Spectral Methods in Data Science

There are lots of applications of Numerical Linear Algebra in data science. One could easily make the case that the Singular Value Decomposition (SVD) and iterative eigensolvers are some of the most important algorithms in industry. Principal Component Analysis (PCA) is one of the most common spectral methods in data science; using 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 paper 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.

The rest of this paper is organized as follows

- Definitions
- Overview of Spectral Clustering
- Observations from clustering synthetic data

Definitions

We try to keep with these conventions in our code as well. 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 $\mathcal{L} = T^{\frac{1}{2}} L T^{\frac{-1}{2}} = I - T^{\frac{1}{2}} A T^{-\frac{1}{2}}$, 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 $T = diag\{d_1, \cdots, d_n\}$ 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 $g \in \mathbb{R}^{|V|}$

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

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

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

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

This is the same difference operator (up to the scaling by T) 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. When a matrix has a large condition number it can be pre-conditioned for better performance in solvers, particularly for iterative solvers. The multiplication of L on the left and right by T might be seen as a form of Jacobi (diagonal) pre-conditioning. We know the matrix is singular, but it might help with eigenvalue condition numbers. We perform some experiments below to investigate this.

Notice 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]

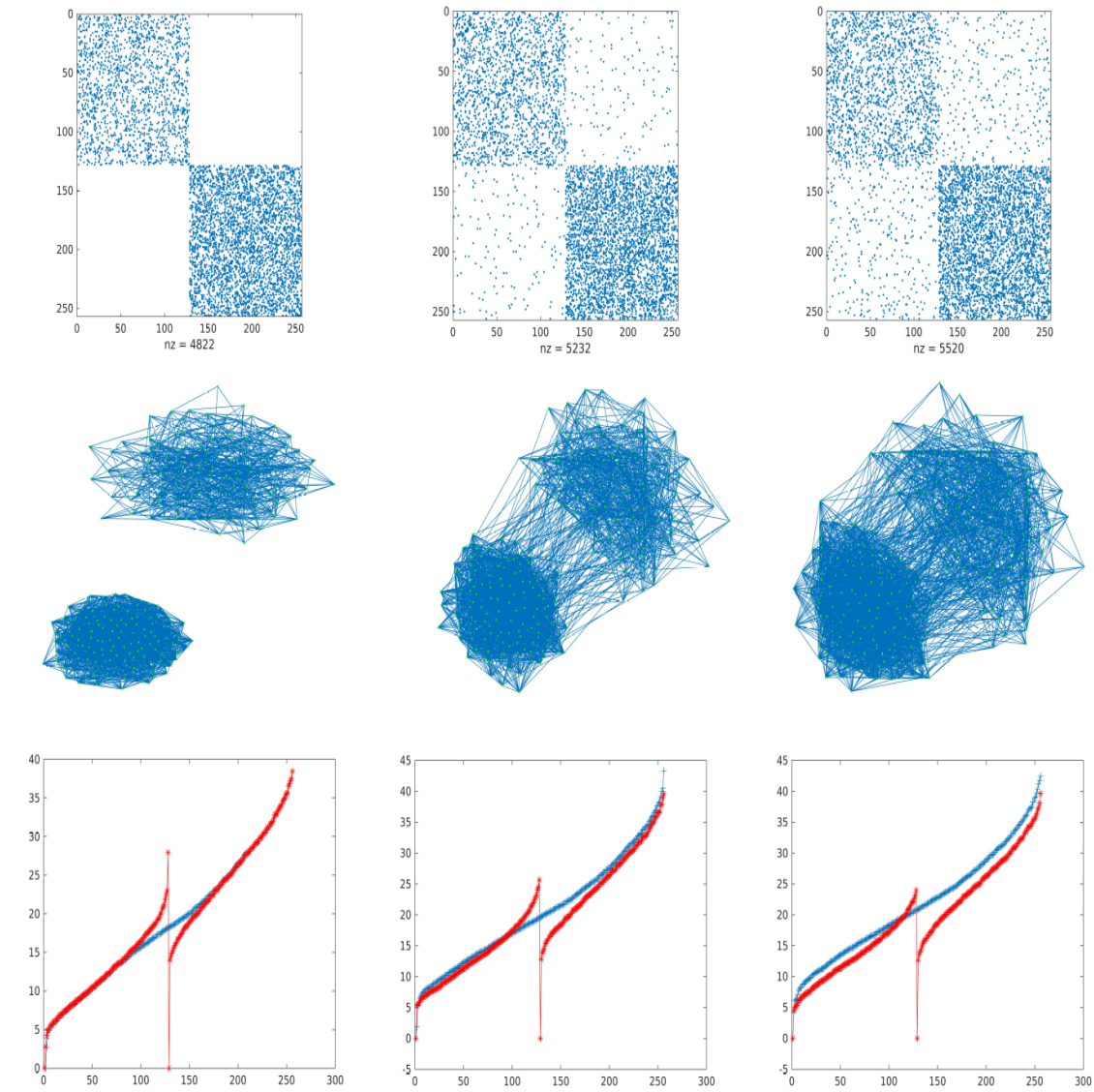
- Calculate similarity matrix A
- Calculate Laplacian L
- Calculate first k eigenvectors U_k
- Using $U_k(i, j)$ as embedded feature values, cluster in $\mathbb{R}^k$

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.

Experimental Results



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.

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 crosstalk and the eigengap makes intuitive sense and is consistent with the Markov chain interpretation of A. As crosstalk 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.

Bibliography

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

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.