \times N}$	Adjacency matrix
$D \in \mathbb{R}^{N \times N}$	Degree matrix
$L \in \mathbb{R}^{N \times N}$	Graph Laplacian $L = D - A$
$PL \in \mathbb{R}^{N \times N}$	Persistent Laplacian, $PL = PD - PA$
$PA \in \mathbb{R}^{N \times N}$	Adjacency matrix associated with PL
$PD \in \mathbb{R}^{N \times N}$	Degree matrix associated with PL
$\zeta_t$	The weight of the graph for the $t$ -th filtration
$\lambda$	Hyperparameter for the regularized NMF

### 6.2.1 NMF

The original formulation of NMF utilizes the Frobenius norm, which assumes that the noise of the data is sampled from a Gaussian distribution. Let  $X \in \mathbb{R}^{M \times N}$  be a nonnegative data matrix. In scRNA-seq, this is the gene-count matrix. The goal of NMF is to find the decomposition  $X \approx WH$ , where both  $W \in \mathbb{R}^{M \times p}$  and  $H \in \mathbb{H}^{p \times N}$  are nonnegative. Here  $p$  is the rank of the decomposition. The minimization function is given as the following

$$\min_{W, H} \|X - WH\|_F^2, \quad \text{s.t. } W, H \geq 0 \quad (6.1)$$

where  $\|A\|_F^2 = \sum_{i,j} a_{ij}^2$ .  $W$  is the basis, which are often times called the meta-genes in scRNA-seq. Lee et al. proposed a multiplicative updating scheme, which preserves the nonnegativity [1]. For the  $t + 1$ th iteration,

$$w_{ij}^{t+1} = w_{ij}^t \frac{(XH^T)_{ij}}{(WHH^T)_{ij}} \quad (6.2)$$

$$h_{ij}^{t+1} = h_{ij}^t \frac{(W^T X)_{ij}}{(W^T W H)_{ij}} \quad (6.3)$$

Although the updating scheme is simple and effective in many biological data applications, scRNA-seq data is sparse and contains large amount of noise. Therefore, a model that is more robust to noise is necessary for feature selection and dimensionality reduction

### 6.2.2 rNMF

The robust NMF (rNMF) utilizes the  $l_{2,1}$  norm, which assumes that the noise of the data is sampled from a Laplace distribution, which may be more suitable for a count-based data matrix, like scRNA-seq. The minimization function is given as the following

$$\min_{W, H} \|X - WH\|_{2,1}, \quad \text{s.t. } W, H \geq 0,$$

where  $\|A\|_{2,1} = \sum_j \|\mathbf{a}_j\|_2$ . Because  $l_{2,1}$ -norm utilizes summation over the  $l_2$  distance between the original cell feature and the reduced feature, the effect of the outlier will not dominate the loss

function as much as the Frobenius norm formulation. RNMF has the following updating scheme

$$w_{ij}^{t+1} = w_{ij}^t \frac{(XQH^T)_{ij}}{(WHQH^T)_{ij}} \quad (6.4)$$

$$h_{ij}^{t+1} = h_{ij}^t \frac{(W^T XQ)_{ij}}{(W^T WHQ)_{ij}}, \quad (6.5)$$

where  $Q_{jj} = 1/\|X - W\mathbf{h}_j\|_2$ .

### 6.2.2.1 GNMF and rGNMF

Manifold regularization has been widely utilized in scRNA-seq. Let  $G(V, E, W)$  be a graph, where  $V = \{\mathbf{x}_j\}_{j=1}^N$  is the set of vertices,  $E = \{(\mathbf{x}_i, \mathbf{x}_j) | \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x}_j) \cup \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i)\}$  is the set of edges, and  $W$  is the weight associated with the edges. Here,  $\mathcal{N}_k(\mathbf{x}_j)$  denotes the  $k$ -th nearest neighbors of vertex  $j$ . For the weight between vertex  $i$  and  $j$ , denoted  $\omega_{ij}$ , we chose a decaying function with the following properties

$$\begin{aligned} \omega_{ij} &\rightarrow 0 \text{ as } \|\mathbf{x}_i - \mathbf{x}_j\| \rightarrow \infty \\ \omega_{ij} &\rightarrow 1 \text{ as } \|\mathbf{x}_i - \mathbf{x}_j\| \rightarrow 0 \end{aligned} \quad (6.6)$$

A common choice for such function is the radial basis function. For example, the heat kernel

$$\omega_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma}\right), \quad (6.7)$$

where  $\sigma$  is the scale of the kernel. We can then represent the weights as an adjacency matrix  $A$

$$A_{ij} = \begin{cases} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma}\right) & \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i) \\ 0, & \text{otherwise.} \end{cases}$$

We can now construct the graph regularization term,  $R_G$ , by looking at the distance  $\|\mathbf{h}_i - \mathbf{h}_j\|^2$  and the adjacency matrix.

$$\begin{aligned}
R_G &= \frac{1}{2} \sum_{i,j} A_{ij} \|\mathbf{h}_i - \mathbf{h}_j\|^2 \\
&= \sum_i D_{ii} \mathbf{h}_i^T \mathbf{h}_i - \sum_{i,j} A_{ij} \mathbf{h}_i^T \mathbf{h}_j \\
&= \text{Tr}(HDH^T) - \text{Tr}(HAH^T) \\
&= \text{Tr}(HLH^T).
\end{aligned}$$

Here,  $L$  and  $D$  are the Laplacian and the degree matrix, given by  $L = D - A$  and  $D_{ii} = \sum_j A_{ij}$ , respectively.  $\text{Tr}(\cdot)$  denotes the trace of the matrix. Utilizing the regularization parameters,  $\lambda \geq 0$ , we get the objective function of GNMF

$$\min_{W,H} \|X - WH\|_F^2 + \lambda \text{Tr}(HLH^T). \quad (6.8)$$

and the objective function for rGNMF

$$\min_{W,H} \|X - WH\|_{2,1} + \lambda \text{Tr}(HLH^T). \quad (6.9)$$

### 6.3 Topological NMF

While graph regularization improves the traditional NMF and rNMF, the choice of  $\sigma$  and vastly change the result. Furthermore, graph regularization only captures a single scale, and may not be able to capture the mutliscale geometric information in data. In this section, we only show the construction of persistent Laplacian. The theoretical formulation of persistent Laplacian can be found in section 2.4.

#### 6.3.1 TNMF and rTNMF

For scRNA-seq data, we calculate the 0-persistent Laplacian using the Vietoris-Rips (VR) complexes by increasing the filtration distance. We can then take a weighted sum over the 0-persistent Laplacian induced by the changes in the filtration distance. For persistent Laplacian

enhanced NMF, we will provide a computationally efficient algorithm to construct the persistent Laplacian matrix.

Let  $L$  be a Laplacian matrix induced by some weighted graph, and note the following

$$L = \begin{cases} l_{ij}, & i \neq j \\ -\sum_{j=1}^N l_{ij} & i = j. \end{cases}$$

Then, let  $l_{\max} = \max_{i \neq j} l_{ij}$ ,  $l_{\min} = \min_{i \neq j} l_{ij}$  and  $d = l_{\max} - l_{\min}$ . The  $t$ -th Persistent Laplacian  $L^t$ ,  $t = 1, \dots, T$  is defined as  $L^t = \{l_{ij}^t\}$ , where

$$l_{ij}^t = \begin{cases} 0 & l_{ij} \leq (t/T)d + l_{\min} \\ 1 & \text{otherwise} \end{cases} \quad (6.10)$$

$$l_{ii}^t = -\sum_{i \neq j} l_{ij}^t. \quad (6.11)$$

Then, we can take the weighted sum over the all the persistent Laplacians

$$PL := \sum_{t=1}^T \zeta_t L^t. \quad (6.12)$$

Unlike the standard Laplacian matrix  $L$ , PL captures the topological features that persists over different filtration, thus providing a multiscale view of the data that standard Laplacian lacks. Here,  $\zeta_t$  is the hyper-parameter and must be chosen. Then, the PL regularized NMF, which we call topological nonnegative matrix factorization (TNMF) is defined as,

$$\|X - WH\|_F^2 + \lambda \text{Tr}(H^T(PL)H) \quad (6.13)$$

and the robust topological NMF (rTNMF) is defined as

$$\|X - WH\|_{2,1} + \lambda \text{Tr}(H^T(PL)H). \quad (6.14)$$

### 6.3.2 Multiplicative Updating scheme

The updating scheme follows the same principle as the standard GNMF and rGNMF.

**TNMF** For TNMF, the Lagrangian function is defined as

$$\mathcal{L} = \|X - WH\|_F^2 + \lambda \text{Tr}(H^T(PL)H) + \text{Tr}(\Phi W) + \text{Tr}(\Psi H) \quad (6.15)$$

$$= \text{Tr}(X^T X) - 2\text{Tr}(XH^T W^T) + \text{Tr}(WHH^T W^T) + \lambda \text{Tr}(H^T(PL)H) + \text{Tr}(\Phi W) + \text{Tr}(\Psi H). \quad (6.16)$$

Taking the partial with respect to  $W$ , we get

$$\frac{\partial \mathcal{L}}{\partial W} = -2H^T XH + 2WHH^T + \Phi. \quad (6.17)$$

Using the KKT condition  $\Phi_{ij}w_{ij} = 0$ , we get the following

$$(-2XH^T)_{ij}w_{ij} + (2WHH^T)_{ij}w_{ij} = 0. \quad (6.18)$$

Therefore, the updating scheme is

$$w_{ij}^{t+1} \leftarrow w_{ij}^t \frac{(XH^T)_{ij}}{(WHH^T)_{ij}}. \quad (6.19)$$

For updating  $H$ , we take the derivative of the Lagrangian function with respect to  $H$

$$\frac{\partial \mathcal{L}}{\partial H} = -2W^T X + 2W^T WH + 2\lambda H(PL) + \Psi. \quad (6.20)$$

Using the Karush–Kuhn–Tucker (KKT) condition, we have  $\Psi_{ij}h_{ij} = 0$  and obtain

$$-2(W^T X + \lambda H(PA))_{ij}h_{ij} + 2(W^T WH + \lambda H(PD))_{ij}h_{ij} = 0, \quad (6.21)$$

where  $PL = PD - PA$  and  $PD_{ii} = \sum_{i \neq j} PA_{ij}$ . The updating scheme is then given by

$$h_{ij}^{t+1} \leftarrow h_{ij}^t \frac{(W^T X + \lambda H(PA))_{ij}}{(W^T WH + \lambda H(PD))_{ij}}. \quad (6.22)$$

### 6.3.2.1 rTNMF

For the updating scheme for rTNMF, we utilize the fact that  $\|A\|_{2,1} = \text{Tr}(AQA^T)$ , where  $Q_{ii} = \frac{1}{2\|A_i\|_2}$ . The Lagrangian is given by

$$\mathcal{L} = \|X - WH\|_{2,1} + \lambda \text{Tr}(H^T(PL)H) + \text{Tr}(\Phi W) + \text{Tr}(\Psi H) \quad (6.23)$$

$$= \text{Tr}((X - WH)Q(X - WH)^T) + \lambda \text{Tr}(H^T(PL)H) + \text{Tr}(\Phi W) + \text{Tr}(\Psi H) \quad (6.24)$$

$$= \text{Tr}(XQX^T) - 2\text{Tr}(WHQ) + \lambda \text{Tr}(H^T(PL)H) + \text{Tr}(\Phi W) + \text{Tr}(\Psi H), \quad (6.25)$$

where  $Q_{ii} = \frac{1}{\|\mathbf{x}_j - W\mathbf{h}_j\|}$ . Taking the partial with respect to  $W$ , we get

$$\frac{\partial L}{\partial W} = -(XQH^T) + WHQH^T - \Phi. \quad (6.26)$$

Using the KKT conditions  $\Phi_{ij}w_{ij} = 0$ , we get

$$-(XQH^T)_{ij}w_{ij} + (WHQH^T)_{ij}w_{ij} = 0, \quad (6.27)$$

which gives the updating scheme

$$w_{ij}^{t+1} \leftarrow w_{ij}^t \frac{(XQH^T)_{ij}}{(WHQH^T)_{ij}}. \quad (6.28)$$

For  $H$ , we take the partial with respect to  $H$ .

$$\frac{\partial L}{\partial H} = -W^T XQ + W^T WHQ + 2\lambda H(PL) + \Psi. \quad (6.29)$$

Then, using the KKT conditions  $\Psi_{ij}h_{ij} = 0$ , we get

$$(-W^T XQ - 2\lambda H(PA))_{ij}h_{ij} + (W^T WHQ + 2\lambda H(PD))_{ij}h_{ij} = 0, \quad (6.30)$$

where  $PL = PD - PA$  and gives the updating scheme

$$h_{ij}^{t+1} \leftarrow h_{ij}^t \frac{(W^T XQ + 2\lambda H(PA))_{ij}}{(W^T WHQ + 2\lambda H(PD))_{ij}}. \quad (6.31)$$

### 6.3.3 $k$ -NN induced Persistent Laplacian

One major issue with TNMF and rTNMF is that the parameters  $\{\zeta_t\}_{t=1}^T$  have to be chosen. For the parameters, we let  $\zeta_t \in \{0, 1, 1/2, \dots, 1/T\}$  for a total of  $T + 1$  parameters. Therefore, the number of parameters that needs to be chosen increases exponentially as the number of filtration  $T$  increases. Therefore, we propose an approximation to the original formulation using  $k$ -NN induced PL.

Let  $\mathcal{N}_t(\mathbf{x}_j)$  be the  $t$ -nearest neighbors of sample  $\mathbf{x}_j$ . First, we define the  $t$ -persistent directed adjacency matrix  $\tilde{A}^t$  as

$$\tilde{A}^t = \{\tilde{a}_{ij}^t\}, \quad \tilde{a}_{ij}^t = \begin{cases} 1 & \mathbf{x}_j \in \mathcal{N}_t(\mathbf{x}_i) \\ 0 & \text{otherwise.} \end{cases} \quad (6.32)$$



Then, the  $k$ -NN based directed adjacency Laplacian is the weighted sum of  $\{A^t\}$

$$\tilde{A} := \sum_{t=1}^T \zeta_t \tilde{A}^t. \quad (6.33)$$

The undirected persistent adjacency matrix can be obtained via symmetrization

$$PA = \tilde{A} + \tilde{A}^T - \tilde{A} \otimes \tilde{A}^T,$$

where  $\otimes$  denote Hadamard product. Then, the PL can be constructed using the persistent degree and persistent adjacency matrices

$$PL = PD - PA, \quad PD_{ii} = \sum_{j \neq i} PA_{ij}. \quad (6.34)$$

One advantage of utilizing the  $k$ -NN induced persistent Laplacian is that the parameter space is much smaller. We can set  $\zeta_t \in \{0, 1\}$ , where  $\zeta_t = 0$  would ‘turn-off’ the particular neighbor’s connectivity. In essence, the number of parameters will be reduced to  $2^T$ , a significant decrease from  $(T + 1)^T$  of the original formulation. We call the  $k$ -NN induced TNMF as  $k$ -TNMF and the  $k$ -NN induced rTNMF as  $k$ -rTNMF.

## BIBLIOGRAPHY

- [1] Daniel Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13, 2000.
- [2] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. *IEEE Transactions on knowledge and data engineering*, 25(6):1336–1353, 2012.
- [3] Weixiang Liu, Nanning Zheng, and Qubo You. Nonnegative matrix factorization and its applications in pattern recognition. *Chinese Science Bulletin*, 51:7–18, 2006.
- [4] Deguang Kong, Chris Ding, and Heng Huang. Robust nonnegative matrix factorization using l21-norm. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 673–682, 2011.
- [5] Qiu Xiao, Jiawei Luo, Cheng Liang, Jie Cai, and Pingjian Ding. A graph regularized non-negative matrix factorization method for identifying microrna-disease associations. *Bioinformatics*, 34(2):239–248, 2018.
- [6] Peng Wu, Mo An, Hai-Ren Zou, Cai-Ying Zhong, Wei Wang, and Chang-Peng Wu. A robust semi-supervised nmf model for single cell rna-seq data. *PeerJ*, 8:e10091, 2020.
- [7] Zhenqiu Shu, Qinghan Long, Luping Zhang, Zhengtao Yu, and Xiao-Jun Wu. Robust graph regularized nmf with dissimilarity and similarity constraints for scrna-seq data clustering. *Journal of Chemical Information and Modeling*, 62(23):6271–6286, 2022.
- [8] Wei Lan, Jianwei Chen, Qingfeng Chen, Jin Liu, Jianxin Wang, and Yi-Ping Phoebe Chen. Detecting cell type from single cell rna sequencing based on deep bi-stochastic graph regularized matrix factorization. *bioRxiv*, pages 2022–05, 2022.
- [9] Jin-Xing Liu, Dong Wang, Ying-Lian Gao, Chun-Hou Zheng, Jun-Liang Shang, Feng Liu, and Yong Xu. A joint-l2, 1-norm-constraint-based semi-supervised feature extraction for rna-seq data analysis. *Neurocomputing*, 228:263–269, 2017.
- [10] Na Yu, Ying-Lian Gao, Jin-Xing Liu, Juan Wang, and Junliang Shang. Robust hypergraph regularized non-negative matrix factorization for sample clustering and feature selection in multi-view gene expression data. *Human genomics*, 13(1):1–10, 2019.

## CHAPTER 7

### APPLICATION IN SINGLE CELL RNA SEQUENCING

#### 7.1 Preprocessing of Single Cell RNA Sequencing data using Correlated Clustering and Projection

##### 7.1.1 Introduction

In this section, we propose a computationally efficient and interpretable dimensionality reduction algorithm for scRNA-seq data called correlated clustering and projection (CCP) [1]. CCP begins by clustering genes based on their similarity and then uses the flexibility rigidity index (FRI) [2] to nonlinearly project each gene cluster into a super-gene, which is a measure of accumulated gene-gene correlations among cells. Unlike traditional nonlinear reduction methods, CCP bypasses matrix diagonalization, allowing users to select the number of super-genes, which is beneficial for machine learning and deep learning tasks. Furthermore, similar to NMF's meta-genes, super-genes are all nonnegative and highly interpretable. We validated CCP's performance on 14 scRNA-seq datasets by varying the number of super-genes and conducting support vector machine classification and  $k$ -means clustering.

Additionally, we have validated the performance of a novel evaluation metric for dimensionality reduction, called the Reside-Similarity-Index (RSI) [1]. RSI evaluates the intra-cluster similarity of cell types or clusters, and compares it to their inter-cluster residual score. As RSI only requires one set of labels, which can be computed from  $k$ -means, it can measure the performance of dimensionality reduction for both clustering and classification tasks, without requiring knowledge of the true labels. Furthermore, by analyzing the relationship between samples, RSI allows for a deeper understanding of the quality of the dimensionality reduction algorithm. We have verified the effectiveness of RSI alongside CCP on both clustering and classification tasks, and introduced the R-S plot as a novel visualization technique for data containing multiple cell types.

### 7.1.2 Results

Table 7.1 Accession ID, source organism, and the counts for samples, genes, cell types and normalization for 14 datasets.

Accession ID	Reference	Organism	Samples	Genes	Cell types	Normalization
GSE45719	Deng [3]	Mouse	300	22431	8	RPKM
GSE59114	Kowalczyk [4]	Mouse	1428	8422	6	TPM
GSE67835	Darmanis [5]	Human	420	22084	8	CPM
GSE75748 cell	Chu [6]	Human	1018	19097	7	TPM
GSE75748 time	Chu [6]	Human	758	19189	6	TPM
GSE82187	Gokce [7]	Mouse	705	18840	10	TPM
GSE84133 h1	Baron [8]	Human	1937	20125	14	TPM
GSE84133 h2	Baron[8]	Human	1724	20125	14	TPM
GSE84133 h3	Baron[8]	Human	3605	20125	14	TPM
GSE84133 h4	Baron[8]	Human	1308	20125	14	TPM
GSE84133 m1	Baron[8]	Mouse	822	14878	13	TPM
GSE84133 m2	Baron[8]	Mouse	1064	14878	13	TPM
GSE89232	Breton [9]	Human	957	20689	4	TPM
GSE94820	Villani [10]	Human	1140	26593	5	TPM

CCP was benchmarked against PCA on 14 datasets, and the dataset details can be found in Table 7.1. The data was normalized using either reads per kilobase of transcript per million (RPKM), transcript per million (TPM) or counts per million (CPM). For each dataset, CCP was used to obtain the number of super-genes as  $N = 50, 100, 150, 200, 250$ , and  $300$ . The parameters  $\kappa$  and  $\tau$  of the exponential kernel were searched over  $\kappa = 1, 2$  and  $\tau = 1, 2, \dots, 6$  and were set to  $\tau = 6$  and  $\kappa = 2$  for the exponential kernel. To test the reduction, 20 random seeds were used for CCP and PCA, and for each reduction, 30 random initializations of  $k$ -means were used to obtain cluster labels. After obtaining the cluster labels, ARI and NMI were computed by comparing the results to the labeled cell types, and the averages were visualized. For each figure, the red and blue lines represent CCP and PCA, respectively, and the star and dot markers indicate ARI and NMI, respectively.

#### 7.1.2.1 CCP Benchmark

Figure 7.1 shows the performance of CCP and PCA on three datasets, GSE67835, GSE75748 time, and GSE59114 data. For GSE67835, CCP outperforms PCA in all the dimensions we have

tested. For GSE75748 time, CCP outperforms PCA for 50 super-genes and above. GSE75748 time shows an increase in performance as the number of gene dimensions increased. PCA exhibits instability as  $N$  increases, which is noticeable from their decrease in performance from  $N = 50$  to 150 for both datasets. CCP does not perform well on GSE59114 because both ARI and NMI are less than 0.3 for all the dimensions we have tested. CCP's performance may be poor due to low intrinsic dimensionality of GSE59114. In other words, the number of gene clusters is inherently small, leading to redundant clusters. GSE59114, in particular, only has 8,422 genes, whereas other data have over 15,000 genes.

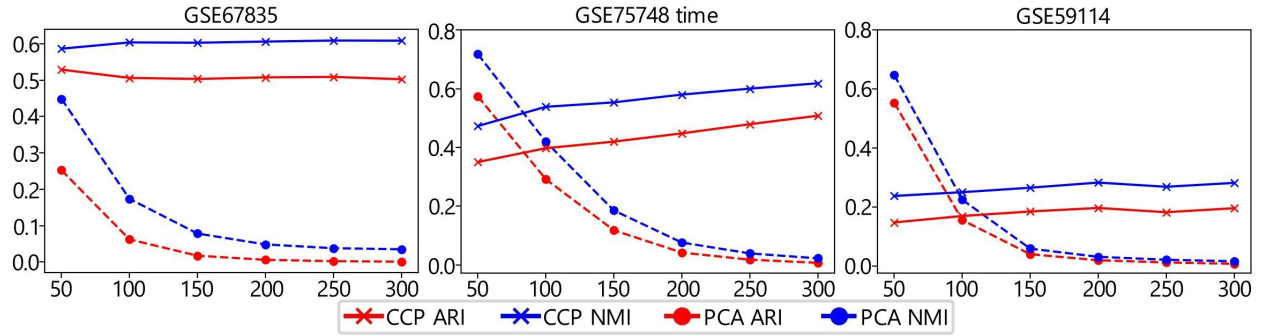
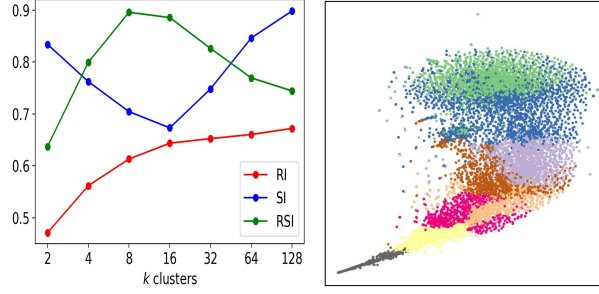


Figure 7.1 ARI and NMI of the clustering results of CCP and PCA on GSE67835 , GSE75748 time and GSE59114 data. The red and blue lines correspond to CCP and PCA, respectively. A total of 20 random initializations were used to test the reduction, and for each reduction, a total of 30 random initializations were used to obtain the clustering results from  $k$ -means clustering. The averages of the ARI and NMI were obtained. For CCP, all the test utilize  $\tau = 6$  and  $\kappa = 2$  for the exponential kernel.

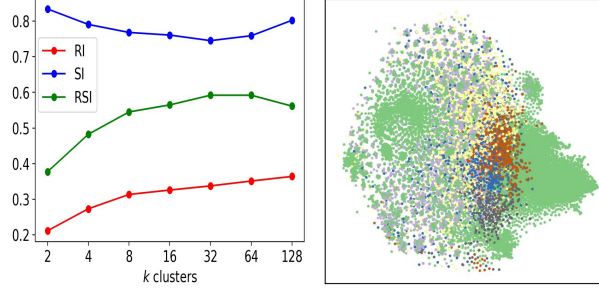
In order to verify CCP's performance, the residue similarity index (RSI) was calculated for the  $k$ -means clustering result of the gene partitioning in CCP. Figure 7.2 shows the RSI of the  $k$ -means clustering on the genes at various numbers of cell clusters ( $k$ ). The top row shows the clustering result for GSE59114, which had poor CCP performance, and the bottom row shows the clustering result for GSE67825, which had good CCP performance. For each number of clusters, 10 random initializations were used for the  $k$ -means clustering, and the averages of the RI, SI, and RSI were obtained. The red, blue, and green lines correspond to RI, SI, and RSI, respectively. RSI can be used to check the quality of the clustering, where the peak in RSI suggests the optimal number of clusters, in the case of CCP, the intrinsic dimensionality of the data. The right column shows

the 2D visualization of the gene using t-SNE. The samples were colored according to their cluster labels. The t-SNE visualization of GSE59114 shows the  $k$ -means clustering result when  $k = 8$  was selected. The t-SNE visualization of GSE67835 shows the  $k$ -means clustering result when  $k = 64$  was selected. Seven of the 64 clusters were colored, and the green samples are the rest of the genes.

Notice that in GSE59114, there is a noticeable peak in the RSI score at  $k = 8$  clusters, whereas in GSE67835, the peak is flat and occurs at about  $k = 32-64$  clusters. This suggests that the intrinsic dimensionality is about 8 for GSE59114, which is unfavorable for CCP. On the other hand, the intrinsic dimension of GSE67835 is much higher, which is more suitable for CCP. Notice that the clusters have distinct boundaries, supporting the relatively low dimensionality of the data. On the other hand, the GSE67835 data is not well-clustered even at  $k = 64$ . Notice that the orange and blue genes have some outliers, and the purple genes are not well-clustered. This suggests that the number of optional gene clusters is larger, which suggests high gene dimensionality and favors CCP.



(a) GSE59114



(b) GSE67835

Figure 7.2 RI, SI, and RSI of the gene clustering of GSE59114 and GSE67835.  $k$ -means clustering was performed with  $k = 2, 4, 8, 16, 32, 64$ , and 128 cell clusters. For each number of clusters, 10 random initializations were utilized, and the averages of RI, SI and RSI were obtained. The red, blue, and green line corresponds to RI, SI, and RSI, respectively. We use t-SNE to visualize the genes in 2D. For GSE59114  $k = 8$  clusters were obtained, and the cells were colored according to their cluster assignment. For GSE67835,  $k = 64$  cell types were obtained. Seven random cell types were colored, and the rest of cell types were colored in green.

### 7.1.2.2 Residue-Similarity Index comparison

Residue-similarity index (RSI) has been shown to correlate with classification accuracy in [1]. In this section, we use RSI for classification and clustering on the 14 datasets from Table 7.1. We use CCP to process each dataset with the same parameters as the previous section with 20 random initializations. For classification, we use 5-fold cross-validation with 10 random seeds and the support vector machine to predict cell types. We use balanced accuracy (BA) to measure the performance of the classification. Then, using the same 5-fold cross-validation, we calculate RSI, where we obtain the RI, SI, and RSI from the test set, similar to [1]. For clustering, we compute RSI for PCA and CCP using the  $k$ -means clustering labels and the true labels. Additionally, using the  $k$ -means clustering labels, we compute the Silhouette score to compare the results with RSI. Full details of the benchmark procedure can be found in 7A.2.1 of the Supporting Materials.

