

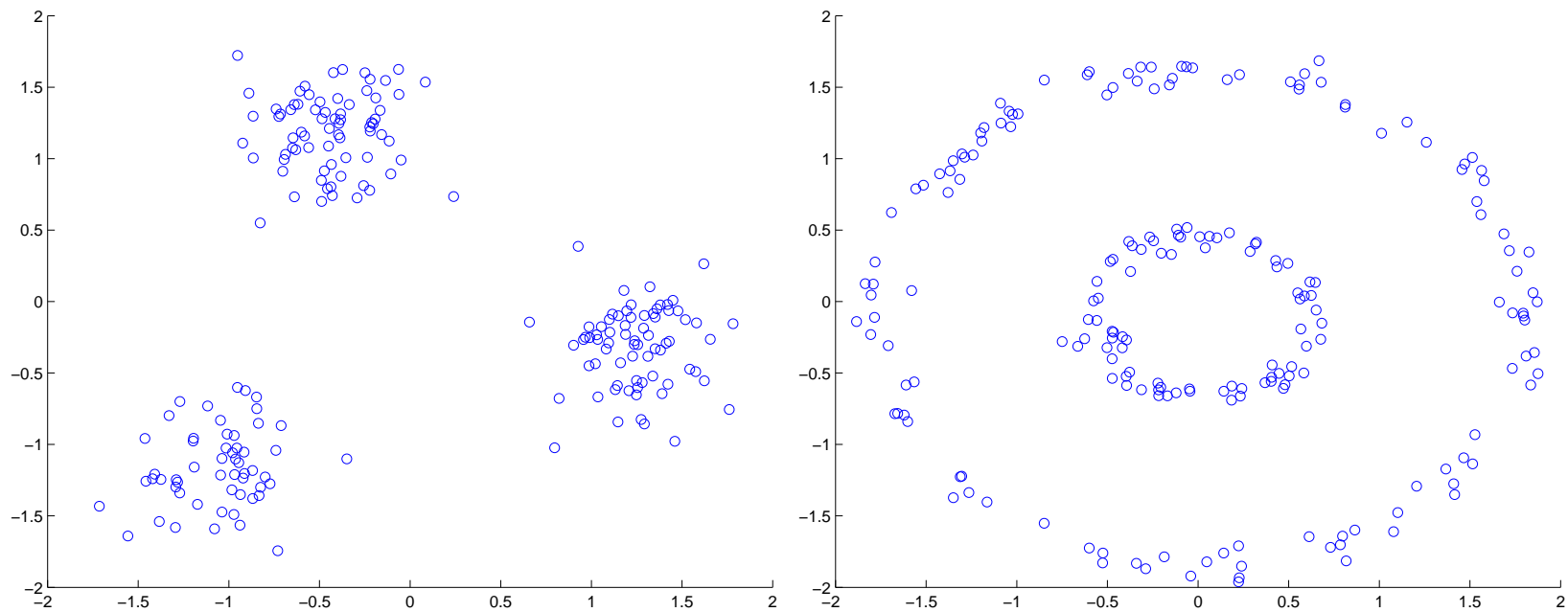
# Segmentation using eigenvectors: a unifying view

Yair Weiss (UC Berkeley)

Greg Hamerly  
CSE 254  
University of California, San Diego

Tuesday, April 16, 2002

# What are the major groups?

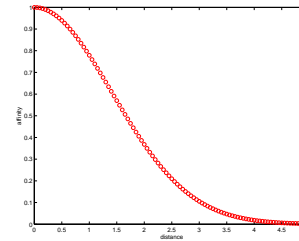


## First, some definitions

**Input data:** data set  $X = \{x_i\}$  where  $1 \leq i \leq n$ . Each  $x_i$  has  $d$  dimensions, so  $X \subset \mathbb{R}^{n \times d}$

