To t-SNE or not to t-SNE: An Examination of t-SNE's Perplexities

Tristan Anacker

Department of Mathematical Sciences
Montana State University

May 3, 2019

A writing project submitted in partial fulfillment
of the requirements for the degree

Master of Science in Statistics

APPROVAL

of a writing project submitted by

Tristan Anacker

This writing project has been read by the writing project advisor and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the Statistics Faculty.

May 3, 2019	Mark Greenwood Writing Project Advisor
May 3, 2019	Mark Greenwood Writing Projects Coordinator

Abstract

t-Distributed Stochastic Neighborhood Embedding (t-SNE) is a dimension reduction technique used to visualize high dimensional data. In application of the t-SNE algorithm the user is required to input a parameter named perplexity. t-SNE visualizations differ on the perplexity input, however prior suggestions for using t-SNE only provide rough guidelines in selecting the input value. This paper explores one method of selecting the perplexity input value; minimizing the Kullback-Leibler (KL) divergence. To do so, three simulated data sets with varying degrees of high dimensional structure and a subset of the MNIST data set are explored. Visual inspection is used to compare t-SNE maps created using the minimization of the KL divergence method to manual selection. Through this analysis it is shown that minimization of the Kullback-Leibler divergence does not necessarily lead to the best t-SNE map both at a given perplexity and across perplexities.

I. Introduction

The importance of statistics lies in its ability to extract information in the face of uncertainty. Traditionally, science has relied upon natural observation and theory to create questions and subsequent hypotheses. Following that, data are gathered and analyzed through statistical methods to arrive at a conclusion. Thus, the value of statistics has been based on its inferential abilities. This model of science, while important, is quickly being circumvented by a new approach. One where both the answers and the questions are data driven. When this new model is properly applied, one set of data spawns a question and another set is used to formulate conclusions. The exploratory techniques of this statistical paradigm have merit in their abilities to extract these questions, as they are intended to gain insight into data. Such techniques will become increasingly relied upon as time progresses and the amount of quantifiable information grows. Today's world is a vast network of data generators where technology has transformed most actions of everyday life into data that need to be explored.

The size (number of observations, N) and dimension (number of measured attributes, Q) of data sets are only going to increase as time progresses. Increasing the size of a data set creates issues in terms of computing power. Whereas, when dimension increases, an entirely new set of challenges present themselves. These problems are often summarized as *the curse of dimensionality*. The curse of dimensionality, in lay terms, states that things do not act as expected in high dimensions. To illustrate this peculiar phenomenon, consider a circle and a sphere, both with the same radius. Which has the larger volume? Clearly the sphere. This would intuitively suggest that if the radius were held constant, as the dimension of a circle increases, the volume would also increase. This however is not the case. Examining the equation for the volume of a hypersphere, $V_d(R) = \frac{\pi^{d/2}}{\Gamma(d/2+1)} * R^d$, where d is the dimension of the sphere, and R is radius, indicates that if the radius is held constant at 1, the volume of the hypersphere will be maximized at a dimension of 5. Then as the dimension increases the volume decreases (Zaki and Meria, 2014). This curse of dimensionality does not stop there, but creates a host of problems that must be overcome. One way to handle such issues is to use dimension reduction techniques, which, as the name suggests, reduce the dimensionality of the data set being analyzed.

The focus of this paper is on one dimension reduction technique, t-Distributed Stochastic Neighborhood Embedding (t-SNE). t-SNE was first introduced by van der Maaten and Hinton (2008). The authors originally intended t-SNE to be used as a tool for visualizing high dimensional data in a lower dimensional projection, but the technique has been extended to other applications including classification and image retrieval (Xie et al; 2011). t-SNE has become a vastly popular technique, evident by the 7,498 citations the original paper has on Google Scholar, as of 4/6/2019. However, this does not necessarily mean it operates without any issues across all these applications. Along with describing the inner workings of t-SNE, this paper will explore one of the more interesting, and less well-explained aspects of t-SNE, the user defined parameter, *perplexity*. The method of choosing an input value for perplexity is not explained well, rather it is stated "performance of SNE [and it turn t-SNE] is fairly robust to changes in the perplexity" (van der Matten and Hinton, 2008). However in some applications, the choice of perplexity will lead to differing results. This paper will explore one potential method for choosing the perplexity input; selecting perplexity based on minimizing the Kullback-Leibler divergence.

To accomplish these tasks this paper contains the following sections: A general explanation of the t-SNE algorithm; a brief comparison to another dimension reduction technique; a description of the data used in this analysis; methodology behind the procedures that were used to examine the perplexity parameter; and finally, the results of the analysis and a discussion of both the perplexity parameter and t-SNE in general are considered.

II. t-SNE Algorithm

The intent of t-SNE is to produce a low dimensional projection that retains the local structure of the high dimensional data that are being projected. That is, t-SNE aims to produce a visualization in either two or three dimensions that retains the groupings or other structures in the data that were present in the high dimensional space (van der Matten and Hinton, 2008). In general the t-SNE algorithm can be broken up into four rough steps. First, it randomly projects the high dimensional data points into either two or three dimensional space. Then, similarities between each point within both the high dimensional space and the low dimensional space are calculated. Once the similarities are obtained, the associated joint distributions that are formed by these similarities are compared using Kullback-Leibler divergence. Finally, gradient descent is used to adjust the lower dimensional projections with the aim of minimizing the Kullback-Leibler divergence between the joint distributions associated with the high and low dimensional similarities (van der Matten and Hinton, 2008).

An optional (though generally used) initialization step before running t-SNE is to run principle component analysis (PCA) to reduce the number of dimensions in the data. In doing so, some of the noise in the data is removed and calculation of the distances between each point is sped up (van der Matten and Hinton, 2008). In the experiments ran in van der Matten and Hinton (2008), all data were preprocessed using PCA to reduce the number of dimensions to 30. Then the associated PC scores for the 30 dimensions were extracted and used as variables in their application of t-SNE. In the R packaged used here (Krijthe, 2015), this option is controlled through the *pca* argument.

To calculate the similarities between the high dimensional data points a Gaussian

similarity of the form $p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\Sigma_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}$ is used (van der Matten, 2014). The $p_{j|i}$ can be interpreted as the probability that point i is the closest neighbor, in terms of Euclidean distance, to point j in the high dimensional space (van der Matten and Hinton, 2008). To simplify the optimization step, a symmetric restriction is imposed such that $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$, where N is the size of the data. It is also assumed that $p_{i|i} = 0$ (van der Matten, 2014). Of special importance to this paper is the variance term σ_i^2, which is calculated by using a binary search that involves the user defined perplexity parameter (van der Matten and Hinton, 2008).

To calculate the similarities between the projected low dimensional data points a Cauchy distribution (t-distribution with 1 degree of freedom) of the following form $q_{ij} = \frac{(1+||y_i - y_j)||^2)^{-1}}{\Sigma_{k \neq l}(1+||y_k - y_l)||^2)^{-1}}$ is used (van der Matten, 2014). The q_{ij} can be interpreted as the probability that point i is the closest neighbor, in terms of Euclidean distance in the low dimensional projection, to point j (van der Matten and Hinton, 2008). It is again assumed that $q_{ii} = 0$ (van der Matten, 2014). A Cauchy is used in place of a Gaussian distribution in the lower dimensional similarity calculation to mitigate the potential for overcrowding that is present in the earlier version of t-SNE named Stochastic Neighborhood Embedding (SNE) (van der Matten and Hinton, 2008). Since there is more area in the tails of a Cauchy distribution in comparison to a Gaussian, observations that are close together are more spread out, alleviating the overcrowding issue that was present in the prior version (van der Matten, 2014).

Once the similarities have been calculated within both the high dimensional data and associated lower dimensional projection, the associated joint distributions that these similarities define are compared using Kullback-Leibler divergence. Kullback-Leibler divergence is defined as $KL(P||Q) = \Sigma_{i \neq j} p_{ij} log(\frac{p_{ij}}{q_{ij}})$ (van der Matten and Hinton, 2008). By restricting the distance between points to be symmetric, t-SNE differs in more than one way from its predecessor, SNE. The optimization can be done on a single Kullback-Leibler divergence that compares the joint distribution of the high dimensional similarities to the joint distribution of the low dimensional similarities. In comparison, SNE, which did not require the pairwise similarities to be symmetric, uses the sum of Kullback-Leibler divergences as the associated objective function (van der Matten and Hinton, 2008).

Using the Kullback-Leibler divergence as the objective function, t-SNE iteratively updates the lower dimensional projection using gradient descent. The goal of the optimization is to make the joint distribution of the low dimensional similarities as close to the joint distribution of the high dimensional similarities, and thus minimizing the Kullback-Leibler divergence (van der Matten and Hinton, 2008).

