ACADEMIC REVIEW OPEN ACCESS

The Value of Manifold Learning Algorithms in Simplifying Complex Datasets for More Efficacious Analysis

Muhammad Amjad[1]

1. McMaster University, Honours Integrated Science, Class of 2023

Received 1 October 2020
Accepted 3 November 2020
Published 4 December 2020

SUMMARY

Suppose you wished to determine the probability of a basketball player scoring a three-point shot. Analyzing previous scoring data would guide you in arriving at a logical prediction. Now, imagine that you aimed to predict the weather. Analyzing previous weather data may not be nearly as useful given that there are an unfathomable number of logged entries that each encompass a multitude of characteristics such as temperature, wind speed, season, etc. An alternative to inefficiently relating data points based on a few overlapping characteristics would be to simply reduce the complexity of the entire set. Rather than predicting the weather based on dozens or hundreds of factors, you would only utilize the few that are the most impactful. This is manifold learning – reducing the complexity of data sets for a more efficient examination. Many algorithms are currently being developed and improved to limit characteristics of data entries, while retaining central information. These simplified entries are graphed on a smooth curve and connected. Points that are crowded will be strongly correlated; hence, illuminating a hidden pattern. This paper will explore the importance of manifold learning in data analysis and its applications.

ABSTRACT

Advances in manifold learning have proven to be of great benefit in reducing the dimensionality of large complex datasets. Elements in an intricate dataset will typically belong in high-dimensional space as the number of individual features or independent variables will be extensive. However, these elements can be integrated into a low-dimensional manifold with well-defined parameters. By constructing a low-dimensional manifold and embedding it into high-dimensional feature space, the dataset can be simplified for easier interpretation. In spite of this elemental dimensionality reduction, the dataset's constituents do not lose any information, but rather filter it with the hopes of elucidating the appropriate knowledge. This paper will explore the importance of this method of data analysis, its applications, and its extensions into topological data analysis.

Keywords: Manifold learning, principal component analysis, Isomap

INTRODUCTION

Significant advances in the field of machine learning can be attributed to the readily available high-quality datasets that are vital for training algorithms.[1] These datasets play an integral role in the development of machine learning algorithms that attempt to predict outcomes. This is done by providing computers with large and complex datasets, from which they can deduce patterns. This knowledge of patterns can be used to predict future outcomes. An instance of predictive modelling is in annual weather prediction. Climatologists and meteorologists train machine learning algorithms with previous weather data to recognize patterns[2]. This knowledge is then used to forecast upcoming weather. This predictive process is both a benefit and a burden. Although large sets of data provide algorithms with a breadth of information, they also hinder their efficiency with irrelevant or misleading details. For example, crowding datasets with unnecessary parameters can lead to issues with clarity. This can make subgrouping, and thus making informed conclusions, difficult.[3] On account of this, mathematicians have developed various techniques in alleviating inessential dimensions in an effort to make machine learning algorithms more efficient. This process is known as dimensionality reduction and is crucial to understand before delving into mathematical modeling of complex data.[4]

Dimensionality reduction aims to simplify datasets with the hopes of making them more efficient for machine learning algorithms. Dimensionality reduction can assist with scientific understanding by visualizing complex data. For example, if a dataset consists of 50 dimensions, then there are 50 attributes categorizing each individual point within that collection. This makes the process of finding meaningful relationships very challenging as a great deal of irrelevant information must be filtered first. This irrelevant information is sometimes referred to as noise, large

amounts of additional or meaningless information.[5] If a dimensionality reduction algorithm could be used to convert 50 dimensions into 3, then it would expedite the process of visualizing connections between data points. Apart from visualization, dimensionality reduction may also assist in elucidating underlying forces behind complex datasets. For example, dimensionality reduction can be used to improve the geophysical log data that classifies crystalline rocks.[6] Log functions are used to classify rock types after the machines measure their characteristics, such as surface area, density, porosity, etc. However, many of these log functions may not be of use as several different types of rocks have overlapping characteristics. Shale and claystone, for example, are very similar in grain size.[7] Using log functions that include similar characteristics, such as grain size, when differentiating between shale and claystone would be redundant. With the use of dimensionality reduction algorithms, reduced-log sets that are the most important to the classification process can be used instead. Furthermore, providing feedback to the original algorithm can extend the capabilities of this technique. Machine learning will allow the algorithm to optimize itself by learning from the instances in which it was correct or mistaken.[8] This will allow it to make more statistically advantageous decisions and prioritize what reduced-log functions should be provided to the geological machinery. This instance of dimensionality reduction in machine learning algorithms exemplifies the importance of simplified datasets.

