Numerical Linear Algebra Project: OSU MATH 5603

Bruce B. Campbell

December 4, 2022

Spectral Clustering

There are many applications of Numerical Linear Algebra in data science. One could make the case that the Singular Value Decomposition (SVD) and iterative eigensolvers are among the most important algorithms in statistics and data science. Methods to resolve aspects of data using the SVD and eigensolvers are called spectral methods. For example Principal Component Analysis (PCA) - a common spectral method in data science - uses the SVD to resolve important directions in data space that have the most variance. PCA is useful if the variance describes something were interested in, as it can tell us what that direction is and provide a transformation of the data to a low dimensional space. In this work we are going to explore some of the applications of eigensolver algorithms to a particular type of clustering algorithm in data science called spectral clustering. The idea behind spectral clustering is to transform a data set with n samples and m features, $X \in \mathbb{R}^{n,m}$ to a graph that we can operate on to resolve different clusters in the data.

Definitions

Let $X \in \mathbb{R}^{n,m}$ be our data set. G, a graph is a collection of edges and vertices ($G = (E, V)$. u v are edges and if they are connected we write $u \sim v$ We write $|V|$, $|E|$ for the number of vertices and edges respectively. $A \in \mathbb{R}^{|V|\times|V|}$ is a graph adjacency matrix; $A_{u,v} = 1 : u \sim v$. $\rho(x_i, x_j)$ will denote a metric or similarity function. We write $K_\rho(x_i, x_j)$ for the similarity matrix induced by the similarity measure ρ. Sometimes we don't distinguish between K and A. Indeed, if A is the complete graph and it's a weighted graph weighted by similarity - then $K = A$. Lastly we'll be dealing with diagonal matrices of degree or weight sums - we denote d_v to be the degree of v in the unweighted case or the sum of the sum of the weights $d_v = \sum_u K(x_v, x_u)$ in the weighted case.

Overview of Spectral Clustering

There are many variants of the spectral clustering algorithm. There's a rich history of the techniques, some originating in the parallel solution of PDE's where it's called graph partitioning. We'll see more about graph cuts below. The basic idea of spectral clustering is to use the eigenvectors of a similarity matrix for the data to perform dimension reduction to a low dimensional space

where clustering is performed. If we're interested in regression or classification - that can also be done in the low dimensional space.

There are two key ingredients to forming the affinity matrix; a distance function, and a convention for which pairs to consider. If all or too many pairs are compared, sparse methods might not be possible. We can include k nearest points, all points within ϵ of a sample x_i, or some other criteria. For instance when working with spatial data, the diffusion can be done on a graph formed by a tessellation of the locations. This is exactly how numerical solutions of heat diffusion is done. In spatial biology applications a cell communication distance can be used to form the graph where the distance is some reasonable measure of inter-cellular communication - then all cells within ϵ of a sample x_i will be connected. This creates graphs of communicating cell networks.

Spectral graph theory takes lots of insights from geometry. Discrete analogues of isoperimetry results and heat flow on manifolds are just a few examples. The normalized graph Laplacian is used to aid in consistency between spectral geometry and stochastic processes. The best reference for background material is Chung [1997].

We generally consider connected graphs $G = (E, V)$ in this work, in which case we can define the normalized graph Laplacian as $\mathcal{L} = D^{\frac{1}{2}} L D^{\frac{-1}{2}} = I - D^{\frac{1}{2}} A D^{-\frac{1}{2}}$, where A is the adjacency matrix, L is defined by

$$L(u,v) = \begin{cases} d_v & : u = v \\ -1 & : u \sim v \\ 0 & : u \nsim v \end{cases}$$

and $D = diag\{d_1, \cdots, d_n\}$ where d_v is the degree of vertex v or the sum of weights as described above.

$\mathcal{L}$ is a difference operator on the space of functions $g \in \mathbb{R}^{|V|}$

$$\mathcal{L}g = \frac{1}{\sqrt{d_u}} \sum_{v:u\sim v} (\frac{g(u)}{\sqrt{d_u}} - \frac{g(v)}{\sqrt{d_v}})$$

$$Vol(G) = \sum_{v\in V}^{d_v} = Tr(T) \tag{0.0.1}$$

$$\sigma(\mathcal{L}) \in \mathbb{R}^+ \tag{0.0.2}$$

$$ker(\mathcal{L}) = span\{T^{\frac{1}{2}} 1\} \tag{0.0.3}$$

This is the same difference operator (up to the scaling by D) that describes the coupled mass spring system. It is the discrete analog of the continuous

Laplacian $\Delta f = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2}$ which has eigenfunctions $f(x) = e^{jnx}$ We know from PDE's that the boundary of the domain of the operator determines it's fundamental modes. This is still the case in the discrete setting. We'll see below that the graph Laplacian has eigenvectors that correspond to high and low frequencies.