The user defined parameter *perplexity* is of particular interest to this paper. Mathematically, perplexity is $Perp(P_i) = 2^{H(P_i)}$ where $H(P_i)$, known as Shannon entropy, and is defined as $H(P_i) = -\Sigma_j p_{j|i} log_2 p_{j|i}$, where P_i is the conditional probability distribution of observation i given all other high dimensional points. Perplexity is thus set by the user, and a binary search is used to calculate each σ_i^2 that leads to a conditional probability distributions for each point P_i that has the defined perplexity (van der Matten and Hinton, 2008). As a rough interpretation of what perplexity is, van der Matten and Hinton state "perplexity can

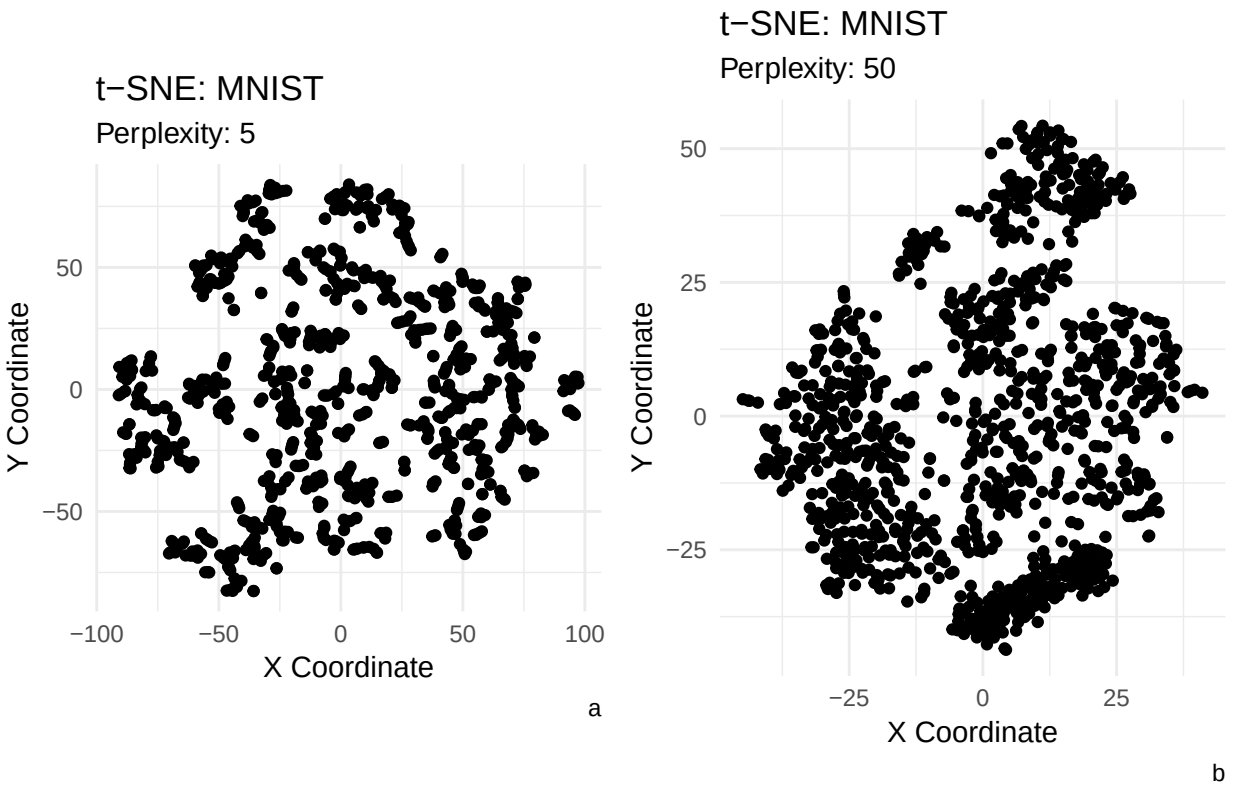

Figure 1: t-SNE maps based on perplexities of 5 (a) and 50 (b) for a sample of n = 1000 observations from the MNIST data set.

be interpreted as a smooth measure of the effective number of neighbors" (van der Matten and Hinton, 2008).

They also state that "The performance of SNE [and in turn t-SNE] is fairly robust to changes in perplexity" (van der Matten and Hinton, 2008). Upon further investigation, this statement does appear overstated. There is typically little qualitative difference between t-SNE maps produced using perplexities that are close or that are the same in large N data sets. However all choice of perplexity will not lead to the same results. Figure 1a presents a t-SNE produced map with the perplexity set to 5, whereas Figure 1b presents a t-SNE map of the same data, using the same initial random projection, with perplexity set to 50. Observation of the two plots yields the conclusion that the two maps differ from one another. This leads to the question of how to set perplexity such that the resulting map is an optimal representation on the true structure in the high dimensional data.

III. t-SNE Comparison

The t-SNE algorithm aims to create a lower dimension representation of high dimensional data. These representations can be thought of as a map of the high dimensional data. As with all map projections, information is inherently lost as the number of dimensions is reduced. An example of this is the Mercator projection, which is responsible for a common

two dimensional map of the earth. The Mercator projection, originally used for sea navigation, retains angles between points. However it distorts the size of objects on the map. Specifically, land masses near the poles are greatly stretched, suggesting they are larger than they actually are and land masses near the equator are compressed, suggesting they are smaller than they actually are (Mercator projection, 2019).

t-SNE aims to retain high dimensional structure by keeping points close in high dimensional space also close in the lower dimensional projection. However, t-SNE loses information such as position of groups relative to one another. That is, how the groups are displayed in the low dimensional space does not necessarily indicate their position with respect to one another in the high dimensional space. In addition, t-SNE maps are arbitrary in reflection, rotation, and scale; these are characteristics shared by most dimension-reduction techniques. Therefore interpretation of these maps must take these aspects into account.

Aside from t-SNE's predecessor, SNE, other algorithms exist that aim to produce lower dimensional maps of the high dimensional data. Two classic techniques are principle component analysis (PCA) and non-metric multidimensional scaling (NMDS). PCA works by projecting the data into the desired dimension such that the resulting projection retains the maximum amount of variability in the new orthogonal axis. For visualization, the desired projection would include scores on the first two principle components. These PCs are the eigenvectors associated with the two largest eigenvalues of the correlation or covariance matrix of the Q variables. This will correspond to the projection that retains the maximum variability (Zaki and Meria, 2014). Unlike t-SNE which aims to keep points in the high dimensional space close in the low dimensional space, PCA aims at keeping points that are far apart in the high dimensional space also far apart in the low dimensional projection (van der Matten and Hinton, 2008). Figure 2a displays a t-SNE map of the MNIST data set with the perplexity defined as 35. Figure 2b displays a map created using the first two principle components of the MNIST data set. The map created using t-SNE correctly delineates some groupings in the data, whereas the map created using PCA does not differentiate any clear groups.

IV. Data Description

To further investigate the performance of t-SNE, four data sets were examined. The first data set was simulated to contain no high dimensional structure, hereafter referred to as "No Structure". The second data set was simulated to have near-perfect high dimensional structure, hereafter referred to as "Clear Structure". The third data set was simulated to have a moderate degree of high dimensional structure, hereafter referred to as "Moderate Structure". The last was a subsample of the classic MNIST data set (LeCun and Cortes, 1999) where 1,000 observations, were randomly sampled from the original 60,000 observations hereafter referred to as "MNIST". The simulated data sets are used to display how changes in the perplexity parameter impact the resulting t-SNE map under controlled circumstances. Alternatively, the MNIST data set was used to display how changes in perplexity impact the resulting t-SNE map when the classes of the data are known, but not the exact high dimensional structure.

To simulate the "No Structure" data set, N = 100 observations were randomly

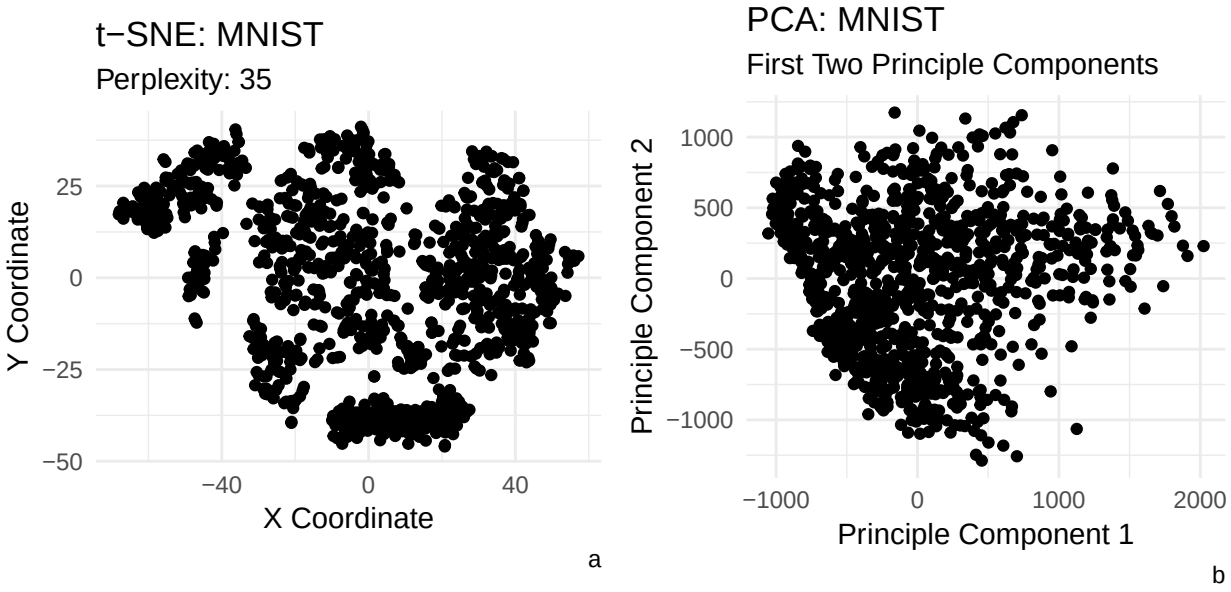

Figure 2: t-SNE (a) map based on a perplexity of 35 and a PCA (b) map using the first two principle components. Both are for a sample of 1000 observations from the MNIST data set. First two PCAs retain 17.63% of the variation in the original 784 variables.

