Spectral Clustering
Implementation and Analysis

AOE 4404: Final Project

Ethan Gaebel
M.S. candidate
Computer Science and Applications

Outline

Data Clustering

- In many fields involving large data sets it is useful to group together subsets of the data to identify similarities
- There are many algorithms used for clustering data

Data Clustering Difficulties

- Compactness vs Connectivity
  - Compact clusters span a small area and are densely populated by data points
    - Usually use distance metrics to determine cluster membership
  - Connected clusters are often found in irregular shapes
    - Usually rely on constructing a graph (as in graph theory, G = (V, E), where V is a set of vertices and E a set of edges) based on some measure of closeness between points
      - More elementary graph theory later

Different View of Problem

- Instead of clustering data points based on distance we could cluster them based on similarity, or closeness
- We can represent these relationships with a graph, which we represent with an adjacency matrix

A Brief Intro to Graph Theory

- Visually graphs are often represented as nodes with lines between them

- Mathematically it is convenient to model them as matrices, where each row and each column corresponds to a vertex
- If there is a 1 at A(i, j) then there is an edge from vertex i to vertex j
- Such a matrix is referred to as an Adjacency Matrix

A Brief Intro to Graph Theory

- A graph is described as follows
  - G = (V, E)
    - V is a set of vertices
    - E is a set of edges
      - Each edge, $e=(v_i, v_j) : v_i, v_j \in V$
- Graphs can be undirected where the existence of the edge (v_i, v_j) implies (v_j, v_i)
  - All the graphs we will work with here will be undirected
- Graphs can be directed where (v_i, v_j) does not imply the existence of the edge (v_j, v_i)

A Brief Intro to Graph Theory

- Visually graphs are often represented as nodes with lines between them

- Mathematically it is convenient to model them as matrices, where each row and each column corresponds to a vertex
- If there is a 1 at A(i, j) then there is an edge from vertex i to vertex j
- Such a matrix is referred to as an Adjacency Matrix

Spectral Graph Theory

- Spectral graph theory is a branch of graph theory that examines the relationship between the eigenvalues and eigenvectors of a graph's adjacency matrix and the underlying graph itself

Spectral Graph Theory

- Spectral graph theory takes special interest in another graph matrix called the Laplacian, L
- First we must introduce the diagonal matrix, D, called the degree matrix
  - The degree of a vertex in a graph is the number of edges that vertex is involved with
  - In the context of an adjacency matrix this means

$$D_{i,j} = \sum_{k=1}^{|V|} A_{i,k}, if\, i = j$$
$$0, otherwise$$

Spectral Graph Theory – The Laplacian

- Recall D is the degree matrix and A is the adjacency matrix
- The unnormalized Laplacian is defined as:
  - L = D – A
- The Laplacian has the very useful property that for every vector f

$$f^T L f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} \left(f_i - f_j \right)^2 \geq 0$$

- Making the Laplacian a positive semi-definite matrix

Spectral Graph Theory – Properties of The Laplacian

- A few more useful properties of the Laplacian (L) are
  - L is symmetric
  - The smallest eigenvalue of L is 0 and the corresponding eigenvector is the 1 vector (more generally, a vector having all the same values)
  - All eigenvalues of L are real-valued and non-negative
- These properties are useful for the linear algebra techniques we will use later

Spectral Graph Theory – Using the Laplacian

- So how does this serve our interest in clustering?
- Translating our data into a graph creates several sub-graphs (depending on how we create our similarity matrix)
  - These sub-graphs contain mapped data points which are similar, so we have our clustering!
  - We just perform a mapping back to our original data set to cluster our points
- How do we create our similarity matrix?

Constructing the Similarity Matrix

- Nearest Neighbor – compare each pair of data points and determine the following:

$$W_{ij} = \|x_i - x_j\| \leq \epsilon$$

- If this is true we place a edge between the two data points in the similarity graph, a 1 in the similarity matrix, otherwise we insert a 0
- Gaussian Kernel – For dimensions > 1 the Gaussian kernel is a regular product of one-dimensional Gaussians; making it separable. This property is useful to us so we compute the value for each edge between our data points using the Gaussian kernel as follows:
  - We omit many of the familiar constant terms since they are being applied uniformly to all of our data

$$W_{ij} = e^{\left(\frac{\|x_i - x_j\|}{(2\sigma^2)}\right)}$$

Constructing the Similarity Matrix

- Clustering results can be dependent on the values of epsilon and sigma depending on how the similarity matrix is constructed
  - Careful attention must be paid to the data and the resulting clustering to ensure good results are obtained