The simplification of large datasets by dimensionality reduction serves to remove repetitive and overlapping information. Take, for example, a large dataset that is governed by 100 dimensions. Given the number of parameters, there is bound to be overlap in the data points. It is possible that out of 10,000 elements, 1,000 will mirror the same behaviour along 15 dimensions. If the goal is to find relationships between data points, then continuous overlapping between parameters will not show any real connection. This is an inherent inefficiency in the machine learning system and can be undone by extracting the integral parameters and excluding derivatives of the same dimensions.[9] It is important to note that reducing the dimensions does not inherently limit the data itself. Suppose there is a large data set of one million participants and one hundred dimensions that indicate character traits. If the dimensions are reduced to three traits, such as integrity, honesty, and responsibility, 97 dimensions still remain for each data point. However, the additional 97 dimensions are virtually emphasized because the system only visualizes relationships between participants for the three traits. Furthermore, there could possibly be infinite dimensions to the dataset, but only one hundred were selected initially, and three left after the reduction. Therefore, the potential classification of the dataset is not bound by the three dimensions set by the algorithm. This high-dimensional space that the data could belong to is known as the feature space.[10] The low-dimensional space that the algorithm reduces the dataset to reflects the applied parameters. The manifold structure that will soon be created will import aspects of both high and low-dimensional space, and will find meaningful relationships from large sets of data.

NOTATION

- The high-dimensional points will be sectioned in matrix X, where the i^{th} row is x_i $(x_1, x_2, ..., x_n)$. Here, n is the number of points in the dataset and can vary widely. Similarly, the low-dimensional parameters will be $y_1, y_2, ..., y_n$ within the matrix Y.[11]
- D is denoted as the dimensionality of the dataset before the reduction algorithm is applied, located in R^D space. Resultantly, d is the dimensionality of the deriving manifold.[12]
- (v_i, λ_i) are eigenvector-eigenvalue pairs for $i = 1, ..., n$. From this $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$. If $v_1, ..., v_d$ are the d eigenvectors in high-dimensional space, then $v_{n-d+1}, ..., v_n$ represent the low-dimensional d eigenvectors. The eigenvectors will change due to the reduced dimensionality of data. The notion of eigenvectors will be elaborated on in the preliminaries section.[13]
- k is the number of nearest neighbors while $N(i)$ is the set of k-nearest neighbors of x_i. This algorithmic concept will be discussed in the preliminaries section.[14]

PRELIMINARY MATHEMATICAL KNOWLEDGE

Eigenvectors are a set of vectors associated with a linear system of equations, such as the ones found in the matrix X and Y established previously.[15] Each of these eigenvectors will have a corresponding eigenvalue. If (v, λ) is an eigenvector-eigenvalue pair of space R, then for any i, (v_i, λ) is an eigenvector-eigenvalue pair of R^i. Similarly, for any k, (kv, λ) is an eigenvector-eigenvalue pair of kR.

The K-nearest neighbors (KNN) algorithm is widely used in pattern recognition and machine learning.[16] It is an example of a supervised machine learning algorithm, which relies on labeled input data to learn and then produce an appropriate output when given new unlabeled data. These algorithms can be used to solve classification or regression problems. KNN assumes that data points with similar characteristics will reside close to one another on a visual plane, such as a manifold structure. Initializing a *k-value* will find the data points closest to a selected starting point. For example, if k is initialized to $k = 3$ and the first data point is defined as $x = 1$, then the algorithm will find the three most similar data points to the *x-value*. Ultimately, the algorithm will find the desired points by taking the

shortest distance between the initialized value and the other values within the dataset. For this example, the three shortest distances will exemplify the three nearest neighbors and those will be outputted. This algorithm proves versatile for classifying the similarities and differences within a dataset.

LINEAR DIMENSIONALITY REDUCTION

Principal Component Analysis (PCA) is a technique for reducing the dimensionality of large datasets while maximizing interpretability and minimizing information loss.[17] Suppose there is a dataset with points that share similar characteristics. These points belong in the same dimensions, but are still unique in their own respects. If the number of characteristics (dimensions) were reduced, then the uniqueness of each point could be more easily observed and quantified. From this, comparisons can be made between each data point along specific parameters to better understand their similarities and differences. For instance, suppose there are three biological cells that each share a common gene;[18] however, the percentage of their genome that consists of that gene is different. If that percentage is the underlying parameter, then each cell can be compared to the other to determine relative similarity (i.e. cell 1-to-cell 2, cell 2-to-cell 3, and so on). This will elucidate which cells are most similar with respect to that specific dimension, yielding several correlational observations amongst samples in the dataset. However, if the number of cells increases linearly, then the number of possible combinations will increase exponentially. This will become increasingly repetitive and inefficient in the long run with large datasets. Instead, a PCA plot is constructed, which will convert the correlations, or lack thereof, among all the cells into a 2-D graph. Cells that are highly correlated will cluster together given that they share very similar characteristics. Perhaps if some cells are very similar in specific characteristics, they will even overlap. These overlapping clusters are exceptionally valuable as they demonstrate great similarity with respect to the dimensionality of the PCA plot. Limited dimensionality overlap provides a comprehensive outlook on how some points react to only a few parameters, which can then be explored further. Once the clusters are identified, the cells contained within can be examined individually to study why correlations were detected by the algorithm. This method is not exclusive to cells and can be applied to any large dataset that needs to be refined by specific dimensions. It is also important to note that the relative distances between clusters is significant. If a specific dimension is prioritized, then clusters spread out along the axes of that dimension will not be as closely correlated because their values will differ greatly. PCA algorithms can be checked for accuracy by initializing them with training data, then providing them with new test data to see if they sort in a similar fashion.[19] It is critical that variance is minimized between similar data points, but maximized for dissimilar ones.[20] Essentially, it is ideal to have tightly packed clusters that are spaced out along the linear subspace. This will be beneficial when grouping data that is similar and separating sets that are not. PCA will render a dataset devoid of any irrelevant features, as no extraneous dimensions will be included by the algorithm. A sample PCA plot that compares three species of *Iris* plants is illustrated in **Figure 1**.

