CSE 519: Data Science
Steven Skiena
Stony Brook University

Lecture 20: Clustering

Supervised / Unsupervised Learning

The methods discussed so far assume class labels or target variables in the training data.

Unsupervised methods try to find structure in the data, by providing labels (clusters) or values (rankings) without a trusted standard.

Semi-supervised methods amplify small amounts of labeled data into more.

Clustering

Clustering is the problem of grouping points by similarity.

Often elements come from a small number of “sources” or “explanations”, and clustering is a good way to reveal these origins.

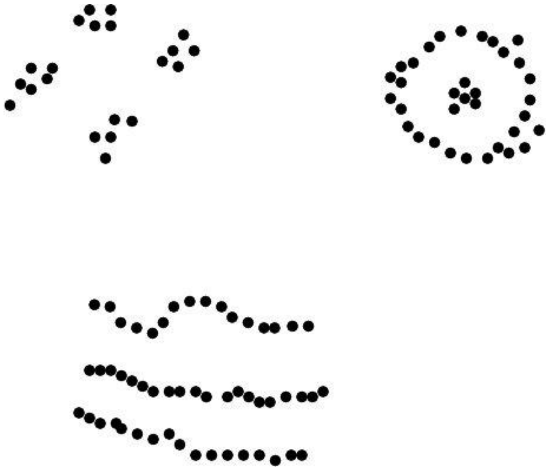
Similarity is defined by some underlying distance function/metric.

How Many Clusters Do You See?

Clustering is an inherently ill-defined problem since they upon context and the eye of the beholder.

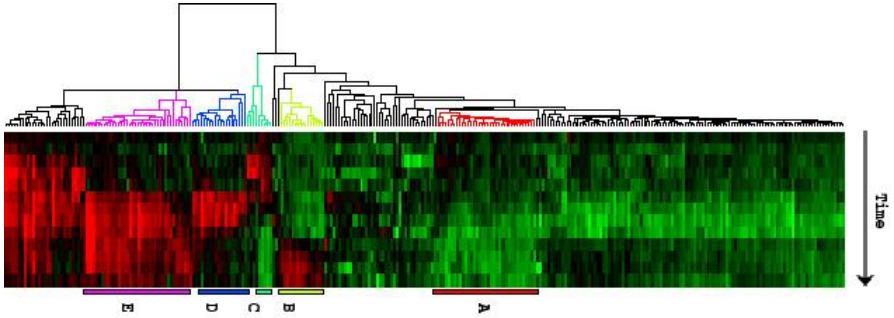
How many do you see?

Compact, circular clusters are natural but not universal.



Clustering Gene Expression Data

- Clustering the columns groups genes active in the same phases of the cell cycle.
- Biological clusterings are often associated with dendograms or phylogenic trees.



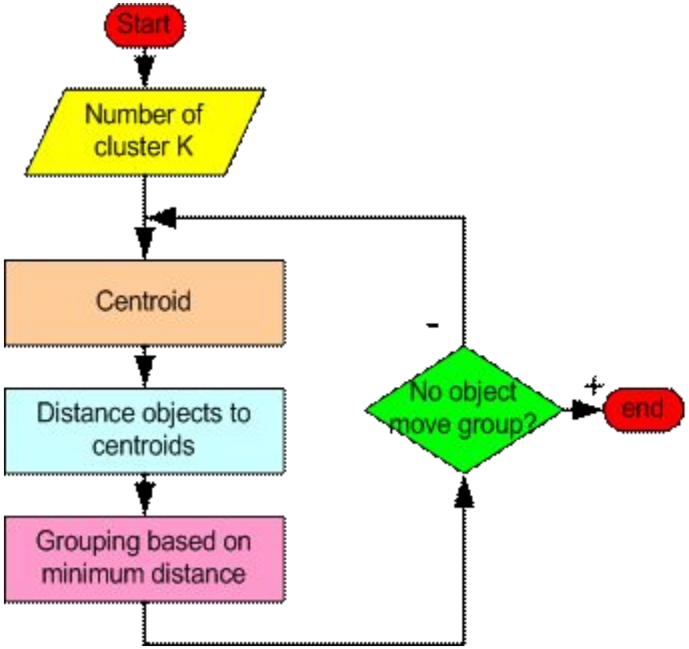
Why Clustering?