**Pairwise affinity matrix:**

$$W(i, j) = e^{-\|x_i - x_j\|^2 / 2\sigma^2}$$

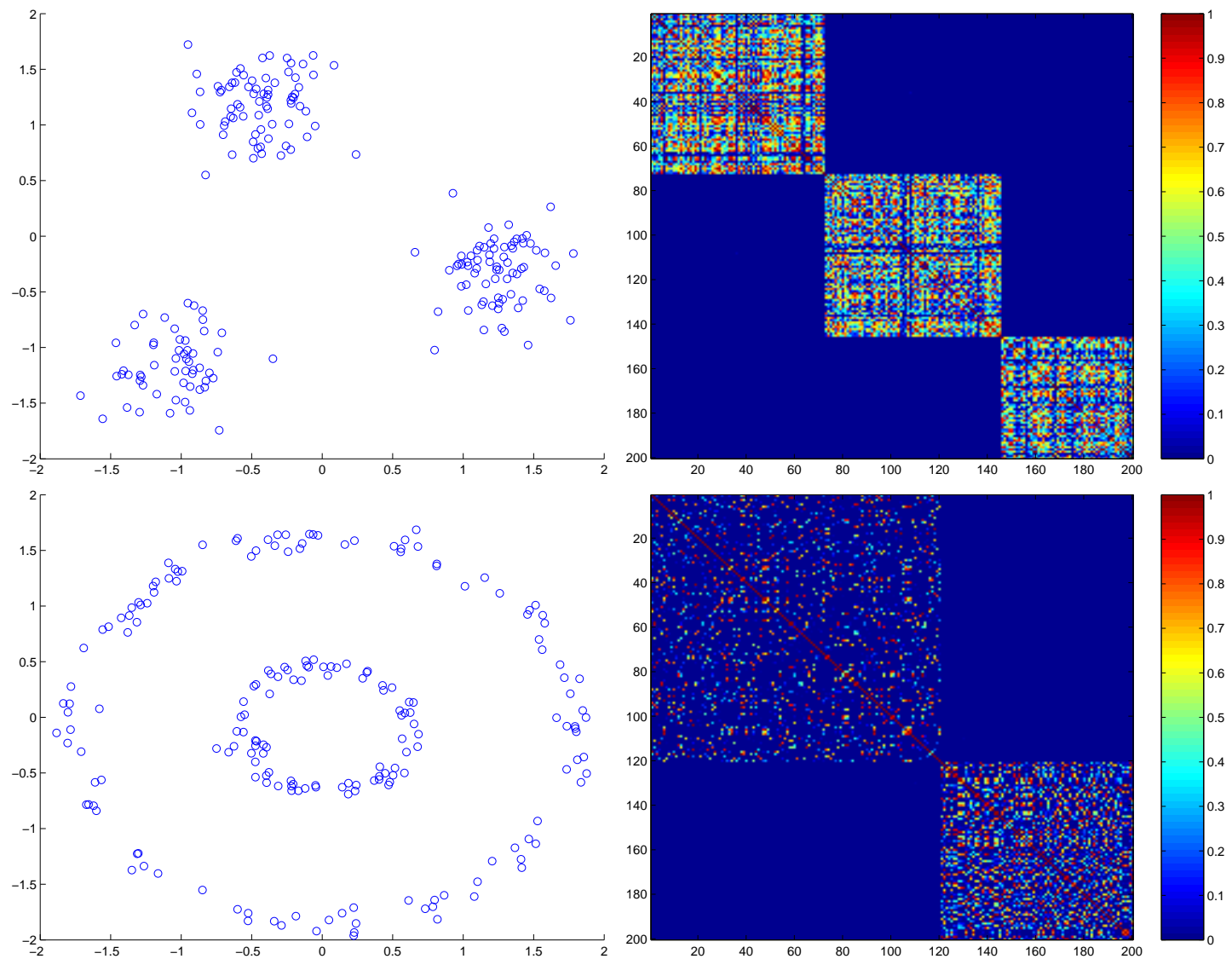


where  $1 \leq i, j \leq n$ .

**Degree matrix:**

$$D(i, i) = \sum_j W(i, j)$$

# The affinity matrix $W$ , for $\sigma = 0.3$



# Work in spectral segmentation

This paper covers...