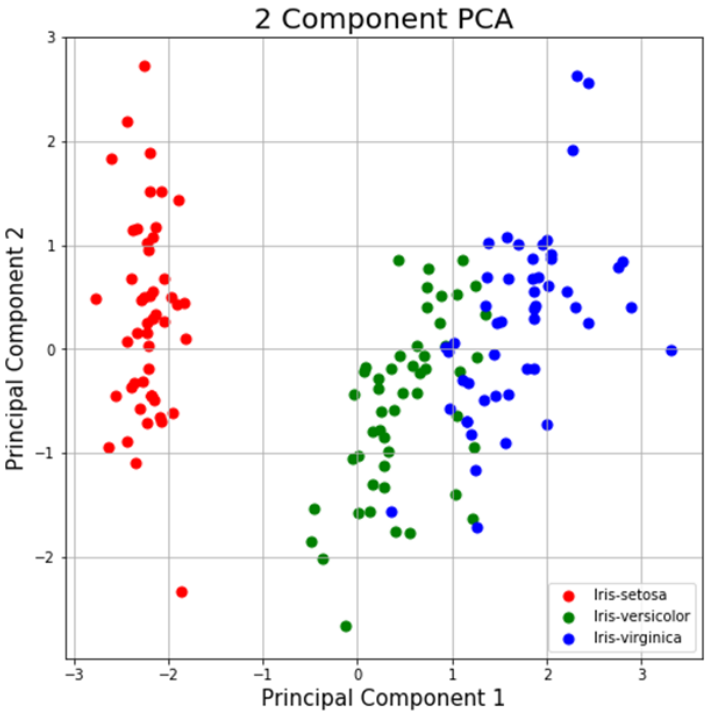

Figure 1. Python was used to perform PCA for three different species of *Iris* plants. Axes of similarities were growth habits and bloom times (arbitrary units). *Iris versicolor* and *Iris virginica* appear to be similar with respect to parameters as they clustered, while *Iris setosa* is different from the other two (see Appendix A).

In order to model the manifold structure, it is important to first develop a maximum variance equation that will segregate unrelated elements in a dataset. Suppose that the matrix $X \varepsilon R^{nxD}$ has rows, which are D-dimensional data points. Along this space of R^D, there must be a linear subspace where the provided dataset has a maximum variance.[21,22]

$$var(XV) \qquad \textbf{Eq. 1}$$

where V is an orthogonal $D \times d$ matrix. If $d = 1$, then there is simply a unit-length vector as the orthogonal matrix will be composed of only one column. This is ideal for constructing the objective equation, as direc-

tion of the maximum variation will be given by the unit vector which contains the projected data points.

$$var(Xv) = E(Xv)^2 - (EXv)^2 \quad \textbf{Eq. 2}$$

To simplify this equation, the dataset can be assumed to be mean-centered, which will eliminate the sum of the vectors within the one-column matrix. This can be seen as a possible limitation of the method, as the process of maximizing variance becomes increasingly more complex as more elements are added to the vector.

$$var(Xv) = E(Xv)^2 \quad \textbf{Eq. 3}$$

Note that $E(Xv)$ is the sum of all $x_i v$. Given that V is an orthogonal matrix, the transpose must be taken for X , making the equation $x_i^T v$ [14].

$$var(Xv) = \sum_i (x_i^T v)^2 \quad \textbf{Eq. 4}$$

In order to conduct matrix multiplication for Equation 4's squared terms, the transpose of the terms must be used.[14] This transpose method is the result of multiplying two vectors.

$$var(Xv) = \sum_i (v^T x_i)(x_i^T v) \quad \textbf{Eq. 5}$$

$$var(Xv) = v^T \left(\sum_i (x_i x_i^T) \right) v \quad \textbf{Eq. 6}$$

Equation 6 must be further maximized by setting v to the largest corresponding eigenvector, as seen in Equation 7. This will yield a manifold curve that is embedded in high-dimensional space, while being low-dimensional itself. This removes the sum component and leaves the X matrix, the orthogonal matrix, and their v respective transposes.

$$var(Xv) = v^T X^T X v \quad \textbf{Eq. 7}$$

MANIFOLDS

This section will focus on the characteristics of a manifold, as well as the ways in which the structure relates to general topology and geometry. As mentioned previously, the goal of this exploration is to obtain a manifold whose parameters lie in low-dimensions while the actual data points are high-dimensional. It is important to remember that the feature space is R^D and the reduced space is R^d. Hence, D will be larger than d as its dimensionality has not been reduced. This distinction and notation are important to remember before the concepts of homeomorphism can be discussed.