In general, we have found no correlation between the Silhouette scores and RSI for clustering results. Additionally, we have found that BA and RSI correlate in classification results.

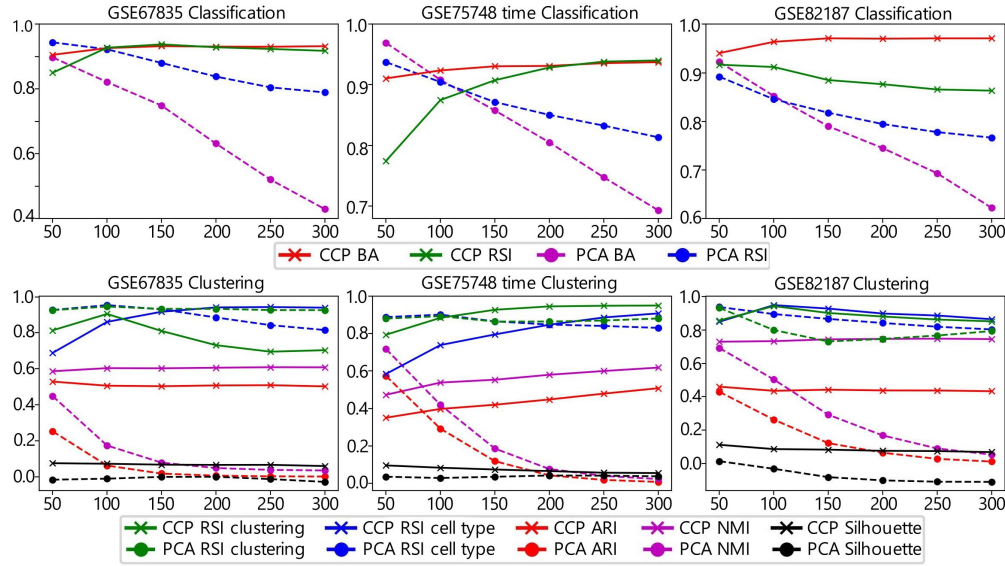


Figure 7.3 Comparison of RSI in classification and clustering problems for GSE67835, GSE75748 time, and GSE82187 data at reduced dimensions  $N = 50, 100, 150, 200, 250$ , and  $300$ . CCP was used to reduce the original data dimension using  $\tau = 6$  and  $\kappa = 2$  for the exponential kernel. The top and bottom rows correspond to the classification and clustering results, respectively. For classification, support vector machine was used. True labels were used to compute the RSI for the 5-fold cross validation. For clustering, RSI were computed using the cluster labels from  $k$ -means clustering and the true labels.

We found that RSI correlates with classification accuracy in many of our tests. Figure 7.3 shows RSI for classification and clustering problems for GSE67835, GSE75748 time, and GSE82187 data. CCP was used to reduce the original data using  $\tau = 6$  and  $\kappa = 2$  for the exponential kernel. The top row corresponds to classification results, and the bottom row corresponds to clustering results. Notice that for classification results, all three datasets show a correlation between BA and RSI. RSI on classification results for GSE67835 shows a plateau at about 150 super-genes, which corresponds to the plateau of the BA. This suggests that the optimal dimension is about 150. RSI on classification results for GSE75748 shows a plateau at about 200 gene clusters, even though BA plateaus at about 150 gene clusters. Even though the accuracy plateaued earlier, this suggests that the optimal dimension is at 200 gene clusters. In addition, since GSE75748 time observes cell differentiation at different times, it is possible that some cells are at different stages in their



cell cycles, as suggested in the literature [6]. This suggests that there are many intermediate stages in cell differentiation. RSI on classification results on GSE82187 shows a small decrease as the number of super-genes increases. This suggests that the optimal dimension is smaller than those of GSE67835 and GSE75748 time. Lastly, RSI decreases for all three datasets when PCA is utilized, which corresponds to the decrease in BA.

For the clustering results, RSI using the  $k$ -means labels and the true cell types are similar. Even though ARI and NMI of PCA decrease as the number of gene clusters increases, RSI remains consistent. This suggests that PCA is not able to differentiate clusters at higher dimensions. CCP, on the other hand, shows a correlation with both RSI scores.

Additional examples of utilizing RSI on classification and clustering problems can be found in Section 7A.2.2 of the Supporting materials.

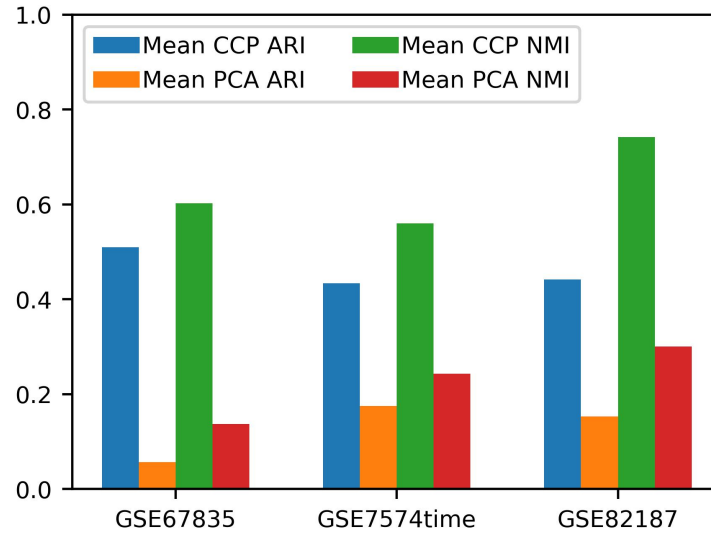


Figure 7.4 Comparison of CCP and PCA clustering on GSE67835, GSE75748 time, and GSE82187 data. CCP was used to reduce the original data dimension using  $\tau = 6$  and  $\kappa = 2$  for the exponential kernel. The blue, orange, green and red bars correspond to mean CCP ARI, mean PCA ARI, mean CCP NMI and mean PCA NMI, respectively. Here, the average was taken over different dimensions.

Figure 7.4 shows the overall clustering performance of CCP and PCA. The bars show the mean ARI and NMI values across the different number of components. Notice that for both ARI and NMI, CCP outperforms PCA.

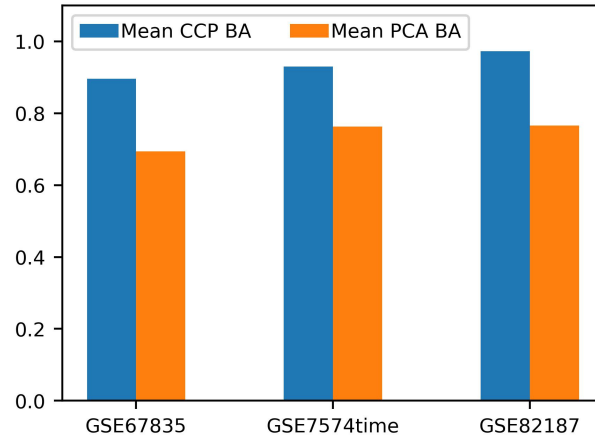


Figure 7.5 Comparison of CCP and PCA classification on GSE67835, GSE75748 time, and GSE82187 data. CCP was used to reduce the original data dimension using  $\tau = 6$  and  $\kappa = 2$  for the exponential kernel. The blue and orange bars correspond to mean BAs of CCP and PCA, respectively. Here, the average was taken over different dimensions.

Figure 7.5 shows the overall classification performance of CCP and PCA. The bars show the mean BAs across the different number of dimensions. Notice that for the mean BA, CCP outperforms PCA.

### 7.1.3 Discussion

#### 7.1.3.1 CCP

Like other dimensionality reduction algorithms, CCP has its advantages and disadvantages. CCP nonlinearly projects each cluster of similar genes into a super-gene. Super-genes are highly interpretable: each super-gene represents a measure of a cluster of genes' accumulated pairwise nonlinear correlations with the same cluster of genes in all other cells for a given cell. Similar to NMF, super-genes are non-negative, which is important for downstream analysis such as differential gene expression analysis.

Since CCP is a data-domain method, it bypasses matrix diagonalization. One limitation of many dimensionality reduction algorithms is their dependence on matrix diagonalization. In scRNA-seq data, the number of genes is typically larger than 5,000, which gives rise to the “curse of dimensionality”. When the number of features are large, every sample may appear to be equidistant from one another, which makes many machine learning algorithms unable to find meaningful

clusters in the data. CCP, on the other hand, partitions the genes into clusters and computes the pairwise gene-gene correlations across all cells, which avoids the curse of dimensionality.

Even though CCP has shown success in many scRNA-seq datasets, it does have limitations. CCP does not perform well for datasets with a low intrinsic dimension. As shown in Figure 7.2, GSE59114 and GSE94228 have a low intrinsic dimension, and as a result, their clustering results also suffered.

In addition, many scRNA-seq datasets are sparse due to low signal-to-noise ratio and dropout events. Therefore, CCP will most likely benefit from data imputation.

### 7.1.3.2 RSI

RSI is a useful tool for assessing the performance of dimensionality reduction for both clustering and classification problems. In the following section, we compare RSI to the traditional clustering metrics, ARI and NMI, and also with the Silhouette score. Then, we discuss RSI and its connection with classification accuracy.

**RSI for clustering** Compared to ARI and NMI, which measure the similarity between two sets of labels, RSI evaluates the performance using only one set of labels. In this study, ARI and NMI were used to compare the true labels with the clustering labels. However, in practice, such true labels may not be available. RSI, on the other hand, can evaluate the effectiveness of clustering without the need for original labels. This is similar to the Silhouette score, which measures the separations between clusters. However, when there are multiple clusters, the Silhouette score becomes difficult to interpret because it measures whether a sample belongs to its current cluster assignment or to the nearest neighboring cluster. Therefore, it is often used to evaluate the optimal number of clusters, rather than evaluating different parameters while fixing the number of clusters. RSI can evaluate the effectiveness of different parameters while fixing the number of clusters.

**RSI for classification** Using RSI for cell types, we have shown that RSI correlates with classification accuracy. Additionally, RI and SI indicate how well the clusters separate from each other. The Area Under the Receiver Operating Characteristic Curve (AUC-ROC) is a metric commonly

used to evaluate classification effectiveness. However, AUC-ROC is a better metric for binary classification problems, and its interpretation is more challenging for multiclass problems. RSI, on the other hand, can handle problems with more than two cell types. Lastly, RSI uses the features and labels to compute the scores, so it can also demonstrate the effectiveness of dimensionality reduction algorithms in conjunction with classification problems.

RSI can also be utilized for visualizing each class or cluster, which we have called Residue-Similarity (R-S) plot. In order to showcase the R-S plot, we compare it with traditional visualization techniques used in scRNA-seq data, namely t-SNE and UMAP. CCP was utilized to reduce the dimensionality. The 5-fold cross-validation was used to divide the data into 5 parts, where 4 parts were used to train the support vector machine classifier, and 1 part to test the classifier. Then, residue and similarity scores were computed for each sample and plotted according to its true number of cell types. Samples were then colored according to their predicted labels from the support vector machine classifier. The x-axis and y-axis correspond to residue and similarity scores, respectively. Both residue and similarity scores range from 0 to 1, where 1 is the most optimal, and the top-right corner indicates well-separated and clustered reduction. However, it is important to note that having a balance of both scores is important, as shown in Hozumi et al. (2022) [1]. For t-SNE and UMAP, the original data was log-transformed, and genes with variance less than  $10^{-6}$  were removed prior to the reduction. Samples were then plotted and colored according to their cell types.

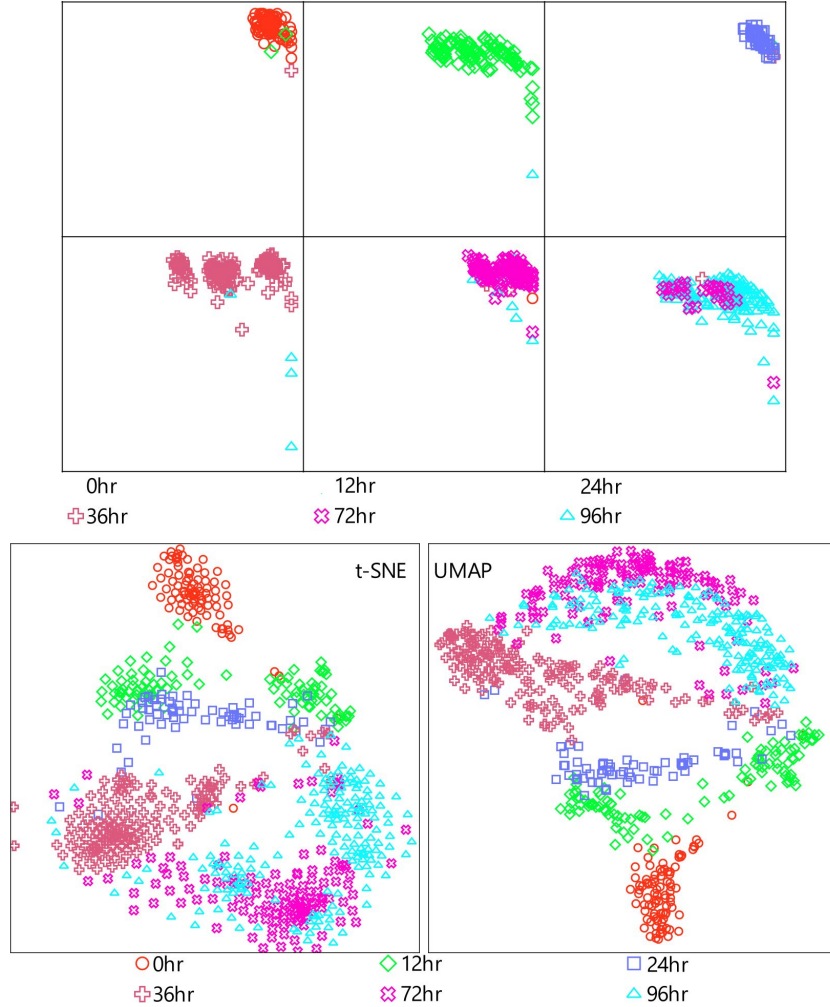


Figure 7.6 R-S plot, CCP assisted t-SNE plot and standard t-SNE plots of GSE75748 time data. CCP was used to reduce the scRNA-seq data to 200 super-genes using  $\tau = 6$  and  $\kappa = 2$ . The 5-fold cross-validation was used to split the data into 5 parts, where 4 were used for training, and 1 part was used for testing the support vector machine classifier. RS scores were computed for the testing set, and all 5 folds were visualized. Each section corresponds to one of the 7 true cell types, and the sample's color and marker correspond to the predicted label from the support vector machine classifier. For t-SNE and UMAP, the data was log-transformed, and any genes with less than  $10^{-6}$  variance were removed before applying the reduction. Samples were colored according to their cell types.

Figure 7.6 shows a comparison between the R-S plot and 2D plots of UMAP and t-SNE for GSE75748 time data. CCP was used to generate 200 super-genes with  $\tau = 6$  and  $\kappa = 2$ . For UMAP and t-SNE plots, the reduction was directly applied to the log-transformed original data. In [6], Chu obtained snapshots at different times of ES cell differentiation from pluripotency to definitive endoderm (ED) over 4 days at 0hr, 12hrs, 24hrs, 36hrs, 72hrs, and 96hrs. Noticeably, cells recorded

at 72hrs and 96hrs are mixed in UMAP and t-SNE plots and are misclassified in the R-S plot. This finding is consistent with [6], where cells from 72hrs and 96hrs were relatively homogeneous. In a biological sense, this may indicate that cell differentiation had mostly completed by 72 hrs, such that not much of the further process of cell differentiation was observed at 96 hrs. In t-SNE and UMAP plots, we can see a similar pattern as the R-S plot. There are 2 subclusters of the 12hr samples. Additionally, the 72hr and 96hr samples form one large cluster, which is consistent with R-S plot's findings. Most notably, there is a large difference between the ES cell at 0hr and ES cells at different times in all visualizations, and there is no misclassification of the 0hr state with cells from 72hr and 96hr states, indicating that the cells have indeed differentiated from the original pluripotent state.

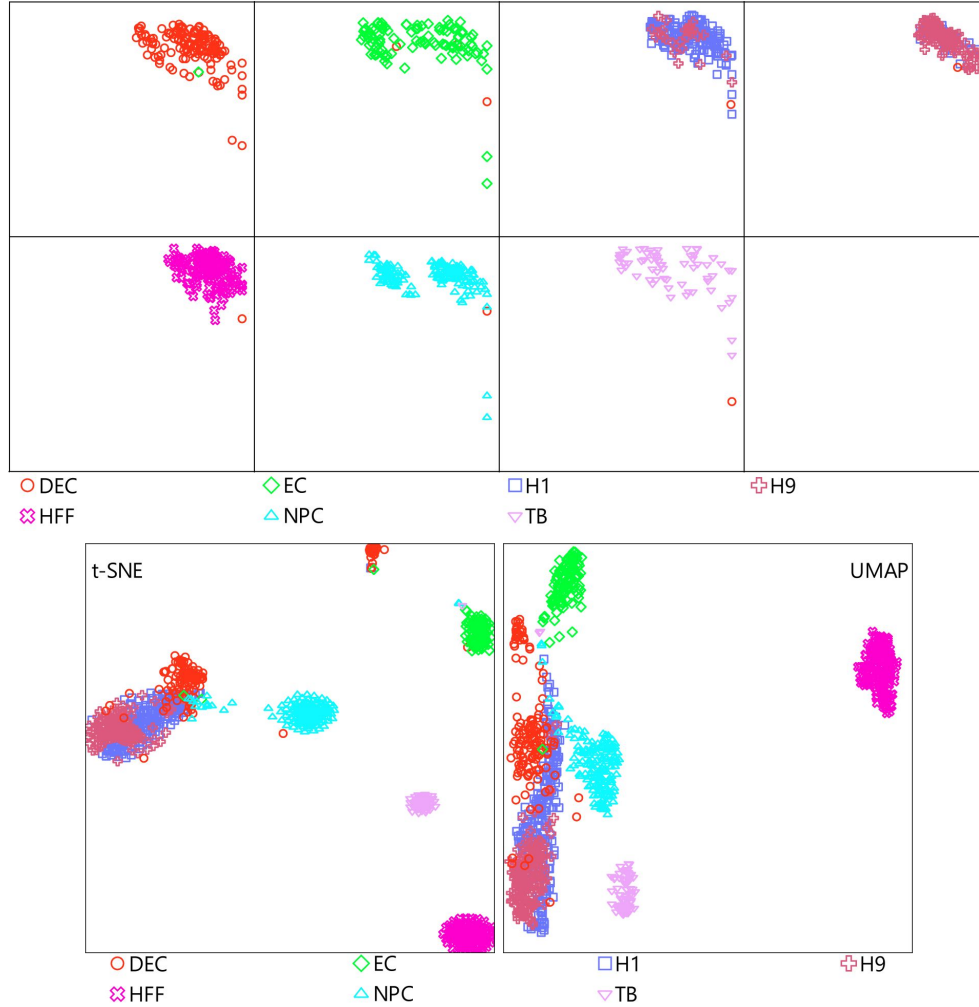


Figure 7.7 RS plot, and CCP assisted UMAP and t-SNE plots of GSE75748 cell. CCP was used to reduce the scRNA-seq data to 100 components using  $\tau = 6$  and  $\kappa = 2$ . 5-fold cross validation was used to split the data into 5 parts, where 4 parts were used for training, and 1 part was used for testing the  $k$ -NN classifier. RS score was computed for the testing set, and all 5 folds were visualized. Each section corresponds to 1 of the 7 true cell types, and the sample's color and marker correspond to the predicted label from the  $k$ -NN classifier. For t-SNE and UMAP, the data was log-transformed and any genes with less than  $10^{-6}$  variance were removed before applying the reduction. Samples were colored according to their cell types.

Figure 7.7 shows a comparison between the R-S plot and 2D plots of UMAP and t-SNE of GSE75748 cell data. CCP was used to reduce the dimension to 100 super-genes with  $\tau = 6$  and  $\kappa = 2$ . In [6], Chu obtained snapshots of lineage-specific progenitor cells that differentiated from H1 human embryonic stem (ES) cells. These differentiated cells include neuronal progenitor cells (NPC), endoderm derivatives cells (DEC), endothelial cells (EC), trophoblast-like cells (TB), human

foreskin fibroblasts (HFF), and undifferentiated H1 and H9 human ES cells. Not surprisingly, all 3 visualizations show that undifferentiated ES cell H1 and H9 are clustered together, indicating that these two ES cells are relatively homogeneous, which agrees with Chu’s findings. In the R-S plot, we see that all but 1 DEC sample are classified incorrectly, whereas in UMAP and t-SNE plots, DEC samples do not form a distinct cluster, and have a super cluster forming with the H1, H9, and DEC cluster. In addition, all 3 visualizations show 2 clusters of NPC samples, but CCP is able to classify NPC samples correctly. Notice that in the R-S plot, there are a few misclassifications of EC and DEC cells, and in UMAP, these two clusters are adjacent to one another. This is consistent with a small number of misclassified EC and DEC shown in the RS plot. Since EC are derivatives of mesoderm, it has been suggested by [11, 12, 13] that mesoderm and DEC may have developed and differentiated from a common progenitor pool.

#### **7.1.4 Conclusion**

CCP is a novel dimensionality reduction method that projects each cluster of similar genes into a super-gene defined as accumulated pairwise nonlinear gene-gene correlations among cells. We have shown that CCP is able to differentiate cell types and also preserve the similarity along trajectory of cellular differentiation. In addition, since CCP works exclusively in the data-domain, it does not rely on matrix diagonalization and its results are easily interpretable. It outperforms PCA for problems having an intrinsically high dimensionality.

We have also shown that RSI is a novel metric for evaluating the effectiveness of dimensionality reduction algorithms. Since it correlates with accuracy but does not rely on knowing the true labels of the data, it can be applied to improve both clustering and classification. In addition, RSI can be used to vary the number of clusters and obtain insight into the optimal number of cell types. This information can be used to filter out data where CCP may not perform well because CCP works best when the intrinsic dimensionality of the data, i.e., the number of gene features, is relatively high. Lastly, the R-S plot is introduced as a new visualization tool that works well for problems with a large number of cell types.



### **7.1.5 Code and Data availability**

All data was processed and is available at <https://github.com/hozumiyu/SingleCellDataProcess>. The code needed to reproduce this paper's result can be found at <https://github.com/hozumiyu/CCP-for-Single-Cell-RNA-Sequencing>.

CCP is made available through our web-server at

<https://weilab.math.msu.edu/CCP/> or through the source code <https://github.com/hozumiyu/CCP>.

Source code of RSI and R-S plot can be found at <https://github.com/hozumiyu/RSI>.

## **7.2 Analyzing scRNA-seq data by CCP-assisted UMAP and t-SNE**

### **7.2.1 Introduction**

The objective of the present work is to explore the utility of CCP for initializing scRNA-seq data. We are particularly interested in its potential application for initializing UMAP and t-SNE, which are among the most successful visualization tools in scRNA-seq analysis. We tested CCP-assisted UMAP and CCP-assisted t-SNE on eight publicly available datasets. CCP performance in assisting UMAP and t-SNE is compared favorably with that of PCA and NMF.

Additionally, we introduce a novel method for handling low-variance (LV) genes. Instead of discarding low-variance genes like many other methods, we group them together into a single category. This grouping is achieved by projecting them into one descriptor using FRI. One of the drawbacks of dropping low-variance genes is that scRNA-seq data often has an unequal number of cell types. Moreover, there are numerous genes with low expression, and removing too many genes may result in overlooking cell outliers. Therefore, LV-gene addresses this issue by consolidating low-variance genes into one descriptor, thereby increasing its predictive power. We found that CCP improves the accuracy of UMAP and t-SNE by over 11% in each case.

### **7.2.2 Methods and Algorithms**

In this section, we describe the construction of LV-gene, which will be on the components. For the rest of the components, refer to section 5.2 for the detail.

### 7.2.2.1 Low variance (LV) genes

Let  $\mathbf{v} = (v_1, \dots, v_I)$  be the variance of the genes, where  $v_i$  is the variance of gene  $\mathbf{z}^i$ , and assume that the variance are sorted in descending order. Then, define the low variance set  $P$  as

$$P = \{i | i > v_c I\}$$

where  $0 \leq v_c \leq 1$  is the cutoff ratio. Then, we can obtain the cell-cell correlation using these low variance genes  $C_{ij}^P$ ,

$$C_{ij}^P = \Phi(\|\mathbf{z}_i^P - \mathbf{z}_j^P\|; \eta^P, \tau, \kappa)$$

where  $\Phi(\|\mathbf{z}_i^P - \mathbf{z}_j^P\|; \eta^P, \tau, \kappa)$  is the generalized exponential function

$$\Phi(\|\mathbf{z}_i^P - \mathbf{z}_j^P\|; \eta^P, \tau, \kappa) = \begin{cases} e^{-\left(\frac{\|\mathbf{z}_i^P - \mathbf{z}_j^P\|}{\eta^P \tau}\right)^\kappa} & \|\mathbf{z}_i^P - \mathbf{z}_j^P\| < r_c^P \\ 0, & \text{otherwise.} \end{cases}$$

$r_c^P$  is taken as the 3-standard deviation of the pairwise distances, and  $\eta^P$  is the average minimum distance

$$\eta^P = \frac{\sum_{m=1}^M \min_{\mathbf{z}_j^P} \|\mathbf{z}_m^P - \mathbf{z}_j^P\|}{M}.$$

Using the correlation function, CCP projects  $|P|$  genes into a super-gene using FRI for  $i$ th sample,

$$x_i^P = \sum_{m=1}^M w_{im} \Phi(\|\mathbf{z}_i^P - \mathbf{z}_m^P\|; \eta^P, \tau, \kappa),$$

where  $w_{im}$  are the weights.

For CCP, we compute the LV-gene first, and use the correlated partition algorithm on the remaining genes.

## 7.2.3 Results

### 7.2.3.1 Data preprocessing

We have tested CCP-assisted UMAP and tSNE visualization on 20 publicly available data. Table 7.2 displays information including the Gene Expression Omnibus (GEO) accession ID [14, 15], the reference, data dimensions, and cell composition for each dataset. Additionally, data from the scziDesk paper [16] was utilized and can be accessed from their supporting materials. The Qx and Qs data correspond to Smart-seq2 and 10x genomic data from Quake et al. [17]. Notably, the GSE84133 human dataset encompasses all human patient data from Baron et al. [8]. Detailed statistics for each data can be found in Table 7B.1 in the supporting materials.

Table 7.2 Dataset name, reference, dimensions and cell type composition.

Dataset [Ref]	Size (cells x genes)	Cell Composition
GSE75748cell [6]	1018 x 19097	7 clusters: 138, 105, 212, 162, 159, 173, 69
GSE75748time [6]	758 x 19189	6 clusters: 92, 102, 66, 172, 138, 188
GSE82187 [7]	705 x 18840	10 clusters: 107, 18, 21, 71, 48, 7, 334, 13, 43, 43
GSE67835 [5]	420 x 22084	8 clusters: 18, 62, 20, 110, 25, 16, 131, 38
GSE84133 H1 [8]	1937 x 20125	14 clusters: 110, 51, 236, 872, 214, 120, 130, 13, 70, 14, 8, 92, 5, 2
GSE84133 H2 [8]	1724 x 20125	14 clusters: 3, 81, 676, 371, 125, 301, 23, 2, 86, 17, 9, 22, 6, 2
GSE84133 H3 [8]	3605 x 20125	14 clusters: 843, 100, 1130, 787, 161, 376, 92, 2, 36, 14, 7, 54, 1, 2
GSE84133 H4 [8]	1303 x 20125	14 clusters: 2, 52, 284, 495, 101, 280, 7, 1, 63, 10, 1, 5, 1, 1
GSE84133 M1 [8]	822 x 14878	13 clusters: 2, 4, 4, 9, 343, 85, 236, 72, 14, 4, 17, 29, 3
GSE84133 M2 [8]	1064 x 14878	13 clusters: 8, 3, 10, 182, 551, 133, 39, 67, 27, 4, 19, 18, 3
GSE84133 human [8]	8569 x 20125	14 clusters: 958, 284, 2326, 2525, 601, 1077, 252, 18, 255, 55, 25, 173, 13, 7
Muraro [18]	2122 x 19046	9 clusters: 21, 812, 193, 101, 219, 245, 3, 80, 448
Romanov [19]	2881 x 21143	7 clusters: 267, 240, 356, 48, 898, 1001, 71
Qx Bladder [17]	2500 x 23341	4 clusters: 1203, 1167, 57, 73
Qx Limb Muscle [17]	3909 x 23341	6 clusters: 461, 320, 1330, 308, 1136, 354
Qx Spleen [17]	9552 x 23341	5 clusters: 6886, 1930, 42, 464, 230
Qs Diaphragm [17]	870 x 23341	5 clusters: 78, 81, 31, 241, 439
Qs Limb Muscle [17]	1090 x 23341	6 clusters: 71, 35, 141, 45, 258, 540
Qs Lung [17]	1676 x 23341	11 clusters: 57, 53, 25, 90, 113, 35, 693, 65, 85, 37, 423
Qs Trachea [17]	1350 x 23341	4 clusters: 206, 113, 201, 830

To normalize the data, we began by normalized the counts by using the average median gene count of each cell. Let  $X \in \mathbb{R}^{M \times N}$  be the data, with  $M$  cells and  $N$  genes. Each row (cell) was divided by its row sum, followed by multiplication by the median row sum to obtain a normalized count matrix. Finally, a log transformation using  $\log 1p$  was applied to achieve the final normalized count.

