Final Project

Johns Hopkins University — 625.714.81(SDEs)

Jack Moody

jmoody11@jhu.edu

1 Diffusion Maps

As data sets continue to grow larger, it is increasingly important to develop tools for proper data analysis that are effective and cost efficient. One common solution to this problem is a diffusion map. Diffusion maps are particularly helpful in non-linear data cases since it can handle non-linear dimensional reduction unlike other common dimension reduction algorithms such as Principle Component Analysis (PCA).

A diffusion map is a dimensionality reduction or "feature extraction" algorithm. It is often implemented by taking a data set of high dimensionality and using dimension reduction to identify intrinsic characteristics of the data by discovering the underlying manifold of the data. The manifold can be defined as the lower-dimensional constrained "surface" upon which the data is embedded. In essence, in order to find this manifold, the diffusion map builds a neighborhood graph on the data based on the distances between nearby points (typically based on Euiclidian distance).

The manifold method assumes that the data points lie on a manifold $\mathcal{M}$ and attempts to encode structure through differential operators on $\mathcal{M}$. The first step in constructing this method usually requires the construction of a neighborhood graph with similarity weights derived from a kernel function. Then a differential operator or "Laplacian" is constructed and the dominant eigenvectors are used to organize the data and define the manifold.

A common example of how the differential operator (as seen in [1])is defined is:

$$\mathcal{L}f = \Delta f - 2(1-\alpha)\nabla f \cdot \frac{\nabla q}{q} \tag{1}$$

Where Δ is the Laplace Beltrami operator, ∇ is the gradient operator and q is the sampling density. The normalization parameter α, which is typically between 0.0 and 1.0, determines how much q is allowed to bias the operator $\mathcal{L}$.

The kernel approximation comes into play when thinking about the the sampling density, q. Depending on the size of q, the kernel will be computed. The larger the kernel the accuracy is increased but will take more computation time. Diffusion maps are typically created using either isotropic, aniostropic, local kernels, or both local and isotropic kernels.

To generalize, here is a pseudocode example for a generic algorithm [3]:

Given: Dataset X consisting of rows x_i

1. Compute the kernel matrix k with elements k_{ij}

$$k_{ij} = k(x_i, x_j) = exp(-\frac{d(x_i, x_j)^2}{\epsilon}) \tag{2}$$

2. Obtain the transition matrix P by normalizing K

$$p_{ij} = \frac{1}{d_x} k_{ij} \text{ with } d_x = \sum_i k_{ij} \tag{3}$$

3. Diagonalize P and sort eigenvalues and corresponding left eigenvectors in descending order

4. The set of n eigenvectors span a space of reduced dimensionality (as long as $n < N$) in which the dataset can be efficiently represented.

Now, and important secondary key property of diffusion maps, is that with slight modifications to the kernel, the diffusive terms encoded within the kernel and the bias from the sampling density, q, can be combined in order to approximate the generator of a Markov process in the case of gradient flow where the gradient terms are determined by the gradient of q. This is interesting because it allows for the approximation of a dynamical system generator based purely on the geometry of the data. An example of that type of process will be discussed shortly from [2].

2 Fokker-Planck Equation and Operator

This leads well to a discussion on the Fokker-Planck Equations (FPE) and it's subsequent backwards/ forwards equations and operators. Within statistical mechanics, the FPE is a partial differential equation (PDE) that describes the time evolution of the probability distribution function of the velocity of a particle under the influence of various drag forces and random forces, such as Brownian motion.

2.1 Examples

Forward Kolmogorov
Imagine an Ito process driven by a standard Weiner process, W_t, in a single dimension x.

This process could be described using the following Stochastic Differential Equation (SDE):

$$dX_t = \mu\left(X_t, t\right) dt + \sigma\left(X_t, t\right) dW_t \tag{4}$$

where drift $\mu\left(X_t, t\right)$ and diffusion coefficient $D\left(X_t, t\right) = \sigma^2\left(X_t, t\right)/2$. From this, the FPE for this probability density p(x,t) of this random variable X_t is:

$$\frac{\partial}{\partial t}p(x,t) = -\frac{\partial}{\partial x}[\mu(x,t)p(x,t)] + \frac{\partial^2}{\partial x^2}[D(x,t)p(x,t)] \tag{5}$$

This solution is often known as the Fokker-Planck operator or the Forward Kolmogorov equation.