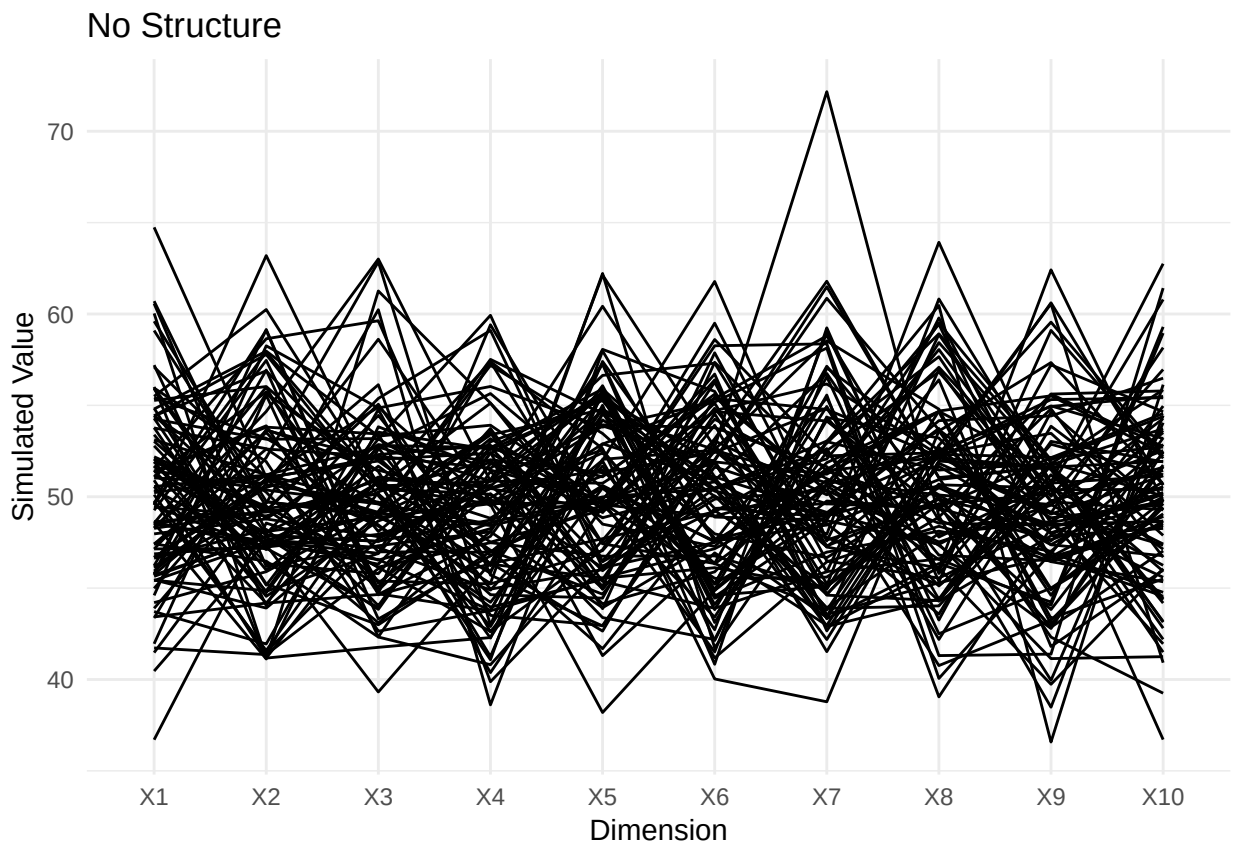

Figure 3: Modified Parallel Coordinates Plot of the No Structure data set displaying each simulated observation across the 10 dimensions.

generated from a Q = 10 dimensional multivariate normal distribution where each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 5\vec{0}_{10x1}$ and $\Sigma = 25I_{10x10}$. Figure 3, a modified parallel coordinates plot (Schloerke et al; 2018), visualizes each of the 100 simulated observations, across the 10 dimensions on the original scale. It displays no inherent structure across the 10 dimensions.

The "Clear Structure" data set was simulated to have three distinct groups. The first group was generated by sampling 10 observations from a 10 dimensional multivariate normal distribution, such that each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 2\vec{0}_{10x1}$ and $\Sigma = 25I_{10x10}$. The second group was generated by sampling 30 observations from a 10 dimensional multivariate normal distribution, such that each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 5\vec{0}_{10x1}$ and $\Sigma = 25I_{10x10}$. The third group was generated by sampling 60 observations from a 10 dimensional multivariate normal distribution such that each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 8\vec{0}_{10x1}$ and $\Sigma = 25I_{10x10}$. Figure 4, a modified parallel coordinates plot, visualizes the value for each of the 100 simulated observations, across the 10 dimensions on the original scale. It displays the three distinct groups.

The "Moderate Structure" data set was also simulated to have three groups, but with less definite boundaries. The first group was generated by sampling 10 observations from a 10 dimensional multivariate normal distribution such that each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 4\vec{1}_{10x1}$

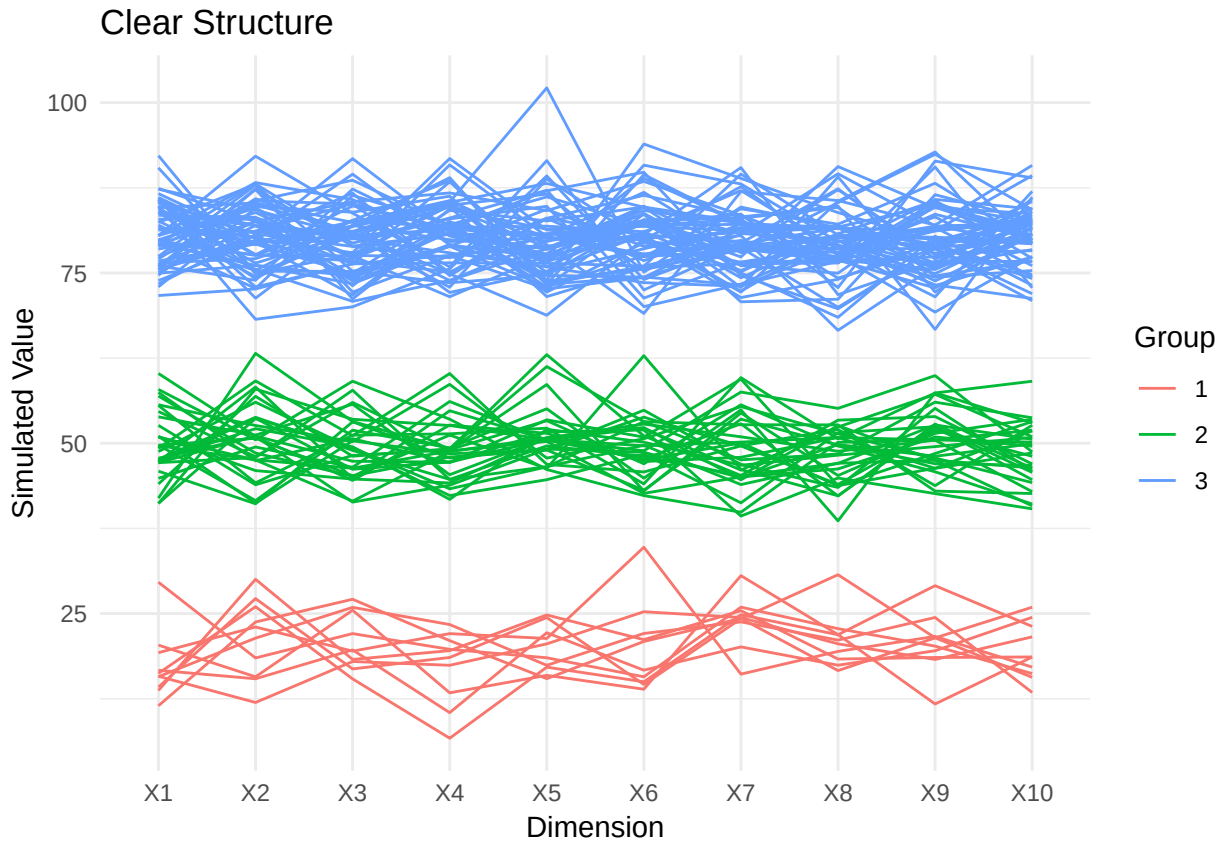

Figure 4: Modified Parallel Coordinates Plot of the Clear Structure data set displaying each simulated observation across the 10 dimensions.