In our benchmarking process, we employed CCP with parameters  $\tau = 6$  and  $\kappa = 2$  to reduce the dimensions to 300 super-genes. Additionally, we utilized  $v = 0.8$  to generate the LV-gene. Clustering was performed using the Leiden algorithm, and we evaluated the quality of clustering using ARI, NMI and ECM by comparing the obtained clusters with the cell types provided by the original authors. Visualizations were generated using Scanpy’s implementation of UMAP and tSNE. In order to reduce the computation load for datasets exceeding 2,000 samples, we utilized subsampling.

### 7.2.3.2 Visualization

Preprocessing of scRNA-seq data is a key step for visualization. Figure 7.8 shows an example of CCP-assisted tSNE visualization and the original tSNE visualization of the Baron dataset [8]. The original data has 20,125 genes, and aggressively reducing the original dimension to 2 dimensions by tSNE leads to poor visualization. In CCP-assisted tSNE, CCP was utilized to reduce the original genes into 300 super-genes, which were further reduced to 2 dimensions with tSNE for visualization. Obviously, CCP-assisted tSNE significantly improves the visualization quality in this case. We further showcase CCP-assisted visualization on the dataset described in Table Table 7.2. We provide additional comparison with PCA-assisted and NMF-assisted visualization in Section 7B.1.2 of the supporting materials.

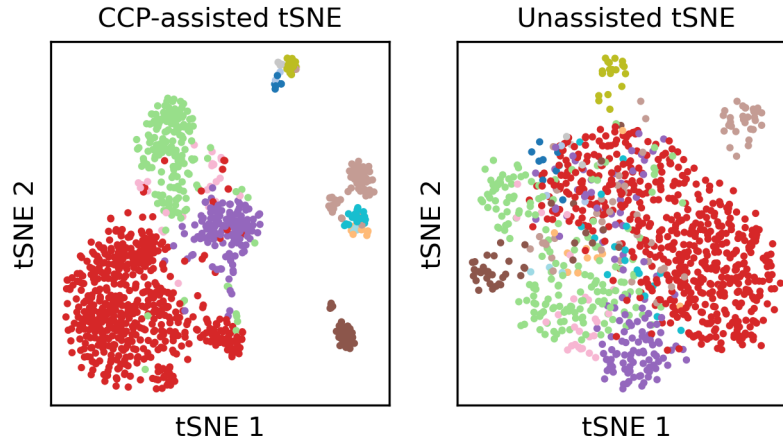


Figure 7.8 TSNE visualization of GSE84133 mouse2 data. The left and right figures show the tSNE assisted and unassisted tSNE visualization.

Figure 7.9 shows the comparison of CCP-assisted UMAP and tSNE with standard UMAP and tSNE visualization on Quake dataset. Each row corresponds to one of the 5 datasets, and the columns correspond to CCP-assisted UMAP, CCP-assisted tSNE, standard UMAP and standard tSNE visualization. The samples were colored according to the true cell type.

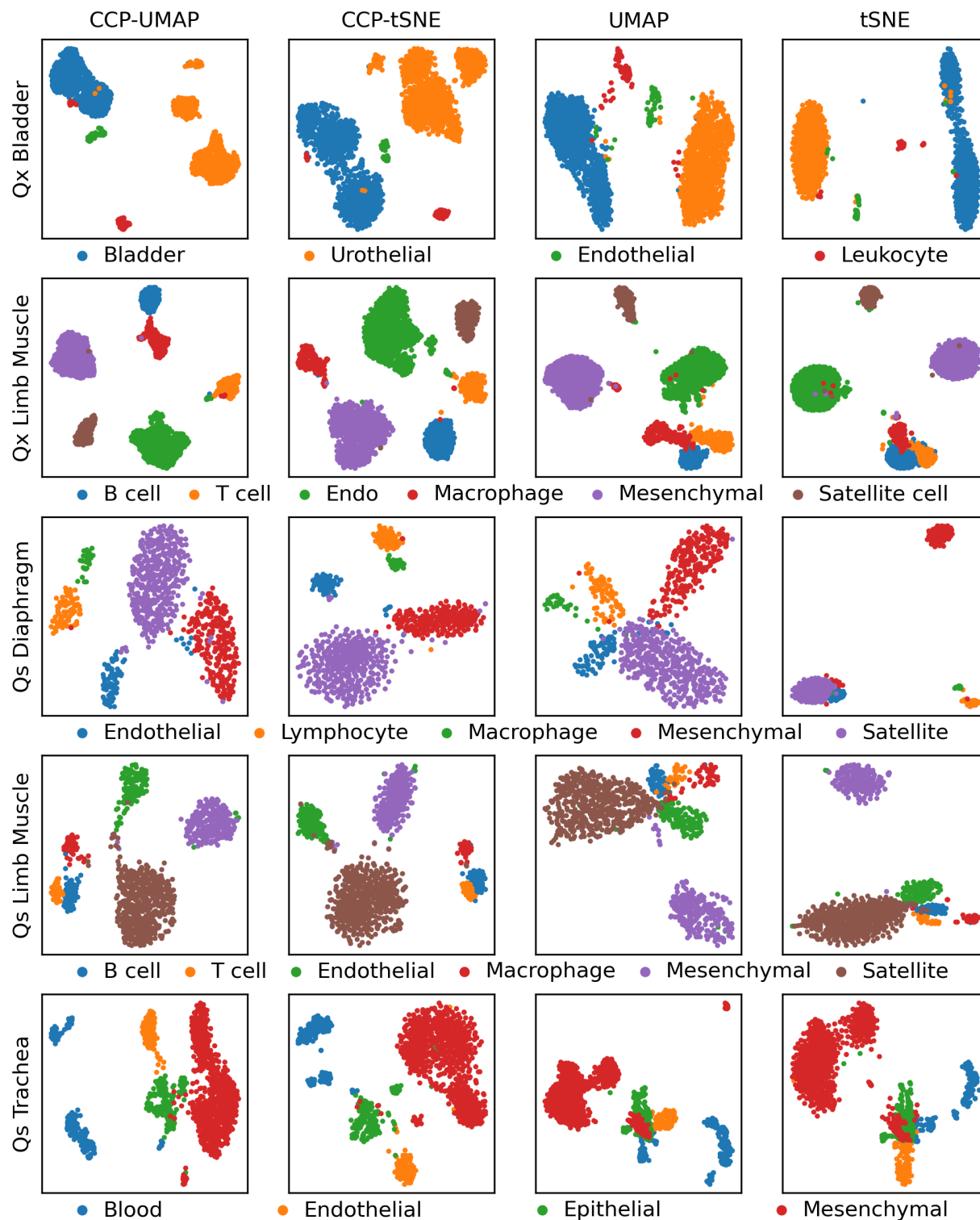


Figure 7.9 Comparison of CCP-assisted UMAP and tSNE with standard UMAP and tSNE visualization on Quake dataset. The rows correspond to Qx Bladder, Qx Limb Muscle, Qx Diaphragm, Qs Limb Muscle and Qs Trachea. Qx indicates scRNA-seq obtained used 10x genomic platform, and Qs indicate data obtained from SmartSeq2 platform. CCP was used to reduced the dimension to 300 super-genes. UMAP and tSNE were utilized to further reduce the dimension to 2 to obtain the visualization. Samples were colored according to the cell types provided by the original authors.

CCP improves the overall visualization of the Quake dataset. In Qx Bladder data, CCP-assisted UMAP and tSNE show three subclusters of bladder urothelial cells, whereas the standard UMAP and tSNE show only one cluster. However, the standard visualization show leukocytes and endothelial cells within the bladder urothelial cell cluster. In Qs Diaphragm dataa, CC:-assisted UMAP and tSNE show a 5 distinct clusters corresponding to each cell types. However the UMAP visualization do not differentiate the 5 cell types. The standard tSNE visualization show poor clustering, where satellite cell, mesenchymal cell and endothelial cell form a supercluster. In Qs Limb Muscle cell, all visualization show a supercluster of B cell and T cell. CCP-assisted visualization show a clear distinction between the B-T cell supercluster and macrophages, whereas the standard visualization show a supercluster of B cell, T cell, macrophages and endothelial cells. In the Qs Trachea data, the standard UMAP and tSNE visualization show a subpopulation of mesenchymal cell within the epithelial cell, where as CCP-assisted counterparts do not.

Figure 7.10 show the comparison of CCP-assisted UMAP and tSNE with standard UMAP and tSNE visualization on GSE75748 cell, GSE75748 time, GSE67835 and GSE82187 dataset. The columns correspond to CCP-assisted UMAP, CCP-assisted tSNE, standard UMAP and standard tSNE visualization. The samples were colored according to the true cell type.

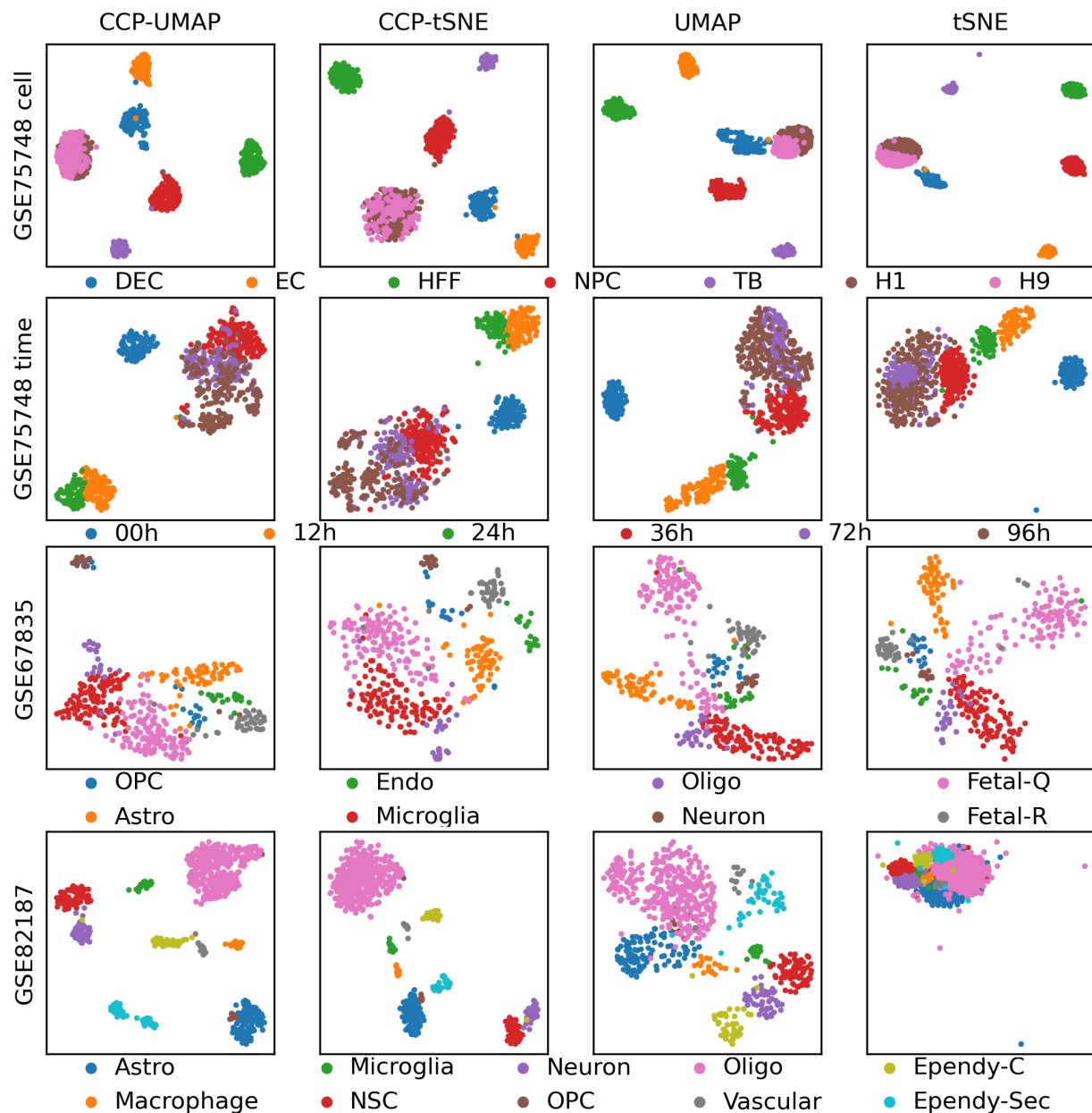


Figure 7.10 Comparison of CCP-assisted UMAP with PCA-assisted, NMF-assisted and standard UMAP visualization on GSE75748 cell, GSE75748 time, GSE67835 and GSE82187. CCP was used to reduced the dimension to 300 super-genes. UMAP and tSNE were utilized to further reduce the dimension to 2 to obtain the visualization. Samples were colored according to the cell types provided by the original authors.

In GSE75748 cell data, all visualization is similar. In [6], Chu obtained snapshots of lineage-specific progenitor cells that differentiated from H1 human embryonic stem (ES) cells and compared the gene profiles with undifferentiated H1 and H9 human ES cells as control. Most notably, H1 and H9 clustered together, which is consistent with our visualization. In GSE75748 time, all



visualization is comparable. Chu et al [6] obtained snapshot of ES cell differentiation from pluripotency to definitive endoderm over the time period 0hr, 12hr, 24hr, 36hr, 72hr and 96hr. Chu noted the cells sequenced at 72hr and 96hr show relatively similar expression profiles, suggesting that the differentiation has completed by 72hr. We see from our visualization that the 72hr and 96hr cells form a cluster. 12hr and 24hr cells form a cluster, and 0hr cells form its own cluster, indicating that there is a clear distinction between the undifferentiated and the cells undergoing differentiation. IN GSE67835, CCP-assisted visualization and its counter part have comparable result. Most notably, neurons cell from a distinct cluster in CCP-assisted visualization, whereas it does not in the standard visualization. In GSE82187 data, CCP-assisted UMAP and tSNE show a significant improvement over standard UMAP and tSNE visualization. Aside from astrocytes and OPC, all cell types form its own cluster. Standard UMAP and tSNE fail to show significant clustering of the different cell types.

Figure 7.11 show the comparison of CCP-assisted UMAP and tSNE with standard UMAP and tSNE visualization on Baron human dataset [8]. The rows correspond to the patients, and the columns correspond to CCP-assisted UMAP, CCP-assisted tSNE, standard UMAP and standard tSNE visualization. The The samples were colored according to the true cell type.

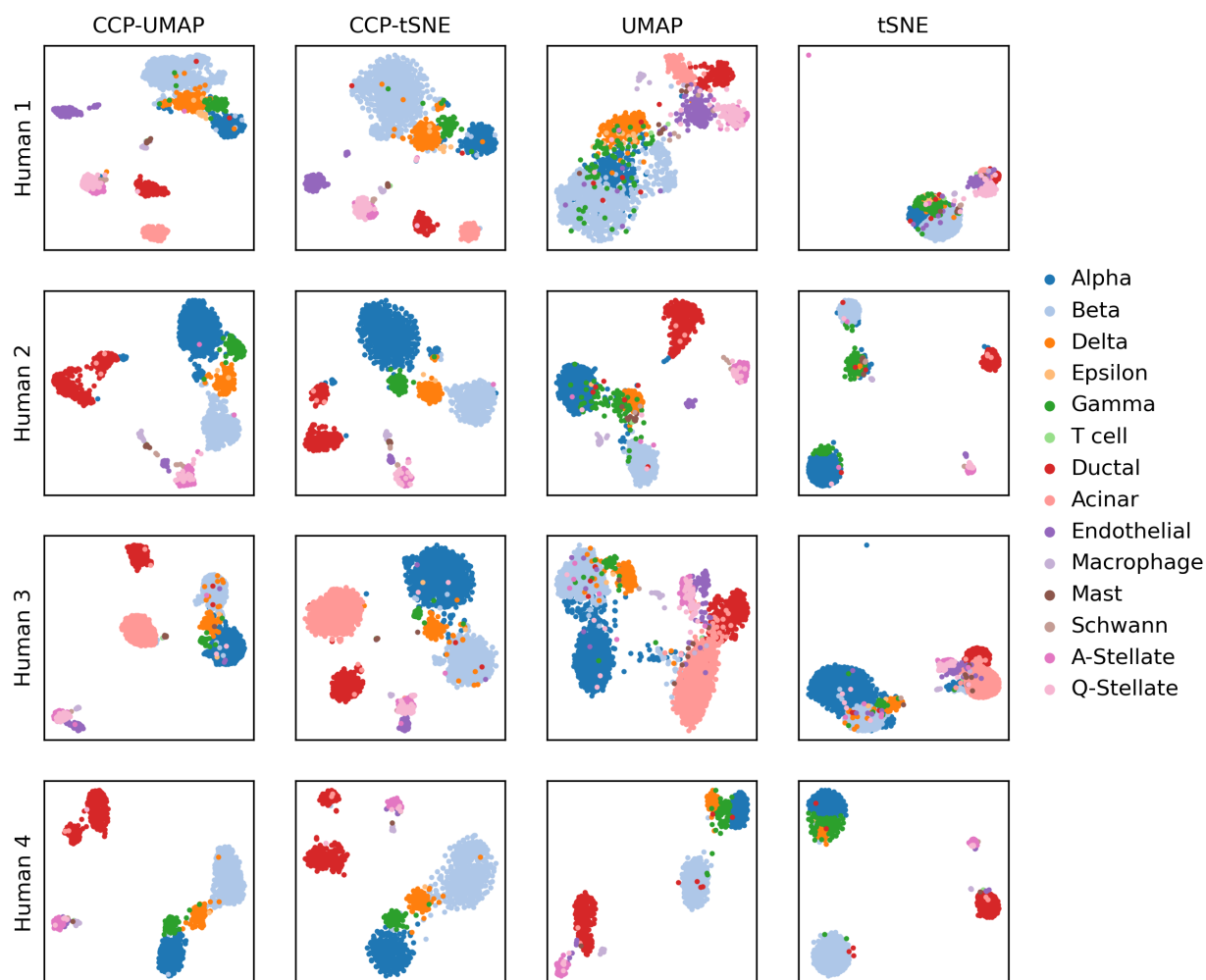


Figure 7.11 Comparison of CCP-assisted UMAP with PCA-assisted, NMF-assisted and standard UMAP visualization on GSE84133 human dataset. Each row corresponds to 1 of the 4 patients. CCP was used to reduced the dimension to 300 super-genes. UMAP and tSNE were utilized to further reduce the dimension to 2 to obtain the visualization. Samples were colored according to the cell types provided by the original authors.

Overall, CCP-assisted visualizations show stronger clustering. In standard UMAP and tSNE visualizations across all patients, we noticed superclusters with unclear boundaries. Conversely, CCP-assisted visualizations display well-defined boundaries between cell types. Most notably is the clear differentiation of quiescent stellate (Q-Stellate) cells, alpha cells, and ductal cells across all patients, which is a distinction that isn't as evident in the standard visualizations. Figure 7.12 show the comparison of CCP-assisted UMAP and tSNE with standard UMAP and tSNE visualization on Baron mouse dataset [8]. The rows correspond to the patients, and the columns correspond to

mouse 1 and 2, and the columns correspond to CCP-assisted UMAP, CCP-assisted tSNE, standard UMAP and standard tSNE visualization.. The The samples were colored according to the true cell type.

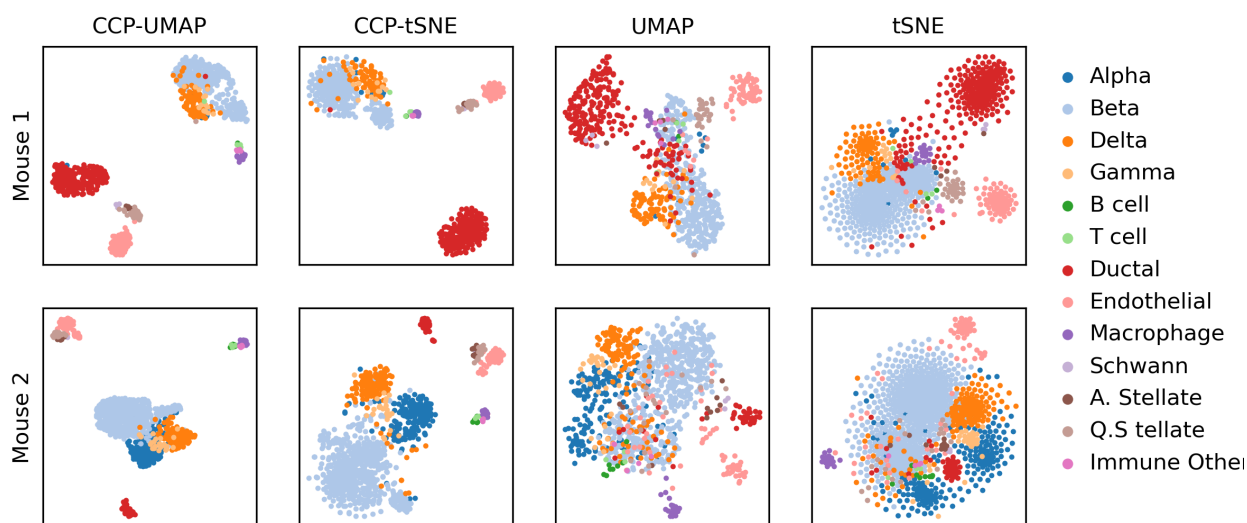


Figure 7.12 Comparison of CCP-assisted UMAP with PCA-assisted, NMF-assisted and standard UMAP visualization on GSE84133 human dataset. The rows correspond to mouse 1 and 2. CCP was used to reduced the dimension to 300 super-genes. UMAP and tSNE were utilized to further reduce the dimension to 2 to obtain the visualization. Samples were colored according to the cell types provided by the original authors.

CCP-assisted visualizations demonstrate significantly stronger clustering for both mouse samples. In the standard visualizations, beta cells are scattered among other cell types. Furthermore, in the data from mouse 2, alpha cells do not form a distinct cluster. Conversely, CCP-assisted visualizations distinctly cluster all cell types. Regarding mouse 1, the CCP-assisted visualization does not form a cluster for gamma cells, potentially due to the limited number of available gamma cells.

Figure 7.13 show the visualization of Murano, Romanov and Qs Lung data. The columns correspond to CCP-assisted UMAP, CCP-assisted tSNE, standard UMAP and standard tSNE visualization. The samples were colored according to the true cell type.

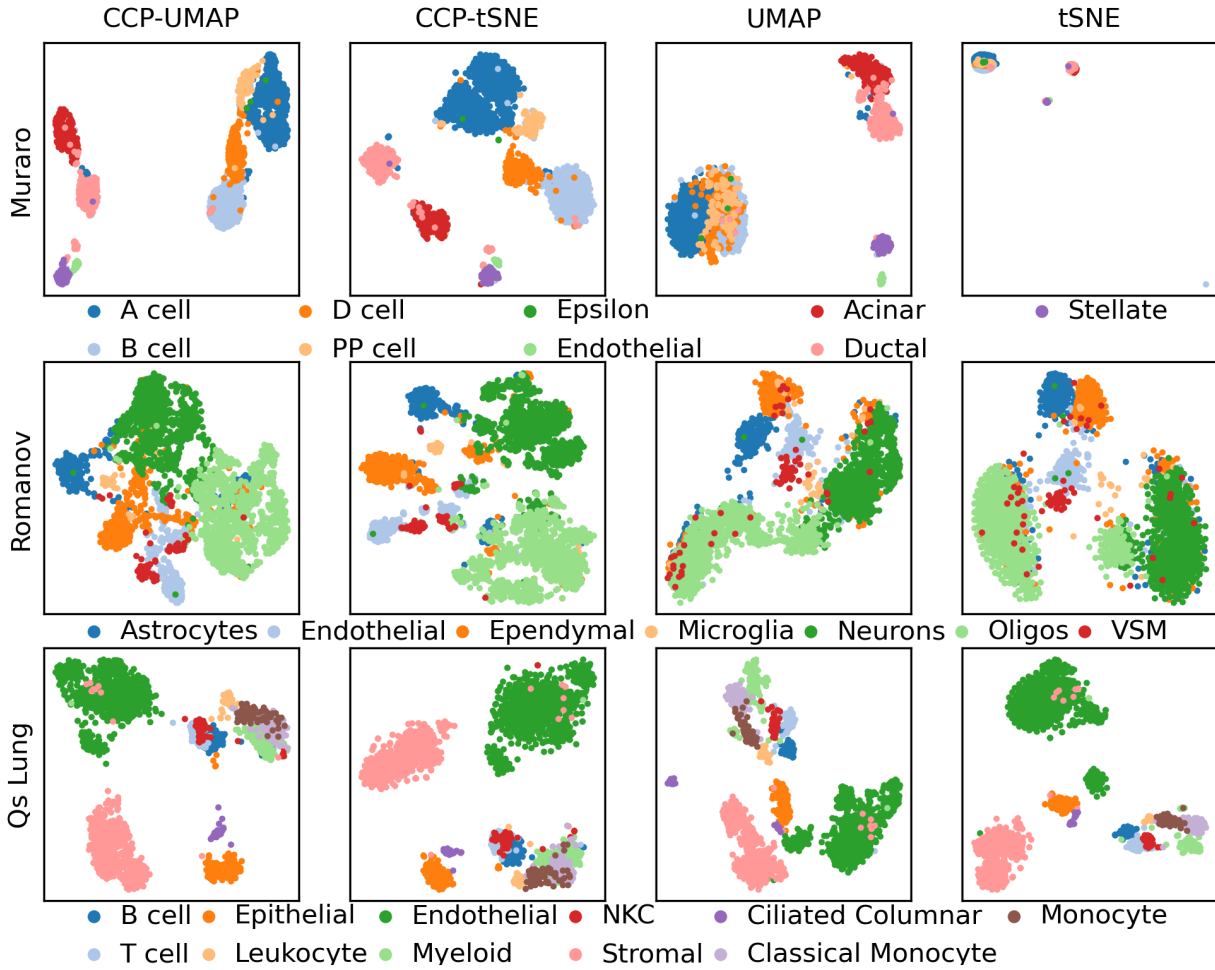


Figure 7.13 Comparison of CCP-assisted UMAP with PCA-assisted, NMF-assisted and standard UMAP visualization on Muraro, Romanov and Qs Lung dataset. CCP was used to reduced the dimension to 300 super-genes. UMAP and tSNE were utilized to further reduce the dimension to 2 to obtain the visualization. Samples were colored according to the cell types provided by the original authors.