Homeomorphism is a method of deformation that is unique to the study of topological structures. A homeomorphism is a continuous function between topological spaces that also has a continuous inverse function.[23] In the field of topology, functions can be characterized as objects. If two objects are homeomorphic, they can be deformed into each other by continuous and invertible mapping.[23] To better visualize this concept, imagine a donut made of clay. This donut can be morphed into the shape of a coffee cup by moulding the surfaces, and without cutting or adding any new clay. These two objects are topologically equivalent. A manifold is a topological space that resembles that of Euclidean space, but only locally.[22] For the purposes of this exploration, visualize Euclidean space as a standard coordinate axis, much like the x-axis on the cartesian plane. That is to say, the vicinity of each point can have a coordinate axis which behaves Euclidean, but may not be. For instance, the surface of a sphere is a manifold. For any point on the surface of the sphere, there can exist a local coordinate system. However, that number line will only be valid for a small section as movement away from that section of the surface will change the angle and thus the straight coordinate system will fail to be of use. On account of this, manifolds are locally homeomorphic to Euclidean space. Mathematically, a *d-dimensional* manifold M is locally homeomorphic with R^d. The local section of a manifold can be labeled as neighborhood N_x for each $x \varepsilon M$, given that each neighborhood is locally homeomorphic with Euclidean space, $f : N_x \rightarrow R^d$. The process of integrating a manifold M in a section of R^d space is known as embedding.[24] Understanding the topology and geometry behind manifolds will allow for its accurate algorithmic implementation.

MANIFOLD LEARNING

This section will discuss how algorithms will learn from manifolds that are constructed with complex datasets. Suppose there is a data set $x_1, \ldots, x_n \varepsilon R^D$ and its dimensionality needs to be reduced for it to be interpreted conclusively. PCA would not work in this case as the data does not lie on a low-dimensional subspace of R^D. Trying to apply PCA would take far too long and be an inefficient use of resources as multiple trial combinations between data points would need to be generated. Instead, the dataset should be interpreted as lying on a *d-dimensional* manifold embedded in R^D.[25] As previously mentioned, d must be a great deal smaller than D for dimensionality reduction to occur. For simplicity, it will be assumed that the manifold is only two -dimensional. This would yield a single coordinate system. The manifold M can be constructed from dataset $x_1, \ldots, x_n \varepsilon R^D$ in a single coordinate system $f : N_x \rightarrow R^d$ as a result of $y_1, \ldots, y_n \varepsilon R^d$, where $y_i = f(x_i)$.[26] An algo-

rithm will learn to assemble a manifold from a dataset through this process, hence, manifold learning. An example of a simple manifold resulting from a training dataset is the surface curve. This curve is a two-dimensional manifold in three-dimensional space. This is topologically identical to that of a sphere, where the surface can be homeomorphic to an S-shaped curve. The curve has length and width, which makes it two-dimensional, but it resides in three-dimensions. That same curve can stretch to a variety of different objects in order to demonstrate the homeomorphic qualities of the structures. The concept of homeomorphism between these topological structures is important because plotted datasets can yield unpredictable shapes, such as a sphere. Homeomorphism demonstrates that the visualizations can be morphed to simplify the relationships, thus deriving meaning from complex data. The ability for a computer to learn a manifold from a set of points proves to be intricate and cerebral, and has countless applications when utilizing a range of datasets.

ISOMAP AND MULTIDIMENSIONAL SCALING

Isomap is a nonlinear dimensionality reduction method, which makes it ideal for the modeling of a manifold by embedding the dataset's information into Euclidean space. Specifically, Isomap computes a quasi-isometric and low-dimensional structure by embedding sets of high-dimensional data points.[27] Quasi-isometry refers to a large-scale geometrical figure's function that ignores the small-scale details. This allows for the easy estimation of a structure's intrinsic geometry based on a rough estimate of each data point's neighbors.[28] Recall that in order to find a data point's similarities with other inputs, it must be grouped with those inputs along a low-dimensional plane. Isomap will employ KNN, which will allow for the algorithm to find the points that are closest to each other, and hence, the most similar in characteristics relative to the parameterized axes. Points that are closest together in the KNN algorithm will then be grouped together in the final manifold. In order to embed the necessary dataset into a manifold, a multidimensional scaling (MDS) algorithm must first be utilized.[29] This is similar to Isomap, where information is visualized by displaying the contents in a distance matrix. Displaying the elements of a matrix along a set of axes will sort the information into groups that are similar, thereby elucidating patterns amongst the dataset. This is another method of nonlinear dimensionality reduction. The importance of MDS is to find interpoint distances in the visualized dataset.[30] Essentially, interpoint distances refers to the distance between two points randomly chosen on a plot. For this exploration, the points will be randomly chosen within a similar cluster, as these points are similar with respect to the parameters. This is done to yield MDS interpoint distances that are similar, if not identical, to the KNN distances in the Isomap method. Isomap will synthesize a manifold and construct lines between the points, while MDS will find the distances between these points. MDS can only be used locally as it uses Euclidean geometry, which is a local coordinate system. As the manifold curves with the addition of more data points, the MDS will not work for comparing data points that are further apart. Hence, MDS can only be used to get distances of neighboring points after Isomap has constructed the manifold. **Figure 2** showcases an example of MDS comparing the presence of two metagenomes in five different pig organs. Greater distances between data points correlates to more significant differences with respect to the metagenomic presence. Ultimately, both methods will prove fruitful in the exploration to generate a topological manifold from a large dataset.