Clustering On the Similarity Graph

- After converting data to similarity graph representation we can use the eigenvectors to perform clustering
- To do this, we create a matrix where the columns correspond to the first K eigenvectors
- Then we run K-Means on the eigenvectors so that each row is treated as a data point
- We obtain the clustering results and for each row in the eigenvectors data is mapped back to the corresponding row index in the data such that if row j in the eigenvectors is assigned to cluster k then row j in the data set is assigned to cluster k

Numerical Methods

- QR Algorithm
- Householder's Method
- K-Means
- Spectral Clustering

QR Algorithm

- Used to compute eigenvectors and eigenvalues for the Laplacian matrix
- Operates on a symmetric tridiagonal matrix, A
  - Non-tridiagonal matrices are transformed using Householder's method, discussed next
- Performs iterative QR factorization on the matrix A

Householder's Method

- Repeatedly multiplies the input matrix A by an orthogonal matrix P to obtain a 'similar' matrix with the same eigenvalues while making the matrix sparse until it becomes tridiagonal
- To preserve eigenvector information we must keep track of the transformations applied to A with a matrix V
  - V = I;
  - …
  - A = PT * A * P
  - V = V * P;
- Accompanying QR algorithm uses this matrix to compute eigenvectors

K-Means

- K-Means is a popular clustering algorithm which clusters data into K clusters, given a particular K
- K-Means does this by solving the following minimization problem, where μ is the center (mean) of a cluster, S_i, and x is a data point in that cluster

$$\underset{\boldsymbol{S}}{argmin} \sum_{i=1}^{K} \sum_{x \in S_i} \left(\left\| x - \mu_i \right\| \right)^2$$

K-Means

- As an algorithm K-means selects initial μ values (means) from the data set and iterates over all data points assigning each data point to the closest cluster by Euclidean distance
- After each assignment of data point to cluster K-means updates the values for all μ
- The algorithm terminates when no further updates are performed

K-Means Initial Means

- K-Means can be very sensitive to the initial assignments of μ
- Many techniques suggest random selection
- Due to the nature of our data, eigenvectors over Laplacians, we see a lot of repetition in the data points being clustered on and to ensure diversity we would like to not select identical means
- K-means++ is a technique to accomplish this

K-Means++

- K-Means++ is a tool used to select higher quality initial values for μ in the K-Means algorithm
- It selects an initial value for μ_1 at random from the dataset
- For each remaining μ_i K-Means++ draws a value according to the probabilities:

$$\frac{(D(x)^2)}{(\sum_{x \in X} D(x)^2)}$$

  - Where X is the entire dataset and D is the distance to the potential μ_i

Formal Spectral Clustering Description

- Spectral Clustering(X, K)

  1. Compute similarity matrix W from X

  2. Compute the Laplacian, L, for W

  3. Find the eigenvectors and eigenvalues for L

  4. Select the first K eigenvectors, $\mu_1, \mu_2, \ldots \mu_k$ and form a matrix U with the K eigenvectors as columns

  5. Run a clustering algorithm with U as the data set

     – We use K-Means

  6. Assign each data point in X to the cluster that the corresponding row in U was assigned to

Stability

- The stability of spectral clustering is highly dependent on the parameters involved in creating the similarity matrix
- Also, stability depends on the data itself, sparse or seemingly random data sets may perform badly
- As stated in the previous section K-Means is susceptible to instability if initial values are chosen badly
  - Using K-Means++ provides a high degree of stability among results

Convergence

- Spectral clustering is guaranteed to converge to a solution although no guarantees regarding optimality can be made

Accuracy

- Clustering accuracy is difficult to measure numerically
- For this study we use sight-checking and ensure data presented supports this form of checking
- Since clustering is often used on raw data points it can be quite intuitive which clusters are correct when looking at the data
  - The problem is hard for computers and easy for humans