In the Muraro dataset, CCP-assisted UMAP exhibits a clear separation of A cells, D cells, B cells, and ductal cells. In contrast, the standard UMAP visualization presents these cells as a supercluster. The standard tSNE visualization is dominated by the outlier B cells, rendering the visualization less interpretable. Regarding the Romanov dataset, all visualizations are relatively similar. CCP-assisted UMAP reveals a distinct cluster of astrocytes and ependymal cells, whereas both the standard UMAP and tSNE display a supercluster of these two cell types. Additionally, CCP-assisted UMAP and tSNE suggest two subclusters of VSM and endothelial cells, which are not discernible in the standard visualization. In the Qs Lung dataset, CCP-assisted and standard

visualizations yield comparable results. While the standard tSNE separates monocytes from classical monocytes, CCP-assisted UMAP and tSNE portray a homogeneous clustering of these two cell types.

### 7.2.3.3 Accuracy

To assess CCP's effectiveness as a primary dimensionality reduction tool for UMAP and tSNE, we conducted clustering using the Leiden algorithm within scanpy. We employed the adjusted Rand index (ARI) and normalized mutual information (NMI) to gauge accuracy by comparing the clustering results with the labels provided by the dataset's authors. It's important to note that these metrics do not measure absolute accuracy due to the absence of a gold standard dataset for scRNA-seq. Additionally, we used the element centric measure (ECM) [20] to evaluate cluster stability.

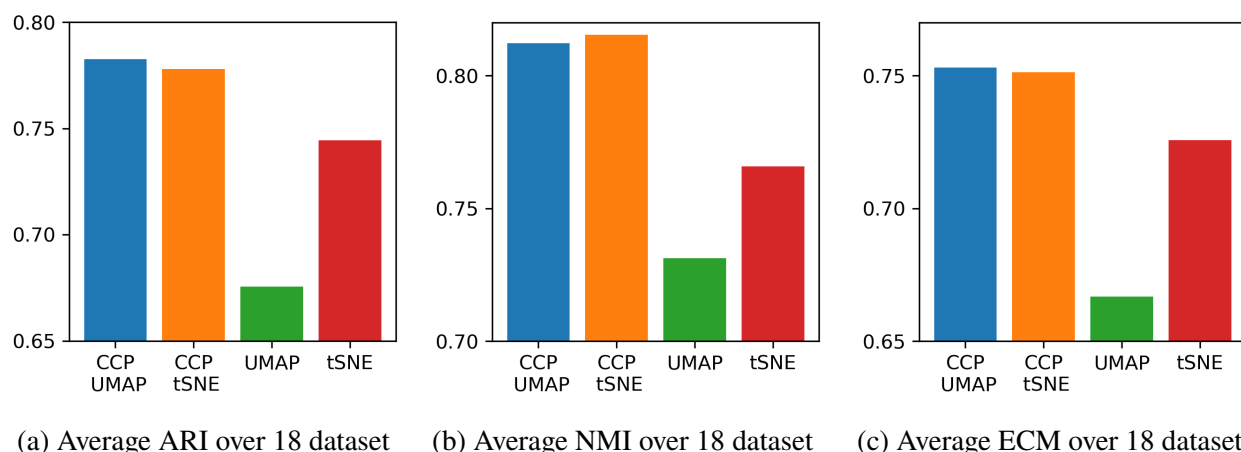


Figure 7.14 The average ARI, NMI, ECM of 18 datasets. 10 random initialization was used to compute CCP, CCP-assisted UMAP, CCP-assisted tSNE, standard UMAP and standard tSNE for each dataset. Leiden clustering was used to obtain the clustering results.

Figure 7.14 show the average ARI, NMI and ECM of CCP-assisted UMAP, CCP-assisted tSNE, UMAP and tSNE across 18 dataset. For each dataset, we conducted 10 random seeds to perform dimensionality reduction, utilizing Leiden clustering to generate clustering labels. These labels were then compared to the annotated cell types provided by the original authors.

CCP-assisted UMAP demonstrates a 24% improvement in ARI, 15% in NMI, and 17% in ECM over standard UMAP. Similarly, CCP-assisted tSNE improves standard tSNE by 11% ARI, 10% NMI

and 8% in ECM. Additionally, CCP-assisted UMAP and tSNE have a higher ECM score, indicating that their clustering is more stable. Notably, both CCP-assisted UMAP and tSNE yield higher ECM scores, indicating more stable clustering. Interestingly, standard tSNE outperforms UMAP. However, UMAP's performance heavily relies on accurately finding nearest neighbors, which can be challenging with noisy, sparse, and high-dimensional data. CCP effectively reduces dimensions, enabling UMAP to find neighbors more effectively and resulting in improved visualization.

For a detailed comparison between CCP-assisted, PCA-assisted, and NMF-assisted visualizations, please refer to Section 7B.1.3 in the supporting materials.

## 7.2.4 Discussion

### 7.2.4.1 Large Data

While CCP proves to be an efficient dimensionality reduction technique for datasets with a large number of features, such as in the case of scRNA-seq data, it may encounter limitations due to the necessity of computing cell-cell correlations for each super-gene. To address this challenge when dealing with larger datasets, we proposed a subsampling approach.

Let  $\mathcal{Z} = \{\mathbf{z}_1, \dots, \mathbf{z}_M\}$  be the training data used to develop a CCP model, and  $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_T\}$  be a new dataset or additional data. Using the training data, gene partitions  $S^n$ , cutoff distance  $r_c^{S^n}$  and the connectivity  $\eta^{S^n}$  are determined. Then, we embed new data to the trained model, utilizing the following modification to Equation 5.15

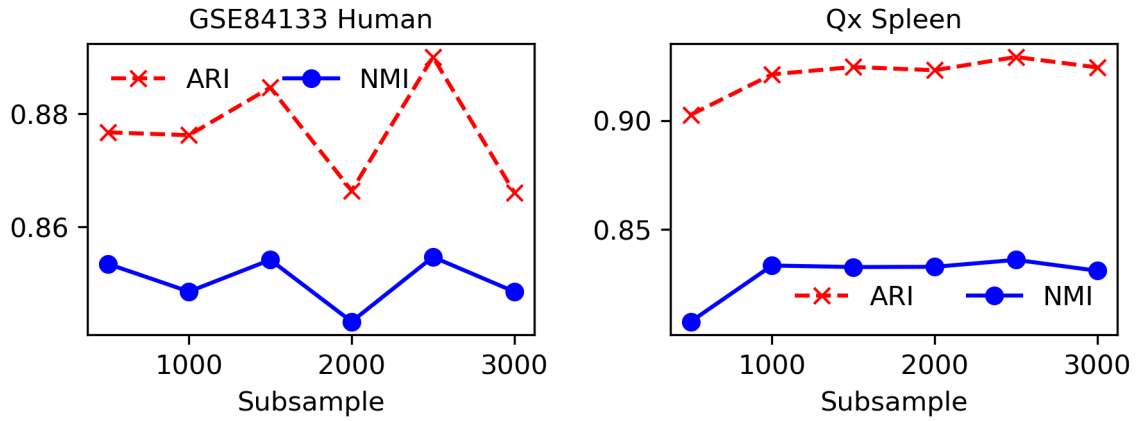
$$x_i^n = \sum_{m=1}^M \Phi(\|\mathbf{y}_i^{S^n} - \mathbf{z}_m^{S^n}\|; \eta^n, \tau, \kappa), \quad (7.1)$$

to obtain appropriate super-genes.

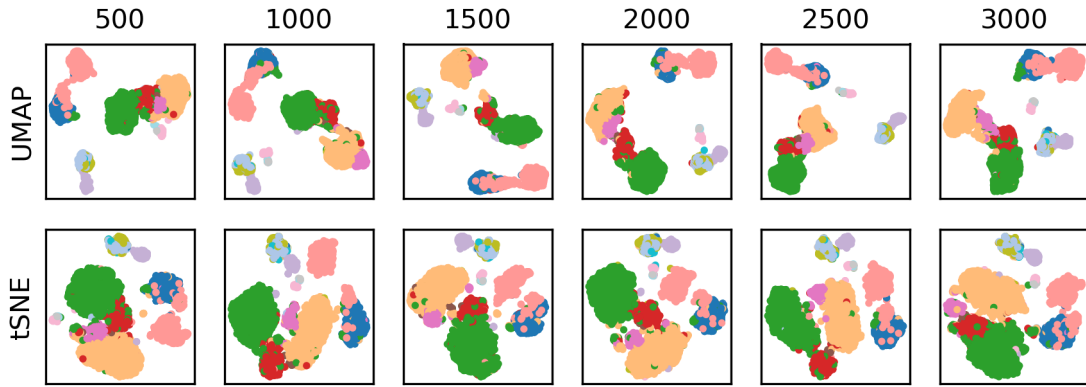
We verified the subsampling approach on GSE84133 human and Qx Spleen data. We combined all four patient's sequencing data into one superset for this analysis. We randomly subsampled 500, 1000, 1500, 2000, 2500, 3000 samples as a training data, and performed the subsampling under 10 random seeds. We projected the testing data using Equation 7.1, followed by Leiden clustering. ARI and NMI were computed, and the average scores are reported in Figure 7.15. Notably, both the GSE84133 human and Qx Spleen datasets exhibited consistent and stable results under varying

number of subsampling.

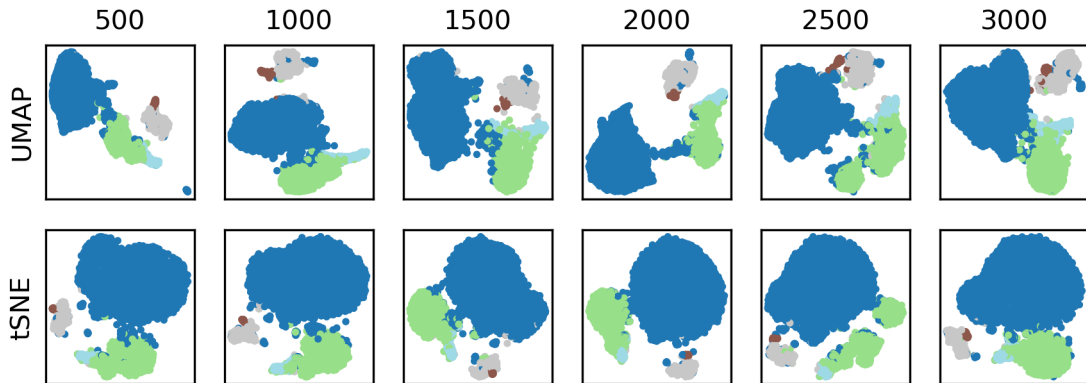
Additionally, we also show the CCP-assisted UMAP and tSNE for both data when subsampling was utilized. Notably, all visualizations were comparable, underscoring the stability of CCP-assisted visualizations even under subsampling.



(a) ARI and NMI under different subsampling



(b) CCP-assisted UMAP and tSNE visualization of GSE84133 human



(c) CCP-assisted UMAP and tSNE visualization of Qx Spleen

Figure 7.15 UMAP and tSNE visualization of GSE84133 human and Qx Spleen data under different number of subsampling. 300 super-genes were generated from CCP, and Leiden clustering was used to obtain the clustering results. (a) ARI and NMI under different subsampling values. Left figure shows the ARI and NMI for GSE84133 Human, where the 4 patient data were combined. Right figure shows the ARI and NMI of Qx Spleen data. (c) CCP-assisted UMAP and tSNE of GSE84133 Human data under different number of subsampling. (d) CCP-assisted UMAP and tSNE of Qx Spleen data under different number of subsampling.

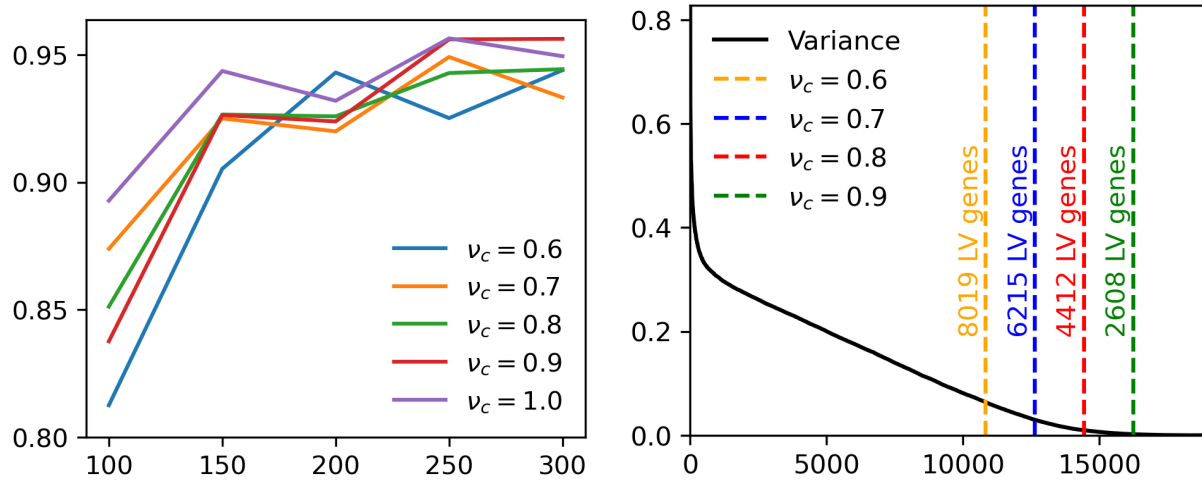


#### 7.2.4.2 Low Variance Genes

We have utilized LV-genes to enhance the predictive power of super-genes with a LV gene cluster. By using a high cutoff ratio, we can reduce the number of genes used in the feature partition, potentially resulting in a lower number of super-genes. To assess the impact of the cutoff ratio on the number of super-genes used for UMAP and tSNE visualizations, we conducted tests using GSE82187 and GSE75748 cell data. The discussion for GSE75748 cell data can be found in Section 7B.2.1 of the supporting materials.

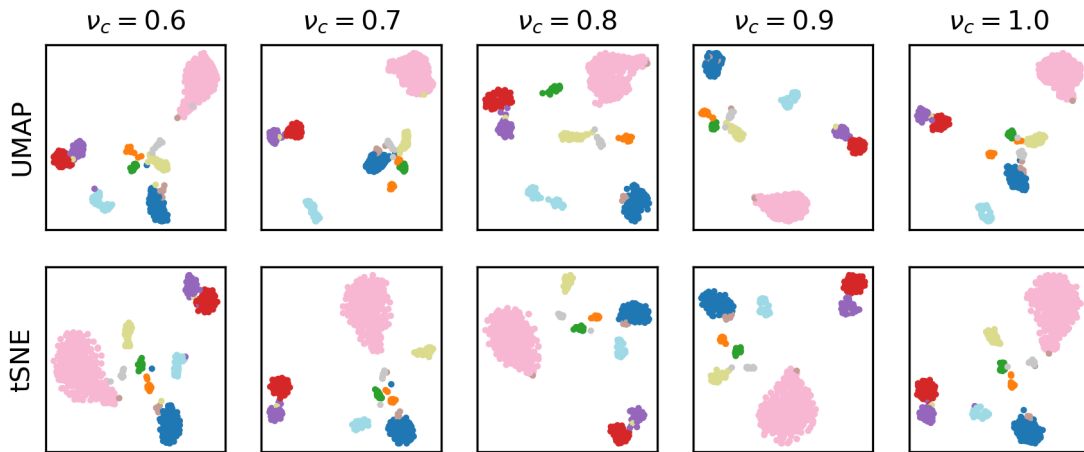
Figure 7.16 show the effect of varying the number of super-genes and the cutoff ratio on the predictive power and visualization of GSE82187 data. We utilized 10 random seeds to generate CCP super-genes using different numbers of super-genes and cutoff ratios. Subsequently, Leiden clustering was applied to obtain cluster labels, and the ARI was computed utilizing the cell labels provided by the original authors. Notably, across all cutoff ratios, the ARI increases with an augmented number of super-genes, plateauing at a comparable level around 300 super-genes. This indicates the robustness of LV-gene.

Figure 7.16(c) show the visualization of CCP-assisted UMAP and tSNE at various cutoff ratio. For the visualization, 300 super-genes were utilized, and UMAP and tSNE was applied to the super-genes to reduce the dimension to 2. Samples were then colored according to the cell types provided by the original authors. Note that all the visualization are comparable, indicating the robustness of LV-gene under different cutoff ratio.



(a) ARI of number of super-genes and  $\nu_c$

(b) Number of genes in LV genes



(c) CCP-assisted UMAP and tSNE visualization

Figure 7.16 Analysis of varying the cutoff ratio  $\nu_c$  on clustering and visualization of GSE82187 data. (a) ARI of leiden clustering when the number of super-genes and cutoff ratio is changed. (b) The number of genes in the LV-gene when  $\nu_c$  is changed. (c) Top and bottom row shows the CCP-assisted UMAP and tSNE visualization, and the columns corresponds to  $\nu_c = 0.6, 0.7, 0.8, 0.9$ . 300 super-genes were used to initialize UMAP and tSNE, and the samples were colored according to the true cell type.

## 7.2.5 Conclusion

CCP is a nonlinear data-domain dimensionality reduction technique that leverages gene-gene correlations to partition genes, and utilizes cell-cell correlation to generate super-genes. Unlike methods that involve matrix diagonalization, CCP can be directly applied as a primary dimensionality reduction tool to complement traditional visualization techniques like UMAP and tSNE. In our

experiments with 18 datasets, CCP-assisted UMAP and CCP-assisted tSNE visualizations consistently outperformed the original UMAP and tSNE. On average, CCP-assisted UMAP improves the standard UMAP visualization by 24% in ARI and 15% in NMI, and CCP-assisted tSNE improves standard tSNE by 11% ARI and 10% NMI. Although the improvement for tSNE visualization is less than the improvement in UMAP, tSNE is sensitive to potential outliers and noise, where the visualization can become uninterpretable. CCP-assisted tSNE consistently show clear visualization in the 18 dataset we have tested. Additionally, CCP-assisted visualization improves PCA-assisted and NMF-assisted visualization in the 18 dataset we have tested. However, CCP comes with some disadvantages. For data with no clear gene-gene correlation, CCP will most likely not perform well. Additionally, although utilizing gene clustering removes the complication with computing distance in high dimensions, when the number of samples becomes large, the cell-cell correlation computation becomes time consuming. We show that subsampling via a training set is an effective approach to enable CCP for dealing with large data. One possible extension for gene clustering is to incorporate prior information, such as using known genes or utilizing known gene regulatory pathways, to guide in the clustering. Additionally, CCP can also be employed in many other single cell contexts, such as spatial transcriptomics and cell-cell communication, and for initializing deep learning methods.

### **7.2.6 Code and Data availability**

All data can be downloaded from Gene Expression Omnibus [14, 15]. The processing files for these data can be found on <https://github.com/hozumiyu/SingleCellDataProcess>. CCP is made available through our web-server at <https://weilab.math.msu.edu/CCP/> or through the source code <https://github.com/hozumiyu/CCP>. The code to reproduce this paper is found at <https://github.com/hozumiyu/CCP-scRNAseq-UMAP-TSNE>.

## 7.3 Topological Non-Negative Matrix Factorization for Single Cell RNA Sequencing Data

### 7.3.1 Introduction

In this work, we introduce PL-regularized NMF, namely the topological NMF (TNMF) and the robust topological NMF (rTNMF). Both TNMF and rTNMF can better capture multiscale geometric information than the standard GNMF and rGNMF. To achieve improved performance, PL is constructed by observing cell-cell interactions at multiple scales through filtration, creating a sequence of simplicial complexes. We can then view the spectra at each complex associated with a filtration to capture both topological and geometric information. Additionally, we introduce  $k$ -NN based PL to TNMF and rTNMF, referred to as  $k$ -TNMF and  $k$ -rTNMF, respectively. The  $k$ -NN based PL reduces the number of hyperparameters compared to the standard PL algorithm.

For the methodology and the algorithm for TNMF, refer to section 6.3. We will utilize the both  $k$ -NN and the cutoff based PL regularized NMF on scRNA-seq, and evaluate the effectiveness of clustering using adjusted rand index (ARI), normalized mutual information (NMI), purity and accuracy (ACC). Additionally, residue-similarity (RS) plot is utilized to visualize the effectiveness of the clustering.

### 7.3.2 Results

#### 7.3.2.1 Benchmark Data

We have performed benchmark on 12 publicly available datasets. The GEO accession number, reference, organism, number of cell types, and number of samples are recorded in Table 7.3. For each data, cell types with less than 15 cells were removed. The data was log-normalized, and then each cell was normalized to have unit length. For GNMF and rGNMF,  $k = 8$  neighbors were used. For TNMF and rTNMF, 8 filtration values were used to construct PL, and for each scale, binary selection  $\zeta_p = 0, 1$  was used. For  $k$ -TNMF and  $k$ -rTNMF,  $k = 8$  was used with  $\zeta_p = 0, 1$ . For each test, double nonnegative singular value decomposition with zeros filled with the average of  $X$  (NNDSDVA) was used for the initialization. For the rank, we chose  $\sqrt{N}$ , where  $N$  is the number of cells. Then  $k$ -means clustering was applied to obtain the clustering results.

Table 7.3 GEO accession code, reference, organism type, cell type, number of samples, and number of genes of each dataset.

Geo Accession	Reference	Organism	Cell type	# of Samples	# of Genes
GSE67835	Dramanis [5]	Human	8	420	22084
GSE75748 time	Chu [6]	Human	6	758	19189
GSE82187	Gokce [7]	Mouse	8	705	18840
GSE84133human1	Baron [8]	Human	9	1895	20125
GSE84133human2	Baron [8]	Human	9	1702	20125
GSE84133human3	Baron [8]	Human	9	3579	20125
GSE84133human4	Baron [8]	Human	6	1275	20125
GSE84133mouse1	Baron [8]	Mouse	6	782	14878
GSE84133mouse2	Baron [8]	Mouse	8	1036	14878
GSE57249	Biase [21]	Human	3	49	25737
GSE64016	Leng [22]	Human	4	460	19084
GSE94820	Villani [10]	Human	5	1140	26593

### 7.3.2.2 Benchmarking PL regularized NMF

In order to benchmark persistent Laplacian regularized NMF, we compared our methods to other commonly used NMF methods, namely the GNMF, rGNMF, rNMF and NMF. For a fair comparison, We omitted supervised or semi-supervised methods. For  $k$ -rTNMF, rTNMF,  $k$ -TNMF, TNMF, GNMF and rGNMF, we set  $\alpha = 1$  for all tests.

Table 7.4 shows the ARI values of the NMF methods for the 12 data we have tested. The bold number indicate the highest performance. Figure 7.17 depicts the average ARI value over the 12 datasets for each method.

Table 7.4 ARI of NMF methods across 12 datasets.

Data	$k$ -rTNMF	rTNMF	$k$ -TNMF	TNMF	rGNMF	GNMF	rNMF	NMF
GSE67835	<b>0.9454</b>	0.9236	0.9306	0.8533	0.9391	0.9109	0.7295	0.7314
GSE64016	<b>0.2569</b>	0.1544	0.2237	0.1491	0.1456	0.1605	0.1455	0.1466
GSE75748time	0.6421	<b>0.6581</b>	0.5963	0.6099	0.6104	0.5790	0.5969	0.5996
GSE82187	<b>0.9877</b>	0.9815	0.9676	0.9809	0.7558	0.7577	0.8221	0.8208
GSE84133human1	0.8310	<b>0.8969</b>	0.8301	0.8855	0.8220	0.7907	0.7080	0.6120
GSE84133human2	<b>0.9469</b>	0.9072	0.9433	0.9255	0.9350	0.9255	0.8930	0.8929
GSE84133human3	0.8504	0.9179	0.8625	<b>0.9181</b>	0.8447	0.8361	0.7909	0.8089
GSE84133human4	0.8712	<b>0.9692</b>	0.8712	<b>0.9692</b>	0.8699	0.8681	0.8311	0.8311
GSE84133mouse1	<b>0.8003</b>	0.7894	<b>0.8003</b>	0.7913	0.7945	0.7918	0.6428	0.6348
GSE84133mouse2	0.6953	0.8689	0.7005	<b>0.9331</b>	0.6808	0.6957	0.5436	0.5470
GSE57249	<b>1.0000</b>	0.9638	<b>1.0000</b>	0.9483	<b>1.0000</b>	<b>1.0000</b>	0.9483	0.9483
GSE94820	<b>0.6101</b>	0.5480	0.4916	0.5574	0.5139	0.5189	0.5440	0.5556

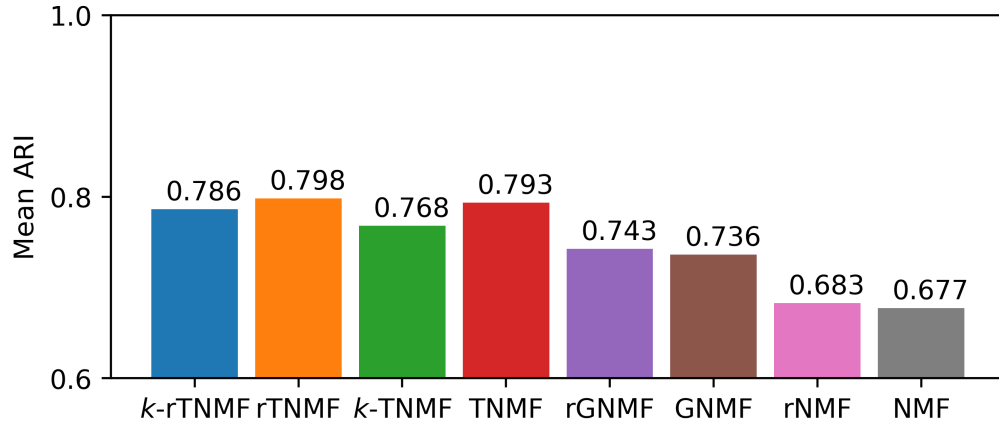


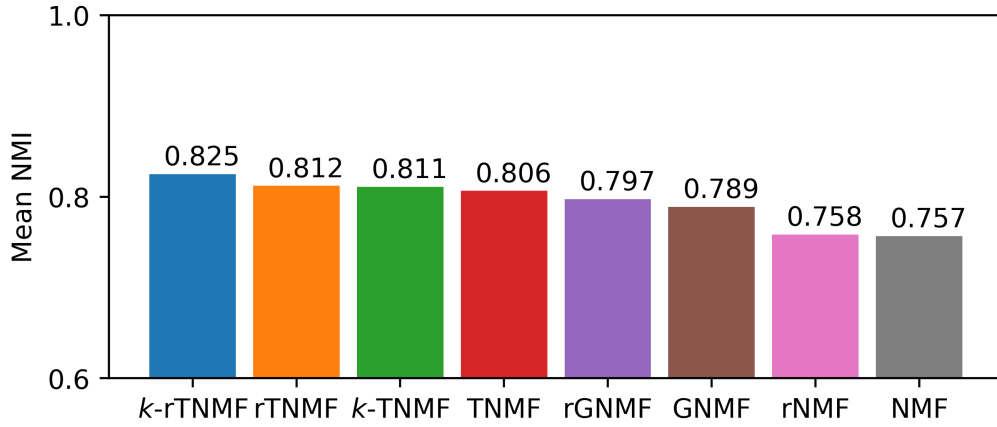
Figure 7.17 Average ARI of *k*-rTNMF, rTNMF, *k*-TNMF, TNMF, rGNMF, GNMF, rNMF and NMF for the 12 datasets.

Overall, PL regularized rNMF and NMF have the highest ARI value across all the datasets. *k*-rTNMF outperforms other NMF methods by at least 0.09 for GSE64016. All PL regularized NMF methods outperform other NMF methods by at least 0.14 for GSE82187. For GSE84133 human 3, both rTNMF and TNMF outperform other methods by 0.07. TNMF improves other methods by more than 0.2 for GSE84133 mouse 2. Lastly, *k*-rTNMF has the highest ARI value for GSE94820. Moreover, rTNMF improves rGNMF by 0.05, and TNMF improves GNMF by about 0.06. *k*-TNMF and *k*-rTNMF also improve GNMF and rGNMF by about 0.03.

Table 7.5 shows the NMI values of the NMF methods for the 12 datasets we have tested. The bold number indicate the highest performance. Figure 7.18 shows the average NMI value over the 12 datasets.