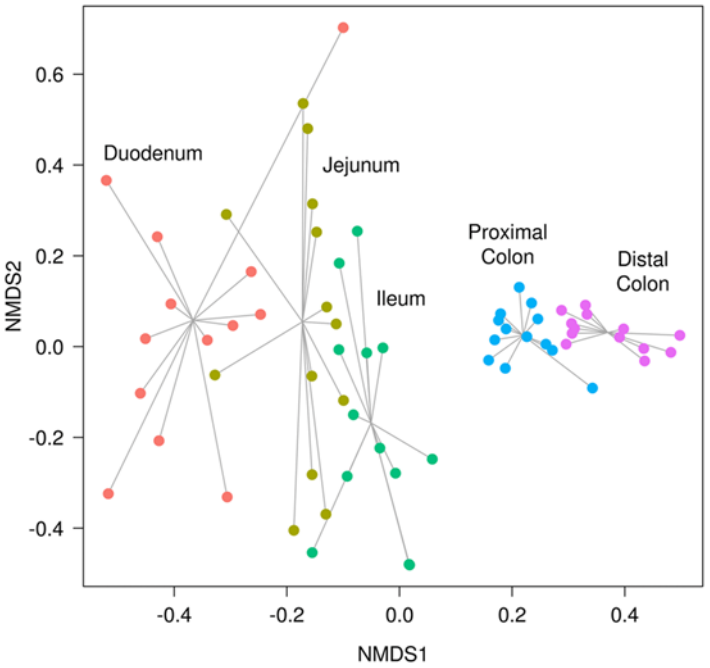

Figure 2. Example of MDS used to plot dissimilarities for two arbitrary metagenomes in five different intestinal sections of a pig. This plot showcases the genomic presence of each metagenome in the different sections. Points clustered together are similar in their metagenomic composition for the two chosen genomes. Interpoint distances are taken between each point and that organ section's average NMDS1 and NMDS2 value. Interpoint distance illustrates how deviated a point is from the other samples in the same section.[31]

Taking distances between adjacent points can be done using Isomap and MDS as the generated distances will be relatively small. However, data points that are further apart will be more difficult as MDS only takes Euclidean distances, which cannot be done between data

points that are on different curved sections of a manifold. Instead, distances between points on a manifold must be taken. To better visualize this, imagine two points on a sphere that are far apart. The distance between these points cannot be calculated, because it would involve passing through the sphere to get the shortest length. Instead, the distance must be taken while still traveling along the surface of the sphere. This will involve tracing from point A to point Z. A computer can do this by taking the distance from point A to some point B that is closer to point Z and also close enough to point A, so that taking the distance does not involve passing through the curved surface of the sphere. Then, the distance between point B and some other point C can be taken, and that distance will add to the previously mentioned distance. Continuously, points will be selected that are separating point A and point Z, until a distance is obtained between the two points far apart. This is known as Dijkstra's algorithm.[32]. This algorithm can be applied to this exploration to construct a manifold. For data points that are far apart, smaller distances between those points can be calculated and then added to yield the cumulative distance. Having MDS calculate local distances and Dijkstra's algorithm quantify larger distances will allow for the construction of a manifold that contains all data points.

Dijkstra's algorithm can be mathematically expressed, and then implemented into the Isomap and MDS algorithm. If x_i, x_j are points on the manifold and $G(x_i, x_j)$ is the distance between them, then there is a chart $f : M \rightarrow R^d$, such that $\|f(x_i) - f(x_j)\| = G(x_i, x_j)$.[33] In this equation, the short distances across high-dimensional Euclidean space will be calculated from neighboring points. This equation will not prove useful however, for distances that stretch far across the manifold structure. This is, again, due to the fact that the manifold is locally linear, and the equation will have to map through paths that have already been delineated. Instead, Dijkstra's algorithm can be used to estimate the distance from one point to another using the pre-existing distances connecting each adjacent point[34]. This will estimate the distances between distant points, which will accompany the information already approximated from the aforementioned equation. Now that Isomap has been utilized, MDS can be put into effect. Recall that MDS will calculate interpoint distances that correspond to the KNN distances calculated using Isomap. In order to use MDS, the calculated Euclidean distances will need to be converted to a Gramian matrix. A Gramian matrix consists of all possible inner products of another matrix.[35] In this particular problem, the starting matrix is $D \varepsilon R^{nxn}$. If X is the matrix of D that is needed, then B is the Gramian matrix; $B = XX^T$, where X is found through spectral decomposition of B into $U\Lambda U^T$.[36] This will yield $X = U\Lambda^{1/2}$. From here, PCA will again be employed to project X onto d-dimensions given that there is a desire to keep the data embedded in a low-dimensional manifold. This will yield XV, where $V \varepsilon R^{nxd}$ is the original matrix containing the dataset. Here, the rows are eigenvalues while the columns are eigenvectors. These eigenvectors $v_1, \ldots, v_d$ are provided by