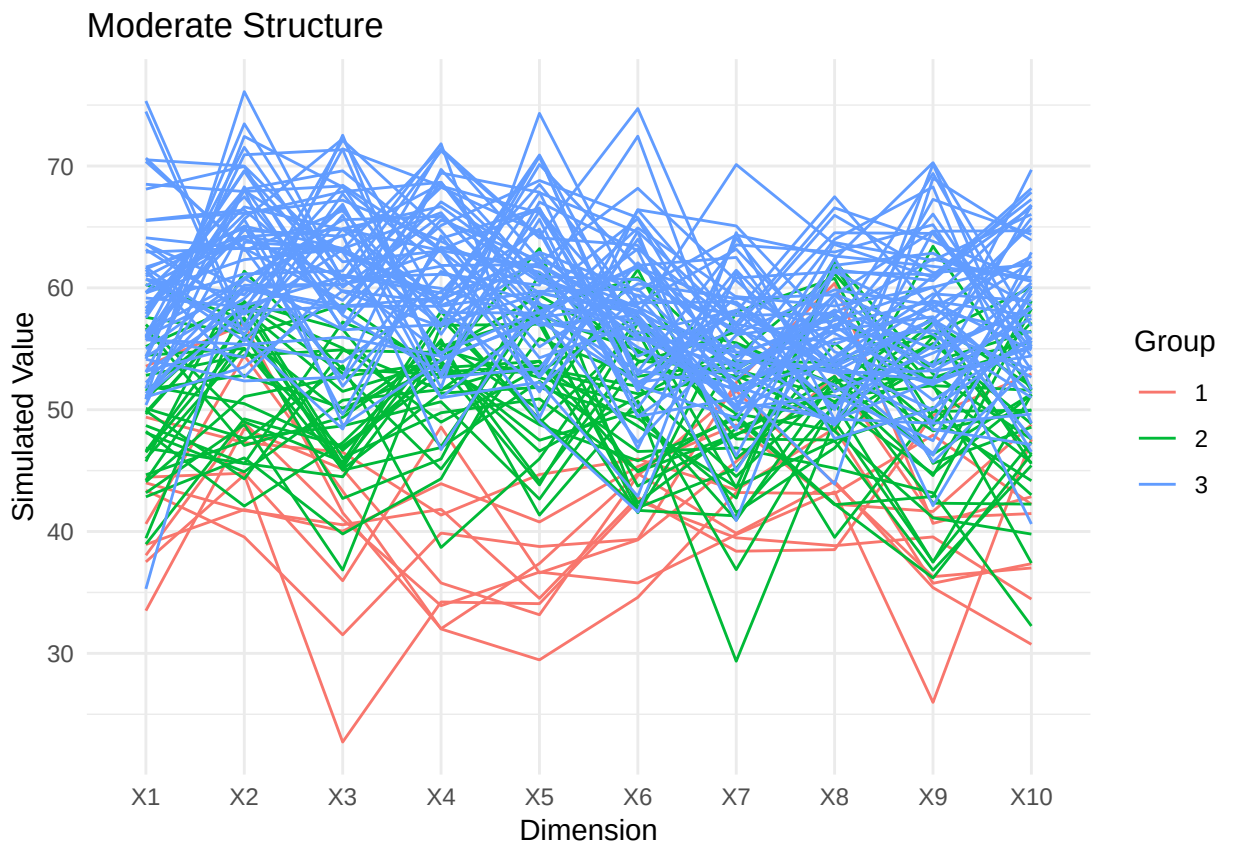

Figure 5: Modified Parallel Coordinates Plot of the Moderate Structure data set displaying each simulated observation across the 10 dimensions.

and $\Sigma = A_{10x10}$, where A has 40s on the main diagonal and 5s on the off diagonals. The second group was generated by sampling 30 observations from a 10 dimensional multivariate normal distribution such that each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 5\vec{0}_{10x1}$ and $\Sigma = A_{10x10}$, where A has 40s on the main diagonal and 5s on the off diagonals. The third group was generated by sampling 30 observations from a 10 dimensional multivariate normal distribution such that each $y_i \sim MVN(\vec{\mu}, \Sigma)$ with $\vec{\mu} = 5\vec{9}_{10x1}$ and $\Sigma = A_{10x10}$, where A has 40s on the main diagonal and 5s on the off diagonals. Figure 5, a modified parallel coordinates plot, visualizes the value for each of the 100 simulated observations, across the 10 dimensions on the original scale. It displays the three groups and indicates some group-level differences but also some overlap across the groups.

The MNIST data set was downloaded from Kaggle.com and it contains 60000 observations and 785 variables. Each observation corresponds to one hand written image of a number between 0 and 9. One of the variables in the data set is an identifier variable that indicates the number that was hand drawn. Whereas the other 784 variables in the data set each correspond to a single pixel in the images and contains the gray-scale value ranging from 0 to 255. t-SNE has a computational complexity of $O(n^2)$ (van der Matten and Hinton, 2008). To make repeated analyses of this data set feasible, 1,000 of the images were randomly sampled and used in this analysis.

V. Methodology

t-SNE uses gradient descent to minimize the Kullback-Leibler divergence between the joint probability distribution of the high dimensional similarities and the joint probability distribution of the low dimensional similarities (van der Matten and Hinton, 2008). Therefore it seemed reasonable to select the optimal perplexity based on optimizing the same objective function, that is, minimizing the Kullback-Leibler divergence.

To test this idea, the t-SNE algorithm was applied using every potential value of perplexity for each of the four data sets, twice. Two runs of the algorithm allows for the assessment of variability due to the random initialization. For each application of t-SNE the Kullback-Leibler divergence of the last iteration was extracted. Then, for each data set, a plot was created that displayed the Kullback-Leibler divergence against perplexity. Based on these plots, a t-SNE map was produced using the perplexity that lead to the t-SNE projection with the smallest Kullback-Leibler divergence. The t-SNE map was compared to the "best" t-SNE map that was found based on manual selection. That is, running through a suite of values of perplexity manually, visually inspecting each associated plot, and selecting the resulting map that appeared "best".

The goal of t-SNE is to produce meaningful visualizations that retain the local structure of the high dimensional data. "Meaningful" differs depending on the data set used. For the "No Structure" simulated data set, there should be no t-SNE map that indicates any groupings in the data. Whereas for the two structured data sets, the more defined the groupings are in the map, the better. For the MNIST data set, there are 10 known groups. The best map would display each known group as distinctly as possible.

This metric of visual comparison is clearly subject to bias. However when the goal is to produce a meaningful visualization, the "eye test" may be what ultimately matters.

All t-SNE maps were estimated in *R* using the package *Rtsne* (Krijthe, 2015). Another package in R that has the ability to run t-SNE is the package *tsne* (Donaldson, 2016). Along with the t-SNE algorithm, the *Rtsne* package provides a few optional parameters: a setting for whether or not to run PCA prior to t-SNE and a "speed up" parameter named *theta*. The theta parameter input can range from 0 to 1 depending on the degree of speedup. It uses the Barnes Hut method as an approximation (van der Matten, 2014), and the closer theta is to one, the greater the approximation. A theta setting of 0 will lead to no approximation, and thus the t-SNE algorithm in its original version. For the purpose of this paper, theta was set at zero for all applications, although variation of results across combinations of perplexity and theta could also be of interest.

VI. Results

Figures 6a through 6d display the Kullback-Leibler (KL) divergence across each value of perplexity for the four data sets analyzed. All four plots indicate an interesting relationship between perplexity and KL divergence, Figures 6b through 6d in particular. Initially, as perplexity increases so does the KL divergence. However, quickly after this initial increase, KL divergence decreases across the remaining values of perplexity. Figure 6a also displays this pattern, however at a perplexity of 1, the KL divergence is lower than at any other value

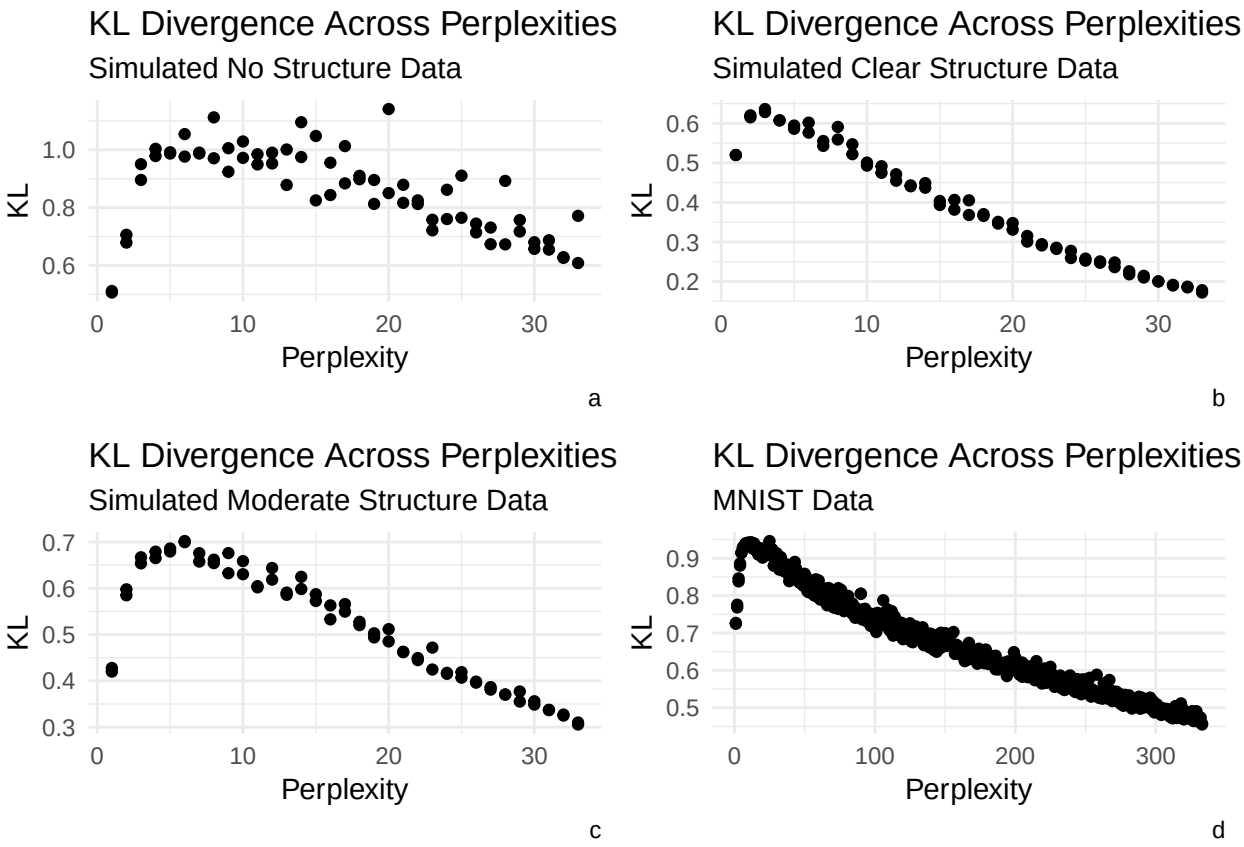

Figure 6: Scatterplots displaying the change in Kullback-Leibler divergence across perplexities for each of the (a) No Structure, (b) Clear Structure, (c) Moderate Structure, and (d) MNIST data sets.

of perplexity. For the other three data sets, the lowest KL divergence occurs at the maximum value of perplexity considered. If minimization of the KL divergence is the goal, it would appear that a perplexity value of 1 should be used for the "No Structure" simulated data set, perplexity values of 33 for both the "Clear Structure" and "Moderate Structure" simulated data sets, and a perplexity of 333 should be used for the reduced MNIST data set. Those last three are the maximum values of perplexity allowed by the algorithm.