There are many nice properties of the Laplacian matrix. It's symmetric, positive semi-definite and diagonally dominant. The implications of which are that the eigenvalues are real and non negative.

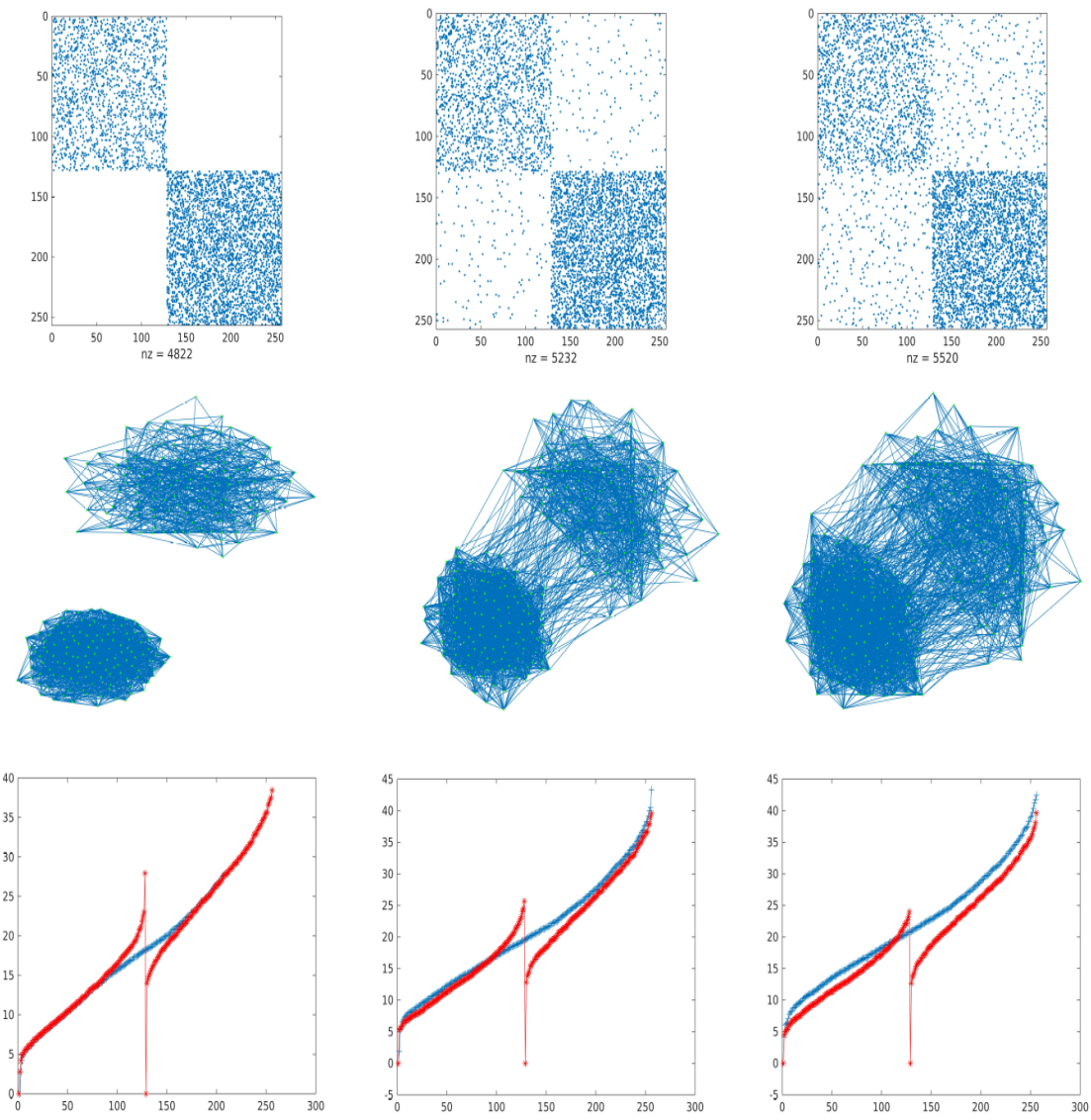
We have defined a discrete version of the traditional Laplacian. There are many interesting comparisons between the discrete setting and continuous geometry. We will see later, the data set we analyze has been prepared to mirror the setting of heat conduction and capture some of the aspects of isoperimetry.