- Scott & Longuet-Higgins (1990)
- Costeira & Kanade (1995)
- Shi & Malik (1997)
- Perona & Freeman (1998)

There has also been significant recent work by

- Kannan, Vempala, Vetta (2000)
- Meila & Shi (2001)
- Ng, Jordan, Weiss (2002)

Other areas:

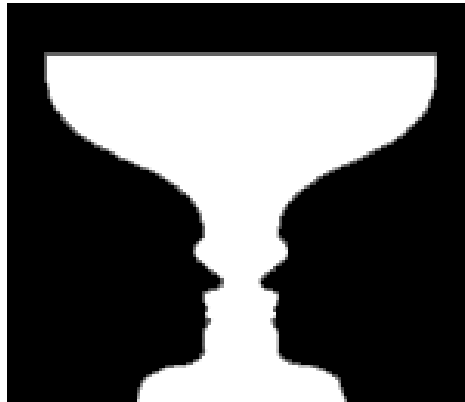
- graph theory (Fiedler, 1975)
- parallel load balancing (Hendrickson, 1993)

# Multiple views of spectral segmentation

**Graph cuts** (Shi & Malik, Kannan & Vempala)

**Random walks on Markov chains** (Meila & Shi)

**Eigensystems** (Perona & Freeman, Ng *et al.*)



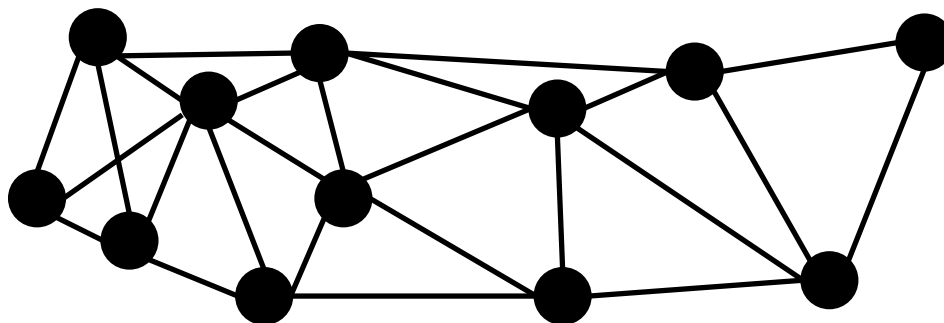
# Motivation 1: finding good cuts in a graph

Intuitively, we want to

- keep together things that are similar
- separate things that are different

Idea: **construct a graph**.

Construct a complete undirected graph  $G$  where each  $x_i$  maps to a unique graph node  $v_i$ . Edges  $(i, j)$  are weighted as  $W(i, j)$ .

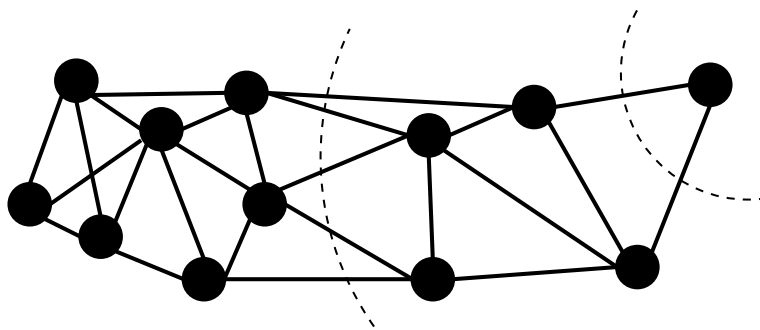


## Now that we have a graph...

Define the **affinity** of two sets of nodes  $A \cup B = V$ .

$$aff(A, B) = \sum_{i \in A, j \in B} W(i, j)$$

This is sometimes called the *graph cut*.



**Maximize intra-segment affinity:**  $aff(A, A), aff(B, B)$

**Minimize inter-segment affinity:**  $aff(A, B)$

By themselves these are not enough!

## Several metrics

Shi & Malik (2000) point out several reasonable metrics:

Average association finds more **coherent segments**:

$$\text{maximize } aass(A, B) = |A| \left( \frac{aff(A, A)}{|A|^2} \right) + |B| \left( \frac{aff(B, B)}{|B|^2} \right)$$

Average cut finds **better splits**:

$$\text{minimize } acut(A, B) = |A| \left( \frac{aff(A, B)}{|A||B|} \right) + |B| \left( \frac{aff(A, B)}{|A||B|} \right)$$

Normalized cut **balances the two**:

$$\text{minimize } ncut(A, B) = \frac{aff(A, B)}{aff(A, V)} + \frac{aff(A, B)}{aff(B, V)}$$

