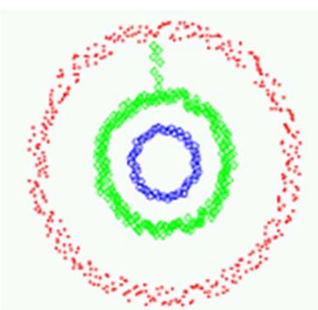
Advanced Introduction to Machine Learning

10715, Fall 2014

Spectral Clustering

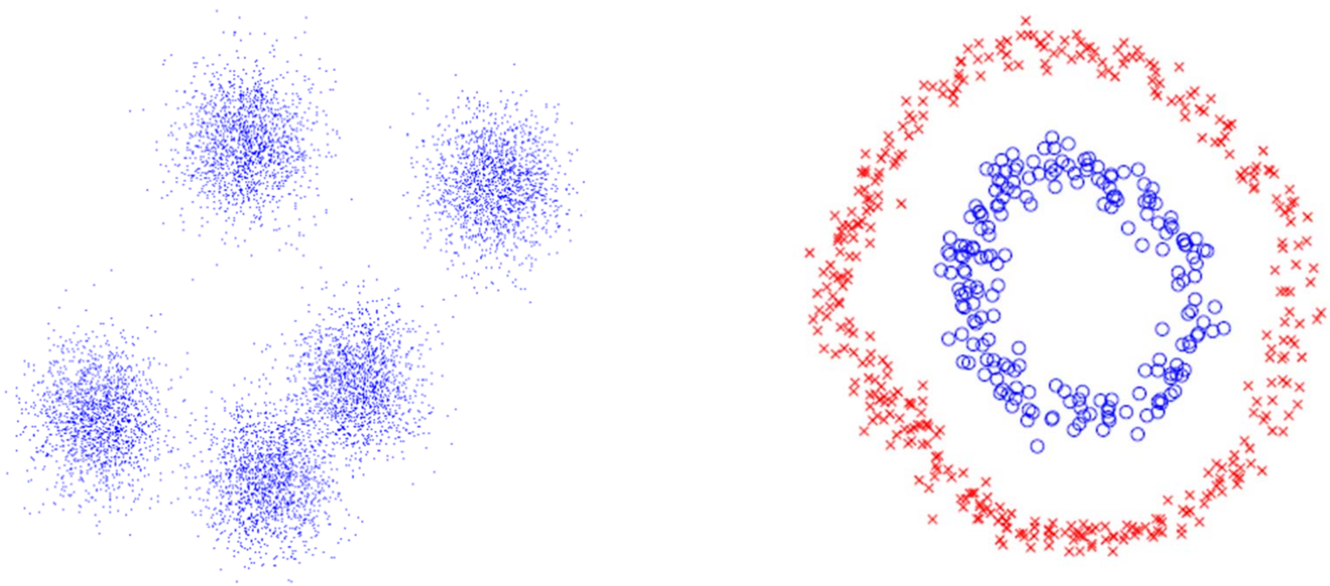
Eric Xing

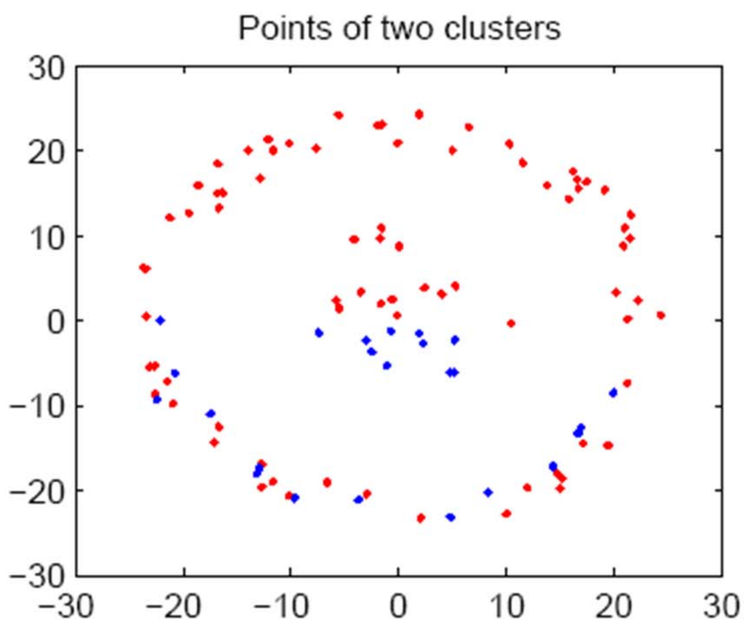
Lecture 20, November 17, 2014

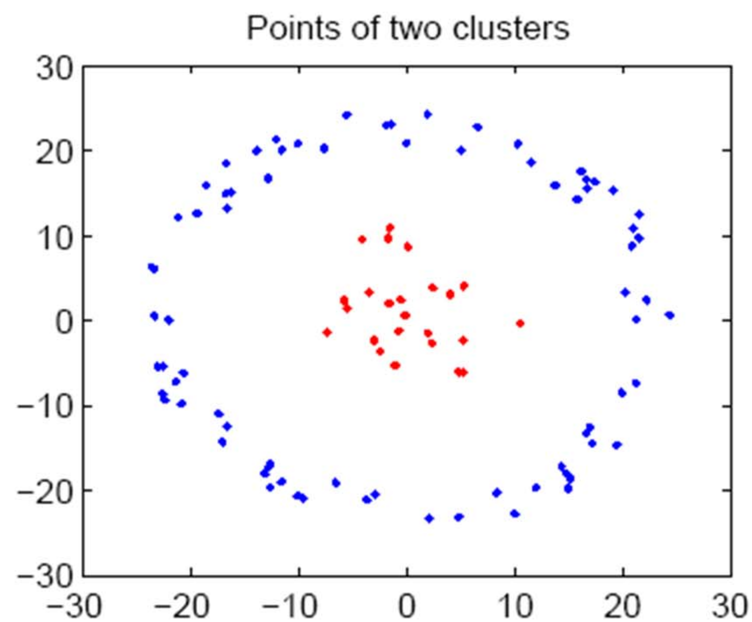


Reading:

Data Clustering

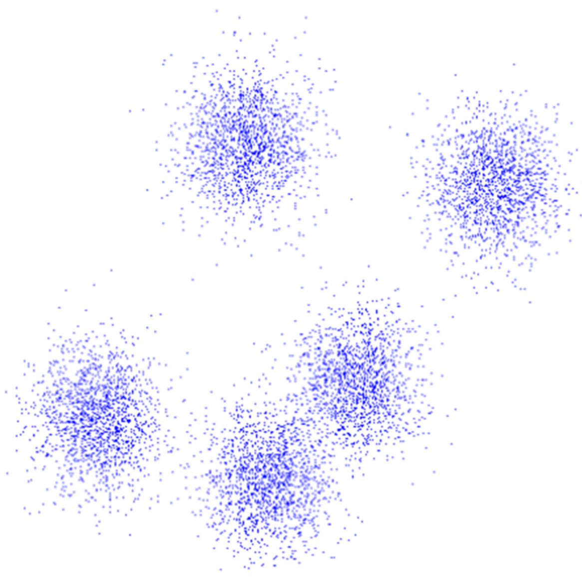



Points of two clusters
30
20
10
0
−10
−20
−30
−30
−20
−10
0
10
20
30

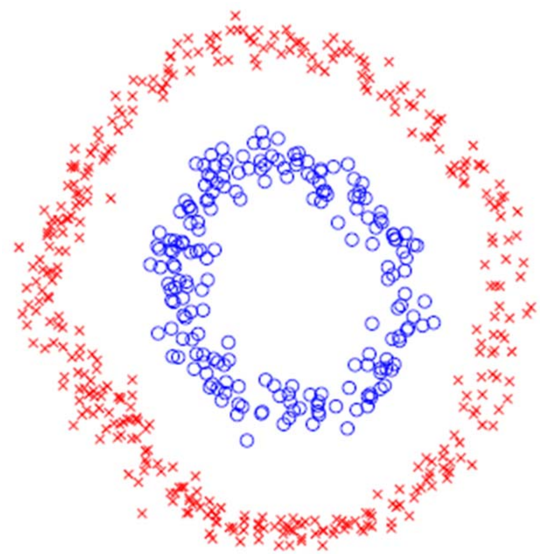

Points of two clusters
30
20
10
0
−10
−20
−30
−30
−20
−10
0
10
20
30

Data Clustering

- Two different criteria
  - Compactness, e.g., k-means, mixture models
  - Connectivity, e.g., spectral clustering