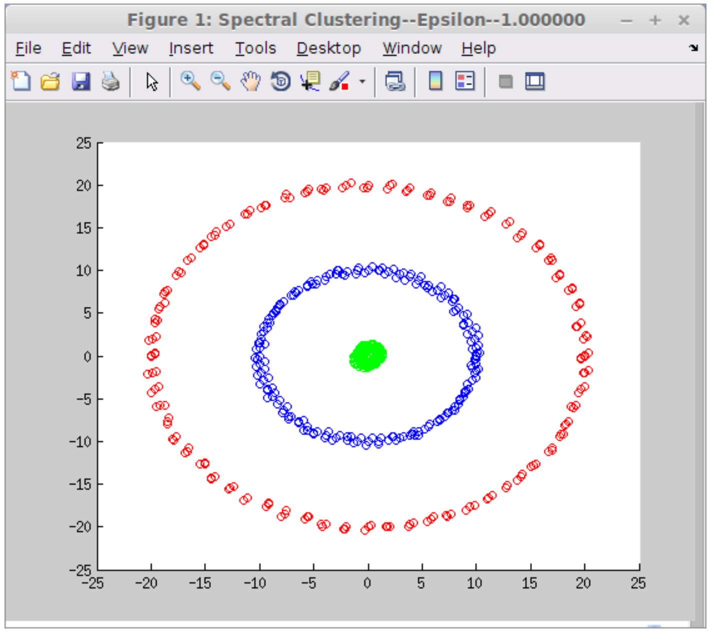
Performance Analysis

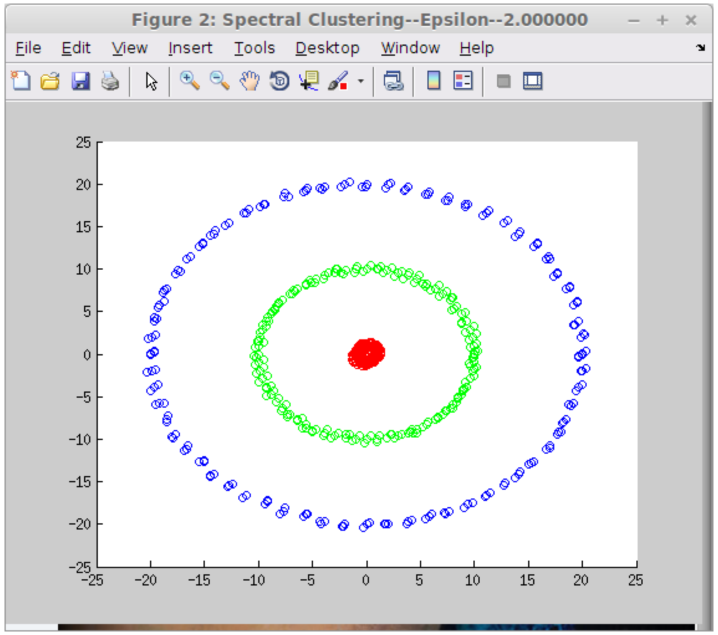
- Spectral clustering was analyzed by running it on 2d and 3d data sets
- Its results were compared to k-means without any spectral transformation
- Many of the results show good performance and others quite poor
  - This illustrates the importance of selecting the correct clustering algorithm for the correct data set

Evaluation

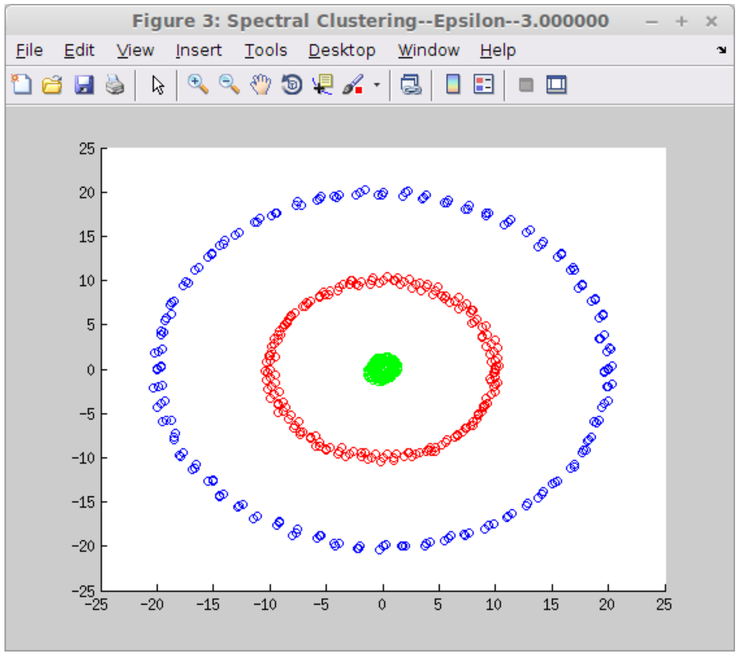
- I compared K-Means to Spectral Clustering on non-convex data
-