## The normalized cut metric

Find  $A, B$  that minimize:

$$ncut(A, B) = \frac{aff(A, B)}{aff(A, V)} + \frac{aff(A, B)}{aff(B, V)}$$

Reducing numerator **minimizes affinity between segments.**

The denominator  $aff(A, V), aff(B, V)$  normalizes the sizes of the segments. But we know that:

$$aff(A, V) = aff(A, A \cup B) = aff(A, A) + aff(A, B)$$

and similarly for  $aff(B, V)$ . So the normalized cut is:

$$ncut(A, B) = \frac{aff(A, B)}{aff(A, A) + aff(A, B)} + \frac{aff(A, B)}{aff(B, B) + aff(A, B)}$$

So normalized cut also **maximizes the affinity within segments.**

## Normalized cut as an eigensystem

$$ncut(A, B) = \frac{aff(A, B)}{aff(A, V)} + \frac{aff(A, B)}{aff(A, V)}$$

Shi & Malik show minimizing Ncut is equivalent to:

$$\min_{\mathbf{x}} ncut(\mathbf{x}) = \min_{\mathbf{y}} \frac{\mathbf{y}^T (\mathbf{D} - \mathbf{W}) \mathbf{y}}{\mathbf{y}^T \mathbf{D} \mathbf{y}}$$

where  $\mathbf{x} \in \{-1, 1\}^n$  is an indicator function for the two segments, and subject to constraints that  $y_i \in \{1, -1\}$  and  $\mathbf{y}^T \mathbf{D} \mathbf{1} = 0$ .

**This is NP-complete.**

## NP-complete is bad

But, if we **relax the constraint** that eigenvectors take discrete values, we can **approximate the solution** with the generalized eigensystem:

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda\mathbf{D}\mathbf{y}$$

Now: take the generalized eigenvector  $y$  for the second smallest  $\lambda$ . The component values will give indications about the segmentation.

Why the second smallest? We'll get to that...

# What is an eigenvector?

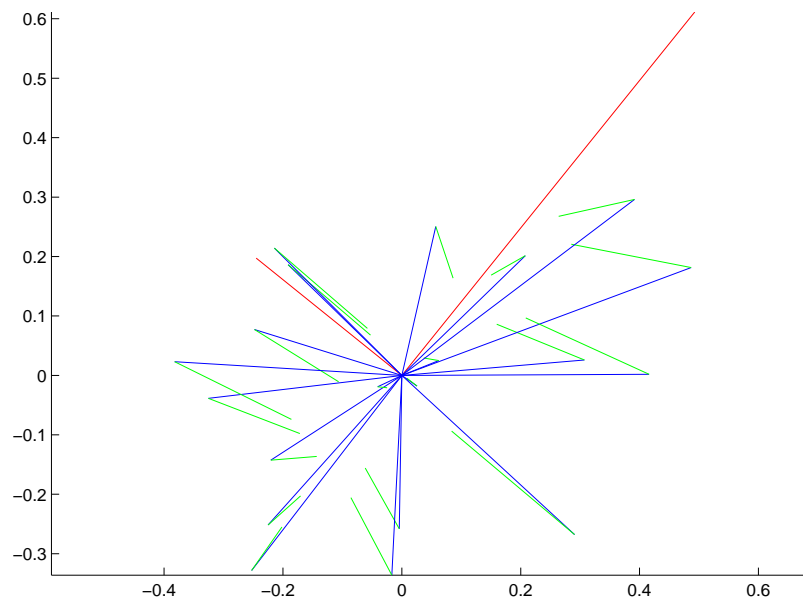
A **characteristic** vector of a matrix. A solution to the equation  $A\mathbf{y} = \lambda\mathbf{y}$  where  $\mathbf{y}$  is called the eigenvector, and  $\lambda$  is the eigenvalue.

An eigenvector gives insight into the global behavior of the system. They are “basins of attraction” for the system.

$$A = \begin{bmatrix} 0.50 & 0.23 \\ 0.23 & 0.60 \end{bmatrix}$$

$$[\mathbf{y}_1 \mathbf{y}_2] = \begin{bmatrix} 0.62 & -0.78 \\ 0.78 & 0.62 \end{bmatrix}$$

$$\lambda_1 = 0.79, \lambda_2 = 0.32$$



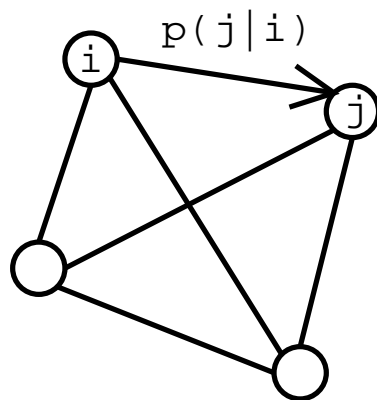
## Motivation 2: random walks

Construct a Markov chain where each data point is a state, connected to all other states with some probability.