We can now describe the basic steps of spectral clustering. We stick with the method defined in Ng et al. [2001]

- Calculate similarity matrix A
- Calculate Laplacian L
- Calculate first k eigenvectors U_k
- Using $U_k(i, j)$ as embedded feature values, cluster in $\mathbb{R}^k$

Consider the case that we have l classes which are completely dissimilar. If our data is arranged the right way - this means our similarity matrix will be block diagonal. We can then make some nice statements about the spectrum and eigenspaces - namely that they are the union of the blocks. This is theorem 5.2.10 and problem 5.2.20 in Watkins [2004]. Now suppose we 'peturb' the inter-class similarity with some noise. We call this class cross-talk in the accompanying Matlab code. We can use the spectral gaps $|\lambda_i - \lambda_j|$ to analyse the stability of the spectrum and eigenspaces to perturbations.

Before we introduce our data sets and analysis results it's worth mentioning an important relationship between $G = (E, V)$ and A. If we specify a random walk on V with probabilities $A_{u,v}$, then the mixing time of this random walk is related to the second eigenvalue governs the mixing time. We can imagine the amount of cross talk is the measure of mixing between classes. One of our aims is to demonstrate this experimentally.



In the figure above we have collected a few examples of a two class scenario. We plotted $\mathcal{L}$, G, and the last row is the 2 spectra of the diagonal blocks in red aligned to the spectrum of the full matrix in blue. The data above for the adjacency matrices A is generated randomly see our Matlab code Campbell [2022].

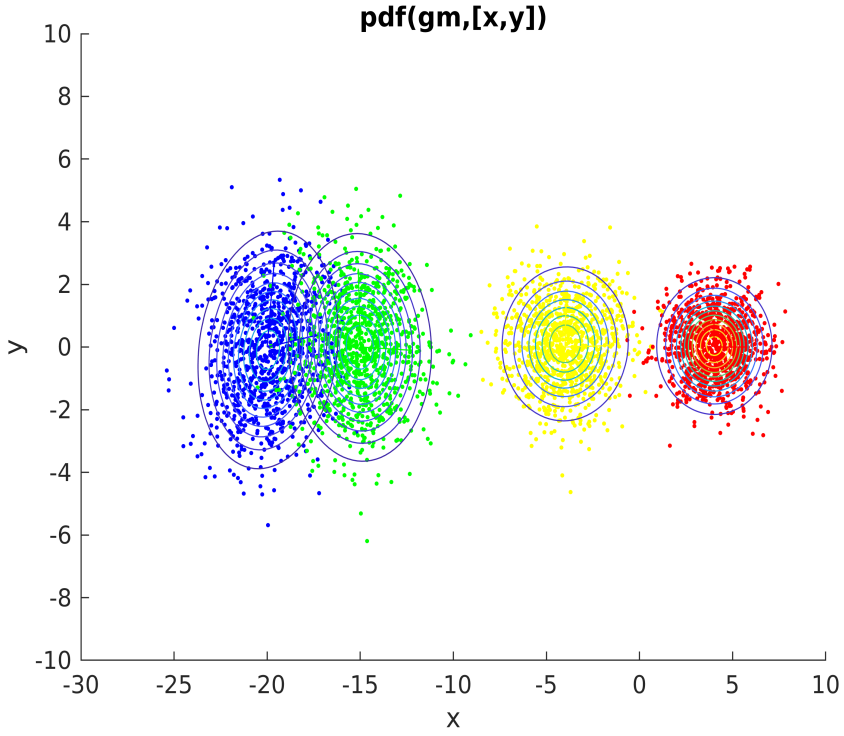
Remember $\mathcal{L}$ is singular with eigenvector $T^{\frac{1}{2}}1$. This is also true for the block diagonal components. We see good agreement on the spectrum being the union

of the components of the blocks when there is no to little crosstalk. On the far right, the union of the spectrum of diagonal blocks is starting to diverge from the spectrum of the full matrix. This is interesting. We calculated the eigengap between 0 and the first non-zero eigenvalue for these Laplacians