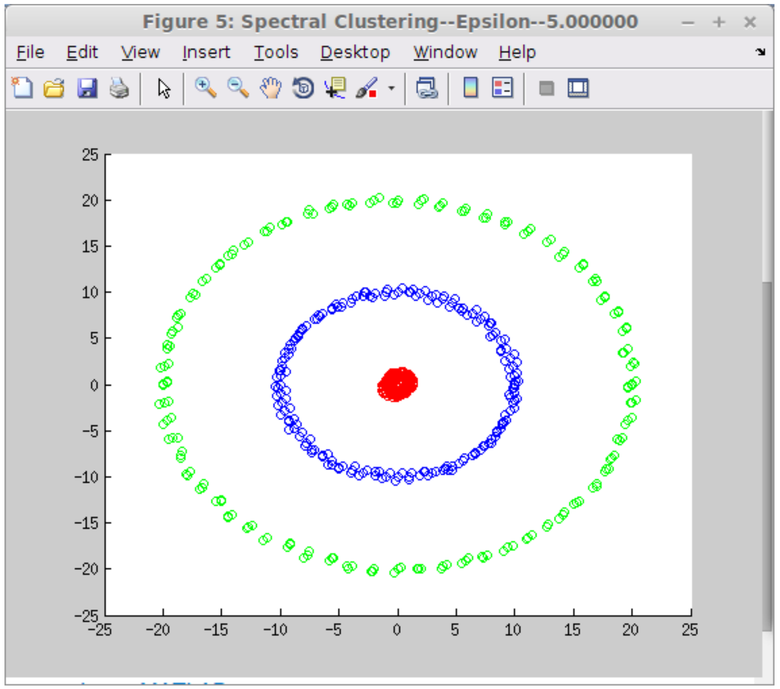
Epsilon Neighborhood



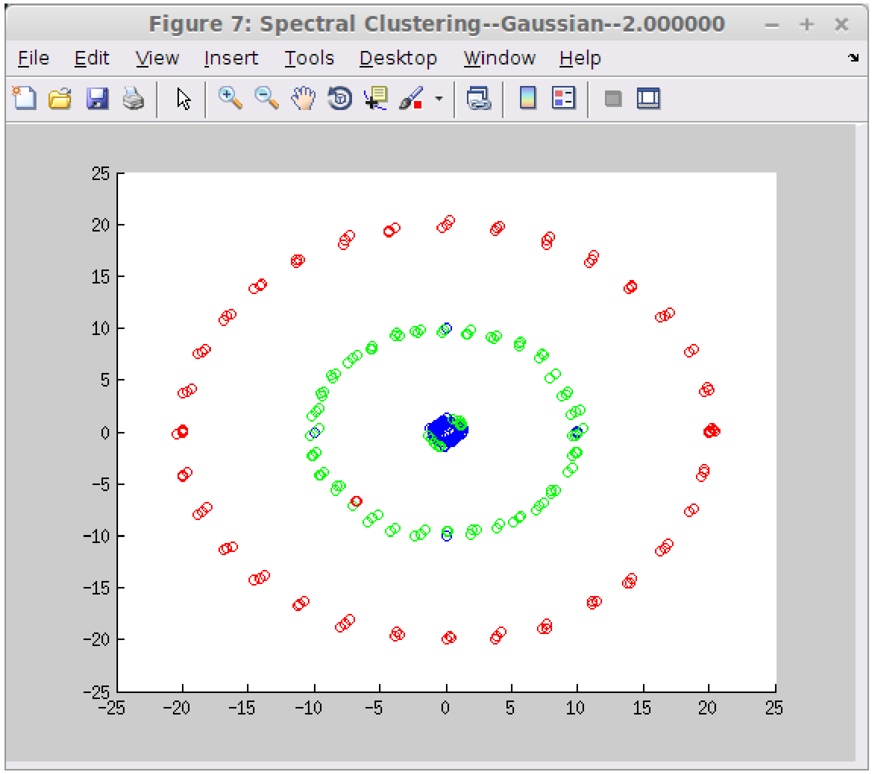


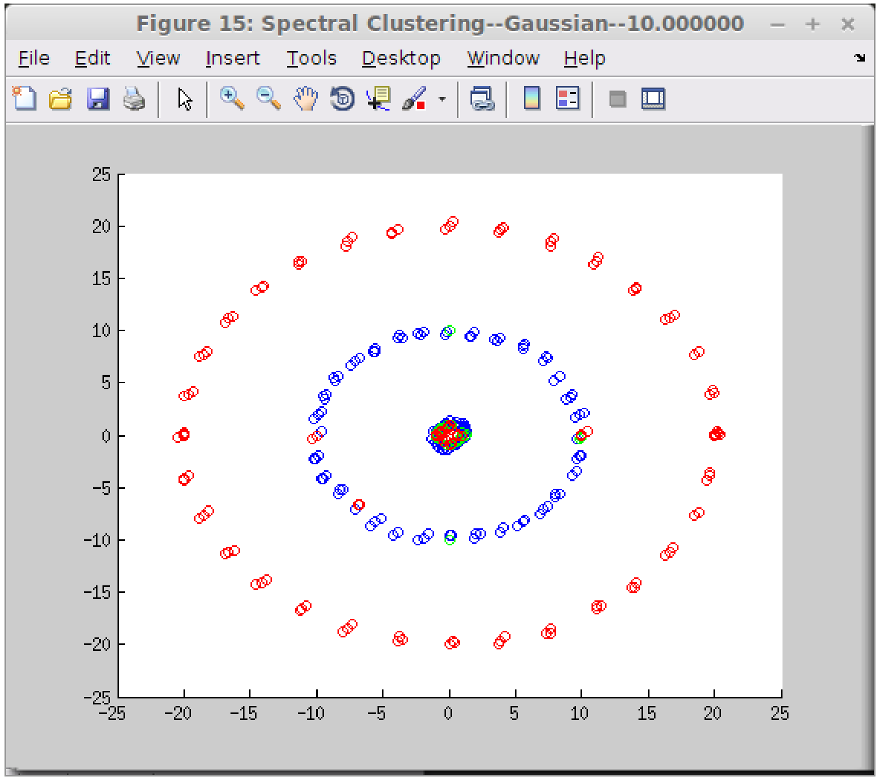
Epsilon Neighborhood