$$X^T X_{v_i} \qquad \textbf{Eq. 8}$$

Substitute the structural decomposition formula mentioned above into Equation 8 and expand terms of matrix and its transpose to simplify.

$$\left(U\Lambda^{\frac{1}{2}}\right)^T \left(U\Lambda^{\frac{1}{2}}\right) v_i \qquad \textbf{Eq. 9}$$

$$(\Lambda^{\frac{1}{2}} U^T U \Lambda^{\frac{1}{2}}) v_i \qquad \textbf{Eq. 10}$$

$$\Lambda v_i \qquad \textbf{Eq. 11}$$

when $v_i = e_i$, e_i is the i^{th} standard basis vector. Hence, $XV = X[e_1 \ldots e_d] = [X]^{nxd}$. Although this seems quite syntactical and complex, it signifies that the dimensionality of the subspace in which the manifold lies is exactly where the matrix X lies. Hence, MDS can be used to find the dimensionality of the manifold space. However, this method of MDS is limited in its capabilities since D's distances are merely an approximation. If the approximation deviates from Isomap's approximations too greatly, this will prevent the formation of the Gramian matrix. Improving the algorithm to accommodate for this mistake can serve as a future extension to this exploration. From these calculations, the distances between data points on a manifold will be approximated, which will prove useful when interpreting the relationship between the dataset's elements. **Figure 3** is a plot of both Isomap, and MDS that utilizes the Dijkstra algorithm. This gives **Figure 3** the capability to construct a three-dimensional manifold with similar points being close together.

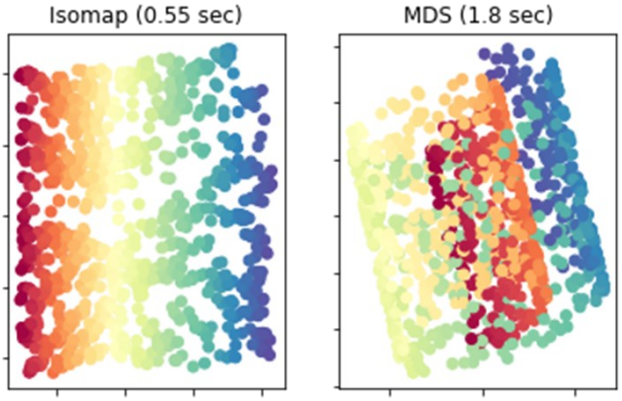

Figure 3. A sample plot of 1000 points reduced to three dimensions of x, y, and z in both Isomap and MDS with computer generation time displayed. Points consist of arbitrary values in three-dimensions in range of figure size. Isomap is the manifold containing all data points while MDS views it from a different angle to capture its depth. Points that are close together are similar with respect to the arbi-

trary parameters of this database plot. Similar/local points are highlighted in similar colours for easy identification of their shared characteristics (see Appendix B).

APPLICATIONS

Isomap and MDS are incredible tools to use in the modeling of manifolds from large and complex datasets. The Isomap effectively reduces the dimensionality of the provided datasets, which allows the manifold to be derived with ease. Nevertheless, this process is not perfect. These algorithms attempt to reduce dimensionality in order to limit the noise in the dataset while maintaining relevant parameters. As a way to prevent losing all vital information, the algorithm may be "generous" in the sense that not all parameters that are deemed irrelevant are removed. Ultimately, some inapt information will be included in the final model, so the process is not entirely efficient. An advantage to using Isomap over MDS is the time difference. Isomap typically takes less than half the time of MDS, which is advantageous for widespread future applications where computers are expected to quickly find relationships in large-scale datasets.[37] For instance, suppose military personnel wanted to find the exact location of a specific target from voice data.[38] Isomap would be ideal for this situation as it is relatively fast and would segregate the audio samples based on similar characteristics, such as languages and speech patterns. This could then be cross-referenced with collected audio from nearby transponders and ultimately assist the military in triangulating the target location.[38] In different circumstances, MDS may be ideal when analyzing voting data and predicting the outcome of elections.[39,40] An MDS algorithm could construct a visualization of voters by parameterizing their voting history. This would work to segregate groups with similar interests. Those that are grouped closely together with small neighboring distances will likely vote in a similar fashion. This can be useful in determining which groups are firm in their position and others who are relatively lenient based on their relative position along the axes. From this, electoral voting campaigns can target specific demographics that are elucidated from this process.[41] These are merely a few applications of manifold learning, an incredible tool that will likely see tremendous growth in research-oriented fields.