In addition to this noticeable trend in KL divergence across perplexity, it should be noted that the KL divergence differs across different runs using the same perplexity. For each data set, t-SNE was ran twice with different seeds for the random initialization for each potential value of perplexity. Figures 6a through 6d also display variability in the KL divergence at the same value of perplexity. That is, different runs using the same perplexity will create different t-SNE visualizations.

Figures 7a and 7b display t-SNE visualizations for the "No Structure" simulated data set. Figure 7a was created using the KL minimization suggested perplexity of one, and Figure 7b was created using a manually selected perplexity of 15. With no prior experience reading a t-SNE map, one may be inclined to pick out many small groups from Figure 7a.

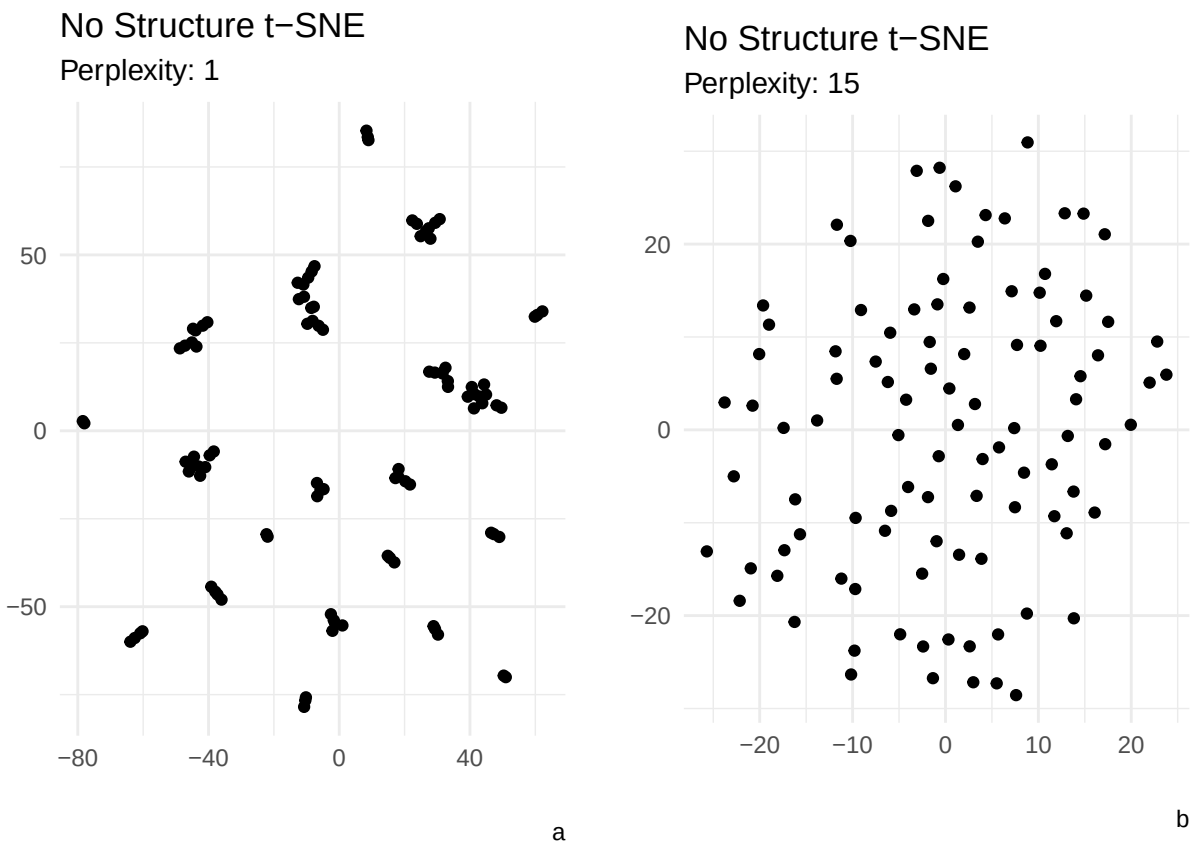

Figure 7: t-SNE maps based on perplexities of 1 (a) and 15 (b) for a sample of n=100 observations from the No Structure data set.

However, with a some experience it becomes clear that maps of this nature, that display small string-like connections, indicate a perplexity value that is too small. Figure 7b, with a perplexity of 15 displays a random cloud of points. In fact, there was no value of perplexity that displayed any sort of grouping (besides the many small string-like groupings) in the low dimensional projection. Thus in the case where there is absolutely no high dimensional structure (at least with Q = 10 dimensions considered here), the choice of perplexity does not seem to have a great impact on the resulting t-SNE visualization. It is possible that the number of dimensions in the original data set could impact these results but that is not explored here.

Figures 8a and 8b display t-SNE visualizations for the "Clear Structure" simulated data set. Figure 8a was created using the KL minimization suggested perplexity of 33, and Figure 8b was created using a manually selected perplexity of nine. Both visualizations display three groups. However the groupings are more clear in the t-SNE map using a perplexity of nine. Figure 8a displays very compact groups. Once again without much experience, one may be inclined to misinterpret the plot by observing only one or two points in one of the groups. However, all the associated points in that group are there, they just have very similar projected values. An interesting point to note about Figures 8a and 8b are the positioning of the groups in relation to one another. In Figure 8a, the large group (which

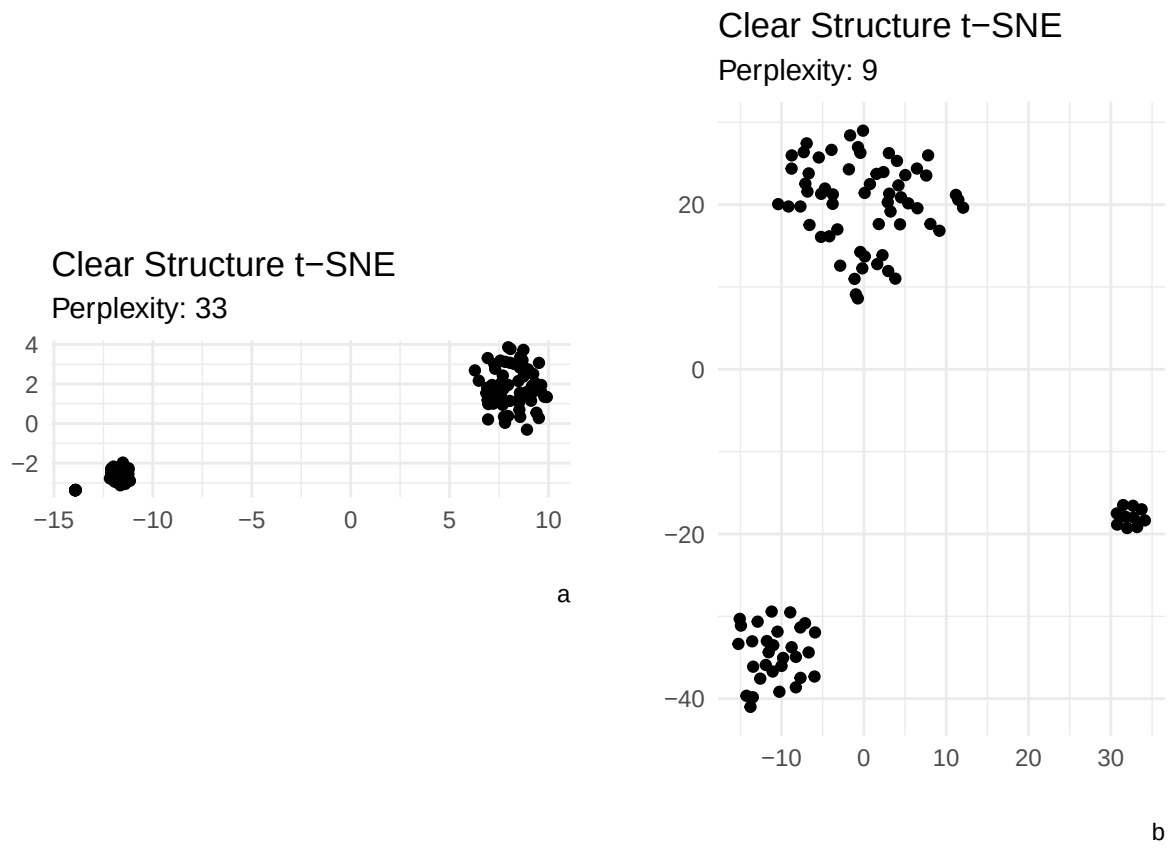

Figure 8: t-SNE maps based on perplexities of 33 (a) and 9 (b) for a sample of n=100 observations from the Clear Structure data set.

was simulated to have a multivariate mean vector of all 80s) is located far away from the other two groups. Whereas the two smaller groups (simulated to have multivariate mean vectors of all 20s and 50s respectively) are relatively close to one another. The projection is clearly not using the means to decide how the groups are positioned with respect to one another, rather the number of observations together or not in a groups. Since perplexity is related to the effective number of neighbors, it is evident with a perplexity of 33, the two smaller groups, which have 10 and 30 points respectively, are pulled closer together, such that each point is close to approximately 33 other points. Whereas in Figure 8b, when the perplexity is set at nine, the groups can spread out as there is no need for the group of size 10 to be close to any other points to meet the perplexity definition. A manual exploration of maps across the perplexities showed that all values of perplexity lead to three groups, besides a perplexity of one. When a perplexity of one was used, the resulting t-SNE map showed the many small-string like groupings that were discussed previously. Thus, it appears that in the "Clear Structure" case, the choice of perplexity does not seem to have a qualitative impact on the resulting t-SNE visualization if it is set to reasonable values.