- Hypothesis development -- how many distinct populations are there in your data?
- Modeling over smaller groups -- build separate predictive models for each cluster.
- Data reduction -- replace/represent each cluster of items by its centroid.
- Outlier detection -- which items are far from cluster centers, or stuck in tiny clusters?

K-Means Clustering

Pick *k* points as centers, then assign all examples to the nearest center.

Recalculate the center, and repeat until sufficiently stable.


Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

Running Time

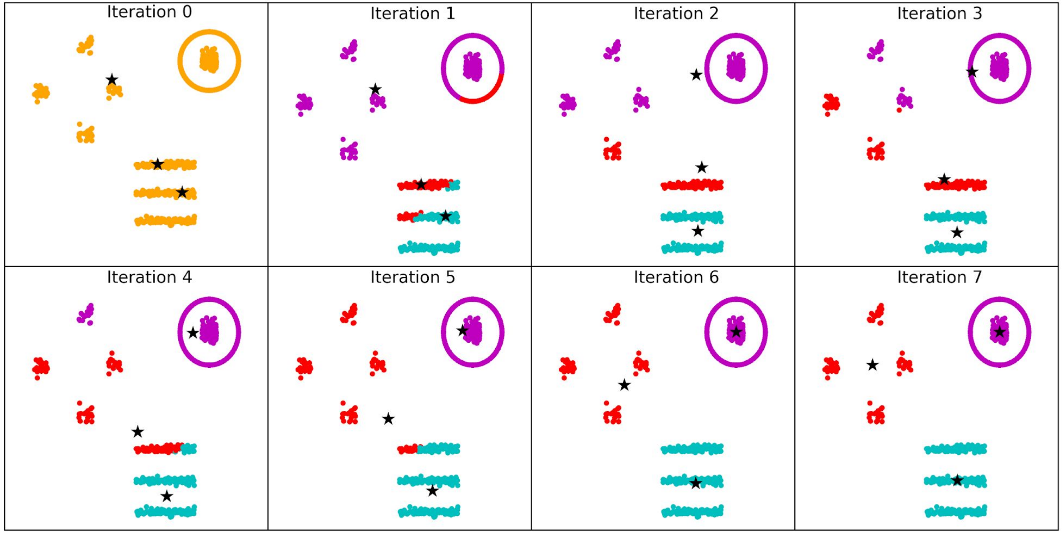
Finding the nearest neighbor among *k* cluster centers takes *O(kd)* per point, or *O(nkd)*.

Finding the centroids of each of k clusters takes *O(nd)* per cluster, or *O(nkd)*.

The number of iterations is usually small, but can be exponential, bounded by the number of partitions since we end when repeat a partition.

K-Means Clustering Example

It can get stuck in local optima, but generally does pretty well.

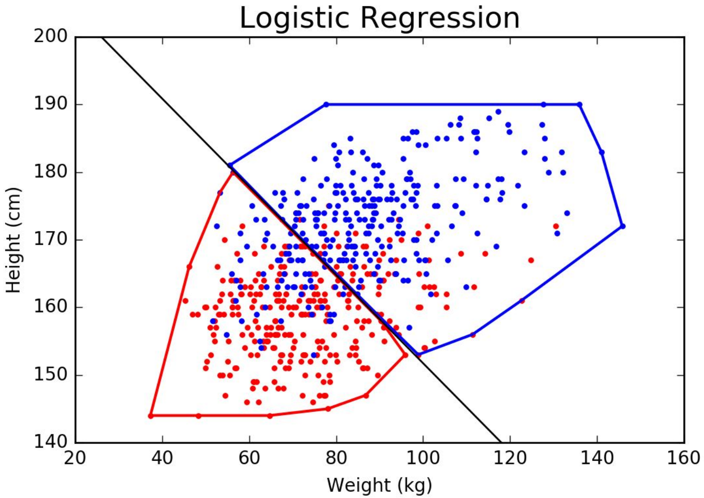


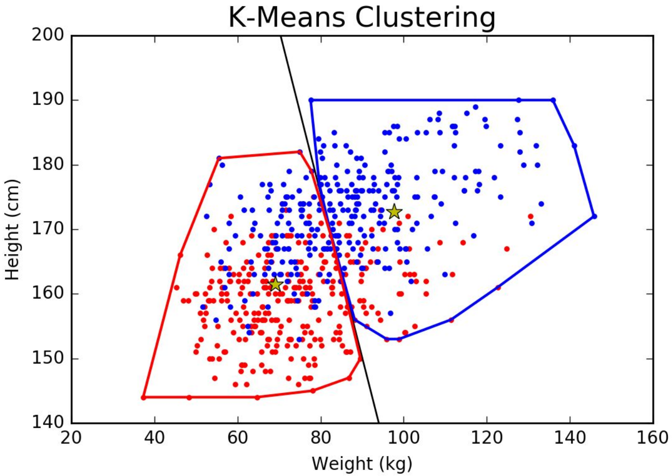
K-Means vs. Logistic Regression

K-means: 240w / 112m red, 174m / 54w blue

Logistic: 229w / 63m red, 223m / 65w blue

But K-means was unsupervised!