LIMITATIONS

A number of limitations occurred throughout this literature review, all of which were a result of the manifold learning method. As mentioned in the Isomap and multidimensional scaling section, the Isomap and MDS distances cannot deviate too much from one another, or else a Gramian matrix will not form. This was conclusive across all papers, but was rather vague and unexplained, leading to more questions. The Jupyter Notebook simulations in **Figure 3** were created using an open source dataset, but it is questionable what the output would have been if the KNN and MDS distances did not correspond. The cited literature also did not make note of this dilemma, or how it could be resolved. Additionally, the papers discussed that the KNN and MDS values can only deviate slightly to yield an adequate Gramian matrix. This is yet again a vague phrase that begs an explanation. What amount or percent can the results of the two methods deviate in order to still yield a functioning Gramian matrix? In order to improve the algorithm for future use, and combat the dilemma of different KNN and MDS distances, a thorough understanding of the underlying causes must be explored. An additional limitation that arose during this exploration concerned the foundational concept of dimensionality reduction. Many papers noted that computers will reduce dimensions somewhat sparingly to avoid the loss of vital information. Not all parameters that are deemed irrelevant are removed. This is simple, but rather vague from the perspective of programming. How do algorithms rank irrelevant pieces of information to preserve knowledge from the dataset? Quantifying the ideas behind this concept would be beneficial to further understand the nuances of the manifold learning algorithm. Moreover, it would be favourable to provide information on the handling of irrelevant information before it is integrated into the final model. Addressing these vagaries would significantly improve ideas around this concept. Although there are some limitations in this literature review, the general concepts of manifold learning were easily understood and appear to be extraordinary in their future implications.

CONCLUSION

This exploration on manifold learning has proved to be advantageous as it works to untangle the complexities behind geometric interpretations of data, while illuminating the benefits that the procedure can provide to data analytics. Though manifold learning has become an increasingly popular area of study within machine learning, much is still unknown. This paper explored only two methodologies of manifold learning: Isomap and multidimensional scaling. This is not an exhaustive list as many more exist, and continue to develop. Testing a multitude of algorithms may provide different perspectives and results, which would improve scientific understanding in this area of discipline. Utilizing more real-world data sets would also be of great benefit. It is important to train algorithms to determine whether they provide sensible outcomes, but real-world data should also be used to test if the methodologies can have widespread applications. Per-

haps most real-world datasets do not lie along an embedded manifold. Realizing this now and adjusting the development of these algorithms would be the best option. Manifold learning as a methodology proves to be a promising resource in elucidating meaningful relationships from otherwise complex datasets.

ACKNOWLEDGEMENTS

This paper and the research behind it would not have been possible without the exceptional support of my mentor, Khawar Zaidi. His enthusiasm, knowledge and exacting attention to detail have been an inspiration and kept my work on track from my first encounter with the concepts of manifolds, to the final draft of this paper. I am also grateful for the insightful comments offered by my colleagues in the Bachelor of Integrated Science program at McMaster University. The authors report no conflicts of interest.

REFERENCES

(1) Jackson Q, Landgrebe D. An adaptive classifier design for high-dimensional data analysis with a limited training data set. IEEE Transactions on Geoscience and Remote Sensing. 2001;39(12):2664–79.

(2) Rasouli K, Hsieh WW, Cannon AJ. Daily streamflow forecasting by machine learning methods with weather and climate inputs. Journal of Hydrology. 2012;414-415:284–93.

(3) Saunders B, Sim J, Kingstone T, Baker S, Waterfield J, Bartlam B, et al. Saturation in qualitative research: exploring its conceptualization and operationalization. Quality & Quantity. 2017;52(4):1893–907.

(4) Roweis ST. Nonlinear dimensionality reduction by locally linear embedding. Science. 2000;290(5500):2323–6.

(5) Yang J, Peng W, Ward M, Rundensteiner E. Interactive hierarchical dimension ordering, spacing and filtering for exploration of high dimensional datasets. IEEE Symposium on Information Visualization 2003 (IEEE Cat No03TH8714). 2003.

(6) Konaté AA, Pan H, Ma H, Cao X, Ziggah YY, Oloo M, et al. Application of dimensionality reduction technique to improve geophysical log data classification performance in crystalline rocks. Journal of Petroleum Science and Engineering. 2015;133:633–45.

(7) Folk RL. The Distinction between Grain Size and Mineral Composition in Sedimentary-Rock Nomenclature. The Journal of Geology. 1954Jul;62(4):344–59.

(8) Zemzem W, Tagina M. Cooperative multi-agent systems using distributed reinforcement learning techniques. Procedia Computer Science. 2018;126:517–26.

(9) Kuwana T. Repetition advantage and disadvantage in perceptual identification tasks. Japanese Psychological Research. 2012;55(3):216–28.

(10) Pang Y, Yuan Y, Li X. Effective feature extraction in high-dimensional space. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2008;38(6):1652–6.