With our affinity  $W$  and degree  $D$ , the stochastic matrix is:

$$P = D^{-1}W$$

which is the row-normalized version of  $W$ , so each entry  $P(i, j)$  is a probability of “walking” to state  $j$  from state  $i$ .



Meila & Shi (2000)

## Segmentation as random walks

Probability of a walk through states  $(s_1, \dots, s_m)$  is:

$$p(s_2, \dots, s_m | s_1) = \prod_{i=2}^m P(s_i, s_{i-1})$$

Suppose we **divide the states** into two groups, and **minimize the probability of jumping between** the two groups.

Formulate this as an eigenvector problem:

$$P\mathbf{y} = \lambda\mathbf{y}$$

Where the components of  $\mathbf{y}$  will give the segmentation.

## Segmentation as random walks

$P$  is a stochastic matrix:  $P\mathbf{y} = \lambda\mathbf{y}$

The largest eigenvalue is 1, and its eigenvector is  $\mathbf{1}$ . Not very informative about segmentation.

The second-largest eigenvector is orthogonal to the first, and its components indicate the strongly connected sets of states.

Meila and Shi show that minimizing the probability of jumping between two groups in the Markov chain is equivalent to minimizing  $N_{\text{cut}}$ .

## Relating Random walks to Ncut

**Theorem:** If  $\lambda, \mathbf{y}$  are a solution to  $P\mathbf{y} = \lambda\mathbf{y}$ , then

- $(1 - \lambda)$  is an eigenvalue of  $(D - W)\mathbf{y} = \lambda D\mathbf{y}$
- $\mathbf{y}$  is an eigenvector of  $(D - W)\mathbf{y} = \lambda D\mathbf{y}$

**Proof:**

$$\begin{aligned}(D - W)\mathbf{y} &= \lambda D\mathbf{y} \\ D^{-1}(D - W)\mathbf{y} &= D^{-1}\lambda D\mathbf{y} \\ (D^{-1}D - D^{-1}W)\mathbf{y} &= \lambda D^{-1}D\mathbf{y} \\ (I - P)\mathbf{y} &= \lambda\mathbf{y} \\ I\mathbf{y} - P\mathbf{y} &= \lambda\mathbf{y} \\ P\mathbf{y} &= I\mathbf{y} - \lambda\mathbf{y} \\ P\mathbf{y} &= (1 - \lambda)\mathbf{y}\end{aligned}$$

## Why not the smallest eigenvector in Ncut?

We know to choose the small eigenvectors of  $(D - W)\mathbf{y} = \lambda\mathbf{y}$  because of the connection with random walks.

Recall that Ncut wants to minimize

$$ncut(A, B) = \frac{aff(A, B)}{aff(A, V)} + \frac{aff(A, B)}{aff(B, V)}$$

Then if we take  $A = V, B = \{\}$ , then  $ncut(A, B) = 0$ , which is its minimum. So saying that everything is in the same segment minimizes this term, but it's not interesting.

## Shi & Malik approach (1997)

Each data point encodes texture around and location of a pixel, and the affinity matrix is for texture and region similarity.

- Find the generalized eigenvectors of  $(D - W)\mathbf{y} = \lambda D\mathbf{y}$
- Take  $k$  smallest eigenvectors  $\mathbf{y}_i$  starting with the second smallest.
- Form the matrix of  $k$  column vectors  
$$X = [y_1 y_2 \dots y_k] \in \mathbb{R}^{n \times k}$$
- Normalize the rows:  $X_{ij} = X_{ij} / (\sum_j X_{ij}^2)^{1/2}$
- Treat each row of  $X$  as a point in  $\mathbb{R}^k$ , and cluster using  $k$ -means.