Figures 9a and 9b display t-SNE visualizations for the "Moderate Structure" simulated data set. Figure 9a was created using the KL minimization suggested perplexity of 33 and Figure 9b was created using a manually selected perplexity of 10. In this case, both plots do

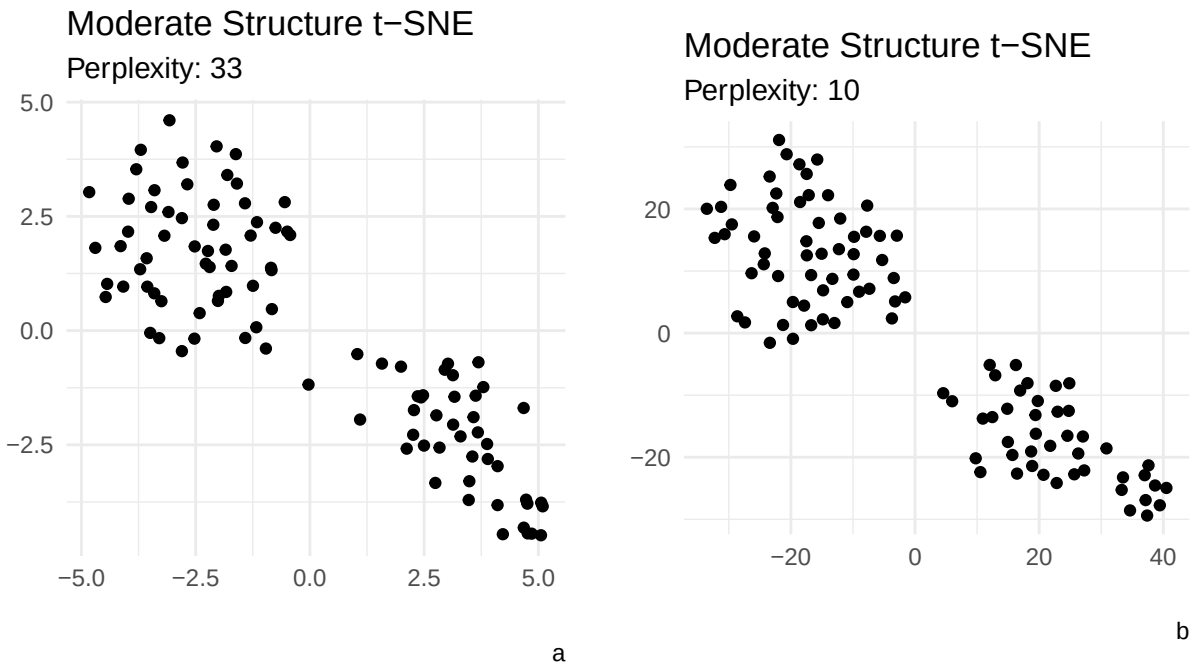

Figure 9: t-SNE maps based on perplexities of 33 (a) and 10 (b) for a sample of n=100 observations from the Moderate Structure data set.

not clearly display three groups. Figure 9a displays two obvious groups, one small and one large. Whereas in Figure 9b, three possibly distinct groups can be seen. In this case then, the choice of perplexity matters. The goal of t-SNE is to project high dimensional points into a low dimensional space while retaining the local structure. Therefore a perplexity of 33 results in a visualization that "fails", as the three groups are not clearly identifiable. Whereas in the visualization created using a perplexity of 10, the three groups are clearly identifiable.

Figures 10a and 10b display t-SNE visualizations for the reduced MNIST data set. Figure 10a was created using the KL minimization suggested perplexity of 333, whereas Figure 10b was created using a manually selected perplexity of 35. Figure 6a displays a random cloud of points and there are no distinguishable groups. Whereas Figure 6b displays some groupings. Without prior knowledge of the groupings, a person may pick out six or seven groups. This is not the 10 classes that are present in these data, but is closer to showing the known groups than Figure 6a that shows one large group.

Figures 11a and 11b also display t-SNE visualizations for the sample of the MNIST data set discussed previously. However they are colored to indicate the grouping that each observed point belongs to. Like Figures 10a and 10b, 11a and 11b were created using perplexities of 333 and 35, respectively. It is important to first examine the non-colored

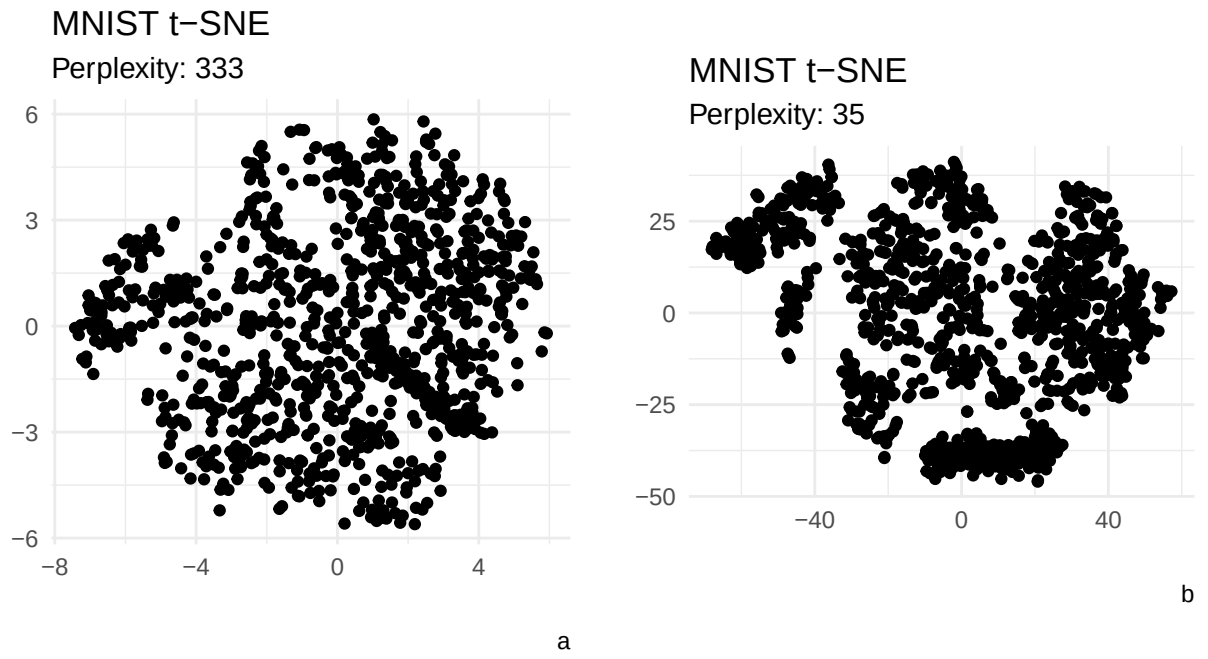

Figure 10: t-SNE maps based on perplexities of 333 (a) and 35 (b) for a sample of n=1000 observations from the MNIST data set.

maps, as to not get mislead by addition of group information. In practice, t-SNE is often used in situations where groups are not known and so coloring the points by known groups is not possible. By adding color to distinguish groupings in Figures 11a and 11g, a general understanding of the accuracy of t-SNE can be examined. Ignoring the fact that Figure 11a is a random cloud of points, the colors indicate that t-SNE generally places points of the same class close to one another. However there is not enough space between the groups to distinguish them from one another without the aid of color. Figure 11b indicates that the groups that were created by the t-SNE algorithm were fairly accurate. For the most part, observations that are close to one another belong to the same class. This method is certainly not perfect, as there are many points that appear to be in groups that actually belong to a different class.