Table 7.5 NMI of NMF methods across 12 datasets.

data	<i>k</i> -rTNMF	rTNMF	<i>k</i> -TNMF	TNMF	rGNMF	GNMF	rNMF	NMF
GSE67835	<b>0.9235</b>	0.8999	0.9107	0.8607	0.9104	0.8858	0.7975	0.8017
GSE64016	0.3057	0.2059	<b>0.3136</b>	0.1869	0.2593	0.2562	0.1896	0.1849
GSE75748time	0.7522	<b>0.7750</b>	0.7159	0.7343	0.7235	0.6971	0.7227	0.7244
GSE82187	<b>0.9759</b>	0.9691	0.9298	0.9668	0.8802	0.8754	0.9124	0.9117
GSE84133human1	<b>0.8802</b>	0.8716	0.8785	0.8780	0.8713	0.8310	0.8226	0.7949
GSE84133human2	<b>0.9363</b>	0.8937	0.9313	0.9070	0.9237	0.9145	0.8835	0.8829
GSE84133human3	0.8500	<b>0.8718</b>	0.8577	0.8677	0.8439	0.8357	0.8215	0.8260
GSE84133human4	0.8795	<b>0.9542</b>	0.8795	<b>0.9542</b>	0.8775	0.8753	0.8694	0.8694
GSE84133mouse1	<b>0.8664</b>	0.8498	<b>0.8664</b>	0.8495	0.8596	0.8565	0.7634	0.7593
GSE84133mouse2	0.8218	<b>0.8355</b>	0.8299	0.8713	0.8005	0.8129	0.7258	0.7272
GSE57249	<b>1.0000</b>	0.9505	<b>1.0000</b>	0.9293	<b>1.0000</b>	<b>1.0000</b>	0.9293	0.9293
GSE94820	<b>0.7085</b>	0.6657	0.6157	0.6716	0.6195	0.6258	0.6624	0.6693

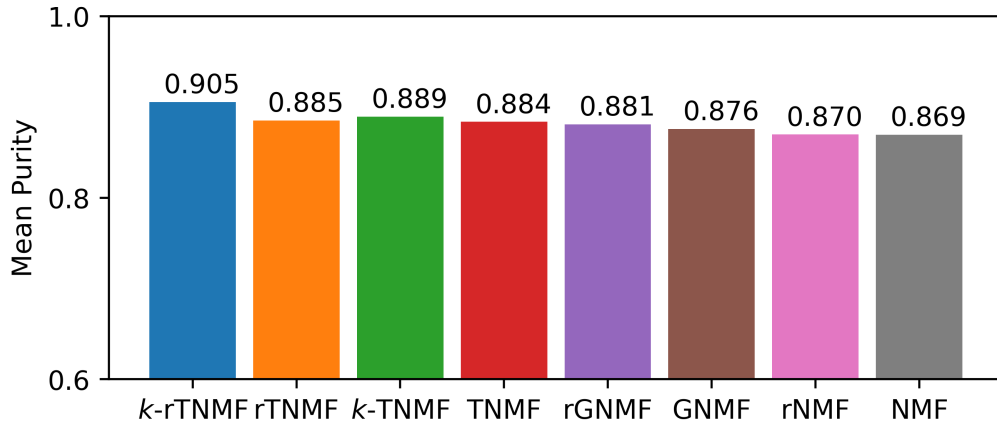
Figure 7.18 Average NMI values of *k*-rTNMF, rTNMF, *k*-TNMF, TNMF, rGNMF, GNMF, rNMF and NMF for the 12 datasets.

Interestingly, *k*-rTNMF and *k*-TNMF on average have higher NMI values than rTNMF and TNMF, respectively. However, all PL regularized methods outperform rGNMF, GNMF, rNMF and NMF. Overall, PL regularized methods outperform other methods. Most noticeably, *k*-rTNMF, rTNMF and TNMF outperform standard NMF methods by 0.06 for GSE82187. Both rTNMF and TNMF outperform rGNMF and GNMF by 0.08 for GSE84133 human 4.

Table 7.6 shows the purity values of the NMF methods for the 12 datasets we have tested. The bold number indicate the highest performance. Figure 7.19 shows the average purity over the 12 datasets.

Table 7.6 Purity of NMF methods across 12 datasets.

data	<i>k</i> -rTNMF	rTNMF	<i>k</i> -TNMF	TNMF	rGNMF	GNMF	rNMF	NMF
GSE67835	<b>0.9643</b>	0.9267	0.9595	0.9024	0.9595	0.9476	0.8726	0.8719
GSE64016	<b>0.6048</b>	0.4913	0.5846	0.5013	0.5339	0.5398	0.5080	0.5050
GSE75748time	<b>0.7736</b>	0.7512	0.7533	0.7454	0.7553	0.7387	0.7467	0.7455
GSE82187	<b>0.9927</b>	0.9895	0.9620	0.9888	0.9620	0.9594	0.9693	0.9692
GSE84133human1	<b>0.9543</b>	0.9357	0.9536	0.9382	0.9490	0.9187	0.9189	0.9099
GSE84133human2	<b>0.9818</b>	0.9614	0.9806	0.9661	0.9777	0.9736	0.9602	0.9600
GSE84133human3	0.9472	0.9485	<b>0.9531</b>	0.9460	0.9452	0.9420	0.9464	0.9466
GSE84133human4	0.9427	<b>0.9882</b>	0.9427	<b>0.9882</b>	0.9427	0.9420	0.9412	0.9412
GSE84133mouse1	<b>0.9565</b>	0.9540	<b>0.9565</b>	0.9540	0.9552	0.9540	0.9309	0.9299
GSE84133mouse2	0.9585	0.9410	<b>0.9604</b>	0.9373	0.9466	0.9507	0.9185	0.9199
GSE57249	<b>1.0000</b>	0.9857	<b>1.0000</b>	0.9796	<b>1.0000</b>	<b>1.0000</b>	0.9796	0.9796
GSE94820	<b>0.7893</b>	0.7462	0.6658	0.7550	0.6421	0.6421	0.7429	0.7531

Figure 7.19 Average purity values of *k*-rTNMF, rTNMF, *k*-TNMF, TNMF, rGNMF, GNMF, rNMF and NMF for the 12 datasets.

In general, PL-regularized methods achieve higher purity values compared to other NMF methods. Purity measures the maximum intersection between true and predicted classes, which is why we do not observe a significant difference, as seen in ARI and NMI. Furthermore, since purity does not account for the size of a class, and given the imbalanced class sizes in scNRA-seq data, it is not surprising that the purity values are similar.

Table 7.7 shows the ACC of the NMF methods for the 12 datasets we have tested. The bold number indicate the highest performance. Figure 7.20 shows the average ACC over the 12 datasets.



Table 7.7 ACC of NMF methods across 12 datasets.

data	$k$ -rTNMF	rTNMF	$k$ -TNMF	TNMF	rGNMF	GNMF	rNMF	NMF
GSE67835	<b>0.9643</b>	0.9243	0.9595	0.9000	0.9595	0.9383	0.8357	0.8364
GSE64016	<b>0.5700</b>	0.4870	0.5502	0.4746	0.4891	0.4537	0.4691	0.4759
GSE75748time	<b>0.7565</b>	0.7438	0.7414	0.6917	0.7355	0.7241	0.6873	0.6875
GSE82187	<b>0.9927</b>	0.9895	0.9599	0.9888	0.8512	0.8514	0.8896	0.8889
GSE84133human1	0.8973	<b>0.9194</b>	0.8974	0.9088	0.8889	0.8364	0.7988	0.7370
GSE84133human2	0.9260	0.9069	0.9242	<b>0.9447</b>	0.9224	0.9177	0.8998	0.8994
GSE84133human3	0.8539	<b>0.9456</b>	0.8597	0.9419	0.8498	0.8228	0.8032	0.8178
GSE84133human4	0.8831	<b>0.9882</b>	0.8831	<b>0.9882</b>	0.8824	0.8816	0.8847	0.8847
GSE84133mouse1	<b>0.8581</b>	0.8542	<b>0.8581</b>	0.8542	0.8555	0.8542	0.7361	0.7311
GSE84133mouse2	0.8232	<b>0.9101</b>	0.8263	0.9305	0.7903	0.8155	0.7239	0.7294
GSE57249	<b>1.0000</b>	0.9857	<b>1.0000</b>	0.9796	<b>1.0000</b>	<b>1.0000</b>	0.9796	0.9796
GSE94820	<b>0.7533</b>	0.7119	0.6482	0.7201	0.6088	0.6107	0.7091	0.7189

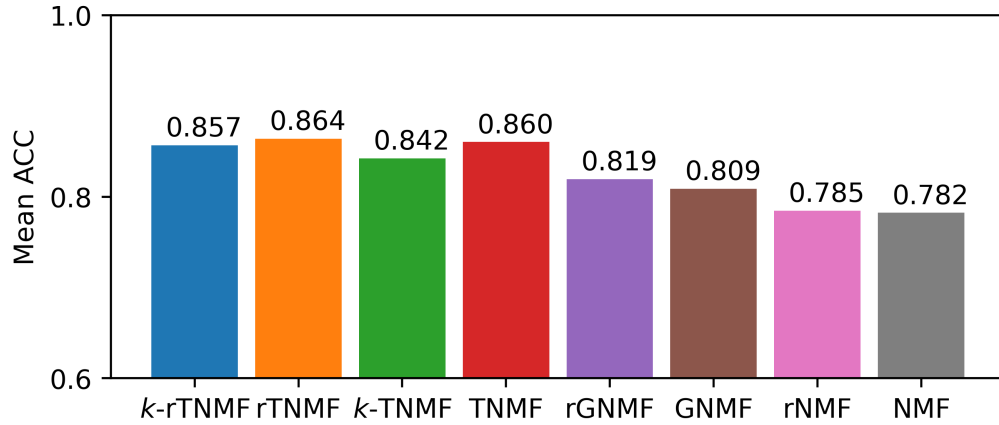


Figure 7.20 Average ACC of  $k$ -rTNMF, rTNMF,  $k$ -TNMF, TNMF, rGNMF, GNMF, rNMF and NMF for the 12 datasets.

Once again, we see that PL regularized methods have higher ACC than other NMF methods. RTNMF and TNMF improves rGNMF and GNMF by 0.05, and  $k$ -rTNMF and  $k$ -TNMF improves rGNMF and GNMF by 0.04. We see an improvement in ACC for both  $k$ -rTNMF and  $k$ -TNMF for GSE64016. All 4 PL regularized methods improve ACC of GSE82187 by 0.1. RTNMF and TNMF improve GSE84133 mouse 2 by at least 0.1 as well.

### 7.3.2.3 Overall performance

Figure 7.21 shows the average ARI, NMI, purity and ACC of  $k$ -rTNMF, rTNMF,  $k$ -TNMF, TNMF, rGNMF, GNMF, rNMF, NMF across 10 datasets. All PL regularized NMF methods

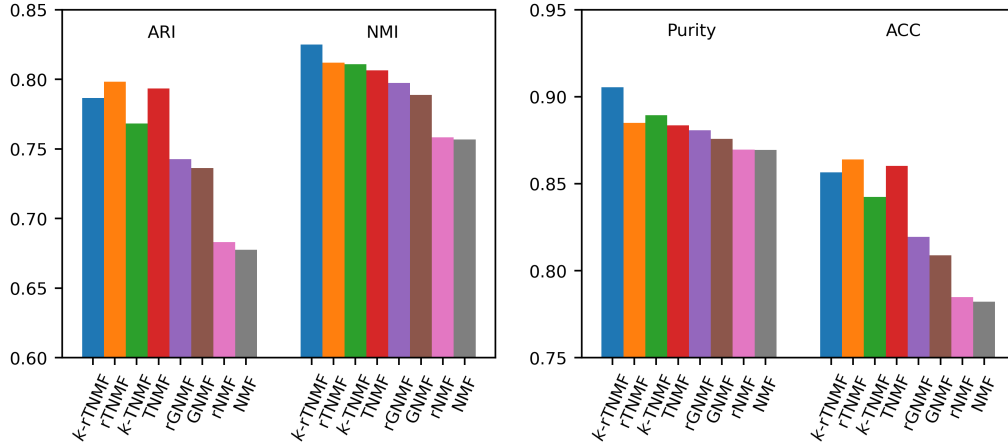


Figure 7.21 Average ARI, NMI, purity and ACC of  $k$ -rTNMF, rTNMF,  $k$ -TNMF, TNMF, rGNMF, GNMF, rNMF, NMF across 12 datasets.

outperform the traditional rGNMF, GNMF, rNMF and NMF. Both rTNMF and TNMF have higher average ARI and purity than the  $k$ -NN based PL counterparts. However,  $k$ -rTNMF and  $k$ -TNMF have higher average NMI than rTNMF and TNMF, respectively.  $k$ -rTNMF has a significantly higher purity than other methods.

### 7.3.3 Discussion

#### 7.3.3.1 Visualization of meta-genes based UMAP and t-SNE

Both UMAP and t-SNE are well-known for their effectiveness in visualization. However, these methods may not perform as competitively in clustering or classification tasks. Therefore, it is beneficial to employ NMF-based methods to enhance the visualization capabilities of UMAP and t-SNE.

In this process, we generate meta-genes and subsequently utilize UMAP or t-SNE to further reduce the data to 2 dimensions for visualization. For a dataset with  $M$  cells, the number of meta-genes will be the integer value of  $\sqrt{M}$ . To compare the standard UMAP and t-SNE plots with the top-NMF-assisted and top-rNMF-assisted UMAP and t-SNE visualizations, we used the default settings of the Python implementation of UMAP and the Scikit-learn implementation of t-SNE. For unassisted UMAP and t-SNE, we first removed low-abundance genes and performed log-transformation before applying UMAP and t-SNE.

Figure 7.22 shows the visualization of PL regularized NMF methods through UMAP. Each row corresponds to GSE67835, GSE75748 time, GSE94820 and GSE84133 mouse 2 data. The columns from left to right are the  $k$ -rTNMF assisted UMAP, rTNMF assisted UMAP,  $k$ -TNMF assisted UMAP, TNMF assisted UMAP and UMAP visualization. Samples were colored according to their true cell types.

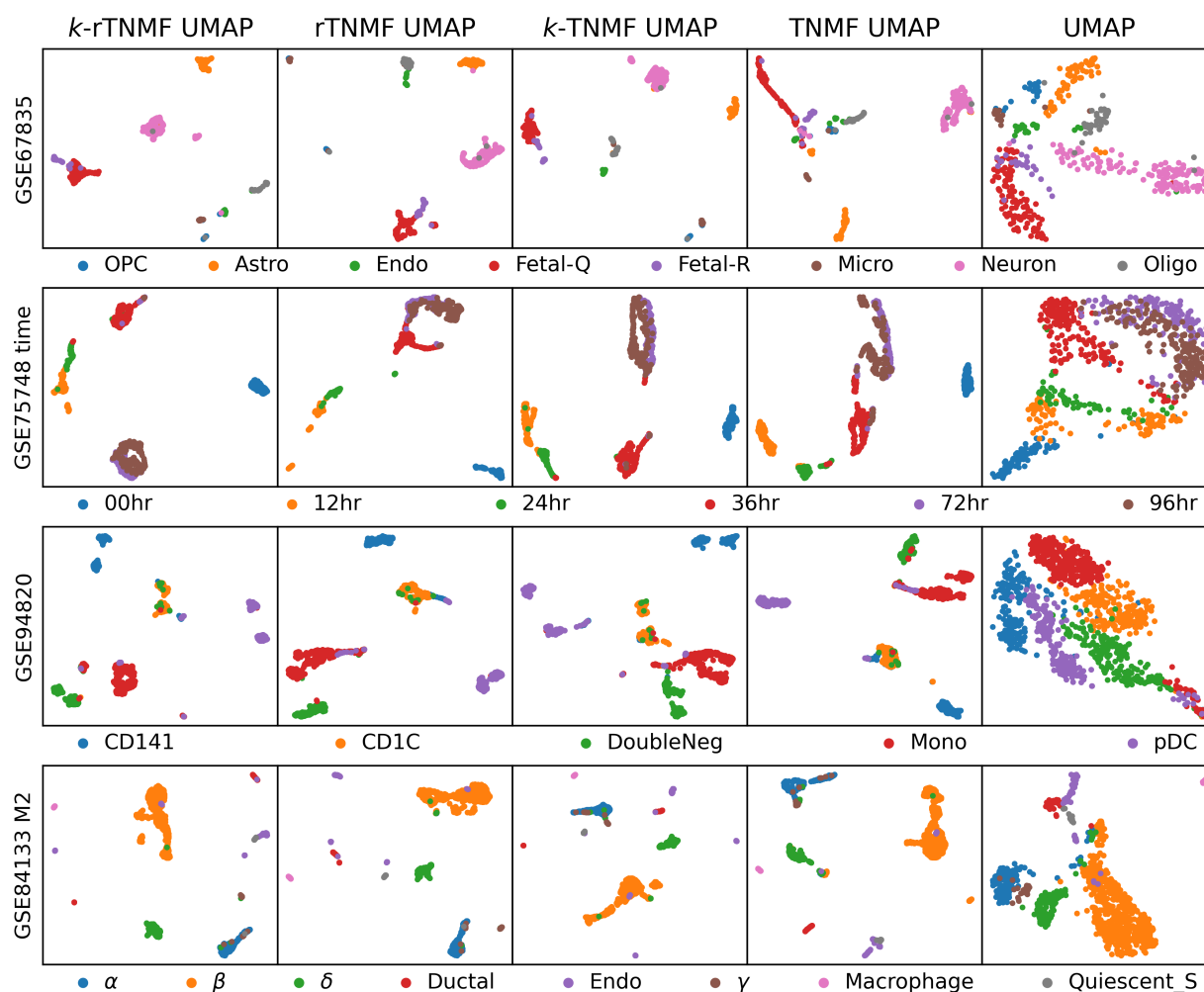


Figure 7.22 Visualization of top-NMF and top-rTNMF meta-genes through UMAP. Each row corresponds to GSE67835, GSE75748 time, GSE94820 and GSE84133 mouse 2 data. The columns from left to right are the  $k$ -rTNMF assisted UMAP, rTNMF assisted UMAP,  $k$ -TNMF assisted UMAP, TNMF assisted UMAP and UMAP visualization. Samples were colored according to their true cell types.

Figure 7.23 shows the visualization of PL regularized NMF through t-SNE. Each row corresponds to GSE67835, GSE75748 time, GSE94820 and GSE84133 mouse 2 data. The columns

from left to right are the  $k$ -rTNMF assisted t-SNE, rTNMF assisted t-SNE,  $k$ -TNMF assisted t-SNE, TNMF assisted t-SNE and t-SNE visualization. Samples were colored according to their true cell types.

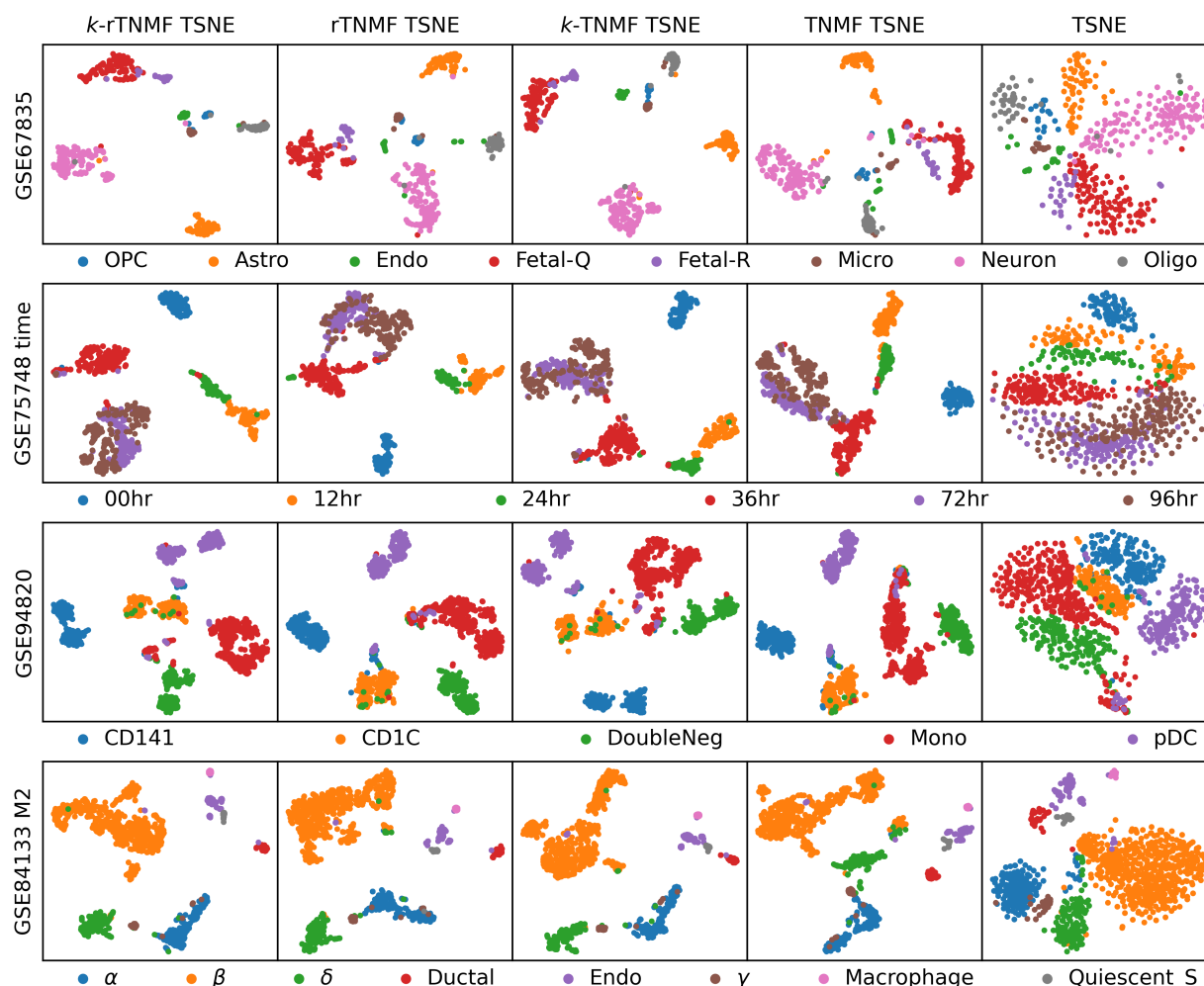


Figure 7.23 Visualization of top-NMF and top-rTNMF meta-genes through t-SNE. Each row corresponds to GSE67835, GSE75748 time, GSE94820 and GSE84133 mouse 2 data. The columns from left to right are the  $k$ -rTNMF assisted t-SNE, rTNMF assisted t-SNE,  $k$ -TNMF assisted t-SNE, TNMF assisted t-SNE and t-SNE visualization. Samples were colored according to their true cell types.

We see a considerable improvement in both top-NMF assisted and top-rTNMF assisted UMAP and t-SNE visualization.

**GSE67835** In the assisted UMAP and t-SNE visualizations of GSE67835, we observe a more distinct cluster, which includes a supercluster of fetal quiescent (Fetal-Q) and fetal replicating (Fetal-R) cells. Darmanis et al. [5] conducted a study that involved obtaining differential gene expression data for human adult brain cells and sequencing fetal brain cells for comparison. It is not surprising that the undeveloped Fetal-Q and Fetal-R cells do not exhibit significant differences and cluster together.

**GSE75748 time** In GSE75748 time data, Chu et al. [6] sequenced human embryonic stem cells at times 0hr, 12hr, 24hr, 36hr, 72hr, and 96hr under hypoxic conditions to observe differentiation. In unassisted UMAP and t-SNE, although some clustering is visible, there is no clear separation between the clusters. Additionally, two subclusters of 12hr cells are observed.

Notably, in the PL-regularized assisted UMAP and t-SNE visualizations, there is a distinct supercluster comprising the 72hr and 96hr cells, while cells from different time points form their own separate clusters. This finding aligns with Chu's observation that there was no significant difference between the 72hr and 96hr cells, suggesting that differentiation may have already occurred by the 72hr mark.

**GSE94820** Notice that in both t-SNE and UMAP, although there is a boundary, the cells do not form distinct clusters. This lack of distinct clustering can pose challenges in many clustering and classification methods. On the other hand, all PL-regularized NMF methods result in distinct clusters.

Among the PL-regularized NMF approaches, cutoff-based PL, rTNMF, and TNMF form a single  $CD1C^+$  ( $CD1C1$ ) cluster, whereas the  $k$ -NN induced PL,  $k$ -rTNMF, and  $k$ -TNMF exhibit two subclusters. Villani et al. [10] previously noted the similarity in the expression profile of  $CD1C1^-CD141^-$  (DoubleNeg) cells and monocytes. PL-regularized NMF successfully differentiates between these two types.

**GSE84133 mouse 2** PL-regularized NMF yields significantly more distinct clusters compared to unassisted UMAP and t-SNE. Notably, the beta and gamma cells form distinct clusters in PL-regularized NMF. Additionally, when PL-regularized NMF is applied to assist UMAP, potential outliers within the beta cell population become visible. Baron et al. [8] previously highlighted heterogeneity within the beta cell population, and we observe potential outliers in all visualizations.

#### **7.3.3.2 RS analysis**

Although UMAP and t-SNE are excellent tools for visualizing clusters, they may struggle to capture heterogeneity within clusters. Moreover, these methods can be less effective when dealing with a large number of classes. Therefore, it is essential to explore alternative visualization techniques.



Figure 7.24 RS plots of GSE67835 data. The columns from left to right correspond to  $k$ -rTNMF, rTNMF,  $k$ -TNMF, and TNMF. Each row corresponds to a cell type. For each section, the x-axis and y-axis correspond to the S-score and R-score, respectively. K-means was used to obtain a cluster label, and the Hungarian algorithm was used to map the cluster labels to the true labels. Each sample was colored according to their true labels.

In our approach, we visualize each cluster using RS plots as described in subsection 4.1.1. RS plots depict the relationship between the residue score (R score) and similarity score (S score) and have proven useful in various applications for visualizing data with multiple class types

[23, 24, 25, 26, 27].

Figure 7.24 shows the RS plots of PL-regularized NMF methods for GSE67835 data. The columns from left to right correspond to  $k$ -rTNMF, rTNMF,  $k$ -TNMF, and TNMF, while the rows correspond to the cell types. The x-axis and y-axis represent the S-score and R-score for each sample, respectively. The samples are colored according to their predicted cell types. Predictions were obtained using k-means clustering, and the Hungarian algorithm was employed to find the optimal mapping from the cluster labels to the true cell types.

We can see that TNMF fails to identify OP cells, whereas  $k$ -rTNMF, rTNMF, and  $k$ -TNMF are able to identify OPC cells. Notably, the S-score is quite low, indicating that the OPC did not form a cluster for TNMF. For fetal quiescent and replicating cells,  $k$ -rTNMF correctly identifies these two types, and the few misclassified samples are located on the boundaries. RTNMF is able to correctly identify fetal replicating cells but could not distinguish fetal quiescent cells from fetal replicating cells. The S-score is low for neurons in both rTNMF and TNMF, which shows a direct correlation with the number of misclassified cells.

#### 7.3.4 Conclusion

Persistent Laplacian-regularized NMF is a dimensionality reduction technique that incorporates multiscale topological interactions between the cells. Traditional graph Laplacian-based regularization only represents a single scale and cannot capture the multiscale features of the data. We have also shown that the  $k$ -NN induced persistent Laplacian outperforms other NMF methods and is comparable to the cutoff-based persistent Laplacian-regularized NMF methods. However, PL methods do come with their downside. In particular, the weights for each filtration must be determined prior to the reduction. If there are  $T$  filtrations, then the hyperparameter space is  $(T + 1)^T$ . However,  $k$ -NN induced PL reduces the number of parameters to  $2^T$ . In addition, we have shown that we can achieve a significant improvement even if we limit the hyperparameter space to  $2^T$ . We would like to further explore possible parameter-free versions of topological NMF. Additionally, NMF methods are not globally convex, but we have shown that with NNDSVDA initialization, our methods perform the best. One possible extension to the proposed methods is to incorporate



higher-order persistent Laplacians in the regularization framework, which will reveal higher-order interactions. In addition, we would like to expand the ideas to tensor decomposition, such as Canonical Polyadic Decomposition (CPD) and Tucker decomposition, multimodal omics data, and spatial transcriptomics data.