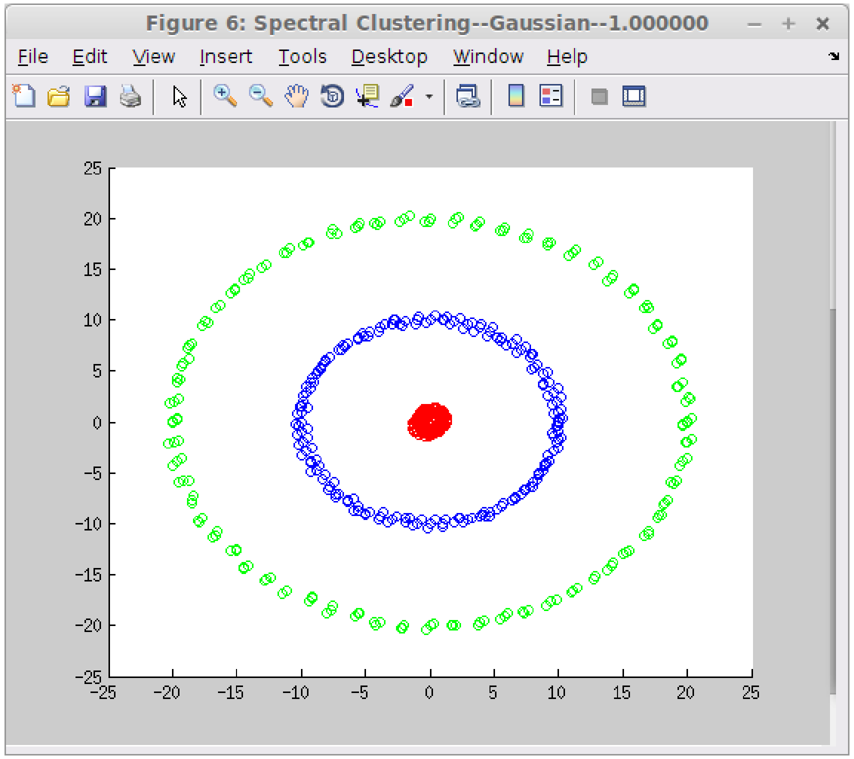


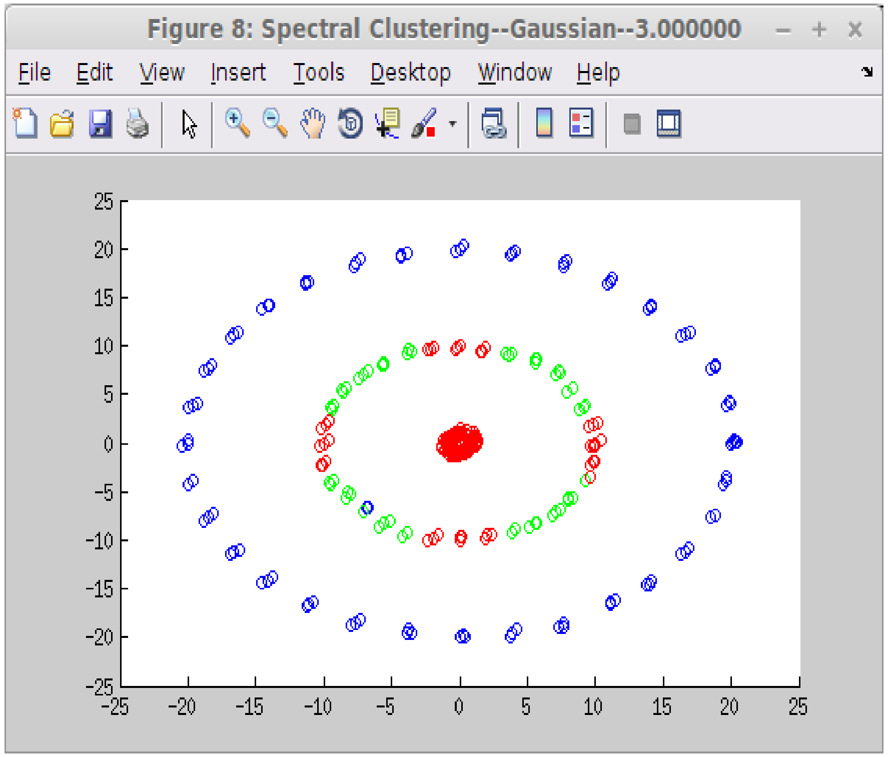
Gaussian Kernel





Gaussian Kernel





K-Means