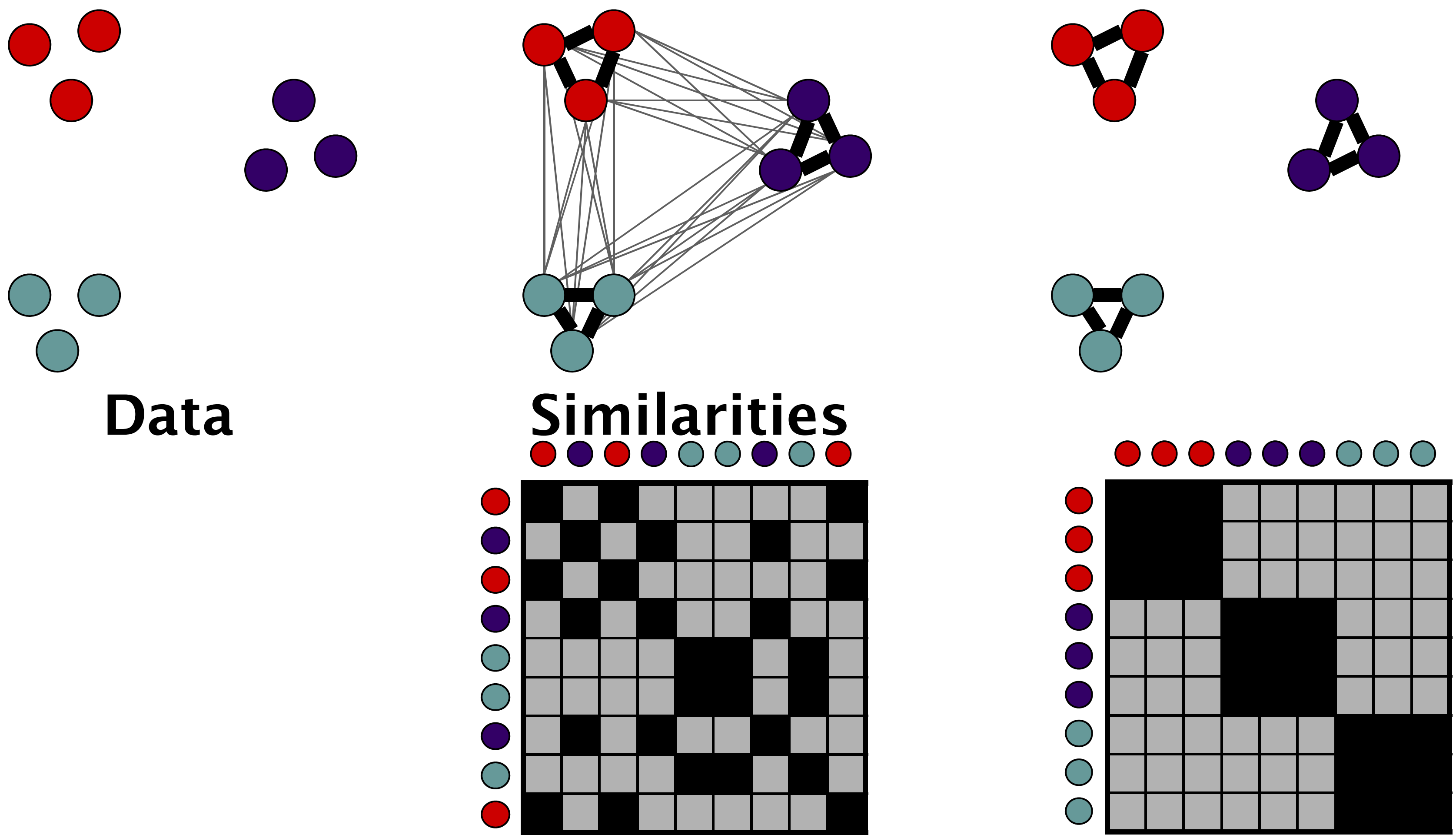


Compactness



Connectivity

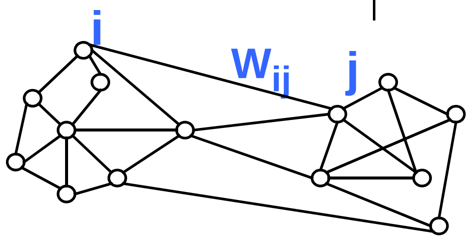
Spectral Clustering



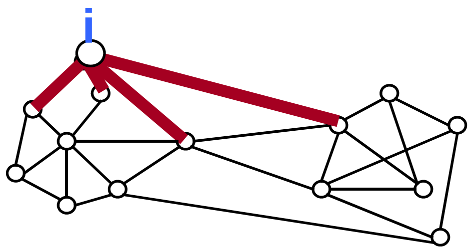
Weighted Graph Partitioning

- Some graph terminology
  - Objects (e.g., pixels, data points)

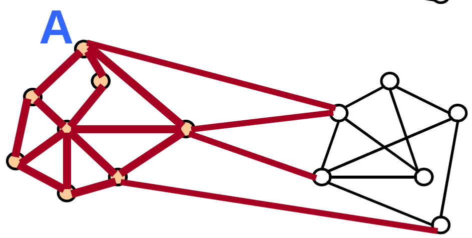
    $i \in I$ = vertices of graph G



  - Edges (ij) = pixel pairs with $W_{ij} > 0$



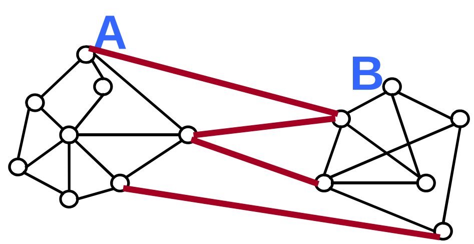
  - Similarity matrix $\mathbf{W} = [\ W_{ij}\ ]$



  - Degree

    $d_i = \Sigma_{j \in G} S_{ij}$

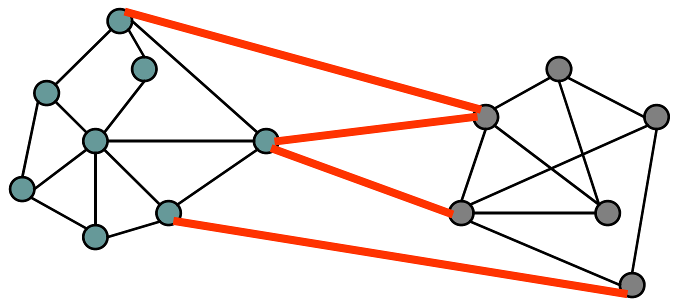
    $d_A = \Sigma_{i \in A} d_i$ degree of A $\subseteq$G



  - Assoc(A,B) = $\Sigma_{i \in A} \Sigma_{j \in B} W_{ij}$

Cuts in a Graph

- (edge) cut = set of edges whose removal makes a graph disconnected



- weight of a cut:

  cut(A, B) = $\Sigma_{i\in A}\,\Sigma_{j\in B}\,W_{ij}$=Assoc(A,B)

- Normalized Cut criteria: minimum cut(A,Ā)

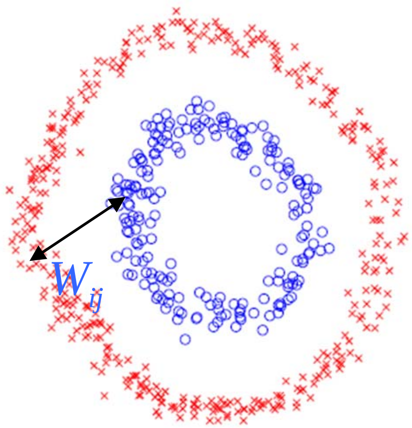
$$\mathrm{Ncut}(A,B) = \frac{\mathrm{cut}(A,B)}{d_A} + \frac{\mathrm{cut}(A,B)}{d_B}$$

More generally:

$$\mathrm{Ncut}(A_1, A_2 \ldots A_k) = \sum_{r=1}^{k} \left(\frac{\sum_{i\in A_r,\, j\in V\setminus A_r} W_{ij}}{\sum_{i\in A_r,\, j\in V} W_{ij}} \right) = \sum_{r=1}^{k} \left(\frac{\mathrm{cut}(A_r, \bar{A}_r)}{d_{A_r}} \right)$$