### **7.3.5 Data availability and code**

The data and model used to produce these results can be obtained at <https://github.com/hozumiyu/TopologicalNMF-scRNAseq>.

## BIBLIOGRAPHY

- [1] Yuta Hozumi, Rui Wang, and Guo-Wei Wei. Ccp: Correlated clustering and projection for dimensionality reduction. *arXiv preprint arXiv:2206.04189*, 2022.
- [2] Kelin Xia, Kristopher Opron, and Guo-Wei Wei. Multiscale multiphysics and multidomain models—flexibility and rigidity. *The Journal of chemical physics*, 139(19):11B614\_1, 2013.
- [3] 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.
- [4] Monika S Kowalczyk, Itay Tirosh, Dirk Heckl, Tata Nageswara Rao, Atray Dixit, Brian J Haas, Rebekka K Schneider, Amy J Wagers, Benjamin L Ebert, and Aviv Regev. Single-cell rna-seq reveals changes in cell cycle and differentiation programs upon aging of hematopoietic stem cells. *Genome research*, 25(12):1860–1872, 2015.
- [5] 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*, 112(23):7285–7290, 2015.
- [6] Li-Fang Chu, Ning Leng, Jue Zhang, Zhonggang Hou, Daniel Mamott, David T Vereide, Jeea Choi, Christina Kendzioriski, Ron Stewart, and James A Thomson. Single-cell rna-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm. *Genome biology*, 17:1–20, 2016.
- [7] Ozgun Gokce, Geoffrey M Stanley, Barbara Treutlein, Norma F Neff, J Gray Camp, Robert C Malenka, Patrick E Rothwell, Marc V Fuccillo, Thomas C Südhof, and Stephen R Quake. Cellular taxonomy of the mouse striatum as revealed by single-cell rna-seq. *Cell reports*, 16(4):1126–1137, 2016.
- [8] 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, et al. 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.
- [9] Gaëlle Breton, Shiwei Zheng, Renan Valieris, Israel Tojal da Silva, Rahul Satija, and Michel C Nussenzweig. Human dendritic cells (dcs) are derived from distinct circulating precursors that are precommitted to become cd1c+ or cd141+ dcs. *Journal of Experimental Medicine*, 213(13):2861–2870, 2016.
- [10] Alexandra-Chloé Villani, Rahul Satija, Gary Reynolds, Siranush Sarkizova, Karthik Shekhar, James Fletcher, Morgane Griesbeck, Andrew Butler, Shiwei Zheng, Suzan Lazo, et al. Single-cell rna-seq reveals new types of human blood dendritic cells, monocytes, and progenitors.

*Science*, 356(6335):eaah4573, 2017.

- [11] Pengzhi Yu, Guangjin Pan, Junying Yu, and James A Thomson. Fgf2 sustains nanog and switches the outcome of bmp4-induced human embryonic stem cell differentiation. *Cell stem cell*, 8(3):326–334, 2011.
- [12] Adam Rodaway, Hiroyuki Takeda, Sumito Koshida, Joanne Broadbent, Brenda Price, James C Smith, Roger Patient, and Nigel Holder. Induction of the mesendoderm in the zebrafish germ ring by yolk cell-derived tgfbeta family signals and discrimination of mesoderm and endoderm by fgf. *Development*, 126(14):3067–3078, 1999.
- [13] Shinsuke Tada, Takumi Era, Chikara Furusawa, Hidetoshi Sakurai, Satomi Nishikawa, Masaki Kinoshita, Kazuki Nakao, Tsutomu Chiba, and Shin-Ichi Nishikawa. Characterization of mesendoderm: a diverging point of the definitive endoderm and mesoderm in embryonic stem cell differentiation culture. *Development*, 2005.
- [14] Ron Edgar, Michael Domrachev, and Alex E Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic acids research*, 30(1):207–210, 2002.
- [15] 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, et al. Ncbi geo: archive for functional genomics data sets—update. *Nucleic acids research*, 41(D1):D991–D995, 2012.
- [16] Liang Chen, Weinan Wang, Yuyao Zhai, and Minghua Deng. Deep soft k-means clustering with self-training for single-cell rna sequence data. *NAR genomics and bioinformatics*, 2(2):lqaa039, 2020.
- [17] Nicholas Schaum, Jim Karkanias, Norma F Neff, Andrew P May, Stephen R Quake, Tony Wyss-Coray, Spyros Darmanis, Joshua Batson, Olga Botvinnik, Michelle B Chen, et al. Single-cell transcriptomics of 20 mouse organs creates a tabula muris: The tabula muris consortium. *Nature*, 562(7727):367, 2018.
- [18] Mauro J Muraro, Gitanjali Dharmadhikari, Dominic Grün, Nathalie Groen, Tim Dielen, Erik Jansen, Leon Van Gurp, Marten A Engelse, Françoise Carlotti, Eelco Jp De Koning, et al. A single-cell transcriptome atlas of the human pancreas. *Cell systems*, 3(4):385–394, 2016.
- [19] Roman A Romanov, Amit Zeisel, Joanne Bakker, Fatima Girach, Arash Hellysaz, Raju Tomer, Alan Alpar, Jan Mulder, Frederic Clotman, Erik Keimpema, et al. Molecular interrogation of hypothalamic organization reveals distinct dopamine neuronal subtypes. *Nature neuroscience*, 20(2):176–188, 2017.
- [20] Alexander J Gates, Ian B Wood, William P Hetrick, and Yong-Yeol Ahn. Element-centric clustering comparison unifies overlaps and hierarchy. *Scientific reports*, 9(1):8574, 2019.

- [21] 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.
- [22] Ning Leng, Li-Fang Chu, Chris Barry, Yuan Li, Jeea Choi, Xiaomao Li, Peng Jiang, Ron M Stewart, James A Thomson, and Christina Kendzierski. Oscope identifies oscillatory genes in unsynchronized single-cell rna-seq experiments. *Nature methods*, 12(10):947–950, 2015.
- [23] Yuta Hozumi, Kiyoto A Tanemura, and Guo Wei Wei. Preprocessing of single cell rna sequencing data using correlated clustering and projection. *Journal of chemical Information and Modeling*, accepted, 2023.
- [24] Hongsong Feng and Guo-Wei Wei. Virtual screening of drugbank database for herg blockers using topological laplacian-assisted ai models. *Computers in biology and medicine*, 153:106491, 2023.
- [25] Zailiang Zhu, Bozheng Dou, Yukang Cao, Jian Jiang, Yueying Zhu, Dong Chen, Hongsong Feng, Jie Liu, Bengong Zhang, Tianshou Zhou, et al. Tidal: Topology-inferred drug addiction learning. *Journal of Chemical Information and Modeling*, 63(5):1472–1489, 2023.
- [26] Li Shen, Hongsong Feng, Yuchi Qiu, and Guo-Wei Wei. Svsbi: sequence-based virtual screening of biomolecular interactions. *Communications Biology*, 6(1):536, 2023.
- [27] Sean Cottrell, Rui Wang, and Guowei Wei. PLPCA: Persistent Laplacian enhanced-PCA for microarray data analysis. *Journal of Chemical Information and Modeling*, doi.org/10.1021/acs.jcim.3c01023, 2023.

## APPENDIX 7A

### ADDITIONAL RESULTS FOR PREPROCESSING SINGLE CELL RNA SEQUENCING DATA USING CORRELATED CLUSTERING AND PROJECTION

#### 7A.1 Methods

##### 7A.1.1 Program and Packages

Python v3.8.5 was utilized for the benchmark, and the following Python packages were utilized: NumPy v1.19.2, Scikit-learn v0.23.2. For visualization, Plotly v5.9.0 and Matplotlib v2.5.2 were used. For preprocessing of the scRNA-seq data, instructions can be found on GitHub at <https://github.com/hozumiyu/SingleCellDataProcess>.

##### 7A.1.2 Benchmark Protocol

Before applying dimensionality reduction, a log-transform was performed on the data. PCA and CCP were then applied to the data to generate 50, 100, 150, 200, 250 and 300 components. For CCP,  $\tau = 6.0$  and  $\kappa = 2.0$  was used for the exponential kernel to generate super-genes. For each reduction, 20 random initializations were used to ensure robustness and stability.

For clustering,  $k$ -means was used, where  $k$  was chosen as the number of cell or gene types, provided by the data, indicated in Table 7.1. A total of 30 random initializations was used to compute the clusters, and for each clustering result, ARI, NMI, Silhouette score, and RSI were computed.

For classification, the support vector machine (SVM) with default parameters from Scikit-learn's package was used. We performed 5-fold cross validation, where 4 parts were used for training the SVM and 1 part was used for testing. Ten random seeds were used for each reduction. Since the number of cells within each class can differ greatly in scRNA-seq data, we modified the training and testing sets. We removed any cell type with less than 5 cells and for the training set, we randomly sampled up to five times the average number of samples per cell type to ensure a balanced training set. We computed the balanced accuracy and RSI for each of the classification results.

#### 7A.2 Results

##### 7A.2.1 Additional benchmark

In this section, we compare the performance of CCP and PCA on datasets that were not introduced in the main text. More details on the benchmark can be found in Section subsection 7.1.2.1 of the main text.

Figure 7A.1 shows the performance of CCP and PCA on three datasets: GSE45719, GSE75748 cell, and GSE82187 data. The solid and dashed lines correspond to CCP and PCA, respectively, and the red and blue lines correspond to the ARI and NMI, respectively. For CCP,  $\tau = 6.0$  and  $\kappa = 2.0$  were used for the exponential kernel. For all three datasets, as the number of components increases, ARI and NMI either increase or remain stable for CCP. On the other hand, PCA shows instability in terms of ARI and NMI as the number of components increases. PCA has higher ARI

and NMI values when the number of components is 50, but in higher numbers of dimensions, PCA performs worse.

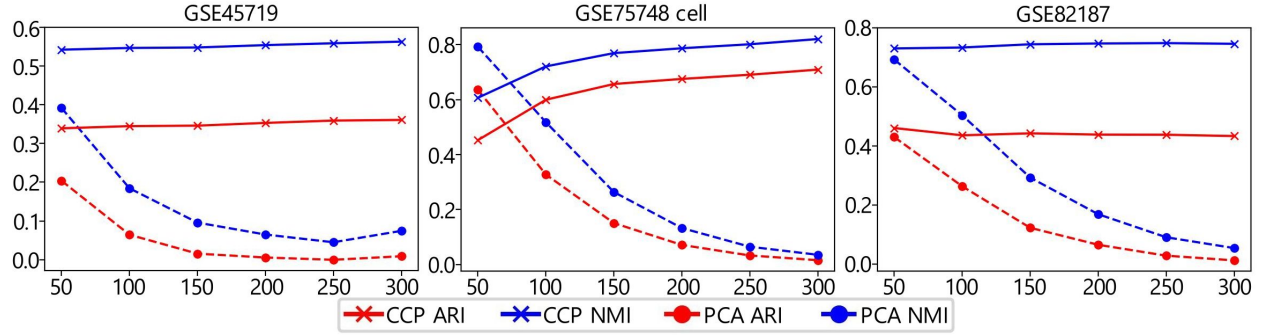


Figure 7A.1 ARI and NMI of the clustering results of CCP and PCA on GSE45719, GSE75748 cell. The red and blue lines correspond to CCP and PCA, respectively. For CCP, all the tests utilize  $\tau = 6$  and  $\kappa = 2$  for the exponential kernels. The  $x$ -axis shows the number of components in the reduction.

Figure 7A.2 shows the performance of CCP and PCA on the 6 datasets, GSE84133h1, GSE84133h2, GSE84133h3, GSE84133h4, GSE84133m1, and GSE84133m2. The solid and dashed lines correspond to CCP and PCA, respectively, and the red and blue lines correspond to the ARI and NMI, respectively. For GSE84133h1, GSE84133h2, and GSE84133h4 data, CCP outperforms PCA when the number of components is larger than 150 for both ARI and NMI. CCP is able to maintain its performance throughout higher dimensions, whereas PCA has a notable drop in its ARI and NMI values when the number of components is larger than 150. For GSE84133h3, PCA outperforms CCP until 300 components. We found that GSE84133h3 data is the only data, where PCA's ARI increased at higher dimensions. For the mouse datasets, GSE84133m1 and GSE84133m2, CCP outperforms PCA when the number of components is higher than 50. There is a slight decrease in ARI and NMI beyond 100 components for CCP, but it is much more stable than PCA.

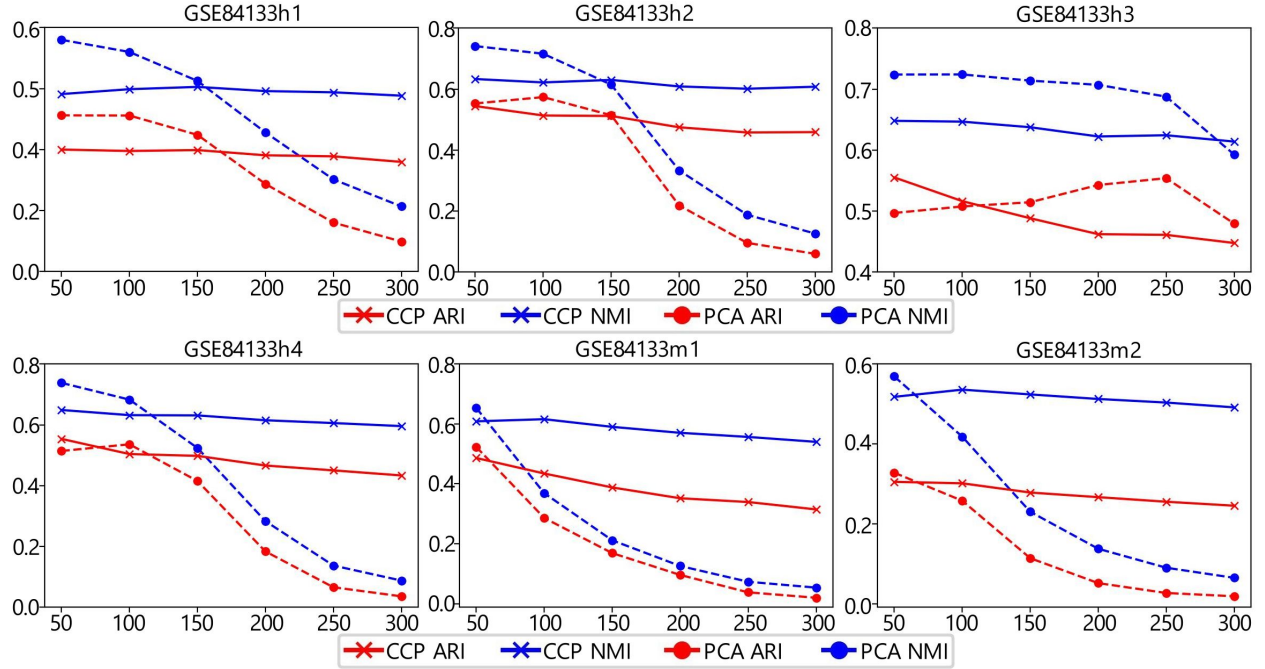


Figure 7A.2 ARI and NMI of the clustering results of CCP and PCA on GSE84133h1, GSE84133h2, GSE84133h3, GSE84133h4, GSE84133m1, and GSE84133m2. The red and blue lines correspond to CCP and PCA, respectively. For CCP, all the test utilize  $\tau = 6$  and  $\kappa = 2$  for the exponential kernels. The x-axis shows the number of components in the reduction.

Figure 7A.3 shows the performance of CCP and PCA on the 2 datasets, GSE89232 and GSE94820. The solid and dashed lines correspond to CCP and PCA, respectively, and the red and blue lines correspond to the ARI and NMI, respectively. For GSE94820, CCP outperforms PCA when the number of components is greater than 50. Both ARI and NMI improve until about 200 components. On the other hand, PCA has a notable drop in its performance from 100 components to 150 components, where the ARI and NMI drop to below 0.1, indicating very low clustering accuracy. CCP has poor performance for GSE89232 at all components tested. Both ARI and NMI are less than 0.2, indicating poor clustering accuracy. Similar to other data, PCA has stability concerns at higher dimensions, where both ARI and NMI become less than 0.1.

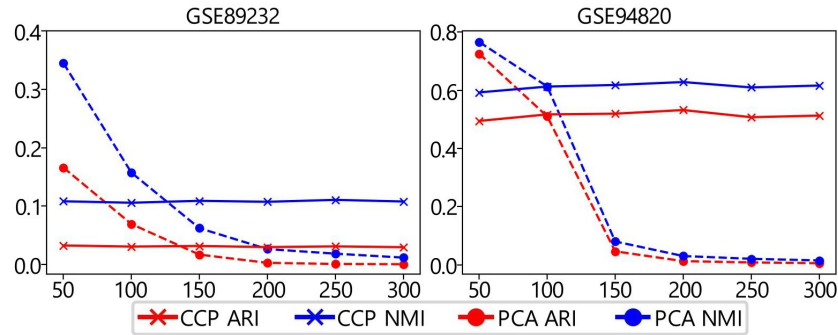
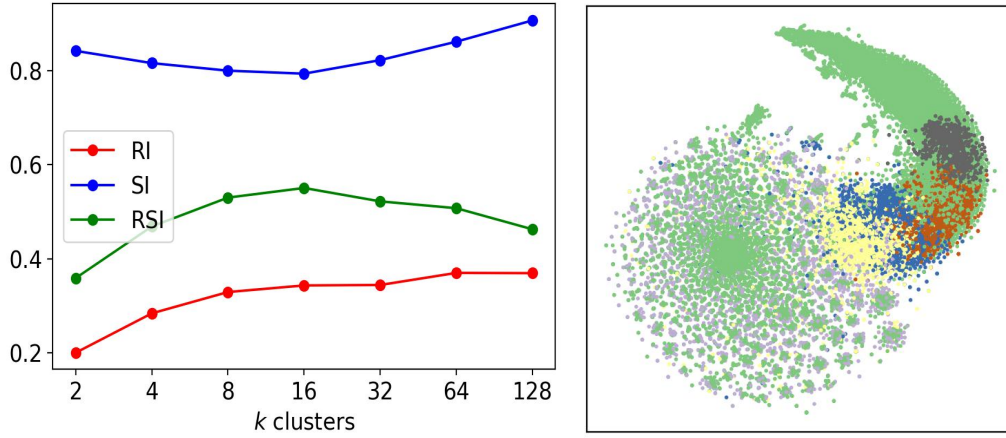


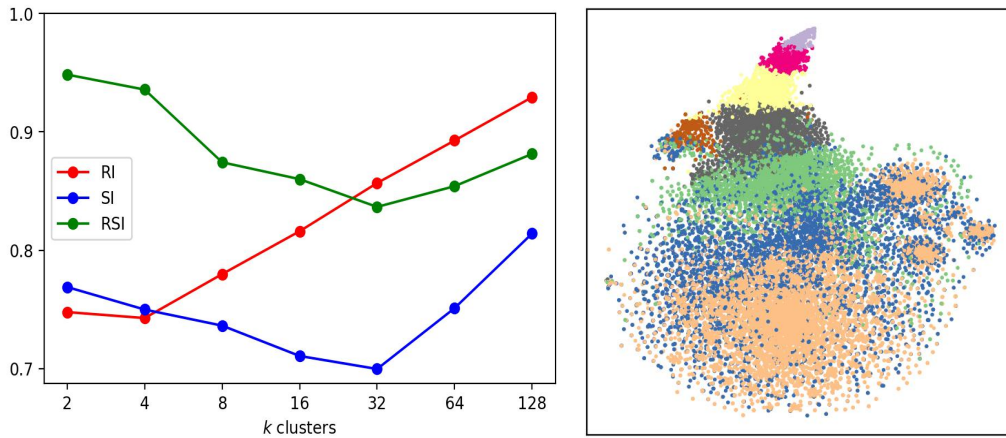
Figure 7A.3 ARI and NMI of the clustering results of CCP and PCA on GSE89232 and GSE94820. The red and blue lines correspond to CCP and PCA, respectively. For CCP, all the test utilize  $\tau = 6$  and  $\kappa = 2$  for the exponential kernels. The x-axis shows the number of components in the reduction.

In order to understand the poor performance of CCP on GSE89232, the RSI was computed on the gene clustering result. Figure 7A.4 shows the RSI plot and t-SNE visualization of GSE89232 and GSE94820 genes. RSI was computed by varying the number of clusters, where the  $x$ -axis shows the number of clusters, and the green, red, and blue lines correspond to the RSI, RI, and SI of the gene clustering results. The RSI stabilizes starting at 16 clusters and gradually decreases as the number of clusters increases. Notice that even though the RSI peaks at 16 clusters, the RI and SI continue to increase, indicating that the clustering may be improving. In addition, t-SNE was used to obtain a 2-dimensional (2D) visualization of the gene types. For GSE94820,  $k = 64$  was used for the  $k$ -means clustering. Seven gene clusters with the largest number of genes were colored, and the rest were colored in green. Notice that purple genes are spread out, indicating that the purple genes do not cluster well. For GSE89232, the RSI is high at a low number of clusters. Compared to the RI and SI of GSE94820, the RI and SI are relatively high at a low number of gene clusters. This may indicate that the clustering is better at a lower number of clusters. In addition, t-SNE was used to visualize the genes with  $k = 8$ . The blue and orange genes are mixed, but they are located at the same general location. The other genes are clustered nicely. This indicates that the optimal number of clusters may be smaller than 8, which suggests that the intrinsic dimensionality of GSE89232 is low. Such a low intrinsic dimensionality is unfavorable for CCP, which may explain its poor performance for this data.





(a) GSE94820



(b) GSE89232

Figure 7A.4 RI, SI, and RSI of the gene clustering of GSE94820 and GSE89232.  $k$ -means clustering was performed with 2, 4, 8, 16, 32, 64, and 128 gene clusters. For each number of clusters, 10 random initializations were utilized, and the average of RI, SI, and RSI was obtained. The red, blue, and green lines correspond to RI, SI, and RSI, respectively. t-SNE was used to visualize the genes in 2D. For GSE94820,  $k = 64$  was used for  $k$ -means clustering, and 7 gene clusters were colored according to their cluster label, while the rest of the genes were colored in green. For GSE89232,  $k = 8$  was used for  $k$ -means clustering, and the genes were colored according to their gene cluster labels.

### 7A.2.2 Additional Residue-Similarity Index comparison

Figure 7A.5 shows performance comparison between CCP and PCA in classification and clustering problems using GSE45719, GSE59114, and GSE89232 data. Top row shows the classification results, where 5-fold cross validation and SVM were used. The bottom row shows the clustering results, where  $k$ -means was used to obtain the labels. For RSI, cluster labels obtained from  $k$ -means and from the true cell types were used.

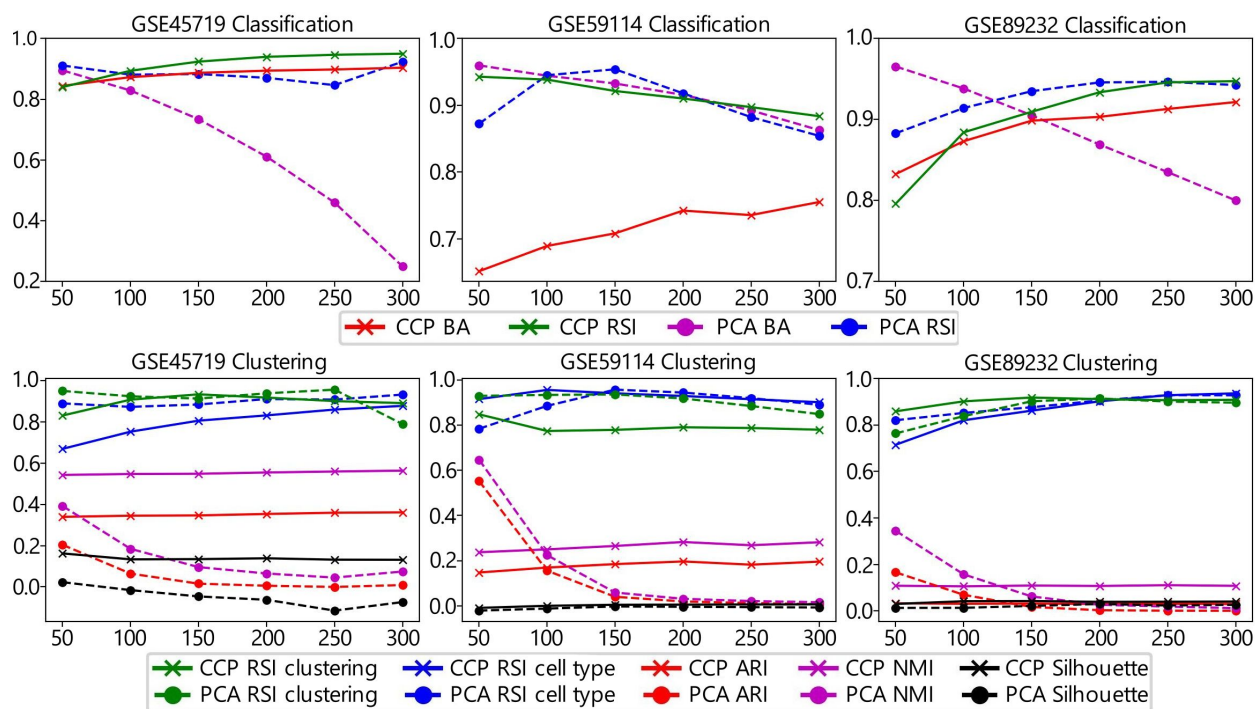


Figure 7A.5 Comparison of RSI on classification and clustering results of GSE45719, GSE59114, and GSE89232 data. The top row shows the classification results, where the solid and dashed lines correspond to the CCP and PCA results, respectively. The  $x$ -axis represents the number of components in the dimensionality reduction. The 5-fold cross-validation was used to evaluate the classification performance using SVM. The bottom row shows the clustering results of the GSE45719, GSE59114, and GSE89232 data. For RSI, the cluster labels obtained from  $k$ -means and the true cell labels provided by the authors were utilized. The solid and dashed lines correspond to the CCP and PCA results, respectively.

There is a noticeable correlation between RSI and BA in the classification results of GSE45719 and GSE89232. PCA's BA decreases in all three datasets. For GSE59114, CCP's result improves as the number of components increases, but is not as good as PCA's result. We can see from the ARI and NMI of GSE59114 that these two values are also low. In addition, from Figure 7A.4, we can see that the gene clustering is poor for GSE59114, which may have caused the poor performance. As for the clustering results, it is evident that the Silhouette score does not correlate with the other metrics used. Interestingly, the RSI using the true cell types and the cluster labels are similar for all the results, which means RSI is a good index for clustering when there are no true labels.

Figure 7A.6 shows performance comparison between CCP and PCA in classification and clustering problems using GSE84133h1, GSE84133h2, and GSE84133h3 data. Top row shows the classification results, where 5-fold cross validation and SVM were used. The bottom row shows the clustering results, where  $k$ -means was used to obtain the labels. For RSI, cluster labels obtained from  $k$ -means and from the true cell types were used.