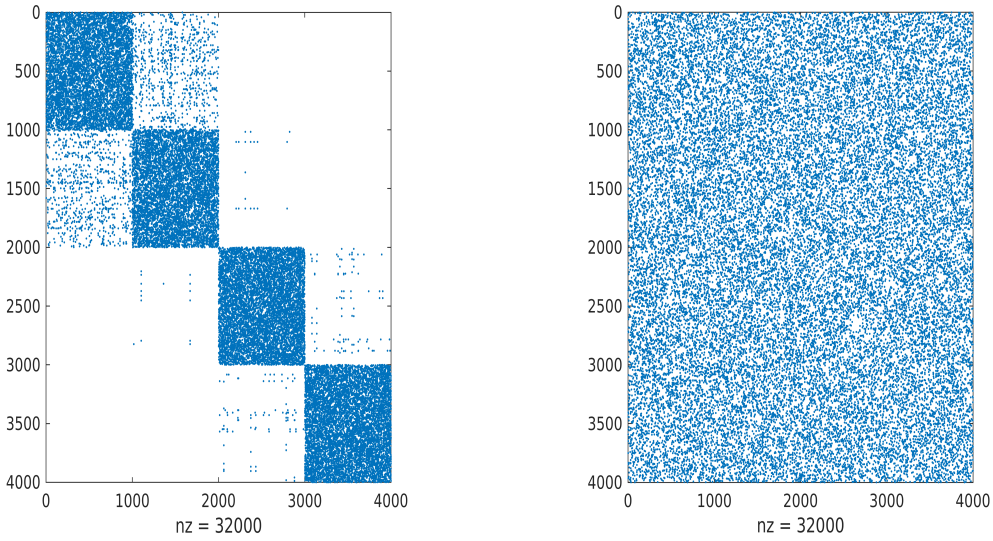
crosstalk probability	eigengap
0	3.73034936275466e-14
0.0100000000000000	1.93400161800437
0.0200000000000000	4.20076779565674
0.0300000000000000	4.65155396750515
0.0400000000000000	7.85252158678672
0.0500000000000000	9.26918323946276
0.0600000000000000	9.96142891808756
0.0700000000000000	10.9737931514795
0.0800000000000000	11.9802338078632
0.0900000000000000	12.6376406303656

We see the relationship between the cross-talk and the eigengap makes intuitive sense and is consistent with the Markov chain interpretation of A. As cross-talk increases we see from the graphs above that the random walk on vertices will be able to cross to other clusters. The more cross talk, the more mixing, and thus the faster convergence to stationarity.

Next we take a synthetic data set through the spectral clustering process up to the calculations of the eigenvalues and eigenvectors. Again see our GitHub repository for the code to replicate the figures and data.



This data was designed to have both noise (outliers) and true cross-talk. The data is a sample from a mixture of four bi-variate random normal variables. Data is not usually given to us in such a nice organized format. To convince ourselves of the utility of the approach in the code we rotate the data in a random direction, and we permute both the rows and columns randomly. The effect of this on the adjacency matrix is below.

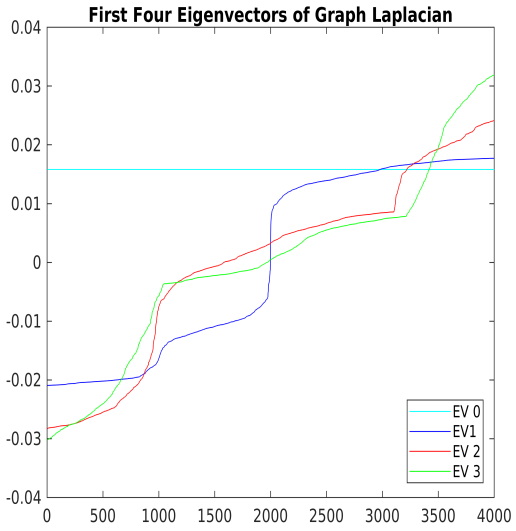


Data analyzed in practice usually arrives out of order, and with no indications which are the important directions. Spectral clustering can help identify underlying hidden structure. Our data is 2 dimensional but the technique works well in high dimensions.