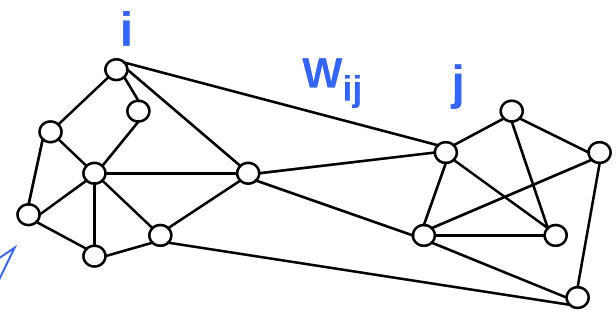
Graph-based Clustering

- Data Grouping



$$W_{ij} = f(d(x_i, x_j))$$

- Image sigmentation



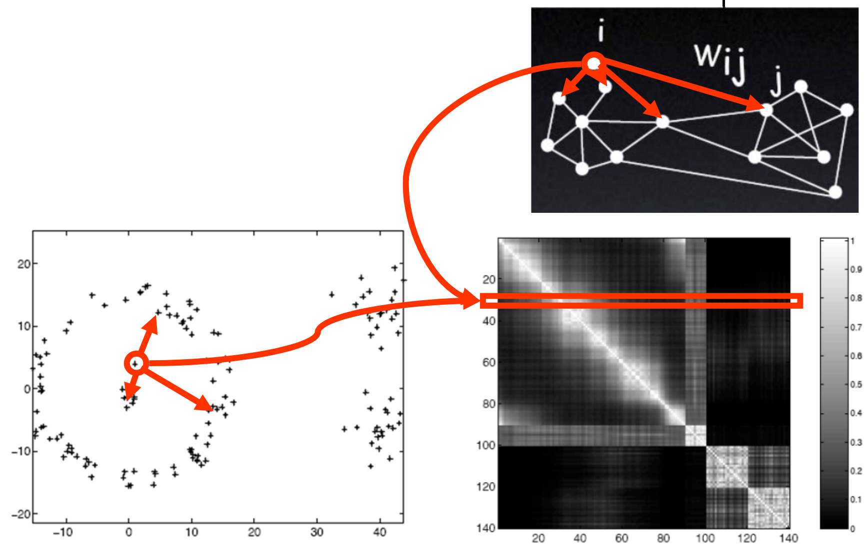
$G = \{V,E\}$

- Affinity matrix: $W = [w_{i,j}]$
- Degree matrix: $D = \mathrm{diag}(d_i)$
- Laplacian matrix: $L = D - W$
- (bipartite) partition vector:

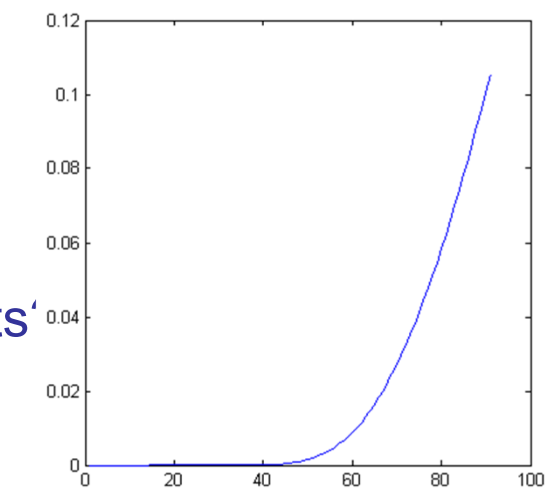
$$x = [x_1, \ldots, x_N]$$
$$= [1,1,\ldots 1,-1,-1,\ldots -1]$$

Affinity Function

$$W_{i,j} = e^{\frac{-\left\| X_i - X_j \right\|_2^2}{\sigma^2}}$$



- Affinities grow as σ grows →



- How the choice of σ value affects the results
- What would be the optimal choice for σ?

Clustering via Optimizing Normalized Cut

- The normalized cut:

$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{d_A} + \frac{\text{cut}(A,B)}{d_B}$$

- Computing an optimal normalized cut over all possible y (i.e., partition) is NP hard

- Transform Ncut equation to a matrix form (Shi & Malik 2000):

$$\min_x Ncut(x) = \min_y \frac{y^T(D-W)y}{y^T Dy}$$

Rayleigh quotient

Subject to: $y \in \{1,-b\}^n$

$$y^T D1 = 0$$

  - Still an NP hard problem

$$Ncut(\mathrm{A},\mathrm{B}) = \frac{cut(\mathrm{A},\mathrm{B})}{\deg(\mathrm{A})} + \frac{cut(\mathrm{A},\mathrm{B})}{\deg(\mathrm{B})}$$

$$= \frac{(1+x)^T(D-S)(1+x)}{k1^T D1} + \frac{(1-x)^T(D-S)(1-x)}{(1-k)1^T D1}; \quad k = \frac{\sum_{x_i>0} D(i,i)}{\sum_i D(i,i)}$$

$$= \ldots$$

Relaxation

$$\min_x Ncut(x) = \min_y \frac{y^T(D-W)y}{y^T D y}$$

Rayleigh quotient

Subject to: $y \in \{1, -b\}^n$

$$y^T D 1 = 0$$

- Instead, relax into the continuous domain by solving generalized eigenvalue system:

$$\min_y \ y^T(D-W)y, \qquad \text{s.t. } y^T D y = 1$$

- Which gives: $(D-W)y = \lambda D y$ ***Rayleigh quotient theorem***
- Note that $(D-W)\mathbf{1} = \mathbf{0}$ so, the first eigenvector is $y_0=1$ with eigenvalue 0.
- The second smallest eigenvector is the real valued solution to this problem!!

Algorithm

1. Define a similarity function between 2 nodes. i.e.:

$$w_{i,j} = e^{\frac{-\|X_{(i)} - X_{(j)}\|_2^2}{\sigma_X^2}}$$

2. Compute affinity matrix (W) and degree matrix (D).

3. Solve $(D-W)y = \lambda D y$

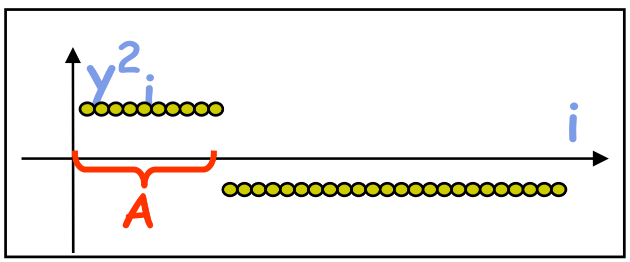
   - Do singular value decomposition (SVD) of the graph Laplacian $L = D - W$

$$L = V^T \Lambda V \quad \Rightarrow \quad y^*$$

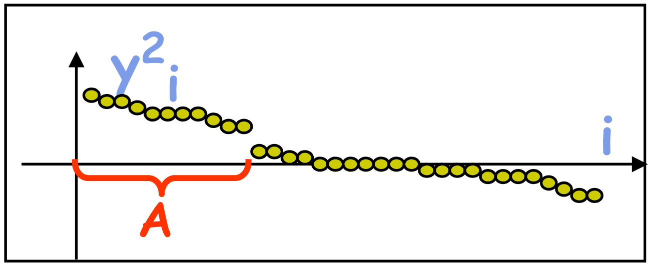
4. Use the eigenvector with the second smallest eigenvalue, y^*, to bipartition the graph.
   - For each threshold k, A_k={i | y_i among k largest element of y*}
     B_k={i | y_i among n-k smallest element of y*}
   - Compute Ncut(A_k,B_k)
   - Output $k^* = \arg\max \mathrm{Ncut}(A_k, B_k)$ and A_{k^*}, B_{k^*}

Ideally ...