VII. Discussion

Simply selecting the perplexity parameter based on minimization of Kullback-Leibler divergence does not lead to the best possible t-SNE visualization. In the "No Structure" simulated data set, the resulting t-SNE map, which used the perplexity suggested by this method, led to many small groupings that appear commonly for values of perplexity that are too small. In the "Clear Structure" simulated data set, the resulting t-SNE map which

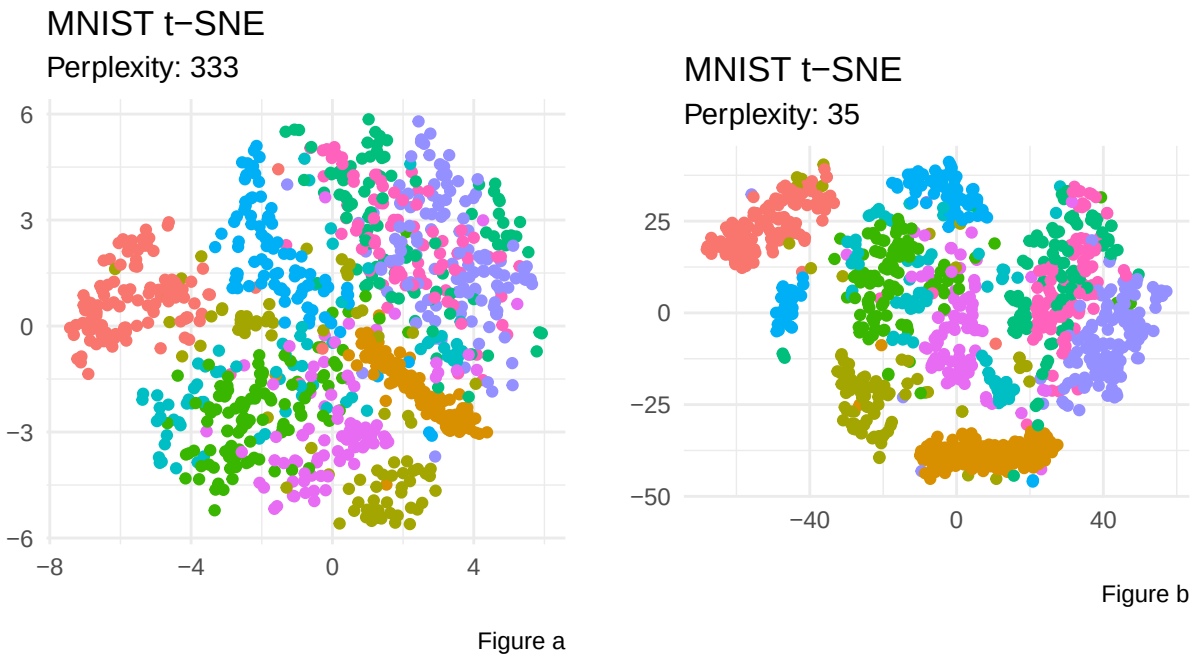

Figure a

Figure b

Figure 11: t-SNE maps based on perplexities of 333 (a) and 35 (b) for a sample of n=1,000 observations from the MNIST data set. Color has been added to display known groups based on the number that were written.

used the perplexity suggested by this method produced a map that correctly identified three groups. However, the proximity of these groups was not in accordance to the global structure of the high dimensional data. In the "Moderate Structure" simulated data set, the resulting t-SNE map which used the perplexity suggested by this method indicated only two groups in these data. Whereas other values of perplexity were able to correctly separate all three groups in the data. In the reduced MNIST data set, the results were similar. The resulting t-SNE visualization using the suggested perplexity indicated one large group; using different perplexities indicated at least some grouping in those data. Thus, selecting perplexity based on minimization of Kullback-Leibler divergence does not appear to be an appropriate method.

Based on these varying results, the question arises: how is perplexity to be chosen? It could be as simple and unsatisfying as a game of guess and check. The original authors provide guidelines that suggest the optimal perplexity parameter will generally be between 5 and 50 (van der Matten and Hinton, 2008). Based on the efforts of this project, this statement does appear generally correct. The method of guess and check falters in its ability to produce results without human interaction. A manual inspection of each produced plot and a judgement call are required for each trial. This is time consuming and likely to lead to biased results. The question that arises by this predicament is how to algorithmically mimic the manual inspection and make a selection based on a less biased criterion.

There is no perfect algorithm, and t-SNE is certainly no exception to the rule. That being said, t-SNE does aim to answer the right questions. Or maybe better yet, produce the right questions. The scientific method relied upon by the current technological world is based on an observation that leads to questions. Traditionally this observation comes from the natural world. However, as time progresses, I believe these observations will be made by observing data. Tools such as t-SNE give a glimpse of the data, they provide insight into a phenomena and spark reason to ask further questions.

VIII. References

Auguie, B. (2017). **gridExtra**: Miscellaneous Functions for "Grid" Graphics. R package version 2.3. https://CRAN.R-project.org/package=gridExtra.

Azzalini, A. and Genz, A. (2016).**The R package 'mnormt'**: The multivariate normal and 't' distributions (version 1.5-5). URL: http://azzalini.stat.unipd.it/SW/Pkg-mnormt.

Donaldson, J. (2016). **tsne**: T-Distributed Stochastic Neighbor Embedding for R (t-SNE). R package version 0.1-3. https://CRAN.R-project.org/package=tsne.

Krijthe, J.H. (2015). **Rtsne**: T-Distributed Stochastic Neighbor Embedding using a Barns-Hut Implementation, URL: https://github.com/jkrijthe/Rtsne.

Mercator projection. (2019, February 04). Retrieved April 30, 2019, from https://en.wikipedia.org/wiki/Mercator_projection.

R Core Team (2018). **R**: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL: https://www.R-project.org/.

Schloerke B., Crowley, J., Cook, D., Briatte, F., Marbach, M., Thoen, E., Elberg, A. & Larmarange, J. (2018). **GGally**: Extension to 'ggplot2'. R package version 1.4.0. https://CRAN.R-project.org/package=GGally.

van der Matten, L.J.P. **Accelerating t-SNE using Tree-Based Algorithms.** *Journal of Machine Learning Research* 15(Oct):3221-3245, 2014.

van der Matten, L.J.P. & Hinton, G.E. **Visualizing Non-Metric Similarities in Multiple Maps.** *Machine Learning* 87(1):33-55, 2012.

van der Matten, L.J.P. & Hinton, G.E. **Visualizing High Dimensional Data Using t-SNE** *Journal of Machine Learning Research* 9(Nov):2579-2605, 2008.

Wickham, H. (2017). **tidyverse**: Easily Install and Load the 'Tidyverse'. R package version 1.2.1. https://CRAN.R-project.org/package=tidyverse.

Xie, B., Mu, Y., Tao, D., Huang, K., & Huang, K. **m-SNE: Multiview stochastic neighbor embedding.** *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics.* Volume 41 Number 4: 1088-1096, August 2011.

Yann Lecun, Y. & Cortes, C. (1999) **The MNIST database of handwritten digits (Images).** http://yann.lecun.com/exdb/mnist/.

Zaki, M. J., & Meira, W. (2014). **Data mining and analysis: Fundamental concepts and algorithms.** New York: *Cambridge University Press.*