Centroids or Center Points?

Centroids are not well defined in clustering non-numerical attributes such as color or gender.

$$C_d = \frac{1}{|S'|} \sum_{p \in S'} p[d]$$

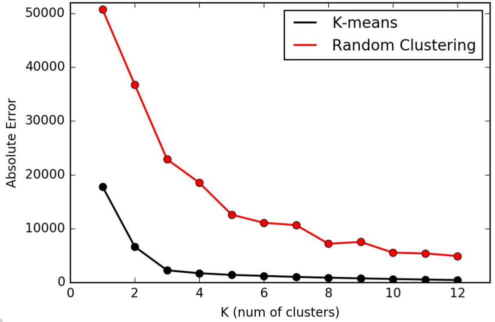
Using the centermost input example as center means we can run k-means so long as we have a meaningful distance function.

How Many Clusters?

The “right” number of clusters is usually unknown prior to clustering.

The SQE of points from their center should generally decrease when adding clusters.

But the SQE should decrease slowly once exceeding the right number of clusters

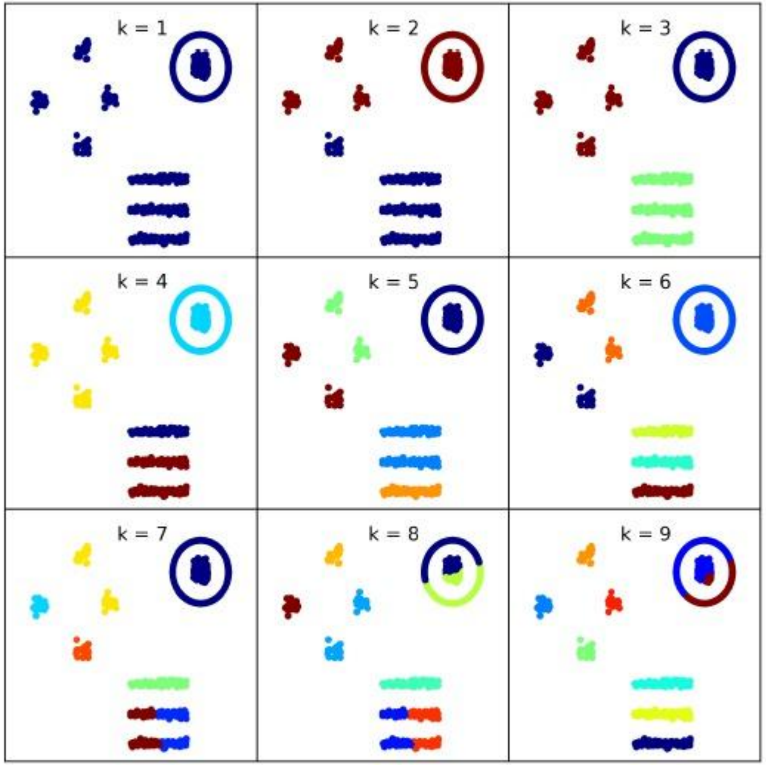


Limitations of K-means

K-means wants round clusters, so it has trouble with:

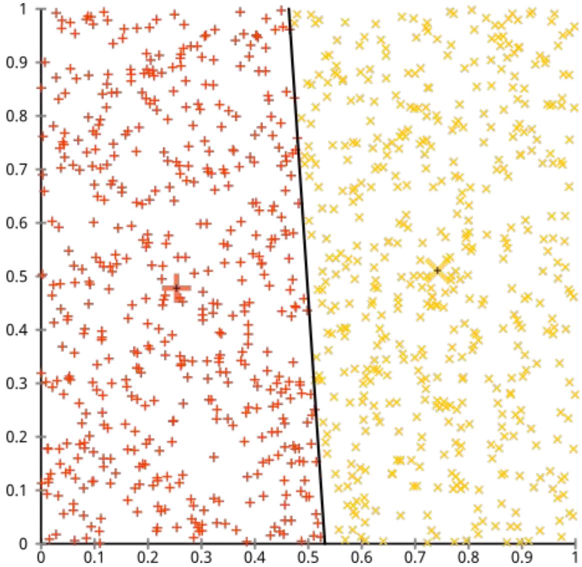
- nested clusters, and
- long thin clusters.