Backwards Kolmogorov
Assume that the state X_t evolves according to the SDE:

$$dX_t = \mu\left(X_t, t\right) dt + \sigma\left(X_t, t\right) dW_t \tag{6}$$

Then its backwards Kolmogorov equation can be described as:

$$-\frac{\partial}{\partial t}p(x,t) = \mu(x,t)\frac{\partial}{\partial x}p(x,t) + \frac{1}{2}\sigma^2(x,t)\frac{\partial^2}{\partial x^2}p(x,t) \tag{7}$$

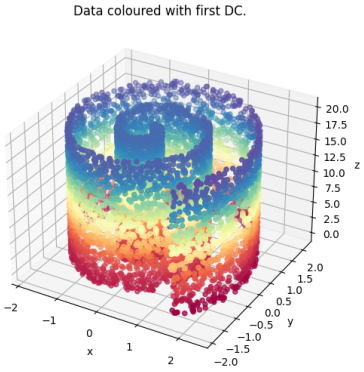
As discussed before, this operator is useful when dealing with time-continuous and state-continuous Markov processes.

3 Canonical Computational Diffusion Maps

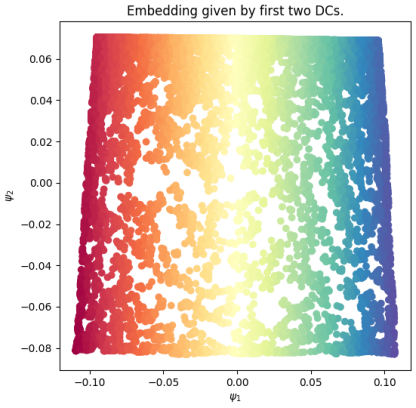
3.1 Swiss Roll

Figure 1 shows what is known in the data science world as the Swiss Roll example, a two dimensional manifold embedded in $\mathbf{R}^3$. The raw data can be seen in the top left figure. The top right shows the diffusion map embedding given by the first two diffusion coordinates. Points are again colored according to the first diffusion coordinate, which seems to parameterize the ϕ direction. We can see that the diffusion map embedding 'unwinds' the Swiss roll. Looking at the bottom left image, a bit more information is squeezed out of the embedding by scaling the points according to the numerical estimate of the sampling density, q, and then coloring the points according to their location in the ϕ direction.

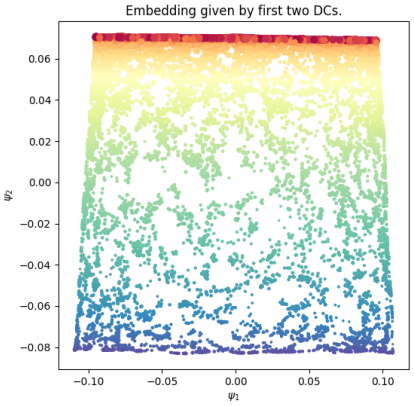
This achieves the real goal we want from a diffusion map, which shows how points that are near the center of the Swiss roll (where data points are tightly together) are closer together in the embedding. In contrast to the points further away from the center, which are more spaced out. Finally, the bottom right figure shows the correlations between the diffusion coordinates and Z and ϕ respectively.



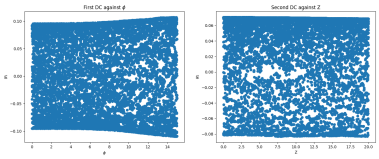
(a) Swill Roll Data Set



(b) sample density,q, distance against z



(c) sample density,q, distance against ϕ



(d) Coordinate correlation against ϕ and against Z

Figure 1: Canonical Swill Roll Example

4 Dynamical System SDE Diffusion Map Example

Due to not being able to implement code for this example, we will just go over the toy model from section 5.3 discussed in [2]:

Let us consider a potential in two dimensions which can fall from the Ornstein-Uhlenbeck process:

$$U(x,y) = \frac{1}{4}x^4 - \frac{25}{12}x^3 + \frac{9}{2}x^2 + 25\frac{y^2}{2} \tag{8}$$

In the x direction, the potential has a double well shape with 2 minima, where one is at x = 0 and the other at x = 4. They are separated by a potential barrier with a maximum at x = 2.25.

As seen in Figure 2, the numerical results of the diffusion map from 1200 points sub-sampled from a stochastic simulation with a potential which generated about 40,000 points. The top right figure shows the potential U(x,0). The top left shows a scatter plot of all the points, which are color coded by the value of the local estimated density p_ϵ, (with $\epsilon = 0.25$). The lower left figure is the first non-trivial eigenfunction plotted vs the first coordinate x. The lower right figure is the first three backward eigenfunctions.