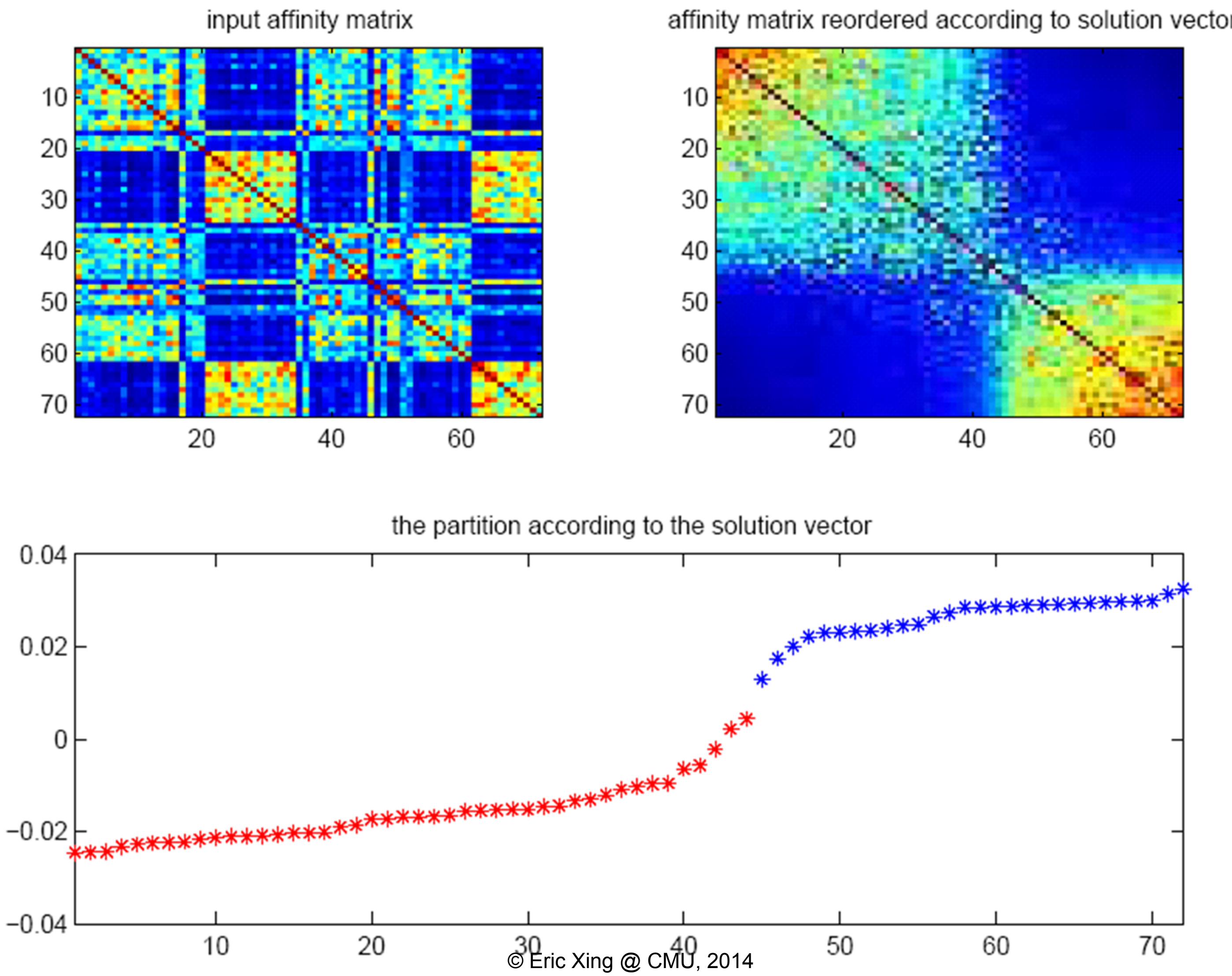
$$Ncut(A,B) = \frac{y^T(D-S)y}{y^T Dy}, \quad \text{with } y_i \in \{1,-b\},\ y^T D1 = 0.$$



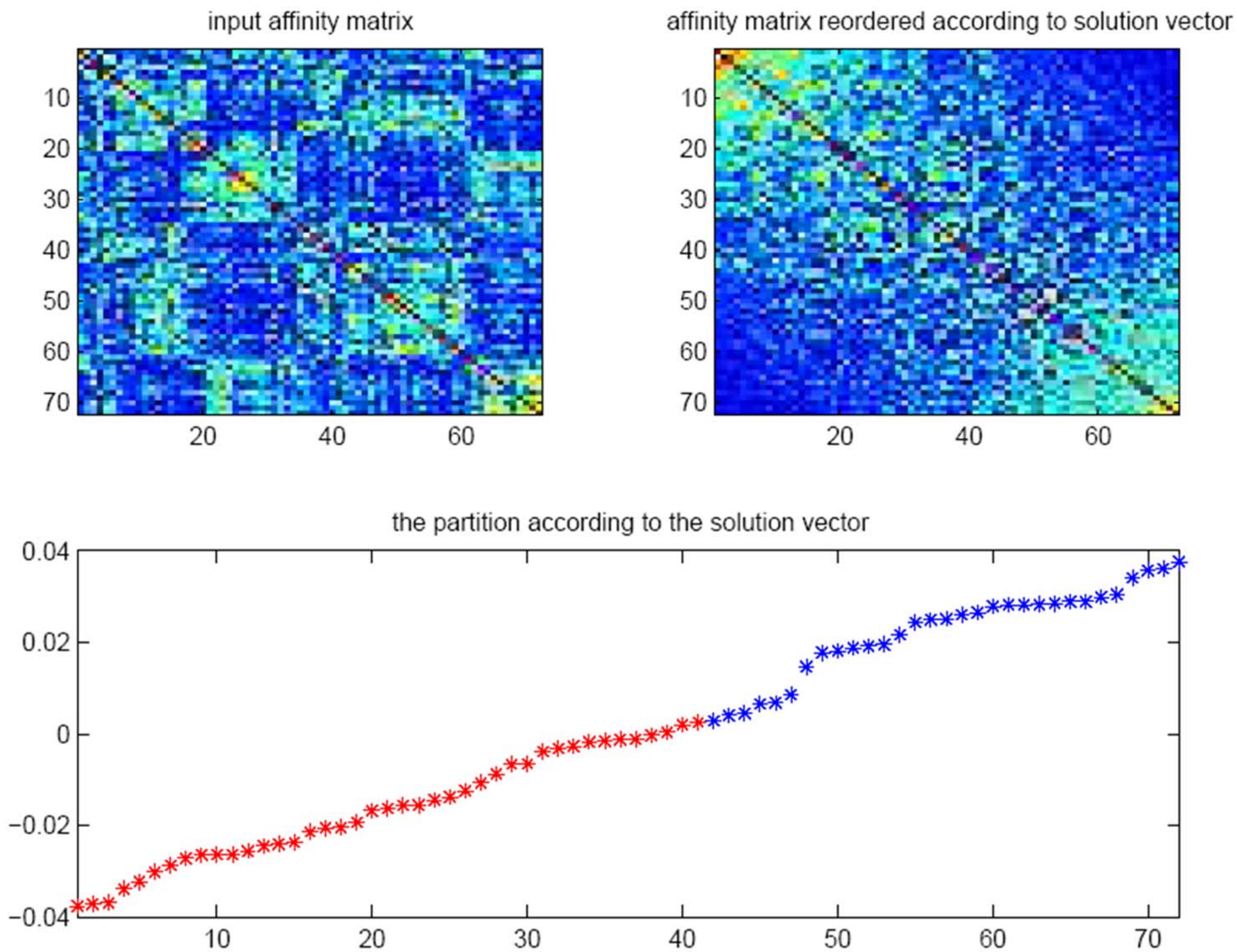
$$(D-S)y = \lambda Dy$$



Example


input affinity matrix
affinity matrix reordered according to solution vector
the partition according to the solution vector
10
20
30
40
50
60
70
0.04
0.02
0
−0.02
−0.04

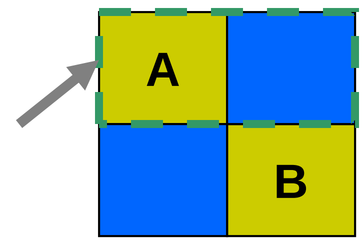
Poor features can lead to poor outcome (xing et al 2002)



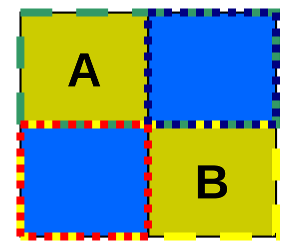
Cluster vs. block matrix

$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{d_A} + \frac{\text{cut}(A,B)}{d_B}$$

$$Degree(A) = \sum_{i \in A, j \in V} W_{i,j}$$



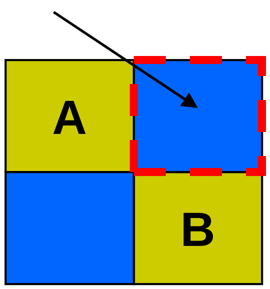
$$Ncut(A,B) = \frac{cut(A,B)}{d_A} + \frac{cut(A,B)}{d_B}$$