Repeated runs help avoid local optima.

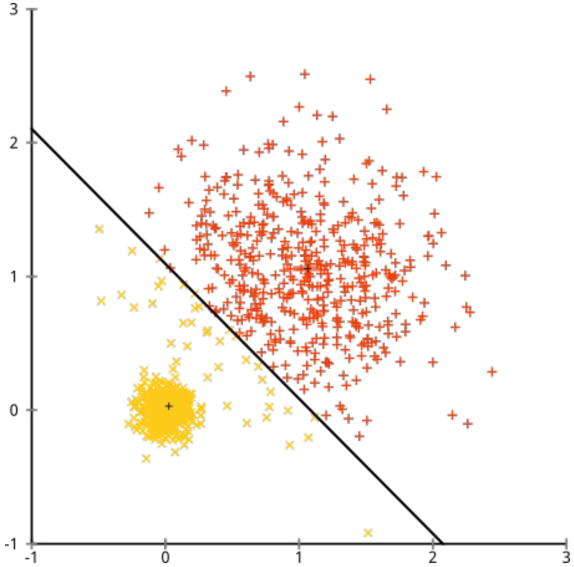


Bad Cases for K-Means

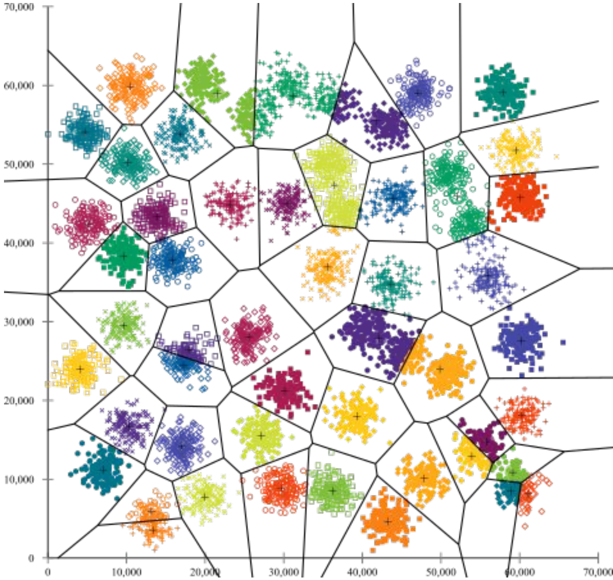
Uniform points



Disparate variances



Local Minima (k=50)



Expectation Maximization (EM)

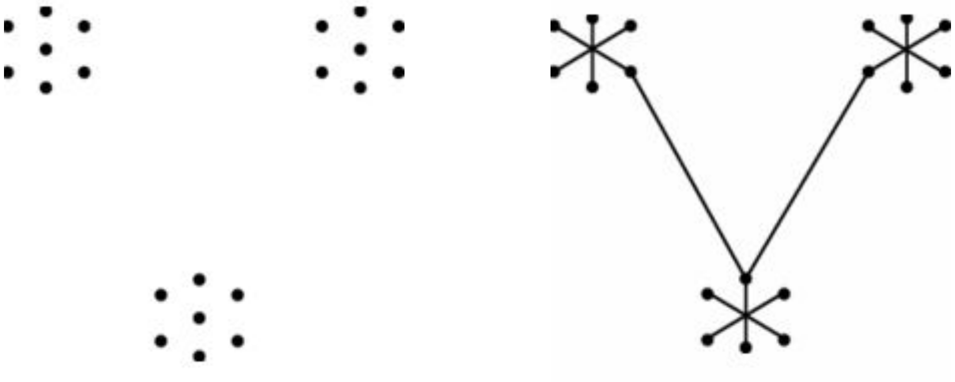
K-means is representative of a class of EM algorithms with two logical steps:

- Assign points to estimated clusters (E-step)
- Use assignments to improve parameter estimates (M-step)

Consider a semi-supervised classifier from few labeled examples but many unlabeled.