## Ng, Jordan, & Weiss approach (2002)

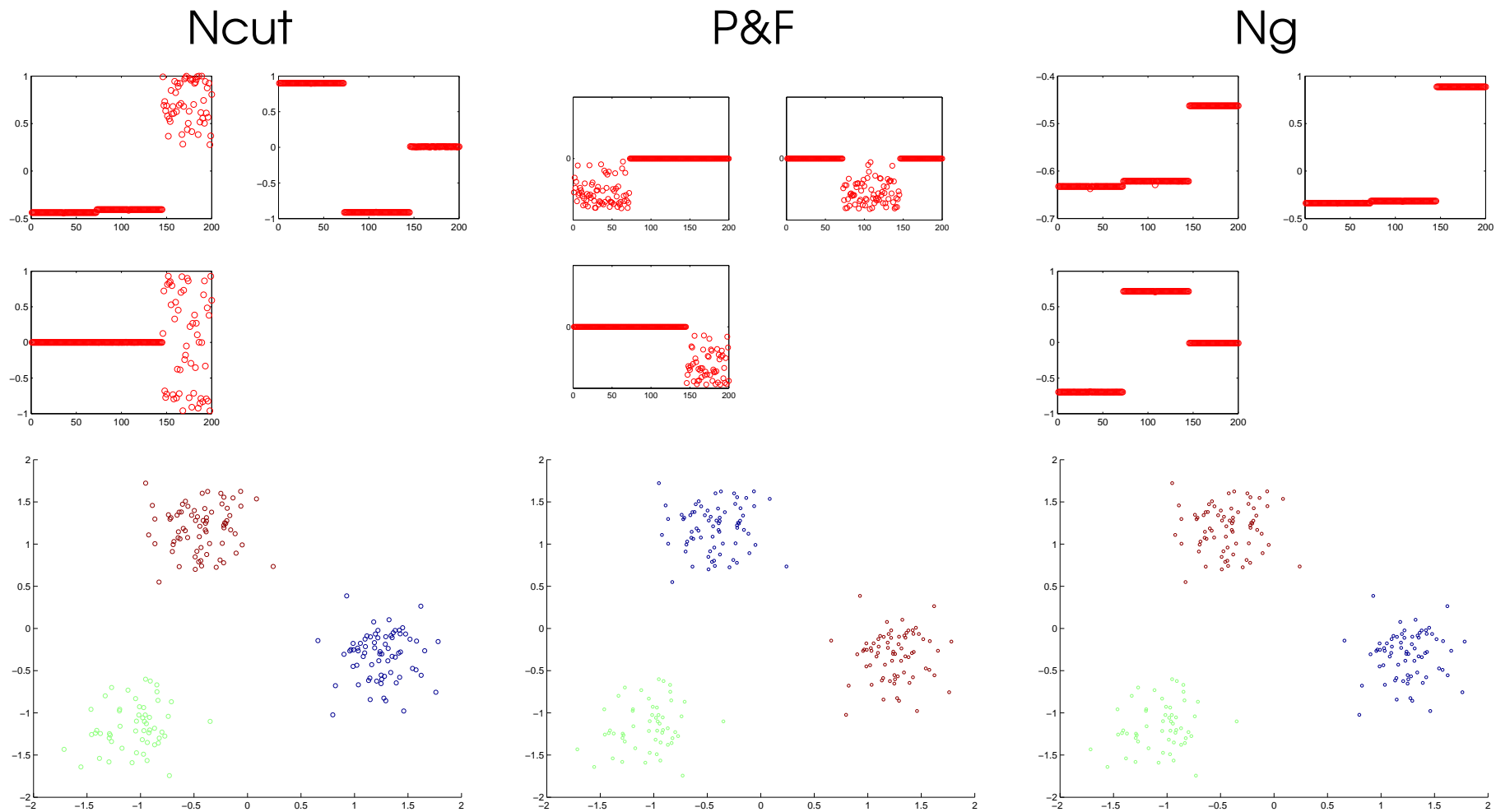
- Form the matrix  $A = W$ , but set  $A_{ii} = 0$
- Construct  $L_{ij} = A_{ij} / \sqrt{D_{ii}D_{jj}}$ . This is normalized by row and column degree.
- Find the  $k$  largest eigenvectors of  $L$ , called  $y_1, \dots, y_k$ .  
Form the matrix of column vectors  $X = [y_1 y_2 \dots y_k] \in \mathbb{R}^{n \times k}$
- Normalize the rows:  $X_{ij} = X_{ij} / (\sum_j X_{ij}^2)^{1/2}$
- Treat each row of  $X$  as a point in  $\mathbb{R}^k$ , and cluster using  $k$ -means.