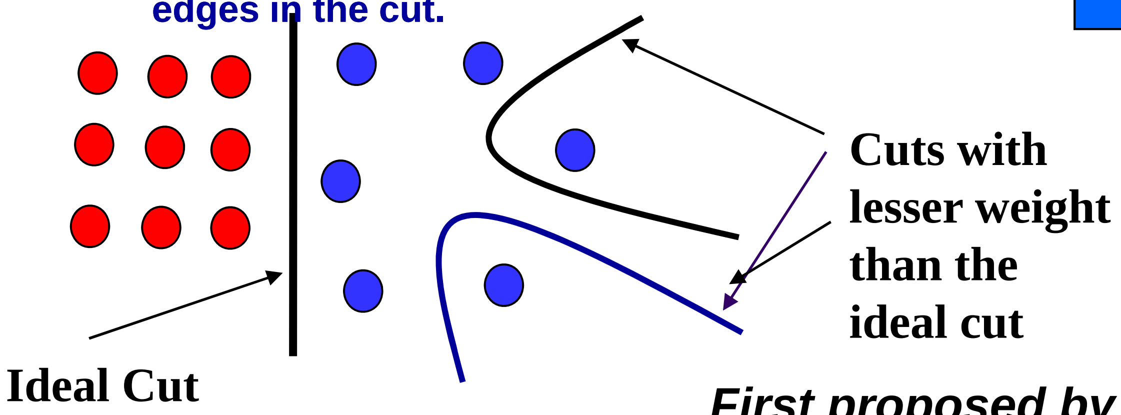
Compare to Minimum cut

- Criterion for partition:

$$\min cut(A,B) = \min_{A,B} \sum_{i \in A, j \in B} W_{i,j}$$

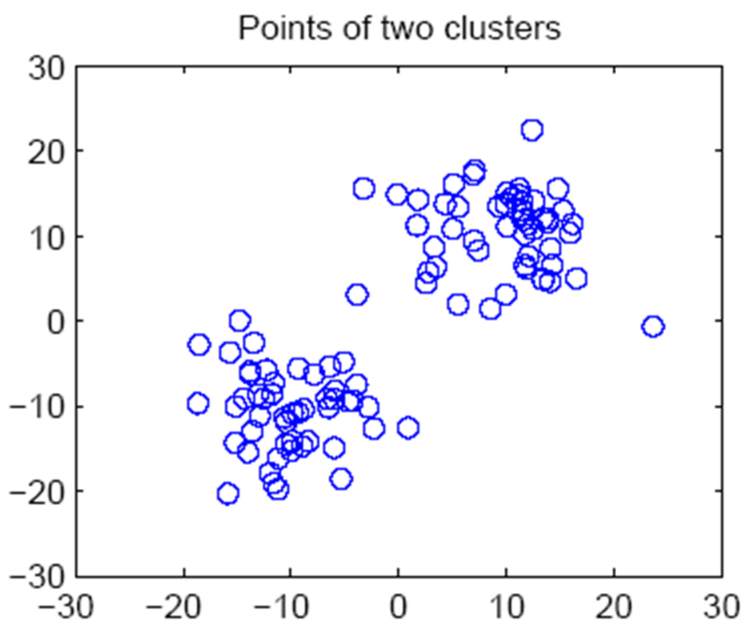


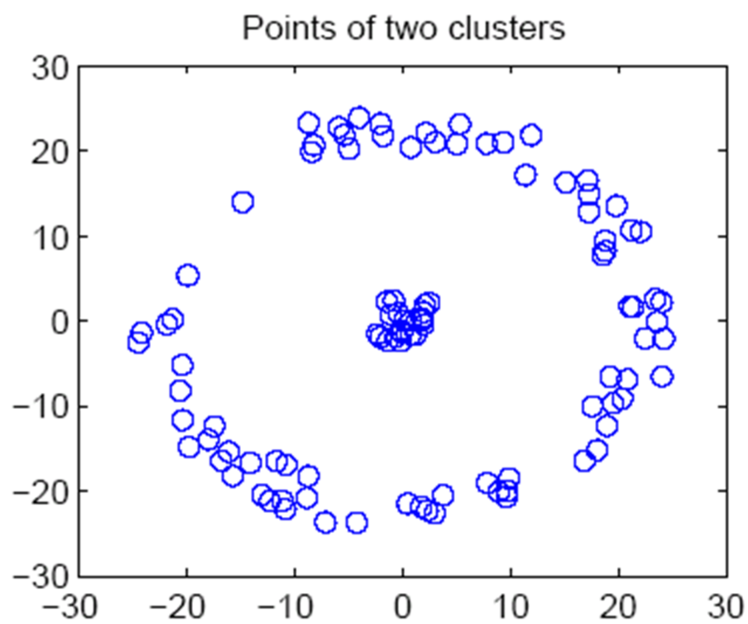
Problem!
Weight of cut is directly proportional to the number of edges in the cut.



First proposed by Wu and Leahy

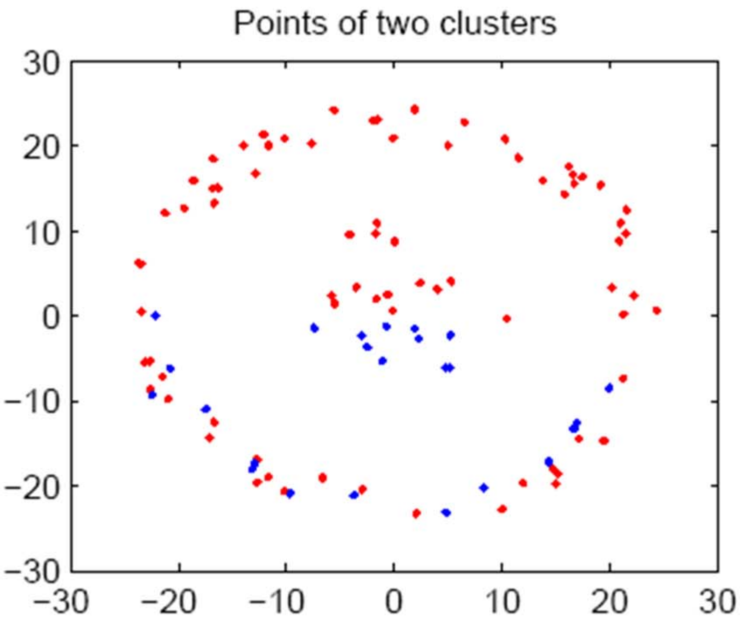
Superior performance?

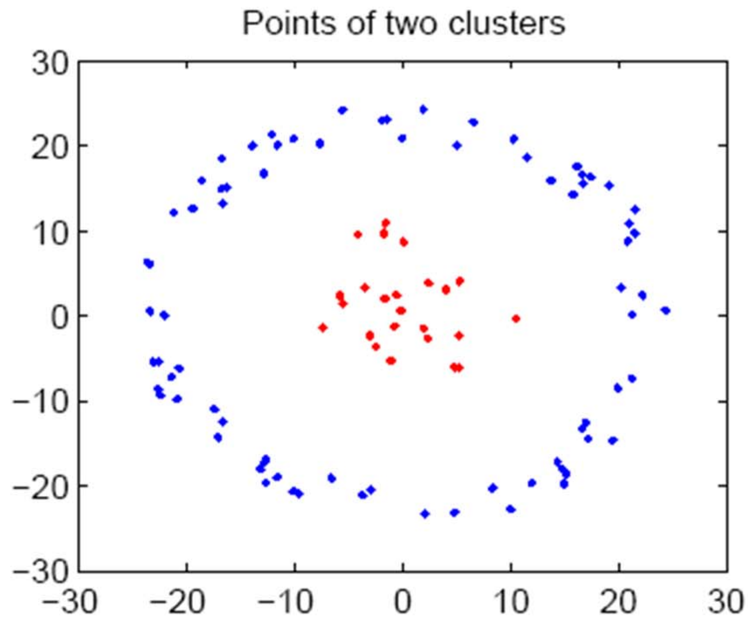




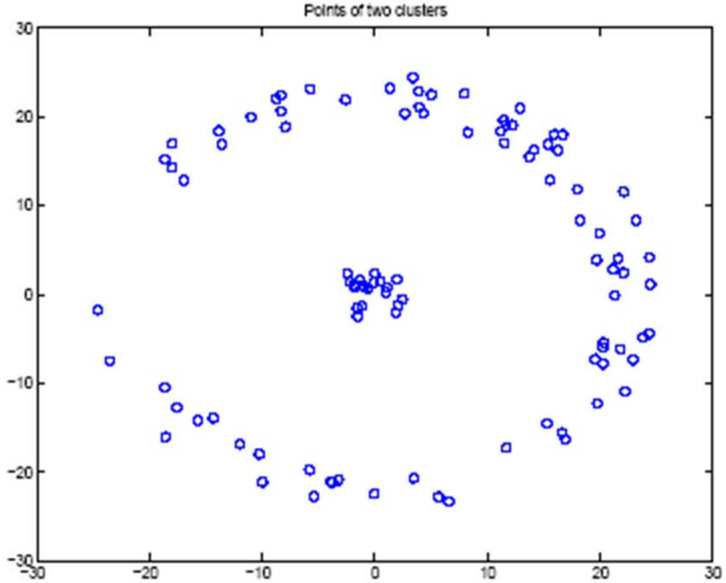
- K-means and Gaussian mixture methods are biased toward convex clusters