Agglomerative Clustering

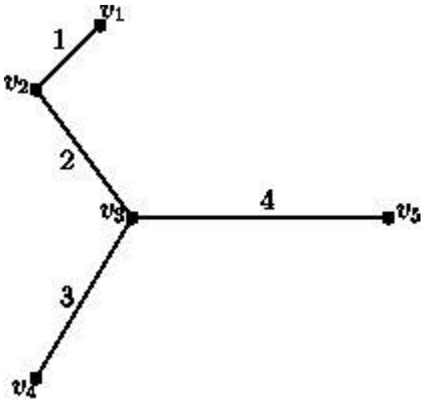
These bottom-up methods merge repeatedly merge the two nearest clusters.

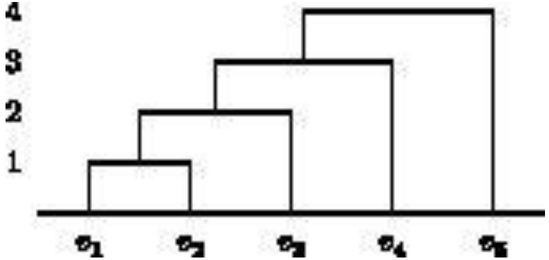


Minimum Spanning tree = single-link clustering

Kruskal’s Algorithm and Dendograms

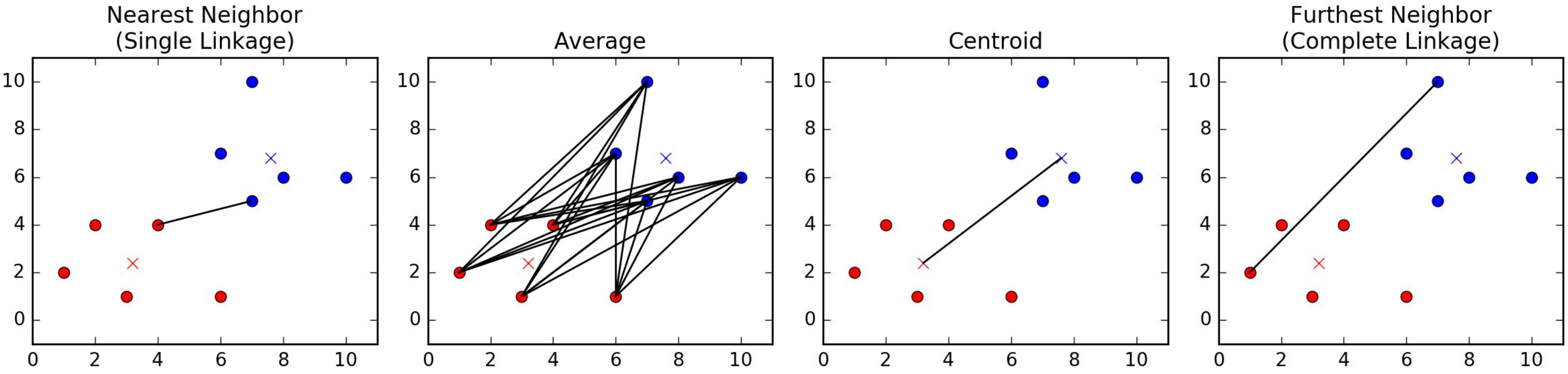
Dendograms are constructed by reflecting the height of the merge as the edge in question, and permuting the vertices so merges take place between neighboring clusters.





What Does Closest Cluster Mean?

The pointwise distance metric is not enough to define distance between clusters:



Linkage Criteria

Nearest neighbor (single link, MST)

$$d(C_1, C_2) = \min_{x \in C_1, y \in C_2} ||x - y||$$

Average link (more robust but expensive)

$$d(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{x \in C_1} \sum_{y \in C_2} ||x - y||$$

Nearest centroid (faster but still robust)

Furthest link (funny but keeps clusters round)

$$d(C_1, C_2) = \max_{x \in C_1, y \in C_2} ||x - y||$$

Agglomerative Clustering Complexity

The run time for clustering is the number of merge steps (n-1) times the cost per merge.

The linkage criteria trades off between speed (O(n) to O(n^2) per iteration) and robustness.

Each merge changes the cost for only 2n of up to n^2 cluster pairs, so we can avoid recomputation.

Advantages of Cluster Hierarchies

- Organization of clusters and subclusters
- Visualization of the clustering process
- Natural measure of distance between clusters.
- Efficient classification of new items: compare against centroids as we march down the tree, in time proportional to height.