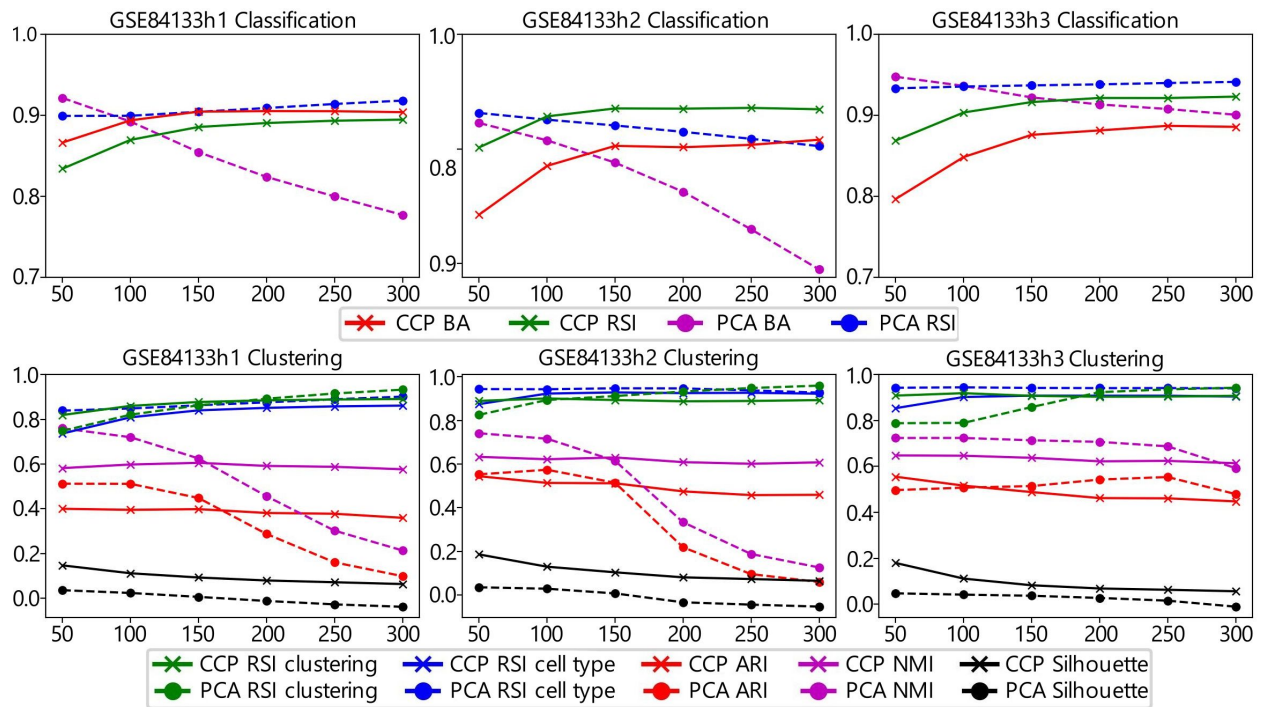


Figure 7A.6 Comparison of RSI on classification and clustering results of GSE84133h1, GSE84133h2, and GSE84133h3 data. The top row shows the classification results, where the solid and dashed lines correspond to CCP and PCA results, respectively. The x-axis represents the number of components in the dimensionality reduction. The classification results were obtained using SVM and verified using 5-fold cross validation. The bottom row shows the clustering results of the GSE45719, GSE59114, and GSE89232 datasets. For RSI, cluster labels obtained from *k*-means and the true cell labels provided by the data were utilized. The solid and dashed lines correspond to CCP and PCA results, respectively.

For classification result of GSE84133h1 and GSE84133h2, CCP outperforms PCA after 100 components, whereas in GSE84133h3, PCA outperforms PCA. However, for all 3 datasets, PCA's BA decreases as the number of components increases. In addition, RSI correlates with BA in all 3 datasets for CCP. For clustering results, Silhouette score shows no relation with ARI, NMI, or RSI. RSI using the true cell types and cluster labels correlates for all 3 datasets. Similar to the classification results, CCP's ARI and NMI of GSE84133h3 are lower than those of PCA's for most of the dimensions.

Figure 7A.7 shows performance comparison between CCP and PCA in classification and clustering problems using GSE84133h1, GSE84133h2, and GSE84133h3 data. Top row shows the classification results, where 5-fold cross validation and SVM were used. The bottom row shows the clustering results, where *k*-means was used to obtain the labels. For RSI, cluster labels obtained from *k*-means and from the true cell types were used.

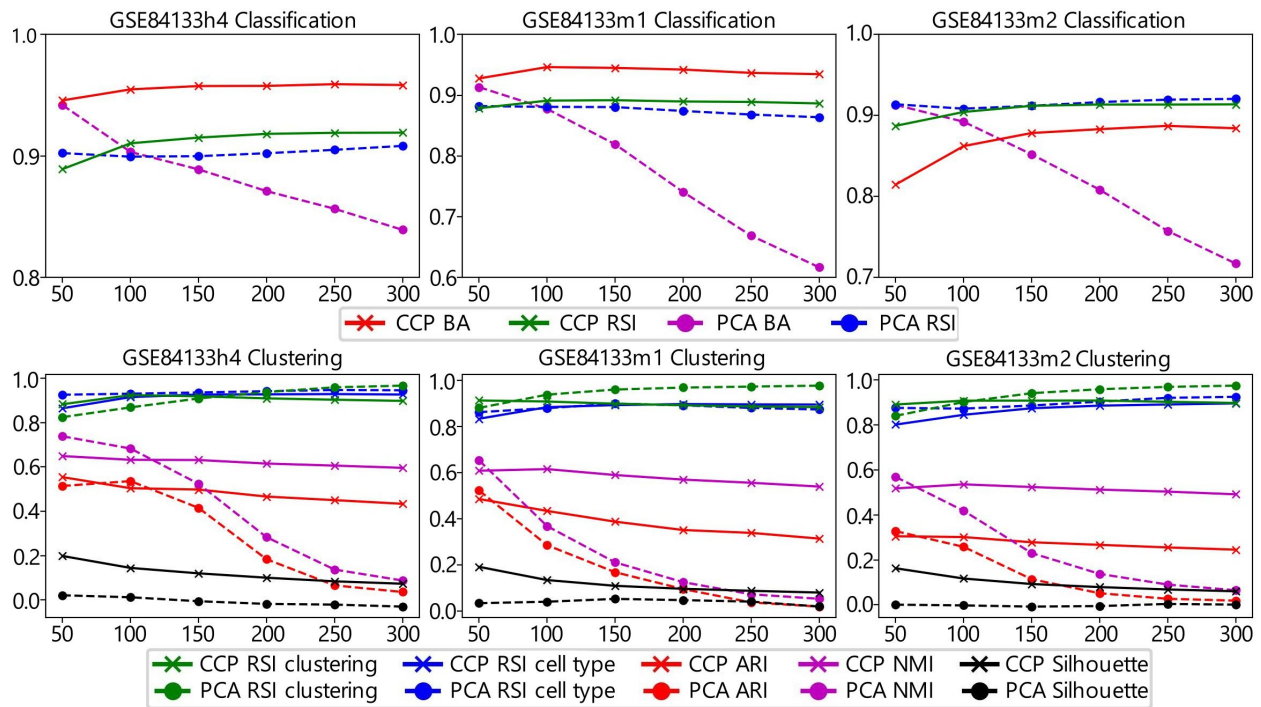


Figure 7A.7 Comparison of RSI on classification and clustering results of GSE84133h4, GSE84133m1, and GSE84133m2 data. Top row shows the classification results, where the solid and dashed lines correspond to CCP and PCA results, respectively. The  $x$ -axis is the number of components in the dimensionality reduction. The 5-fold cross validation was used to verify the classification result obtained from using SVM. The bottom row shows the clustering results of the GSE45719, GSE59114, and GSE89232 data. For RSI, the cluster labels obtained from  $k$ -means and from the true cell labels provided by the data were utilized. The solid and dashed lines correspond to CCP and PCA results, respectively.

For all three datasets, CCP outperforms PCA in classification accuracy when the number of components is larger than 150. CCP's RSI of classification results correlates with BA, whereas PCA's RSI shows less correlation with BA. In addition, for all three datasets, PCA's accuracy decreases as the number of components increases, showing instability in the reduction algorithm. The Silhouette score does not show any correlation with the other metrics. RSI using the true cell types and cluster labels correlates for all three datasets. Lastly, ARI and NMI follow a similar trend as BA, where CCP outperforms PCA at higher numbers of components.

Figure 7A.8 shows performance comparison between CCP and PCA in classification and clustering problems using GSE75748 cell and GSE94820 data. Top row shows classification results, where 5-fold cross validation and SVM were used. The bottom row shows the clustering results, where  $k$ -means was used to obtain the labels. For RSI, cluster labels obtained from  $k$ -means and from the true cell types were used.



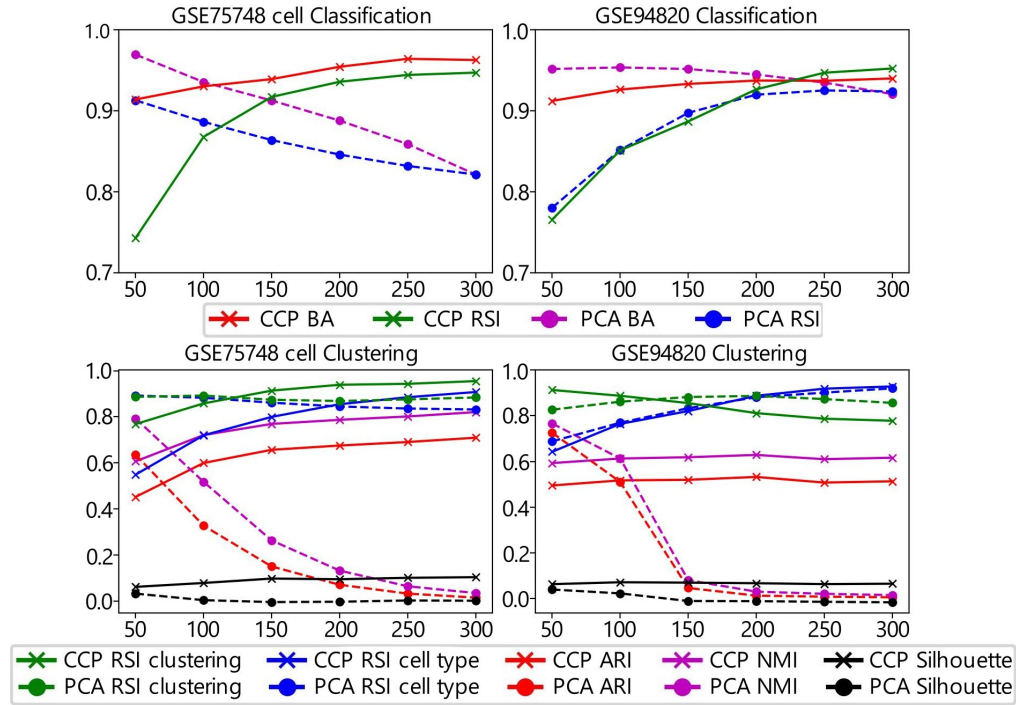


Figure 7A.8 Comparison of RSI on classification and clustering results of GSE75748 cell and GSE94820 data. Top row shows the classification results, where the solid and dashed lines correspond to CCP and PCA results, respectively. The  $x$ -axis is the number of components in the dimensionality reduction. The 5-fold cross validation was used to verify the classification results obtained from using SVM. The bottom row shows the clustering results of the GSE45719, GSE59114, and GSE89232 data. For RSI, the cluster labels obtained from  $k$ -means and from the true cell labels provided by the data were utilized. The solid and dashed lines correspond to CCP and PCA results, respectively.

For all three datasets, CCP outperforms PCA in classification accuracy when the number of components is larger than 150. RSI of CCP's classification results correlates with BA, whereas PCA's RSI does not show any correlation. In addition, for all three datasets, PCA's accuracy decreases as the number of components increases, showing instability in the reduction algorithm. The Silhouette score does not show any correlation with the other metrics. RSI using the true cell types and cluster labels correlates for all three datasets. Lastly, ARI and NMI follow a similar trend as BA, where CCP outperforms PCA at higher numbers of components.

## APPENDIX 7B

### ADDITIONAL MATERIALS FOR ANALYZING SCRNA-SEQ DATA BY CCP-ASSISTED UMAP AND T-SNE

#### 7B.1 Results

##### 7B.1.1 Data statistics

Table 7B.1 show basic statistical analysis of the dataset, namely the sparsity (number of zero expression), max, mean and median expression, and the median sum of the cell expression. For the mean and the median expression, we considered all nonzero expression values.

Table 7B.1 Accession ID, source organism, and the counts for samples, genes, and cell types for fourteen individual datasets.

Dataset	Size (cells x genes)	Sparsity	Max	Mean	Median	Median Row Sum
GSE75748cell [6]	1018 x 19097	49.64	605598.59	470.98	75.00	4404343.22
GSE75748time [6]	758 x 19189	54.69	165308.35	153.61	27.00	1306562.14
GSE82187 [7]	705 x 18840	77.72	5.60	1.70	1.73	7086.69
GSE67835 [5]	420 x 22084	81.40	58272.00	136.71	27.00	505256.00
GSE84133 H1 [8]	1937 x 20125	90.44	4318.00	3.02	1.00	5346.00
GSE84133 H2 [8]	1724 x 20125	90.59	3476.00	2.70	1.00	4889.50
GSE84133 H3 [8]	3605 x 20125	91.30	3071.00	3.35	1.00	4742.00
GSE84133 H4 [8]	1303 x 20125	89.05	4234.00	3.05	1.00	6017.00
GSE84133 M1 [8]	822 x 14878	90.48	3477.00	2.64	1.00	2936.50
GSE84133 M2 [8]	1064 x 14878	87.81	3656.00	3.28	1.00	5631.00
GSE84133 human [8]	8569 x 20125	90.62	4318.00	3.09	1.00	5075.00
Muraro [18]	2122 x 19046	73.02	4501.60	3.52	1.60	18076.53
Romanov [19]	2881 x 21143	85.92	2571.46	2.48	1.26	7373.00
Qx Bladder [17]	2500 x 23341	86.94	4640.55	3.64	1.27	11102.50
Qx Limb Muscle [17]	3909 x 23341	93.57	1124.88	2.74	1.17	4110.00
Qx Spleen [17]	9552 x 23341	94.34	1665.24	2.46	1.07	3244.50
Qs Diaphragm [17]	870 x 23341	91.35	481734.98	247.92	70.45	500405.50
Qs Limb Muscle [17]	1090 x 23341	89.47	682914.40	294.23	78.39	723268.50
Qs Lung [17]	1676 x 23341	89.08	300415.15	245.66	67.56	626066.50
Qs Trachea [17]	1350 x 23341	85.48	730410.52	220.08	61.81	745632.50

##### 7B.1.2 Comparison of CCP, PCA and NMF assisted visualization

In this section, we show the effectiveness of CCP as an initialization of UMAP and tSNE by comparing it to PCA and NMF, 2 of the most utilized algorithm for dimensionality reduction for scRNA-seq data. We perform the same preprocessing procedure as described in Section 7.2.3.1 of the main text.

Figure 7B.1 show the comparison of CCP-assisted, PCA-assisted and NMF-assisted visualization on GSE75748 cell, GSE75748 time, GSE67835 and GSE82187 data. The columns from left to right correspond to CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE.

NMF-assisted visualization do not provide a meaningful clustering result for all data. We can observe the following from the visualization:

- In GSE75748 cell, CCP-assisted and PCA-assisted visualization show similar result.
- IN GSE75748 time data, CCP-assisted and PCA-assisted visualization show similar result. Most notably, both technique show a supercluster of 72hr and 96hr, which is consistent with Chu's findings [6]. There is a cleared distinction between the 72hr-96hr supercluster with 36hr cluster for PCA-assisted visualization. However, CCP-assisted visualization show a clear distinction of 00hr cells from other cells, indicating the clear distinction of undifferentiated cells from cells that have begun or completed differentiating.
- In GSE67835 data, CCP-assisted UMAP and tSNE have the best visualization. PCA-assisted UMAP do no show clear clustering, and PCA-assisted tSNE visualization is dominated by an outlier astrocytes.
- Both CCP-assisted UMAP and tSNE show the best visualization for GSE82187 data. All the cell types show a distinct cluster, whereas PCA-assisted visualization do not.

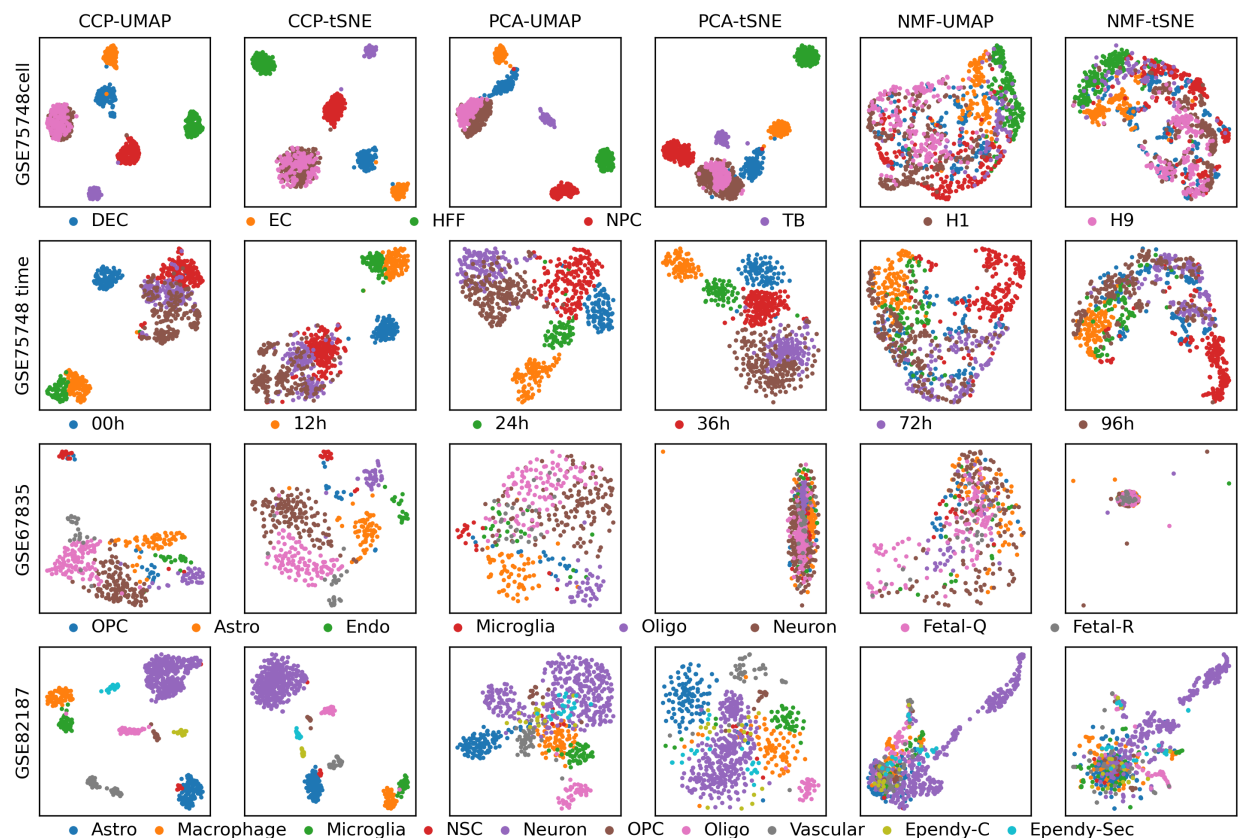


Figure 7B.1 Comparison of CCP-assisted visualization with PCA-assisted and NMF-assisted visualization on GSE75748 cell, GSE75748 time, GSE67835 and GSE82187 data. The columns from left to right show CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. The rows from top to bottom show GSE75748 cell, GSE75748 time, GSE67835 and GSE82187 data. CCP, PCA and NMF were utilized to reduce the dimension to 300, and UMAP and tSNE were used to further reduce the dimension to 2. Sample were colored according to the cell types provided by the original authors.

Figure 7B.2 show the comparison of CCP-assisted, PCA-assisted and NMF-assisted visualization on Quake data. The columns from left to right correspond to CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. Qx indicates scRNA-seq obtained used 10x genomic platform, and Qs indicate data obtained from SmartSeq2 platform.

NMF-assisted visualization do not provide a meaningful clustering result for all data. We can observe the following from the visualization:

- CCP-assisted and PCA-assisted show similar clustering result for both Qx Bladder and Qx Limb data. The most notable difference is that CCP-assisted UMAP show 3 subclusters of bladder urothelial cells.
- CCP-assisted tSNE show a considerable improvement to PCA-assisted tSNE. Macrophages do not form a clustering PCA-assisted tSNE, whereas in CCP-assisted tSNE, it forms a cluster.



Additionally Satellite cells are more dispersed in PCA-assisted tSNE than in CCP-assisted tSNE

- In Qs Limb Muscle, both CCP-assisted and PCA-assisted show a similar result.
- In Qs Trachea data, CCP-assisted show better clustering result from PCA-assisted counterparts. Noticeably, epithelial cells show a distinct cluster in CCP-assisted visualization, where as in PCA-assisted visualization, epithelial cells form 2 clusters, and have many cells mixed with mesenchymal cells.

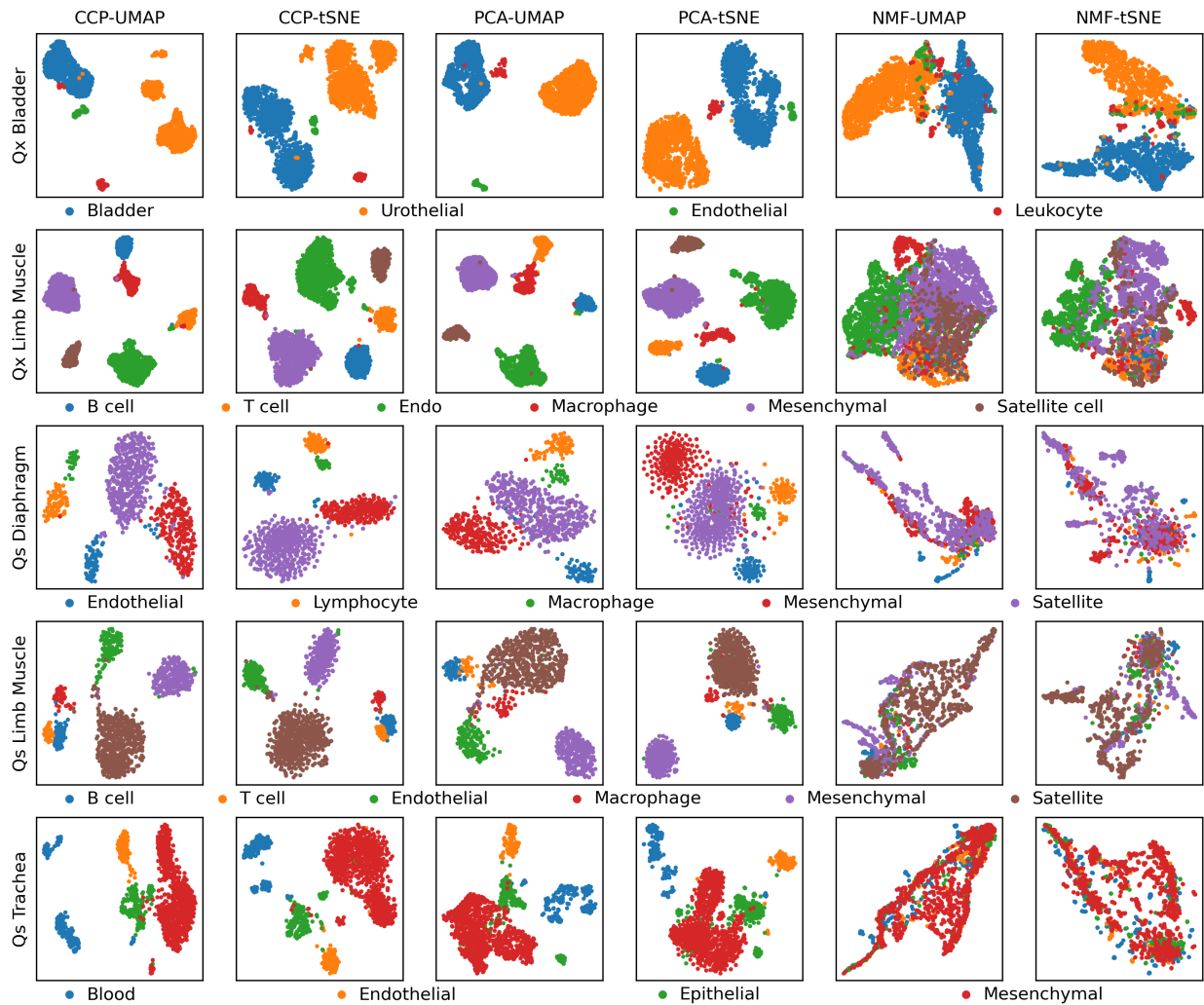


Figure 7B.2 Comparison of CCP-assisted visualization with PCA-assisted and NMF-assisted visualization on Quake data. The columns from left to right show CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. The rows correspond to 1 of the 5 Quake data. Qx indicates scRNA-seq obtained used 10x genomic platform, and Qs indicate data obtained from SmartSeq2 platform. CCP, PCA and NMF were utilized to reduce the dimension to 300, and UMAP and tSNE were used to further reduce the dimension to 2. Sample were colored according to the cell types provided by the original authors.

Figure 7B.3 show the comparison of CCP-assisted, PCA-assisted and NMF-assisted visualization on GSE84133 human data. The columns from left to right correspond to CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. NMF-assisted visualization do not provide a meaningful clustering result for all data. In general, CCP-assisted visualization show the clearest distinction between the cell types. PCA-assisted UMAP and tSNE do not show the distinction between all the cell types.

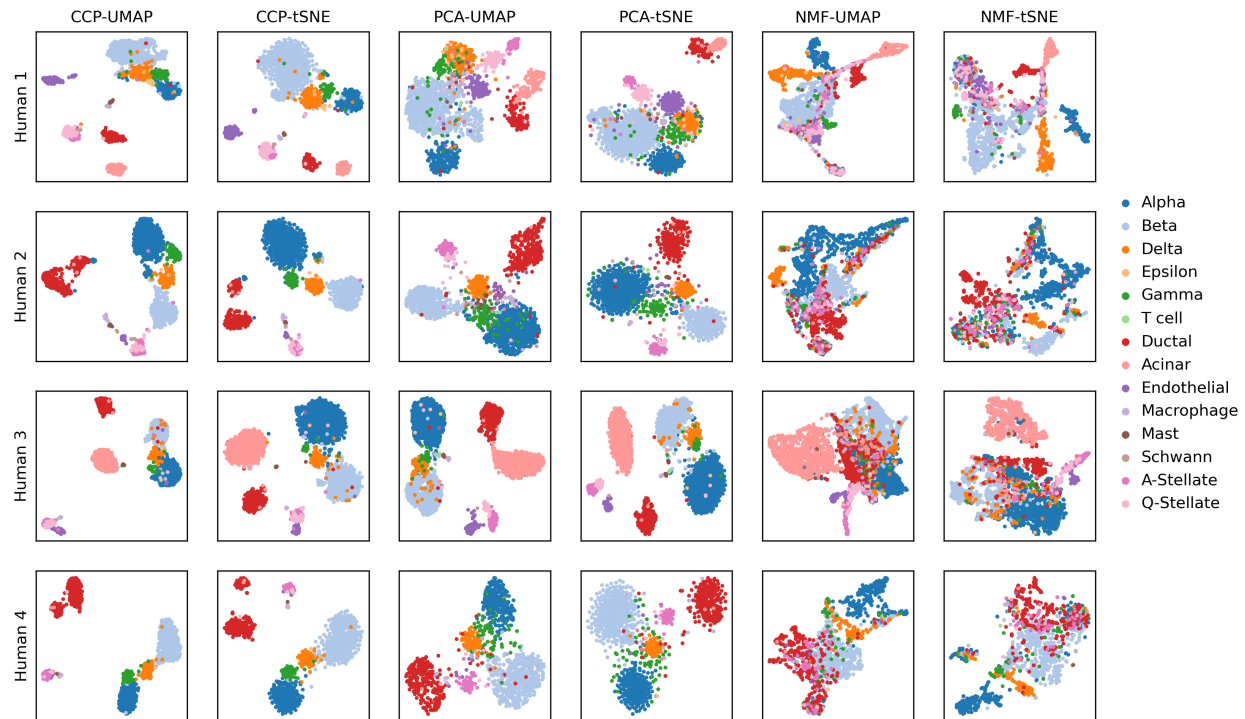


Figure 7B.3 Comparison of CCP-assisted visualization with PCA-assisted and NMF-assisted visualization on GSE84133 human data. The columns from left to right show CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. The rows correspond to one of the four patients. CCP, PCA and NMF were utilized to reduce the dimension to 300, and UMAP and tSNE were used to further reduce the dimension to 2. Sample were colored according to the cell types provided by the original authors.

Figure 7B.4 show the comparison of CCP-assisted, PCA-assisted and NMF-assisted visualization on GSE84133 mouse data. The columns from left to right correspond to CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE.

NMF-assisted visualization do not provide a meaningful clustering result for all data. In general, CCP-assisted visualization show the clearest distinction between the cell types. PCA-assisted UMAP and tSNE do not show the distinction between all the cell types.