Ncut is superior in certain cases

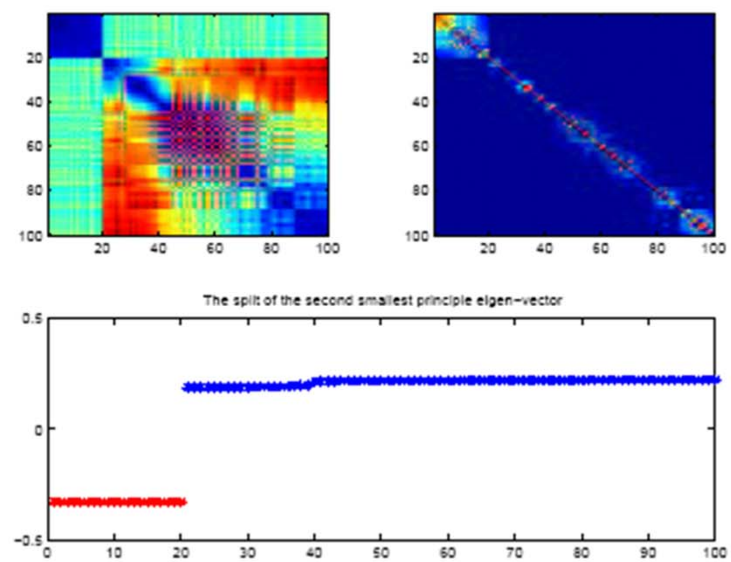




Why?

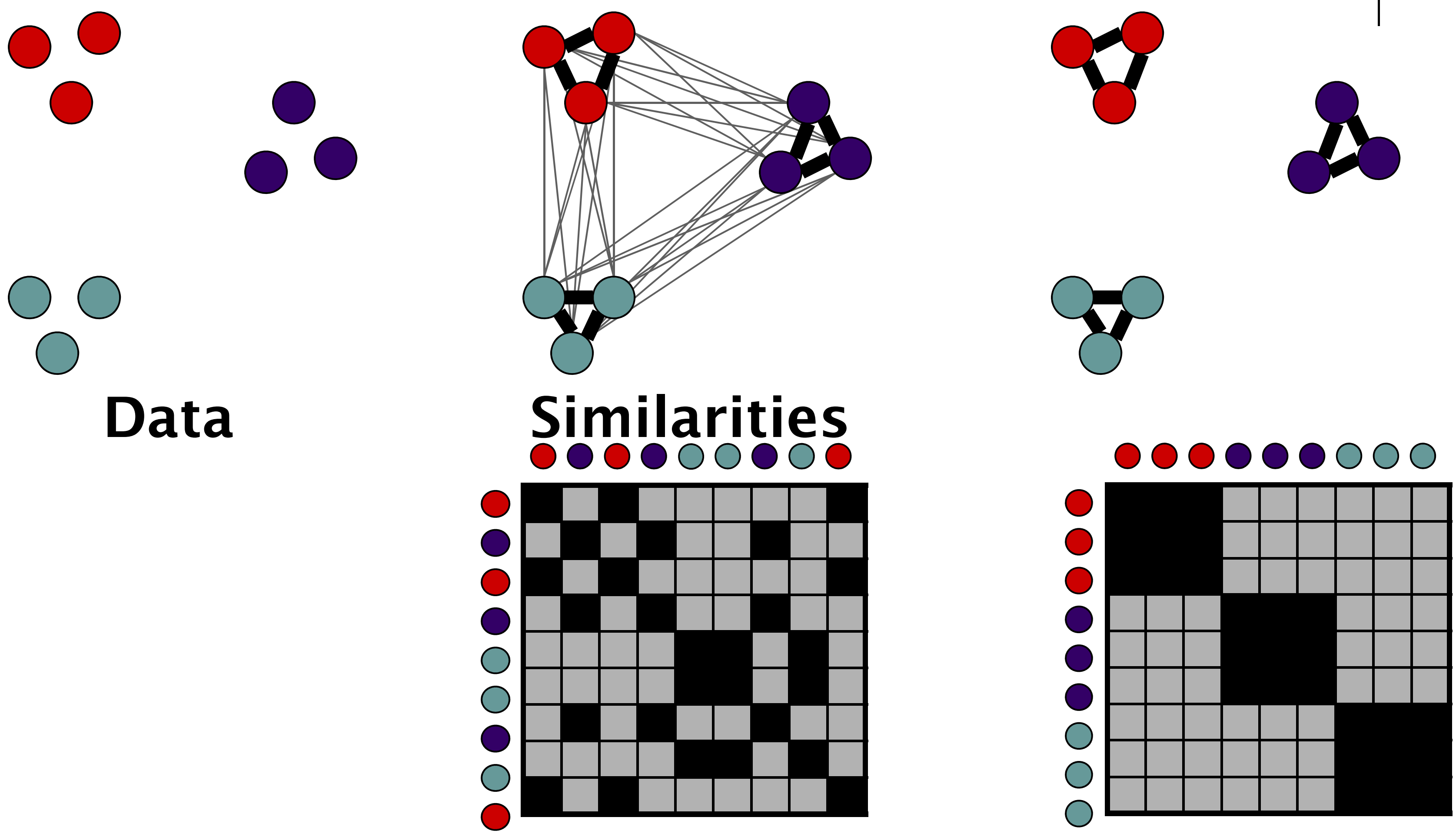

Points of two clusters

(a)


The split of the second smallest principle eigen-vector

(b)

General Spectral Clustering

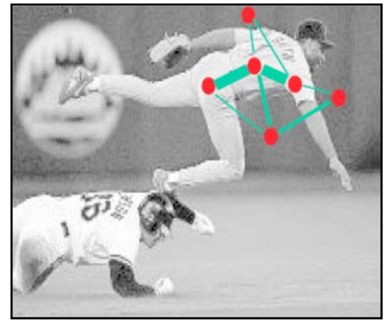


Representation

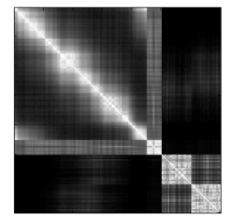
- Partition matrix X:

$$X = [X_1, \ldots, X_K]$$

$$X = \begin{array}{cc} \text{segments} & \\ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \text{pixels} \end{array}$$



- Pair-wise similarity matrix W: $W(i, j) = aff(i, j)$



- Degree matrix D: $D(i,i) = \sum_j w_{i,j}$

- Laplacian matrix L: $L = D - W$

Eigenvectors and blocks

- Block matrices have block eigenvectors:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \xrightarrow{\text{eigensolver}} \underset{\lambda_1 = 2}{\begin{bmatrix} .71 \\ .71 \\ 0 \\ 0 \end{bmatrix}} \quad \underset{\lambda_2 = 2}{\begin{bmatrix} 0 \\ 0 \\ .71 \\ .71 \end{bmatrix}}$$

$\lambda_3 = 0$

$\lambda_4 = 0$

- Near-block matrices have near-block eigenvectors:

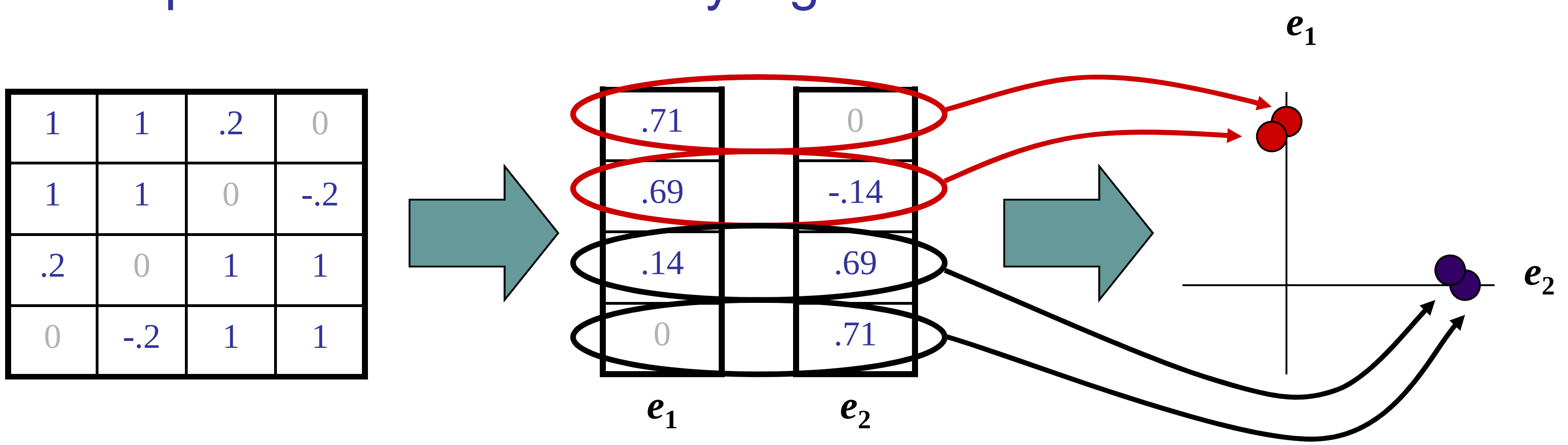
$$\begin{bmatrix} 1 & 1 & .2 & 0 \\ 1 & 1 & 0 & -.2 \\ .2 & 0 & 1 & 1 \\ 0 & -.2 & 1 & 1 \end{bmatrix} \xrightarrow{\text{eigensolver}} \underset{\lambda_1 = 2.02}{\begin{bmatrix} .71 \\ .69 \\ .14 \\ 0 \end{bmatrix}} \quad \underset{\lambda_2 = 2.02}{\begin{bmatrix} 0 \\ -.14 \\ .69 \\ .71 \end{bmatrix}}$$