Which Clustering Algorithm To Use?

There are an enormous number of possible clustering algorithms, but much more important decisions are:

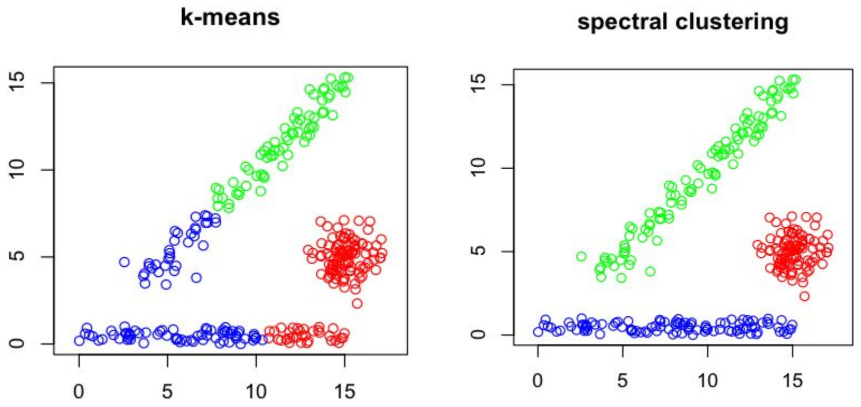
- using the right distance function
- properly normalizing your variables
- appropriately visualizing the final clusters to know whether they are good.

Seeking Connected Clusters

K-means finds centroids and spherical clusters, but not skinny or nested ones.

Single-link agglomerative clustering finds skinny clusters.

But it is easily fooled into merging two clusters by a single close point pair.



Similarity Graphs

Each entry S[i,j] in a similarity matrix S scores how much alike elements i and j are.

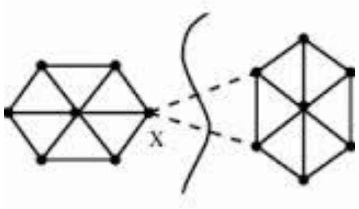
It is essentially an inverse of a distance matrix, and can be computed: $S_{ij} = \exp(-\beta||x_i - x_j||)$

Thus similarity ranges from 0 to 1.

This weighted graph could be made sparse by setting all small terms to zero.

Cuts in Graphs

Clusters in similarity graphs should have small / light edges spanning them.



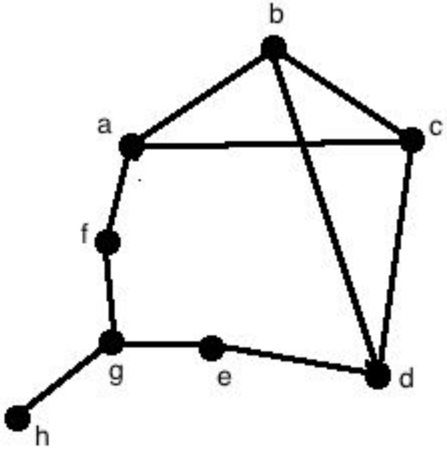
Ideally clusters will have a high weight (ie. sum of internal edges) and a small cut.

Finding Cuts in Graphs

Network flow methods can find the minimum cut in a graph, but not one whose internal weight is large.

The minimum cut will naturally tend to separate isolated vertices, not clusters.

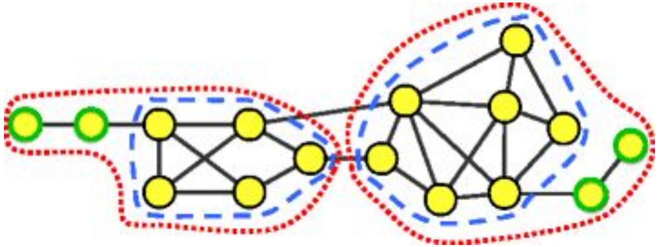
The problem of graph partitioning which seeks balanced clusters is NP-complete, motivating heuristics and other approaches.



Conductance and Eigenvectors

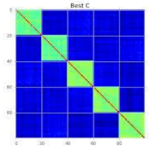
The *conductance* of a cluster *C* is defined as the weight of the cut edges over the weight of the internal edges: *W’(C)/W(C)*.

Low conductance clusters are desirable.