Please take note, that for the bottom left, that despite the variation in the y-variable inside each of the wells, the first eigenfunction ψ_1 is essentially a function of only x, regardless of the value of y. As for the bottom right, all three of the backwards eigenfunction lie on a curve, which indicated that the long time asymptomatics are governed by the passage of time between the two wells and not by the local fluctuations inside them [2].

This shows that within the context of dynamical systems, a diffusion map with appropriate normalization constitute a well structured tool for the analysis of systems exhibiting different time scales. Applications of this technique could include chemical, biological, and physical systems, or even systems in the context of cyber security.

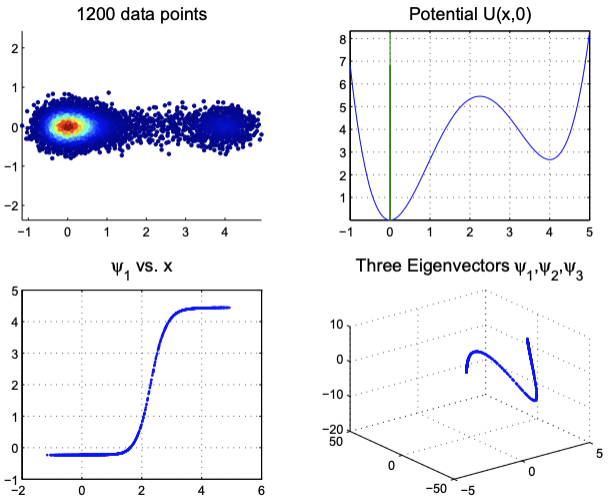


Figure 2: 2 Well Potential from [2]

5 t-distributed Stochastic Neighbor Embedding (t-SNE)

Another rapidly growing area at the intersection of high dimensional data sets and stochastic modeling/ differential equations is the t-distributed Stochastic Neighbor Embedding algorithm (t-SNE), accredited to Laurens van der Maaten and Geoffrey Hinton. t-SNE is a statistical method for visualizing high-dimensional data by giving each data point a location in a two to three dimensional map. From there, each point is modeled in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability. It is important to note that t-SNE is different from a diffusion map, but can be more revealing than diffusion maps when trying to find local manifolds over global manifolds.

Some specific characteristics of t-SNE that separate it from the other methods discussed in this paper are:

1. It is an unsupervised, randomized algorithm that can only be used for visualization
2. Applies a non-linear dimensionality reduction technique where the focus is on keeping the very similar data points close together in lower-dimensional space.
3. Preserves the local structure of the data using student t-distribution to compute the similarity between two points in lower-dimensional space.
4. t-SNE uses a heavy-tailed Student-t distribution to compute the similarity between two points in the low-dimensional space rather than a Gaussian distribution, which helps to address the crowding and optimization problems.
5. It is not impacted by outliers

Specifically, let's discuss how the algorithm achieves this end state [4]:

1. To begin, t-SNE defines joint probabilities p_{ij} that measure the pairwise similarity between objects $\mathbf{x}_i$ and $\mathbf{x}_j$ by symmetrizing two conditional probabilities as follows:

$$p_{j|i} = \frac{\exp\left(-d\left(\mathbf{x}_i, \mathbf{x}_j\right)^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-d\left(\mathbf{x}_i, \mathbf{x}_k\right)^2 / 2\sigma_i^2\right)}, \quad p_{i|i} = 0 \tag{9}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N} \tag{10}$$

2. Then, the bandwidth of the Gaussian kernels σ_i is set such that the perplexity of the conditional distribution P_i equals a predefined perplexity u. The optimal value of σ_i varies per object, and is found using a simple binary search

3. A heavy-tailed distribution is used to measure the similarity q_{ij} between the two corresponding points y_i and y_j in the embedding:

$$q_{ij} = \frac{\left(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|\mathbf{y}_k - \mathbf{y}_l)\|^2\right)^{-1}}, \quad q_{ii} = 0 \tag{11}$$

4. From this embedding, a normalized Student-t kernel is used to measure similarities rather than just using the normalized Gaussian kernel to account for the difference in volume between high and low dimensional spaces. These locations of the embedding points y_i are learned by minimizing the Kullback-Leibler divergence between the joint distributions P and Q:

$$C(\mathcal{E}) = KL(P\|Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{12}$$

   This cost function is non-convex, and is typically minimized by descending along the gradient:

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_{j \neq i} \left(p_{ij} - q_{ij}\right) q_{ij} Z \left(\mathbf{y}_i - \mathbf{y}_j\right) \tag{13}$$

   where the defined normalization terms is $Z = \sum_{k \neq l} \left(1 + \|\mathbf{y}_k - \mathbf{y}_l)\|^2\right)^{-1}$. The evaluation of both joint distributions, P and Q, are $\mathcal{O}\left(N^2\right)$, because their respective normalization terms sum over all N^2 possible pairs of points.