## Perona & Freeman approach (1998)

The most “straightforward” approach.

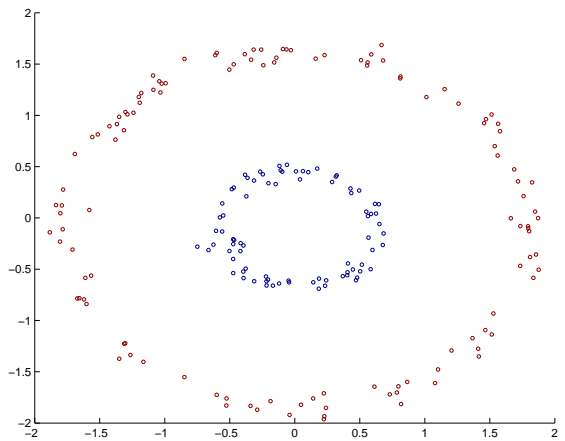
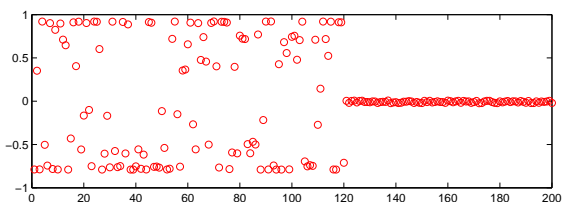
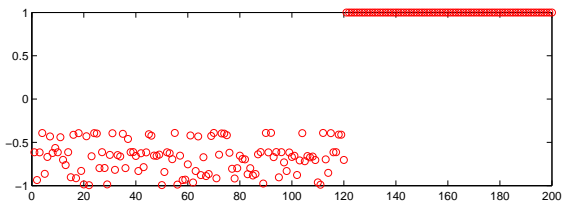
- Compute the eigenvectors  $y_i$  of the affinity matrix  $W$ .
- Form the matrix  $Y = [y_1 y_2 \dots y_k]$
- Treat each row of  $Y$  as a point in  $\mathbb{R}^k$ , and cluster using  $k$ -means.

# Results for three Gaussian ( $\sigma = 0.3$ )

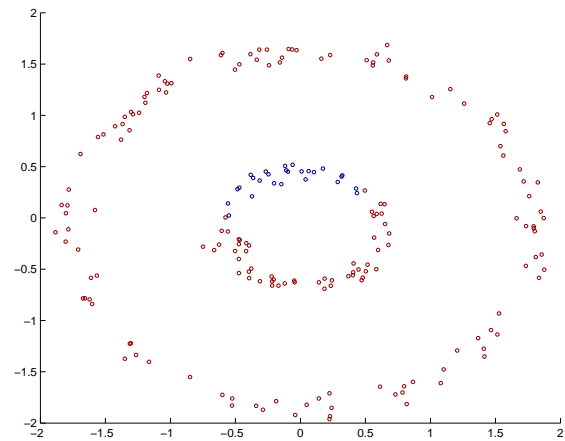
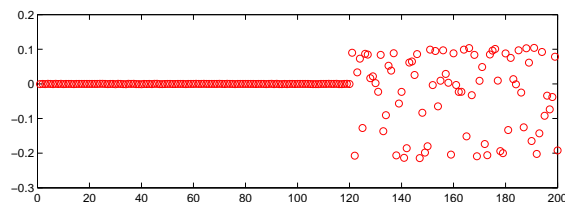
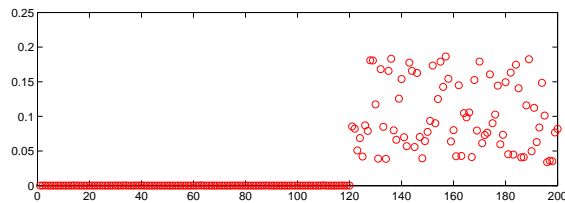