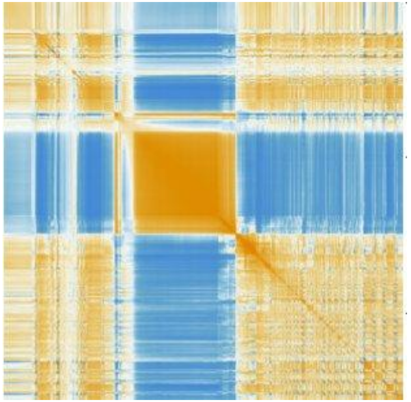


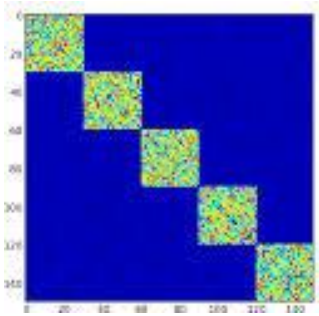
Blocky Matrices and Eigenvectors

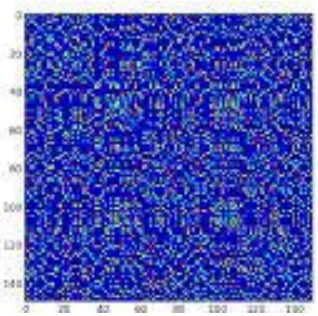
Low conductance clusters correspond to blocky similarity matrices.

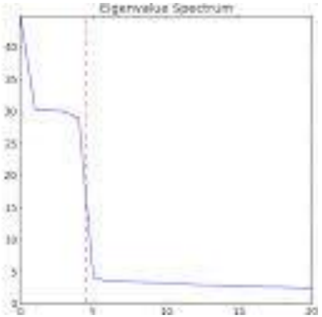


Recall that blocky matrices come from Eigenvector decomposition.





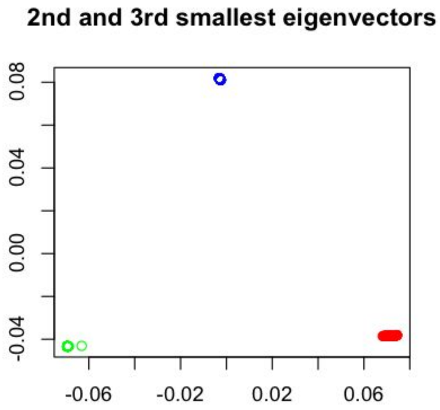




Spectral Clustering

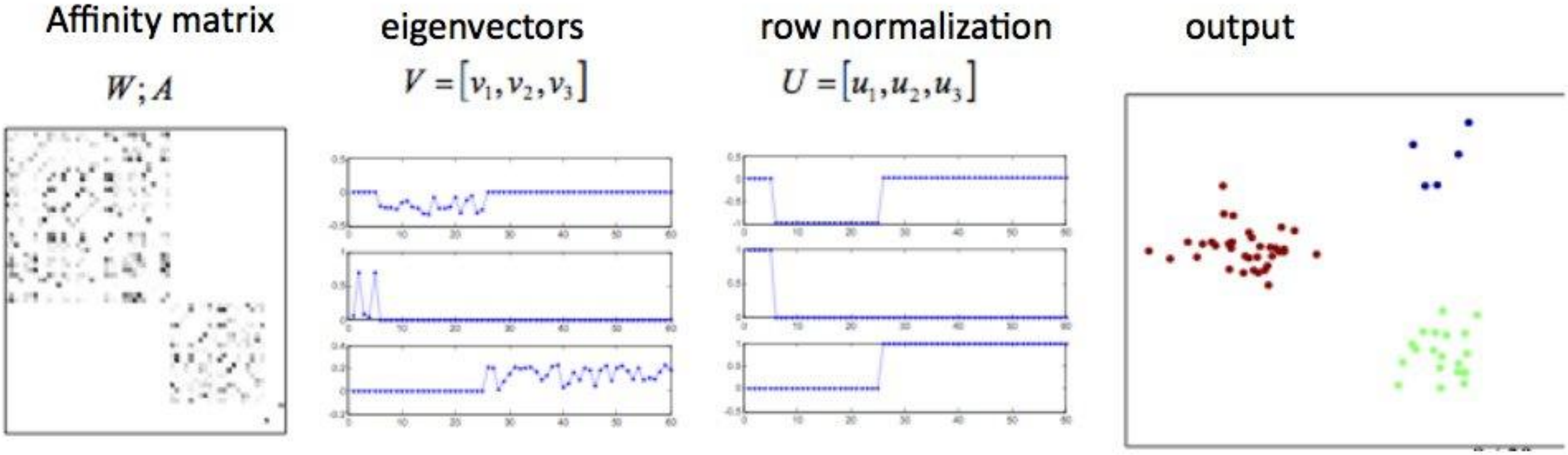
- From similarity matrix *W* computes *L=D-W* (Laplacian) where D is the degree matrix.
- Small Eigenvectors of L define features for each element.