$\lambda_3 = -0.02$

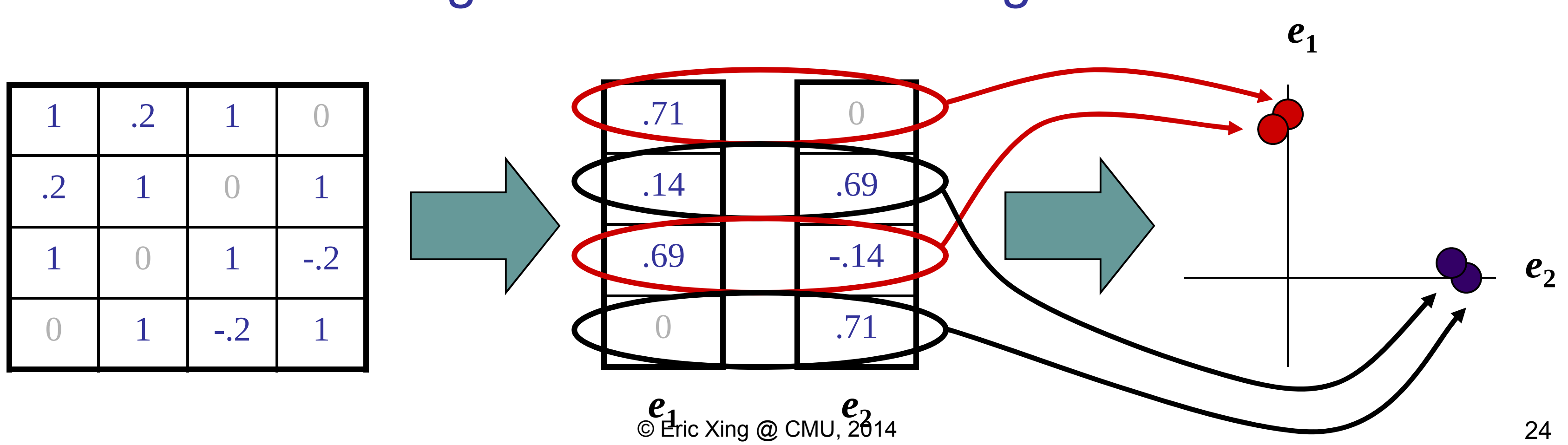
$\lambda_4 = -0.02$

Spectral Space

- Can put items into blocks by eigenvectors:



- Clusters clear regardless of row ordering:



Spectral Clustering

- Algorithms that cluster points using eigenvectors of matrices derived from the data
- Obtain data representation in the low-dimensional space that can be easily clustered
- Variety of methods that use the eigenvectors differently (we have seen an example)
- Empirically very successful
- Authors disagree:
  - Which eigenvectors to use
  - How to derive clusters from these eigenvectors
- Two general methods

Method #1

- Partition using only one eigenvector at a time
- Use procedure recursively
- Example: Image Segmentation
  - Uses 2^{nd} (smallest) eigenvector to define optimal cut
  - Recursively generates two clusters with each cut

Method #2

- Use k eigenvectors (k chosen by user)
- Directly compute k-way partitioning
- Experimentally has been seen to be “better”

Spectral Clustering Algorithm

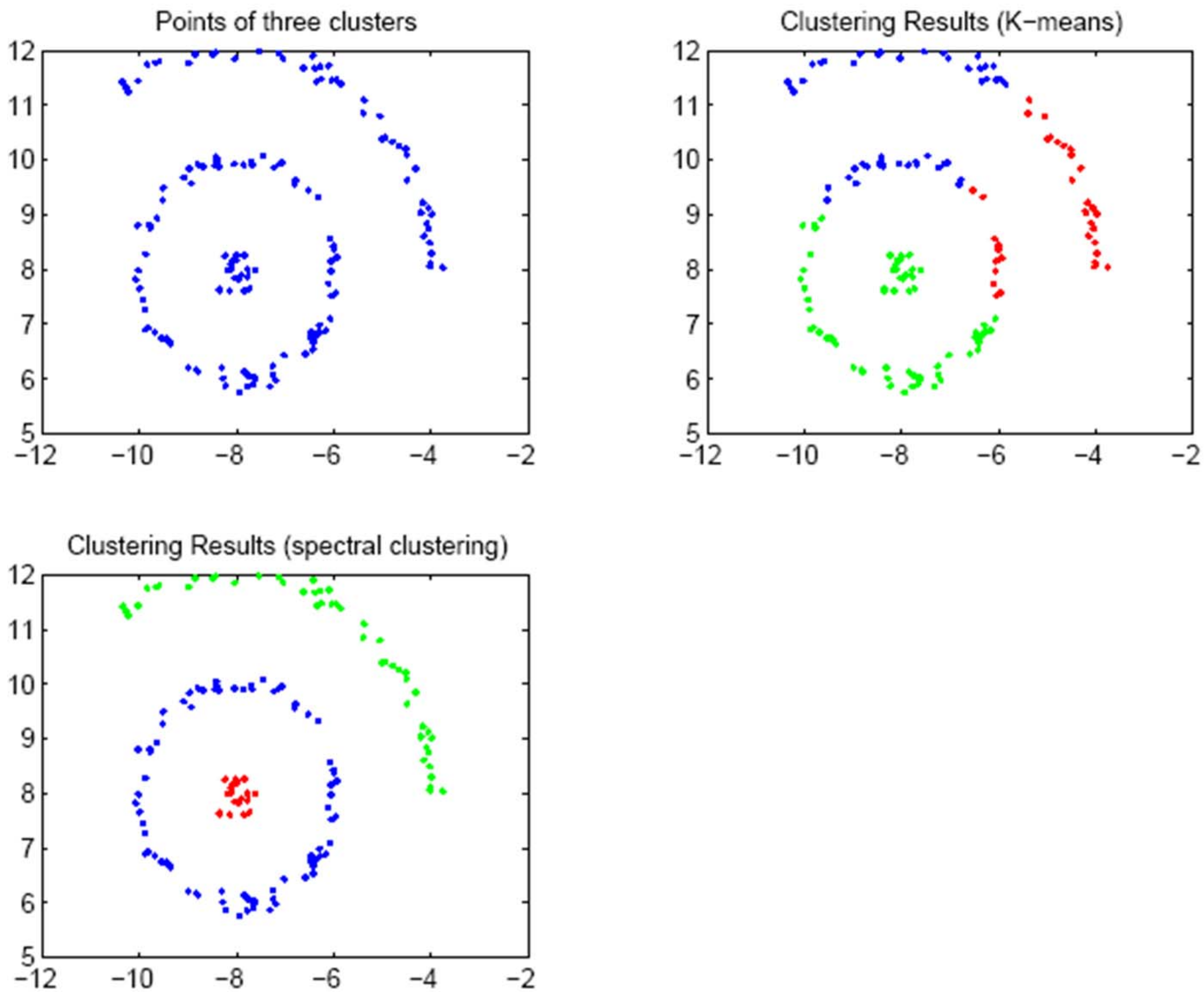
Ng, Jordan, and Weiss 2003

- Given a set of points $S=\{s_1,\ldots s_n\}$
- Form the affinity matrix $w_{i,j} = e^{\frac{-\|S_i - S_j\|_2^2}{\sigma^2}}, \quad \forall i \neq j, \quad w_{i,i} = 0$
- Define diagonal matrix $D_{ii}= \Sigma_\kappa a_{ik}$
- Form the matrix $L = D^{-1/2} W D^{-1/2}$
- Stack the k largest eigenvectors of L to for the columns of the new matrix X:

$$X = \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_k \\ | & | & & | \end{bmatrix}$$