This last point helps us to understand why t-SNE is a better way to visualize data than diffusion maps within specific contexts. Due to the growth of the N^2 term, it limits the total number of data points to just a few thousand before the algorithm becomes cost inefficient. However, the use of the Student-t kernels ensure that the t-SNE focuses on preserving the local data structure. Which can be more helpful in certain situations than seeing the global structure like a typical diffusion map would provide. There is another version of t-SNE, which uses the Barnes-Hut approximation (discussed fully in [5]) which allows for t-SNE to be applied to very large real-world data sets, but there is sadly not enough time to discuss that here.

5.1 t-SNE Computational Example

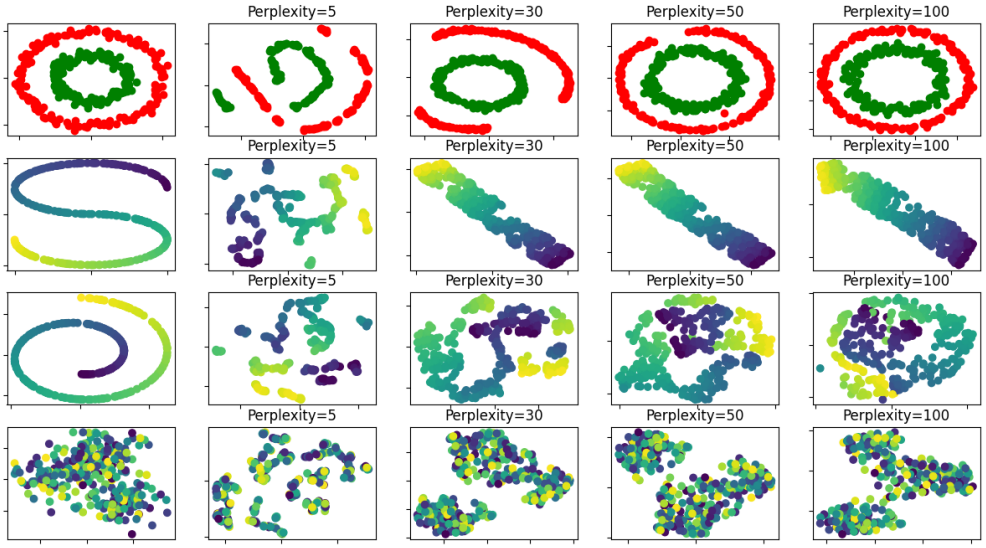


Figure 3: Examples of t-SNE for various example data sets

Above is a t-SNE alrogithm being run in python on various example data sets. It can be easily seen that for different predetermined perplexity values, u, the data can be visualized more easily by t-SNE, but higher perplexity does not always mean better visualization. For example, u = 5 works really well for the blob data set, but terrible for the S-curve. On the other end, u = 100 works great for visualizing the circles and S-curve, but not for the blobs or the Swiss roll. This is an important lesson for t-SNE and general visualization algorithms that experimentation is always needed to ensure the best results are achieved, and to not assume that more strict parameters are always better.

6 Github

Please visit my Github repository here, to see the scripts used in this paper.

References

[1] Ralf Banisch, Zofia Trstanova, Andreas Bittracher, Stefan Klus, and Péter Koltai. Diffusion maps tailored to arbitrary non-degenerate itô processes. October 2017.

[2] R.R. Coifman, I.G. Kevrekidis, S. Lafon, M. Maggioni, and B. Nadler. Diffusion maps, reduction coordinates and low dimensional representation of stochastic systems. 2007.

[3] J. de la Porte, B.M. Herbst, W. Hereman, and S.J. van der Walt. An introduction to diffusion maps. 2006.

[4] Geoffrey Hinton and Laurens van der Maaten. Visualizing data using t-sne. 2008.

[5] Laurens van der Maaten. Barnes-hut-sne. 2013.