# Results for two circle ( $\sigma = 0.3$ )

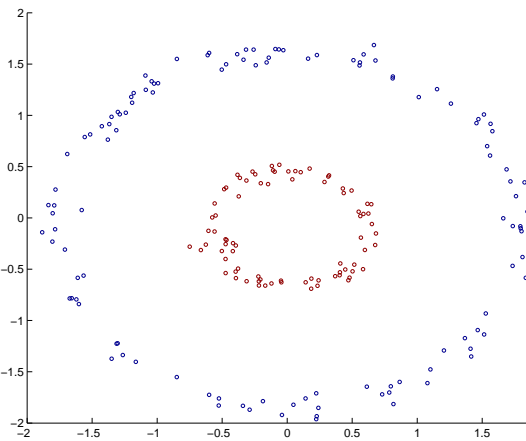
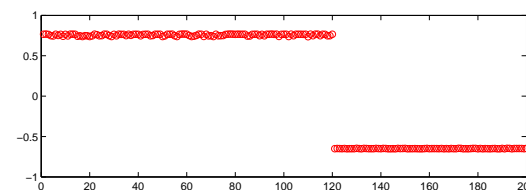
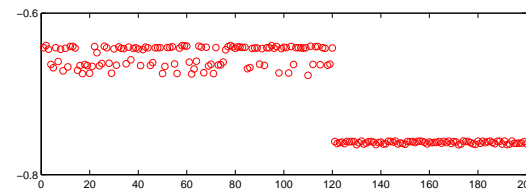
Ncut



P&F



Ng



## Efficiency

Finding the affinity matrix takes  $O(n^2d)$  time.

Finding eigenvectors & eigenvalues takes  $O(n^3)$  time, which is very slow.

Using subsampling, one can reduce  $n$  (Fowlkes *et al.*), at the cost of accuracy.

Using  $k$ -means takes  $O(nk^2)$  time per iteration, since there are  $k$  dimensions and  $k$  centers.

## Affinity matrix normalization

Several different ways of normalizing  $W$ :

- Perona & Freeman don't normalize.
- Shi & Malik: by  $D^{-1}W$  (stochastic formulation, rowsum-normalized).
- Ng *et al.*: as  $D^{-1/2}WD^{-1/2}$  (symmetric matrix).  
Eigenvectors are orthogonal.

Normalization helps regularize the connection between closely-connected segments.

## Why multiple eigenvectors? Why $k$ -means?

Each eigenvector indicates a segmentation based on the values of its components. Therefore, multiple eigenvectors make multiple splits (see results slide).

Using  $k$  eigenvectors is a dimensionality reduction from  $n$  dimensions to the  $k$  most important dimensions.

Generally the similar points will have similar eigenvector components, but  $k$ -means is used to clean up this association.

## Block diagonal matrices

Suppose that there is no affinity between segments, so that

$$W = \begin{bmatrix} W_{11} & 0 & \cdots & 0 \\ 0 & W_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_{kk} \end{bmatrix}$$

Ng *et al.* show that the eigenvectors of  $W_{ii}$  will be the eigenvectors of  $W$ , with zero padding.

## Sensitivity of solutions

What happens if there are small perturbations in the affinity matrix?

Eigenvectors are not stable, but subspaces formed by subsets of eigenvectors are stable (Stewart & Sun).

One comfort: if  $|\lambda_k - \lambda_{k+1}|$  is big, then the top  $k$  eigenvectors will be stable.

Applying to segmentation, this will be the case if there are actually  $k$  clusters in the data (Ng *et al.*).

# Data permutations

Define:

- $W$  is the affinity matrix of data  $\{x_1, \dots, x_n\}$ .
- $\phi(i)$  is a permutation on  $\{1, 2, \dots, n\}$ .
- $W'$  is the affinity matrix of data  $\{x_{\phi(1)}, \dots, x_{\phi(n)}\}$ .

Claims:

- For each eigenvector  $e$  of  $W$ , there is a corresponding permuted eigenvector  $e'$  of  $W'$  where  $e_k = e'_{\phi(k)}$
- $\lambda = \lambda'$ .

Permutations of data order preserve all the matrix values; the dimensions are simply renamed. Therefore, the eigenvectors of the original and permuted matrices are just permutations of each other.

Thank you

## General normalized cut metric

The original ncut metric is for **only two** segments of a graph.

The ***k*-way** ncut metric is as follows:

$$kncut(A, B, \dots) = \frac{cut(A, V - A)}{cut(A, V)} + \frac{cut(B, V - B)}{cut(B, V)} + \dots$$