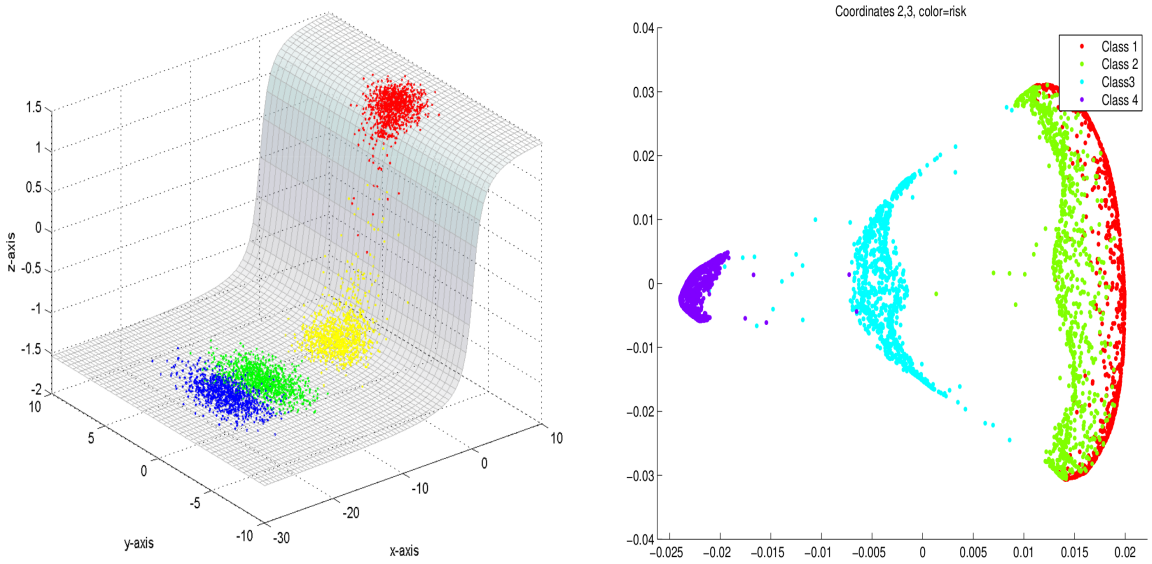
We chose to use a nearest neighbor approach for calculating A. We experimented with using the Gram matrix $X^T X$ and an adaptive approach based on neighborhoods that provides more connections where the data is more dense and got similar results. Below are the first few eigenvectors for the graph Laplacian of our synthetic data.

We see the values of the eigenvectors capture the differences in classes nicely. The same diagram is presented for the two class data that has a low cross-talk. We see the second eigenvector separates the data based on sign alone.

For all the matrices involved we calculated the matrix and eigenvalue condition numbers. Being singular - the matrix condition numbers were very high. The eigenvalue condition numbers for the graph Laplacian were low regardless of the method used to calculate the affinity matrix or which Laplacian was used.



There is a lot of interesting literature extending these methods and investigating convergence properties Lee and Wasserman [2008] Belkin and Niyogi [2008]. We experimented with the diffusion map algorithm of Coifman et al. [2005] on a data set modified with one extra direction to capture a response. Diffusion maps take the approach of running a diffusion on the data to extract a non linear low dimensional embedding for the data that captures the low dimensional geometry. The plot on the right is the low dimensional embedding of the data painted with the cluster memberships.



The steps to calculate Diffusion Maps are very similar to those of spectral clustering but reply on local kernel approaches we touched upon and have nice theoretical properties. The approach takes into account the density of data when calculating L.

We've seen that the eigenvectors of a graph Laplacian calculated from an affinity matrix of similarities can reveal hidden structure in data. This approach to data analysis has many interesting theoretical connections to geometry, probability, and functional analysis.

Bibliography

Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2008. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2007.08.006. URL `https://www.sciencedirect.com/science/article/pii/S0022000007001274`. Learning Theory 2005.

Bruce Campbell. Osu math 5603 numerical linear algebra. `https://github.com/campbell-2589/OSU`$_M ATH_5$603.*git*, 2022.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

Ronald R Coifman, Stephane Lafon, Ann B Lee, Mauro Maggioni, Boaz Nadler, Frederick Warner, and Steven W Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences*, 102(21):7426–7431, 2005.

Ann B. Lee and Larry Wasserman. Spectral connectivity analysis, 2008. URL `https://arxiv.org/abs/0811.0121`.

Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

David S Watkins. *Fundamentals of matrix computations*. John Wiley & Sons, 2004.