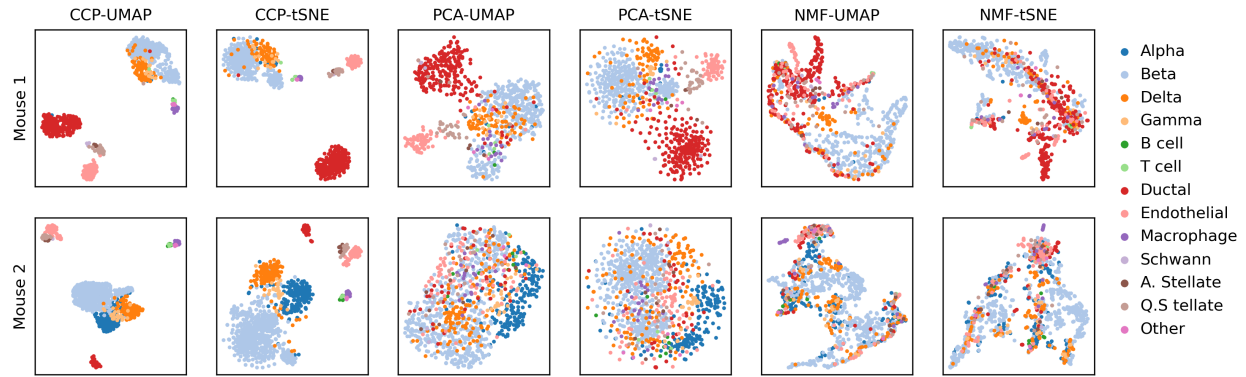


Figure 7B.4 Comparison of CCP-assisted visualization with PCA-assisted and NMF-assisted visualization on GSE84133 human data. The columns from left to right show CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. The rows correspond to mouse 1 and 2. CCP, PCA and NMF were utilized to reduce the dimension to 300, and UMAP and tSNE were used to further reduce the dimension to 2. Sample were colored according to the cell types provided by the original authors.

Figure 7B.5 show the comparison of CCP-assisted, PCA-assisted and NMF-assisted visualization on Muraro, Romanov and Qs Lung data. The columns from left to right correspond to CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE.

NMF-assisted visualization do not provide a meaningful clustering result for all data. CCP-assisted and PCA-assisted visualization show comparable in all data. In PCA-assisted visualization of Qs Lung, it shows a clear distinction between monocytes and ciliated columnar cells, whereas CCP-assisted visualization show a supercluster of these 2 cells. However, the stromal cells in CCP-assisted visualization show a distinct cluster, whereas PCA-assisted show a subcluster within the endothelial cell cluster.

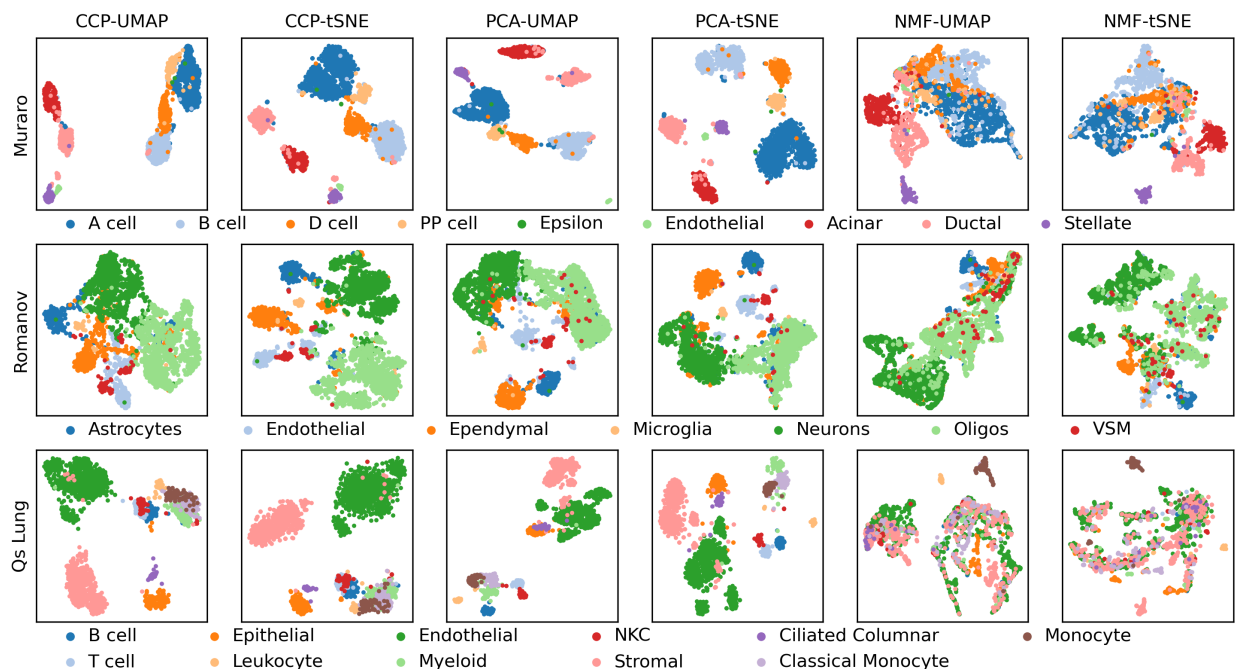
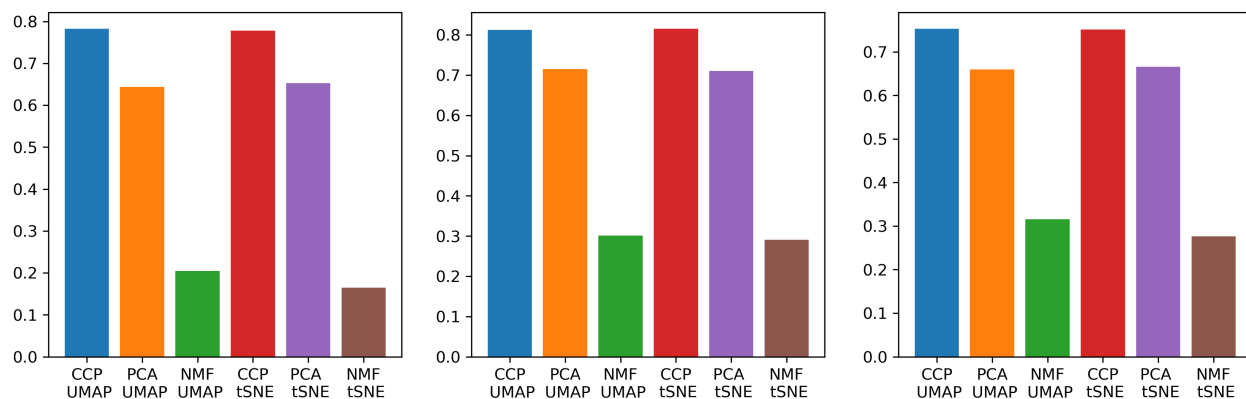


Figure 7B.5 Comparison of CCP-assisted visualization with PCA-assisted and NMF-assisted visualization on Muraro, Romanov and Qs Lung data. The columns from left to right show CCP-assisted UMAP, CCP-assisted tSNE, PCA-assisted UMAP, PCA-assisted tSNE, NMF-assisted UMAP and NMF-assisted tSNE. The rows from top to bottom show Muraro, Romanov and Qs Lung data. CCP, PCA and NMF were utilized to reduce the dimension to 300, and UMAP and tSNE were used to further reduce the dimension to 2. Sample were colored according to the cell types provided by the original authors.

### 7B.1.3 Accuracy

In order to validate CCP's performance, we computed the ARI, NMI and ECM for the 18 dataset. For each data, 10 random seed were utilized to generate CCP, PCA and NMF features, and the reduced features were further reduced to 2D using UMAP and tSNE. Leiden clustering was performed to obtain the clustering results, and ARI, NMI and ECM were computed by comparing the clustering results with the cell types provided by the original authors.

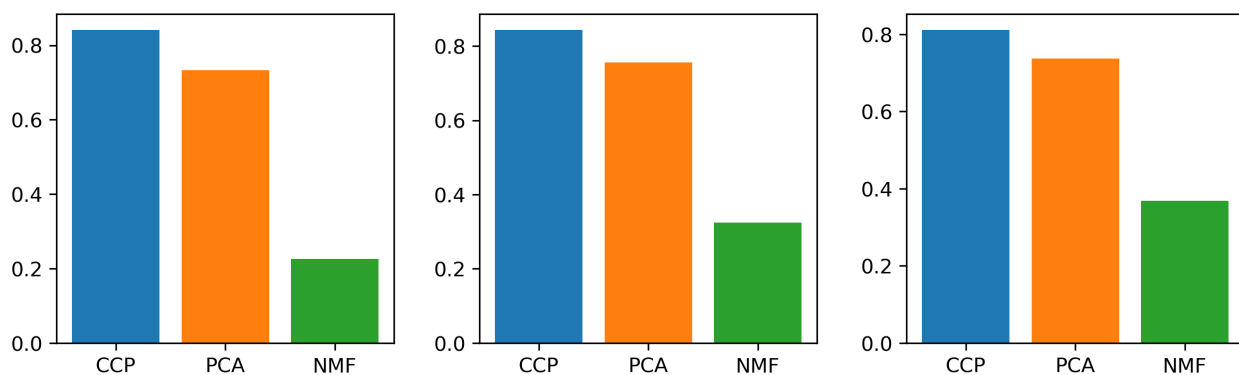
Figure 7B.6 show the average ARI, NMI and ECM of CCP-assisted, PCA-assisted and NMF-assisted UMAP and tSNE over 18 dataset. Notice that CCP outperforms both PCA and NMF. CCP-assisted UMAP improves PCA-assisted UMAP on average by 43% in ARI 22.5% in NMI and 19% in ECM.



(a) Average ARI over 18 dataset (b) Average NMI over 18 dataset (c) Average ECM over 18 dataset

Figure 7B.6 The average ARI, NMI, ECM of 18 datasets. 10 random initialization was used to compute the reduction for each data. Leiden clustering was used to obtain the clustering results.

Additionally, we validated the CCP super-genes with PCA-gene and NMF-gene. Figure 7B.7 show the average ARI, NMI and ECM of CCP, PCA and NMF over 18 dataset. Notice that CCP outperforms both PCA and NMF. Most notably, CCP improves PCA by 29% in ARI, 20% in NMI and 15% in ECM.



(a) Average ARI over 18 dataset (b) Average NMI over 18 dataset (c) Average ECM over 18 dataset

Figure 7B.7 The average ARI, NMI, ECM of 18 datasets. 10 random initialization was used to compute the reduction for each data. Leiden clustering was used to obtain the clustering results.

Figure 7B.7 show the average ARI, NMI and ECM of CCP, PCA and NMF over 18 dataset. Notice that CCP outperforms both PCA and NMF. Most notably, CCP improves PCA by 29% in ARI, 20% in NMI and 15% in ECM.

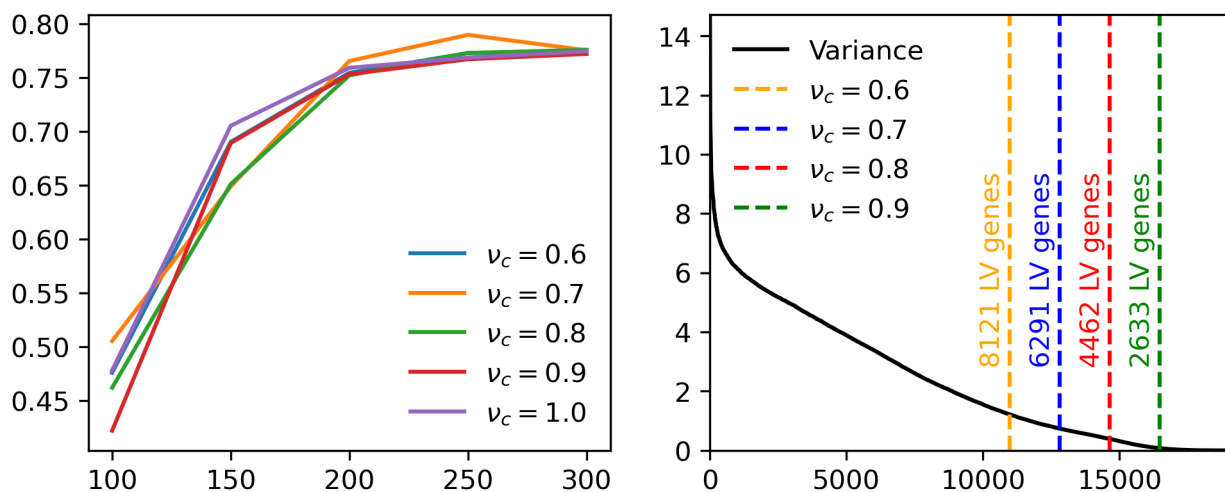
## 7B.2 Discussion

### 7B.2.1 Low variance gene

Figure 7B.8 show the effect of varying the number of super-genes and the cutoff ratio on the predictive power and visualization of GSE75748cell data. We utilized 10 random seeds to generate CCP super-genes using different number of super-genes and cutoff ratio. Then, Leiden clustering

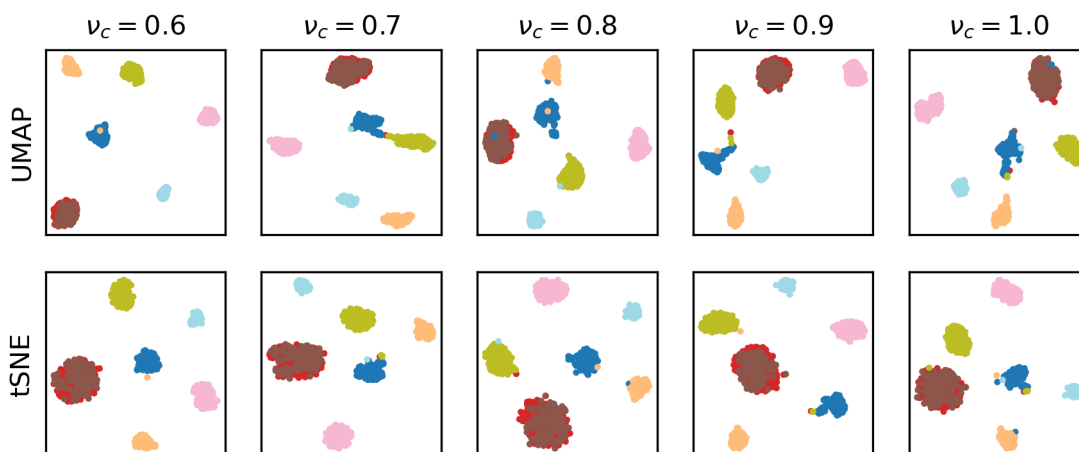
was used to obtain the cluster labels, and the ARI was computed by using the cell types provided by the original authors. Notice that at all cutoff ratio, the ARI increases as the number of super-genes increases, and the ARI is comparable at 300 super-genes. This suggest the robustness of LV-gene. Additionally, we computed the number of genes in the LV-gene cluster at varying cutoff value, and plotted with the variance of the genes in descending order. Notice that at  $v_c = 0.9$  the variance of the genes are relatively small, indicating that the predictive power of these genes may not be high.

Figure 7B.8(c) show the visualization of CCP-assisted UMAP and tSNE at various cutoff ratio. For the visualization, 300 super-genes were utilized, and UMAP and tSNE was applied to the super-genes to reduce the dimension to 2. Samples were then colored according to the cell types provided by the original authors. Note that all the visualization are comparable, indicating the robustness of LV-gene under different cutoff ratio.



(a) ARI of number of super-genes and  $\nu_c$

(b) Number of genes in LV genes



(c) CCP-assisted UMAP and tSNE visualization

Figure 7B.8 Analysis of varying the cutoff ratio  $\nu_c$  on clustering and visualization of GSE75748 cell data. (a) ARI of leiden clustering when the number of super-genes and cutoff ratio is changed. (b) The number of genes in the LV-gene when  $\nu_c$  is changed. (c) Top and bottom row shows the CCP-assisted UMAP and t-SNE visualization, and the columns corresponds to  $\nu_c = 0.6, 0.7, 0.8, 0.9$ . 300 super-genes were used to initialize UMAP and tSNE, and the samples were colored according to the true cell type.

## APPENDIX 7C

### ALGEBRAIC CONNECTIVITY OF FRI IN CCP

#### 7C.1 Motivation

In the original formulation of CCP, for components  $n$ , we have

$$\Phi(|\mathbf{x}_i^{S^n} - \mathbf{x}_j^{S^n}|; r_c^{S^n}, \eta^n, \tau, \kappa) = \begin{cases} \exp\left(-\left[\frac{|\mathbf{x}_i^{S^n} - \mathbf{x}_j^{S^n}|}{\eta^n \tau}\right]^k\right) & \text{if } |\mathbf{x}_i^{S^n} - \mathbf{x}_j^{S^n}| < r_c^{S^n} \\ 0 & \text{otherwise} \end{cases} \quad (7C.1)$$

where  $\eta^{S^n}$  is the algebraic connectivity term defined as

$$\eta^{S^n} = \frac{1}{M} \sum_{m=1}^M \min_{j \neq m} |\mathbf{x}_m^{S^n} - \mathbf{x}_j^{S^n}|. \quad (7C.2)$$

In other words, algebraic connectivity term is defined as the average minimal distance.

However, there are alternative values that  $\eta^{S^k}$  can take, both supervised and unsupervised approach. In this section, I will discuss alternative values, namely the average mean distance, cluster-wide minimal distance and cluster-wide average distance.

#### 7C.2 Formulation of algebraic connectivity

##### 7C.2.1 Average mean distance

The average mean distance can be defined as

$$\eta^n = \frac{1}{M} \sum_{m=1}^M \frac{\sum_{j \neq m} |\mathbf{x}_m^{S^n} - \mathbf{x}_j^{S^n}|}{M} \quad (7C.3)$$

##### 7C.2.2 cluster-wide minimal distance

Let the data be represented as  $\{(\mathbf{x}_m, y_m)\}_{m=1}^M$ , where  $y_m \in \mathbb{Z}_L$  is the class or cluster label for sample  $m$  and  $L$  is the number of class or cluster. Let the data be partitioned in to  $L$  class or cluster  $C_l = \{\mathbf{x}_m | y_m = l\}$ . For the cluster-wide minimal distance, samples in each  $C_l$  obtain its own algebraic connectivity term  $\eta_l^{S^k}$ . For class  $l$ , we have

$$\eta_l^n = \frac{1}{|C_l|} \sum_{\mathbf{x}_m \in C_l} \min_{\mathbf{x}_j \in C_l} |\mathbf{x}_m^{S^n} - \mathbf{x}_j^{S^n}|. \quad (7C.4)$$



### 7C.2.3 cluster-wide average distance

Define  $C_l$  as in the cluster-wide minimal distance. For the cluster-wide average distance, samples in each  $C_l$  obtain its own algebraic connectivity term  $\eta_l^{S_k}$ . For class  $l$ , we have

$$\eta_l^n = \frac{1}{|C_l|} \sum_{\mathbf{x}_m \in C_l} \frac{\sum_{\mathbf{x}_j \in C_l} |\mathbf{x}_m^{S^n} - \mathbf{x}_j^{S^n}|}{|C_l|}. \quad (7C.5)$$

### 7C.3 Results

We utilized the same data as Table 7.1, and performed the same classification procedure as subsection 7A.1.2. Figure 7C.1 show the average balanced accuracy (14) of the 4 methods: the standard (average minimal distance), average mean distance, cluster-wide minimal distance and cluster-wide mean distance.  $\tau = 1$  and  $\kappa = 2$  with exponential kernel was used for all the test.

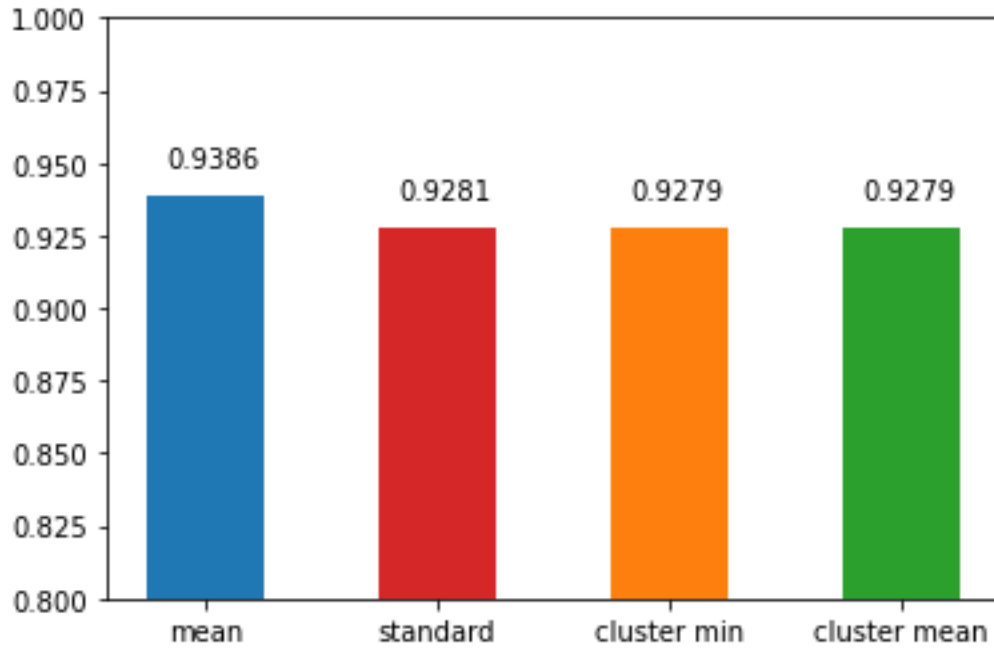


Figure 7C.1 Average balanced accuracy (BA) of different algebraic connectivity. From left to right: standard (average minimal distance), average mean distance, cluster-wide minimal distance and cluster-wide mean distance. The number above each bar indicate the average BA.  $\tau = 1$  and  $\kappa = 2$  with exponential kernel was used for all the test.

We can see that the average mean distance outperforms the other 3 method for scRNA-seq data. However, note that the other choice of algebraic connectivity also perform well. This indicates that CCP is quite stable for classification tasks, and the algebraic connectivity can be data-dependent.

## CHAPTER 8

### DISSERTATION CONTRIBUTIONS AND FUTURE DIRECTIONS

The main contribution in this dissertation is listed as follows:

- In Chapter 3, we introduce large scale clustering method called UMAP-assisted  $k$ -means clustering to clustering SARS-CoV-2 mutation, and analyze world-wide mutational trends. We also developed a novel alignment-free DNA sequence analysis method called  $k$ -mers topology. We show that  $k$ -mers topology outperforms other viral classification algorithm, and show that  $k$ -mers topology can also be used for phylogenetic analysis.
- In chapter 4, We proposed residue-similarity analysis and plot for data with dimensions greater than 3. The definition and the formulation is presented in this chapter. We have compared RS plot with geometric shape of data and topological analysis visualization to validate and compare the results
- In chapter 5, we proposed correlated clustering and projection (CCP) as a dimensionality reduction method for data with high intrinsic dimensions. We benchmark CCP against PCA on standard benchmark datasets
- In Chapter 6, we introduce topological nonnegative matrix factorization (TNMF), which incorporate multiscale topological and geometrical information through persistent Laplacian. We show how PLs are constructed, and show an alternative method called  $k$ -NN induced PL. Then, we prove the updating scheme for TNMF.
- In Chapter 7, we apply CCP and TNMF to scRNA-seq data. We show that both CCP and TNMF improves other dimensionality reduction methods for clustering, classification and visualization.

In the future, I would like to extend my work and explore the following:

- In the  $k$ -mers topology method, we utilized persistent homology to convert varying sequence length to a fixed feature. We would like to utilize tools from natural language processing

models to further enhance our method's performance. Additionally, I am interested in exploring the biological implication of the distribution of  $k$ -mers.

- In the  $k$ -mers topology method, we only utilized persistent homology in the  $k$ -mers method. We would like to extend the work to persistent Laplacians, and other topological methods. Additionally,  $k$ -mers can also be interpreted as nodes of hypergraphs, and I would like to explore the use of hypergraphs in DNA sequencing methods.
- I would like to extend the  $k$ -mers topology to protein sequence classification.
- One down side of topological NMF is that it requires choosing the weights for each filtration. Although  $k$ -NN induced persistent Laplacian can reduce the number of parameters, the number of parameters is still  $2^T$ . In the future I would like to consider a parameter free approach by using consensus methods. Additionally, I would like to extend the work to semi-supervised model.
- I would like to explore the use of residue-similarity score as a minimization function for dimensionality reduction.

The content of the dissertations are mostly adopted from the following publications and preprints.

- Hozumi, Y., & Wei, G.-W (2024).  $K$ -mer Topology for Whole Genome Analysis.
- Hozumi, Y., & Wei, G.-W. (2024). Analyzing single cell RNA sequencing with topological nonnegative matrix factorization. *Journal of Computational and Applied Mathematics*, 115842.
- Hozumi, Y., Tanemura, K. A., & Wei, G.-W. (2023). Preprocessing of single cell RNA sequencing data using correlated clustering and projection. *Journal of Chemical Information and Modeling*.

- Hozumi, Y., Wang, R., & Wei, G.-W. (2022). Ccp: Correlated clustering and projection for dimensionality reduction. ArXiv Preprint ArXiv:2206.04189.
- Hozumi, Y., Wang, R., Yin, C., & Wei, G.-W. (2021). UMAP-assisted K-means clustering of large-scale SARS-CoV-2 mutation datasets. *Computers in Biology and Medicine*, 131, 104264.
- Hozumi, Y., & Wei, G.-W. (2023). Analyzing scRNA-seq data by CCP-assisted UMAP and t-SNE. ArXiv Preprint ArXiv:2306.13750.

These work led to the following publication and preprints, but are not discussed in this dissertation.

- Chen, J., Wang, R., Hozumi, Y., Liu, G., Qiu, Y., Wei, X., & Wei, G.-W. (2022). Emerging dominant SARS-CoV-2 variants. *Journal of Chemical Information and Modeling*, 63(1), 335–342.
- Cottrell, S., Hozumi, Y., & Wei, G.-W. (2023). K-Nearest-Neighbors Induced Topological PCA for Single Cell RNA-Sequence Data Analysis. ArXiv.
- Feng, H., Cottrell, S., Hozumi, Y., & Wei, G.-W. (2024). Multiscale differential geometry learning of networks with applications to single-cell RNA sequencing data. *Computers in Biology and Medicine*, 171, 108211.
- Wang, R., Chen, J., Gao, K., Hozumi, Y., Yin, C., & Wei, G.-W. (2020). Characterizing SARS-CoV-2 mutations in the United States. Research Square.
- Wang, R., Chen, J., Gao, K., Hozumi, Y., Yin, C., & Wei, G.-W. (2021). Analysis of SARS-CoV-2 mutations in the United States suggests presence of four substrains and novel variants. *Communications Biology*, 4(1), 228.

- Wang, R., Chen, J., Hozumi, Y., Yin, C., & Wei, G.-W. (2020). Decoding asymptomatic COVID-19 infection and transmission. *The Journal of Physical Chemistry Letters*, 11(23), 10007–10015.
- Wang, R., Chen, J., Hozumi, Y., Yin, C., & Wei, G.-W. (2022). Emerging vaccine-breakthrough SARS-CoV-2 variants. *ACS Infectious Diseases*, 8(3), 546–556.
- Wang, R., Hozumi, Y., Yin, C., & Wei, G.-W. (2020a). Decoding SARS-CoV-2 transmission and evolution and ramifications for COVID-19 diagnosis, vaccine, and medicine. *Journal of Chemical Information and Modeling*, 60(12), 5853–5865.
- Wang, R., Hozumi, Y., Yin, C., & Wei, G.-W. (2020b). Mutations on COVID-19 diagnostic targets. *Genomics*, 112(6), 5204–5213.
- Wang, R., Hozumi, Y., Zheng, Y.-H., Yin, C., & Wei, G.-W. (2020). Host immune response driving SARS-CoV-2 evolution. *Viruses*, 12(10), 1095.