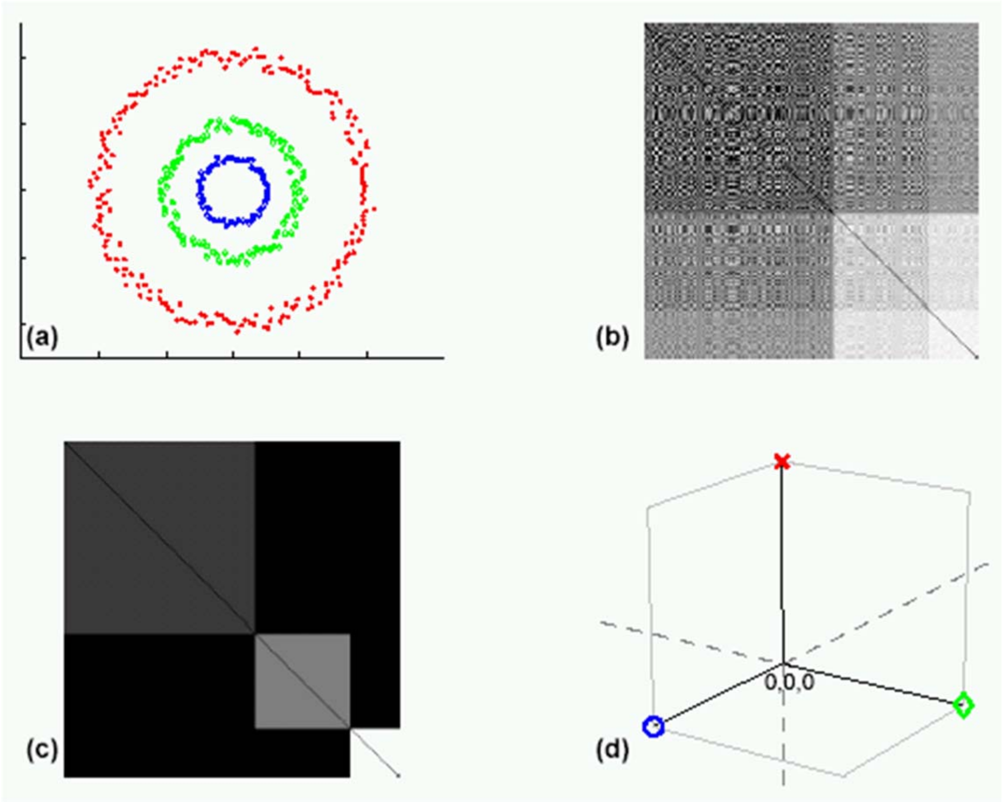
- Renormalize each of X's rows to have unit length and get new matrix Y. Cluster rows of Y as points in R^k

SC vs Kmeans



Points of three clusters
Clustering Results (K−means)
Clustering Results (spectral clustering)
12
11
10
9
8
7
6
5
−12
−10
−8
−6
−4
−2

Why it works?



- K-means in the spectrum space !

More formally …

- Recall generalized Ncut

$$\mathrm{Ncut}(A_1, A_2 \ldots A_k) = \sum_{r=1}^{k} \left(\frac{\sum_{i \in A_r, j \in V \setminus A_r} W_{ij}}{\sum_{i \in A_r, j \in V} W_{ij}} \right) = \sum_{r=1}^{k} \left(\frac{\mathrm{cut}(A_r, \bar{A}_r)}{d_{A_r}} \right)$$

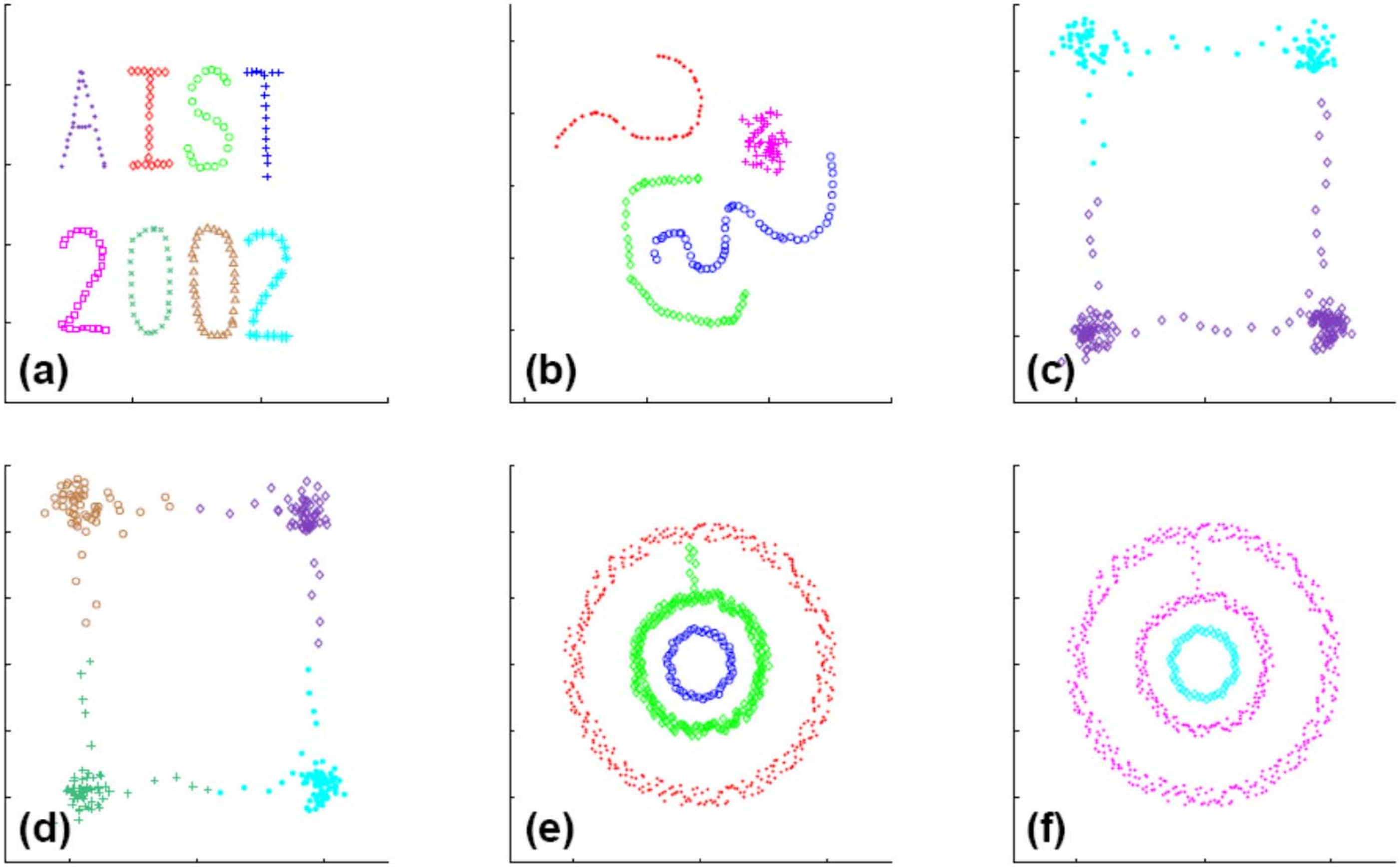
- Minimizing this is equivalent to spectral clustering

$$\min \quad \mathrm{Ncut}(A_1, A_2 \ldots A_k) = \sum_{r=1}^{k} \left(\frac{\mathrm{cut}(A_r, \bar{A}_r)}{d_{A_r}} \right)$$

$$\min \quad Y^{\mathrm{T}} D^{-1/2} W D^{-1/2} Y$$
$$\text{s.t.} \quad Y^{\mathrm{T}} Y = I$$

$$Y = \begin{array}{cc} \text{segments} & \\ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \text{pixels} \end{array}$$

Toy examples



Images from Matthew Brand (TR-2002-42)

User's Prerogative

- Choice of k, the number of clusters
- Choice of scaling factor
  - Realistically, search over σ^2 and pick value that gives the tightest clusters
- Choice of clustering method: k-way or recursive bipartite
- Kernel affinity matrix

$$w_{i,j} = K(S_i, S_j)$$

Conclusions

- Good news:
  - Simple and powerful methods to segment images.
  - Flexible and easy to apply to other clustering problems.

- Bad news:
  - High memory requirements (use sparse matrices).
  - Very dependant on the scale factor for a specific problem.

$$W(i,j) = e^{\frac{-\left\| X_{(i)} - X_{(j)} \right\|_2^2}{\sigma_X^2}}$$