(11) Fan J, Fan Y, Lv J. High dimensional covariance matrix estimation using a factor model. Journal of Econometrics. 2008;147(1):186–97.

(12) Vlachos M, Domeniconi C, Gunopulos D, Kollios G, Koudas N. Non-linear dimensionality reduction techniques for classification and visualization. Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '02. 2002.

(13) Talwalkar A, Kumar S, Mohri M, Rowley H. Large-scale manifold learning. Manifold Learning Theory and Applications. 2011;:121–43.

(14) Han H, Mao B. Fuzzy-rough k-nearest neighbor algorithm for imbalanced data sets learning. 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery. 2010.

(15) Powers DL. Graph partitioning by eigenvectors. Linear Algebra and its Applications. 1988;101:121–33.

(16) Sarkar M. Fuzzy-rough nearest neighbor algorithms in classification. Fuzzy Sets and Systems. 2007;158(19):2134-2152.

(17) Abdi H, Williams L. Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics. 2010;2(4):433-459.

(18) Pierson E, Yau C. ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. Genome Biology. 2015;16(1).

(19) Bing-Kun Bao, Guangcan Liu, Changsheng Xu, Shuicheng Yan. Inductive robust principal component analysis. IEEE Transactions on Image Processing. 2012;21(8):3794-3800.

(20) Harrington P, Vieira N, Espinoza J, Nien J, Romero R, Yergey A. Analysis of variance–principal component analysis: A soft tool for proteomic discovery. Analytica Chimica Acta. 2005;544(1-2):118-127.

(21) Dicker L. Variance estimation in high-dimensional linear models. Biometrika. 2014;101(2):269-284.

(22) FAN L, WANG J. Clustering algorithms for mixed attributes based on rough set. Journal of Computer Applications. 2011;30(12):3377-3379.

(23) Dijkstra J, van Mill J. Erdos space and homeomorphism groups of manifolds. Memoirs of the American Mathematical Society. 2010;208(979):0-0.

(24) Wen G-H, Lu T-H, Jiang L-J, Wen J. Locally linear embedding based on relative manifold. Journal of Software. 2009;20(9):2376–86.

(25) Roe J. Index theory, coarse geometry, and topology of manifolds. Providence, RI: American Mathematical Society; 1996.

(26) Zhang J, Huang H, Wang J. Manifold learning for visualizing and analyzing high-dimensional Data. IEEE Intelligent Systems. 2010;.

(27) Tong Lin, Hongbin Zha. Riemannian manifold learning. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2008;30(5):796-809.

(28) Balasubramanian M. The isomap algorithm and topological stability. Science. 2002;295(5552):7a-7.

(29) Attie O. Quasi-isometry classification of some manifolds of bounded geometry. Mathematische Zeitschrift. 1994;216(1):501-527.

(30) Chen L, Buja A. Local multidimensional scaling for nonlinear dimension reduction, graph drawing, and proximity analysis. Journal of the American Statistical Association. 2009;104(485):209–19.

(31) Chatfield C, Collins AJ. Multidimensional scaling. Introduction to Multivariate Analysis. 2018;:189–211.

(32) Crespo-Piazuelo D, Estellé J, Revilla M, Criado-Mesas L, Ramayo-Caldas Y, Óvilo C, et al. Characterization of bacterial microbiota compositions along the intestinal tract in pigs and their interactions and functions. Scientific Reports. 2018;8(1).

(33) Huang Y-J, Li F-Z. Isospectral Manifold Learning Algorithm. Journal of Software. 2014;24(11):2656–66.

(34) Tang J-T, Wang T, Wang J. Shortest path approximate algorithm for complex network analysis. Journal of Software. 2011;22(10):2279–90.

(35) Chang Y, Hu C, Turk M. Probabilistic expression analysis on manifolds. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004 CVPR. 2004.

(36) Silva CD, Herrmann FJ. Optimization on the hierarchical tucker manifold – applications to tensor completion. Linear Algebra and its Applications. 2015;481:131–73.

(37) Akhbardeh A, Jacobs MA. Comparative analysis of nonlinear dimensionality reduction techniques for breast MRI segmentation. Medical Physics. 2012;39(4):2275–89.

(38) Chu MT, Xu S-F. Spectral decomposition of real symmetric quadratic □-matrices and its applications. Mathematics of Computation. 2009;78(265):293–.

(39) Shen D, Zulch P, Disasio M, Blasch E, Chen G, Wang Z, et al. Manifold learning algorithms for sensor fusion of image and radio-frequency data. 2018 IEEE Aerospace Conference. 2018.

(40) Holmes E. (Supervised Learning) Genre section 1–detecting and characterizing known patterns. Practical Data Mining. 2011;:179–204.

(41) Zhang M, Alvarez RM, Levin I. Election forensics: Using machine learning and synthetic data for possible election anomaly detection. Plos One. 2019; 14(10).

ARTICLE INFORMATION

Senior Editor
Stefano A. Biasi

Reviewers and Section Editors
Caitlin Reintjes, Dalen Koncz, Kushi Malhotra

Formatting and Illustrations
Youssef El-Sayes