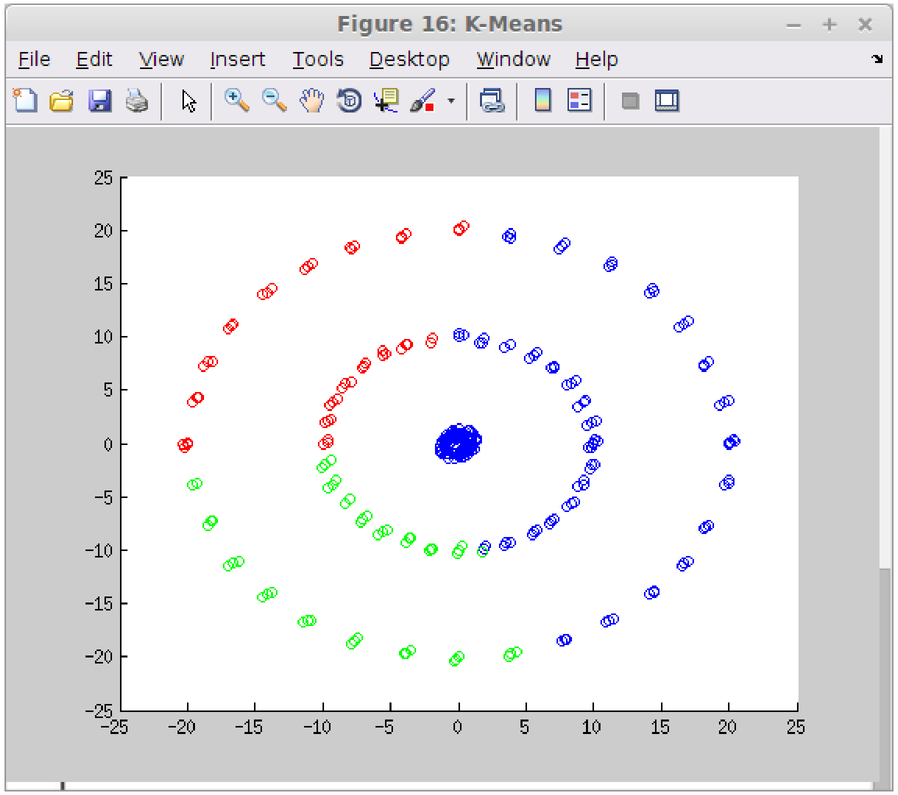


Figure 16: K-Means
File Edit View Insert Tools Desktop Window Help
25
20
15
10
5
0
-5
-10
-15
-20
-25
-25 -20 -15 -10 -5 0 5 10 15 20 25