Performing k-means clustering on this transformed feature space recovered good clusters.



3-class Spectral Example

The eigenvectors define features to cluster on.



Singular Value Decomposition

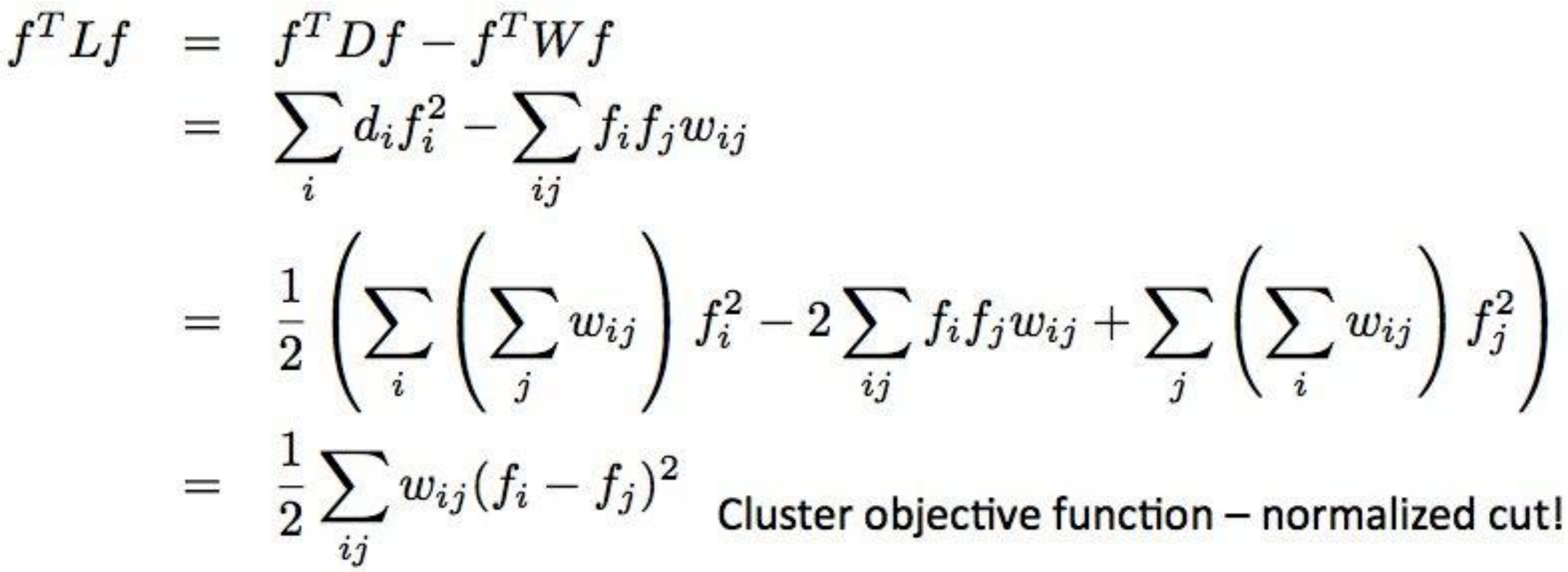
The covariance matrix can be represented by summing the spectrum of Eigenvalues/vectors.

However, just using the vectors associated with the largest Eigenvalues gives a good approximation.

This dimension reduction method is very useful to produce smaller, more effective feature sets.

Why Does Spectral Clustering Work?

Small value eigenvectors f define small cuts.



$$
\begin{aligned}
f^T L f &= f^T D f - f^T W f \\
&= \sum_i d_i f_i^2 - \sum_{ij} f_i f_j w_{ij} \\
&= \frac{1}{2}\left(\sum_i \left(\sum_j w_{ij}\right) f_i^2 - 2\sum_{ij} f_i f_j w_{ij} + \sum_j \left(\sum_i w_{ij}\right) f_j^2\right) \\
&= \frac{1}{2}\sum_{ij} w_{ij}(f_i - f_j)^2
\end{aligned}
$$

Cluster objective function – normalized cut!