IX Appendix:

```
# Settings

knitr::opts_chunk$set(echo = FALSE)
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)
###### Setup

### Packages
library(tidyverse)
library(Rtsne)
library(gridExtra)
library(mnormt)
library(GGally)

### t-sne mnist

mnist <- read_csv("mnist_train.csv")

mnist_small <- mnist %>%
  sample_n(1000, replace = FALSE)


set.seed(34536)

tsne_mnist_1 <- Rtsne(mnist_small[,2:785], pca = TRUE, perplexity = 5, theta = 0)

set.seed(34536)

tsne_mnist_2 <- Rtsne(mnist_small[,2:785], pca = TRUE, perplexity = 50, theta = 0)

tm1 <- ggplot(data = data.frame(tsne_mnist_1$Y)) +
  geom_point(aes(x = tsne_mnist_1$Y[,1], y = tsne_mnist_1$Y[,2])) +
  labs(title = "t-SNE: MNIST",
       subtitle = "Perplexity: 5",
       x = "X Coordinate",
       y = "Y Coordinate",
       caption = "a") +
  theme_minimal() +
  coord_fixed(ratio = 1)

tm2 <- ggplot(data = data.frame(tsne_mnist_2$Y)) +
  geom_point(aes(x = tsne_mnist_2$Y[,1], y = tsne_mnist_2$Y[,2])) +
  labs(title = "t-SNE: MNIST",
```

```
       subtitle = "Perplexity: 50",
       x = "X Coordinate",
       y = "Y Coordinate",
       caption = "b") +
  theme_minimal() +
  coord_fixed(ratio = 1)

grid.arrange(tm1, tm2, nrow = 1)


set.seed(34536)

tsne_mnist_good <- Rtsne(mnist_small[,2:785], pca = TRUE, perplexity = 35, theta = 0)

tmgood <- ggplot(data = data.frame(tsne_mnist_good$Y)) +
  geom_point(aes(x = tsne_mnist_good$Y[,1], y = tsne_mnist_good$Y[,2])) +
  labs(title = "t-SNE: MNIST",
       subtitle = "Perplexity: 35",
       x = "X Coordinate",
       y = "Y Coordinate",
       caption = "a") +
  theme_minimal() +
  coord_fixed(ratio = 1) +
  guides(color = FALSE)


pca1 <- princomp(mnist_small[,2:785])
pca2 <- pca1$scores %>%
  matrix(nrow = 1000) %>%
  data.frame() %>%
  select(c(1, 2))

pcaplot <- ggplot(data = pca2) +
  geom_point(aes(x = pca2$X1, y = pca2$X2)) +
  labs(title = "PCA: MNIST",
       subtitle = "First Two Principle Components",
       x = "Principle Component 1",
       y = "Principle Component 2",
       caption = "b") +
  theme_minimal()+
  coord_fixed(ratio = 1) +
  guides(color = FALSE)
```

```r
grid.arrange(tmgood, pcaplot, nrow = 1)


### Data Generate

# "No Structure"
set.seed(34536)


no_structure <- rmnorm(n = 10, mean = rep(50, 100), diag(25, 100), sqrt=NULL) %>%
  t() %>%
  data.frame()

# "Clear Structure"

set.seed(34536)


g1 <- rmnorm(n = 10, mean = rep(20, 10), diag(25, 10)) %>%
  t() %>%
  data.frame()

g2 <- rmnorm(n = 10, mean = rep(50, 30), diag(25, 30)) %>%
  t() %>%
  data.frame()

g3 <- rmnorm(n = 10, mean = rep(80, 60), diag(25, 60)) %>%
  t() %>%
  data.frame()

yes_structure <- rbind(g1, g2, g3) %>%
  data.frame() %>%
  mutate(Group = c(rep("1", 10), rep("2", 30), rep("3", 60)))

# Middle Structure

set.seed(343446)

g1 <- rmnorm(n = 10, mean = rep(41, 10), matrix(c(rep(5, 100)), nrow = 10) + diag(35, 1
  t() %>%
  data.frame()

g2 <- rmnorm(n = 10, mean = rep(50, 30), matrix(c(rep(5, 900)), nrow = 30) + diag(35, 3
  t() %>%
```

```
  data.frame()

g3 <- rmnorm(n = 10, mean = rep(59, 60), matrix(c(rep(5, 3600)), nrow = 60) + diag(35, 6
  t() %>%
  data.frame()

middle_structure <- rbind(g1, g2, g3) %>%
  data.frame() %>%
  mutate(Group = c(rep("1", 10), rep("2", 30), rep("3", 60)))

ggparcoord(no_structure,
           columns = 1:10,
           scale = "globalminmax") +
  labs(x = "Dimension",
       y = "Simulated Value",
       title = "No Structure")+
  theme_minimal()

ggparcoord(yes_structure,
           columns = 1:10,
           groupColumn = 11,
           scale = "globalminmax") +
  labs(x = "Dimension",
       y = "Simulated Value",
       title = "Clear Structure") +
  theme_minimal()

ggparcoord(middle_structure,
           columns = 1:10,
           groupColumn = 11,
           scale = "globalminmax") +
  labs(x = "Dimension",
       y = "Simulated Value",
       title = "Moderate Structure")+
  theme_minimal()


KL_perp <- function(data, rep_perplexity, max_perplexity) {

  KL <- matrix(c(rep(0, max_perplexity * rep_perplexity)), nrow = max_perplexity)

  for(i in 1:max_perplexity) {
   for(j in 1:rep_perplexity){
        tsne_train <- Rtsne(data, pca = TRUE, perplexity = i, theta = 0.0)
```

```
        KL[i, j] <- tsne_train$itercosts[20]
    }
  }

  KL_gather <- KL %>%
    t() %>%
    data.frame() %>%
    gather(key = "Perplexity", value = "KL", 1:max_perplexity) %>%
    mutate(Perplexity = as.numeric(substring(Perplexity, 2)))

  return(KL_gather)
}


### "No Structure"

set.seed(34536)

KL_gather_nostr <- KL_perp(no_structure, 2, 33)

g_nostr <- ggplot(data = KL_gather_nostr) +
    geom_point(aes(x = Perplexity, y = KL)) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "KL Divergence Across Perplexities",
       subtitle = "Simulated No Structure Data",
       caption = "a") +
  theme_minimal()

### yes structure

set.seed(34536)

KL_gather_yesstr <- KL_perp(yes_structure, 2, 33)

g_yesstr <- ggplot(data = KL_gather_yesstr) +
    geom_point(aes(x = Perplexity, y = KL)) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "KL Divergence Across Perplexities",
       subtitle = "Simulated Clear Structure Data",
       caption = "b") +
  theme_minimal()


### middle structure
```

```r
set.seed(34536)

KL_gather_midstr <- KL_perp(middle_structure, 2, 33)

g_midstr <- ggplot(data = KL_gather_midstr) +
    geom_point(aes(x = Perplexity, y = KL)) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "KL Divergence Across Perplexities",
       subtitle = "Simulated Moderate Structure Data",
       caption = "c")+
  theme_minimal()

### MNIST

set.seed(34536)

#KL_MNIST <- KL_perp(mnist_small[, 2:785], 3, 333)

KL_MNIST <- read_csv("MNIST_perplexity.csv")

KL_MNIST <- KL_MNIST[,2:3]

g_mnist <- ggplot(data = KL_MNIST) +
    geom_point(aes(x = Perplexity, y = KL)) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "KL Divergence Across Perplexities",
       subtitle = "MNIST Data",
       caption = "d")+
  theme_minimal()

### plots
grid.arrange(g_nostr, g_yesstr, g_midstr, g_mnist, nrow = 2)


# No str

set.seed(34536)

tsne_no_str_1 <- Rtsne(no_structure, pca = TRUE, perplexity = 1, theta = 0)

g_nostr1 <- ggplot(data = data.frame(tsne_no_str_1$Y)) +
  geom_point(aes(x = tsne_no_str_1$Y[,1], y = tsne_no_str_1$Y[,2])) +
  labs(title = "No Structure t-SNE",
       subtitle = "Perplexity: 1",
```

```
       caption = "a",
       x = "",
        y = "") +
  theme_minimal()

set.seed(34536)

tsne_no_str_15 <- Rtsne(no_structure, pca = TRUE, perplexity = 15, theta = 0)


g_nostr15 <- ggplot(data = data.frame(tsne_no_str_15$Y)) +
  geom_point(aes(x = tsne_no_str_15$Y[,1], y = tsne_no_str_15$Y[,2])) +
  labs(title = "No Structure t-SNE",
       subtitle = "Perplexity: 15",
       caption = "b",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)

grid.arrange(g_nostr1, g_nostr15, nrow = 1)

# yes str

set.seed(34536)

tsne_yes_str33 <- Rtsne(yes_structure, pca = TRUE, perplexity = 33, theta = 0)

g_yesstr33 <- ggplot(data = data.frame(tsne_yes_str33$Y)) +
  geom_point(aes(x = tsne_yes_str33$Y[,1], y = tsne_yes_str33$Y[,2])) +
  labs(title = "Clear Structure t-SNE",
       subtitle = "Perplexity: 33",
       caption = "a",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)

set.seed(34536)

tsne_yes_str10 <- Rtsne(yes_structure, pca = TRUE, perplexity = 9, theta = 0)

g_yesstr9 <- ggplot(data = data.frame(tsne_yes_str10$Y)) +
```

```r
  geom_point(aes(x = tsne_yes_str10$Y[,1], y = tsne_yes_str10$Y[,2])) +
  labs(title = "Clear Structure t-SNE",
       subtitle = "Perplexity: 9",
       caption = "b",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)

grid.arrange(g_yesstr33, g_yesstr9, nrow = 1)


# mid str

set.seed(34536)

tsne_mid_str33 <- Rtsne(middle_structure, pca = TRUE, perplexity = 33, theta = 0)

g_midstr33 <- ggplot(data = data.frame(tsne_mid_str33$Y)) +
  geom_point(aes(x = tsne_mid_str33$Y[,1], y = tsne_mid_str33$Y[,2])) +
  labs(title = "Moderate Structure t-SNE",
       subtitle = "Perplexity: 33",
       caption = "a",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)

set.seed(34536)

tsne_mid_str10 <- Rtsne(middle_structure, pca = TRUE, perplexity = 10, theta = 0)

g_midstr10 <- ggplot(data = data.frame(tsne_mid_str10$Y)) +
  geom_point(aes(x = tsne_mid_str10$Y[,1], y = tsne_mid_str10$Y[,2])) +
  labs(title = "Moderate Structure t-SNE",
       subtitle = "Perplexity: 10",
       caption = "b",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)
```

```r
grid.arrange(g_midstr33, g_midstr10, nrow = 1)



# mnist

set.seed(34536)

tsne_mnist_333 <- Rtsne(mnist_small[,2:785], pca = TRUE, perplexity = 333, theta = 0)

g_mnist333 <- ggplot(data = data.frame(tsne_mnist_333$Y)) +
  geom_point(aes(x = tsne_mnist_333$Y[,1], y = tsne_mnist_333$Y[,2])) +
  labs(title = "MNIST t-SNE",
       subtitle = "Perplexity: 333",
       caption = "a",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)

set.seed(34536)

tsne_mnist_40 <- Rtsne(mnist_small[,2:785], pca = TRUE, perplexity = 35, theta = 0)

g_mnist40 <- ggplot(data = data.frame(tsne_mnist_40$Y)) +
  geom_point(aes(x = tsne_mnist_40$Y[,1], y = tsne_mnist_40$Y[,2])) +
  labs(title = "MNIST t-SNE",
       subtitle = "Perplexity: 35",
       caption = "b",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme_minimal() +
  coord_fixed(ratio = 1)

grid.arrange(g_mnist333, g_mnist40, nrow = 1)



g_mnist333_col <- ggplot(data = data.frame(tsne_mnist_333$Y)) +
  geom_point(aes(x = tsne_mnist_333$Y[,1], y = tsne_mnist_333$Y[,2], color = as.factor(m
  labs(title = "MNIST t-SNE",
       subtitle = "Perplexity: 333",
       caption = "Figure a",
```

```
       color = "Number",
       x = "",
       y = "") +
  theme(legend.position="none") +
  theme_minimal() +
  coord_fixed(ratio = 1) +
  guides(color = FALSE)

g_mnist40_col <- ggplot(data = data.frame(tsne_mnist_40$Y)) +
  geom_point(aes(x = tsne_mnist_40$Y[,1], y = tsne_mnist_40$Y[,2], color = as.factor(mni
  labs(title = "MNIST t-SNE",
       subtitle = "Perplexity: 35",
       caption = "Figure b",
       color = "Multivariate Mean",
       x = "",
       y = "") +
  theme(legend.position="none") +
  theme_minimal() +
  coord_fixed(ratio = 1) +
  guides(color = FALSE)

grid.arrange(g_mnist333_col, g_mnist40_col, nrow = 1)
```