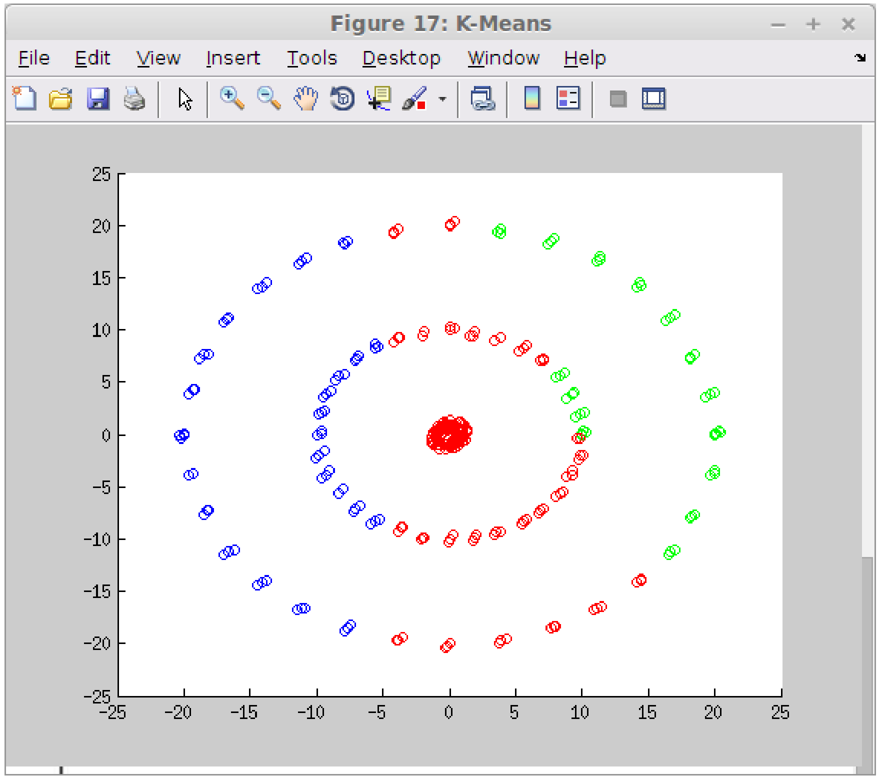


Figure 17: K-Means
File Edit View Insert Tools Desktop Window Help
25
20
15
10
5
0
-5
-10
-15
-20
-25
-25 -20 -15 -10 -5 0 5 10 15 20 25

Comments on Results

- As expected K-Means is outperformed by spectral clustering in this instance of data being more closely associated by its relative relationship than by its global one
- The epsilon neighborhood method is far more stable, providing identical clusterings across values from 1-5
- The Gaussian kernel method provided noisy clusters at best and totally confused clusters at worst

Conclusion

- Spectral clustering outperforms K-Means for cases where data is better described by its relationship to surrounding, or similar, data points
- Spectral
- As the data collected by researchers becomes more complex we expect to see more irregular data of this fashion which will motivate the use of spectral clustering and algorithms like it in the future

References

- Ulrike von Luxburg, “A Tutorial on Spectral Clustering”, Max Planck Institute for Biological Cybernetics, Tubingen, Germany
- David Arthur and Sergei Vassilvitskii, “k-means++: The Advantages of Careful Seeding”
- Madeleine Udell, “Introduction to Spectral Graph Theory”
- “The Gaussian Kernel”, http://www.stat.wisc.edu/~mchung/teaching/MIA/reading/diffusion.gaussian.kernel.pdf.pdf
- “The QR Algorithm”, http://www.math.iit.edu/~fass/477577_Chapter_11.pdf
- Richard L. Burden and J. Douglas Faires “Numerical Analysis” 8th Edition, Thomson

Source Code

- All source code written for this project was done by me, Ethan Gaebel, can be found on BitBucket at:

  https://bitbucket.org/egaebel/spectral-clustering-math-4404-project/src