Statistical Methods in Artificial Intelligence
CSE471 - Monsoon 2015 : Lecture 21

Avinash Sharma

CVIT, IIIT Hyderabad

Lecture Plan

- Revision from Previous Lecture
- Graph Representation
- Graph Partitioning
- Graph Laplacian
- Spectral Embedding
- Spectral Clustering and Demo
- Practical Aspects
- Kernel Kmeans
- Decision Trees (Next Class)

K-means Clustering

- What is a cluster: a group of points whose inter-point distance are small compared to distances to points outside the cluster.
- Cluster centres: $\mu_1, \ldots, \mu_m$.
- Goal: find an assignment of points to clusters as well as a set of mean-vectors μ_k.
- Notations: For each point $\{x_1, \ldots, x_n\} \in \mathbb{R}^D$ there is a binary indicator variable $r_{jk} \in \{0,1\}$.
- Objective: minimize the following distortion measure:

$$J = \sum_{j=1}^{n} \sum_{k=1}^{m} r_{jk} \|x_i - \mu_k\|^2$$

K-means Clustering

- Initialization: Choose m and initial values for $\mu_1, \ldots, \mu_m$.
- First step: Assign the j^{th} point to the closest cluster centre:

$$r_{jk} = \begin{cases} 1 & \text{if } k = \arg\min_l \left\| x_j - \mu_l \right\|^2, \\ 0 & \text{otherwise.} \end{cases}$$

- Second Step: Minimize J to estimate the cluster centres:

$$\mu_k = \frac{\sum_{j=1}^{n} r_{jk} x_j}{\sum_{j=1}^{n} r_{jk}}$$

- Convergence: Repeat until no more change in the assignments.

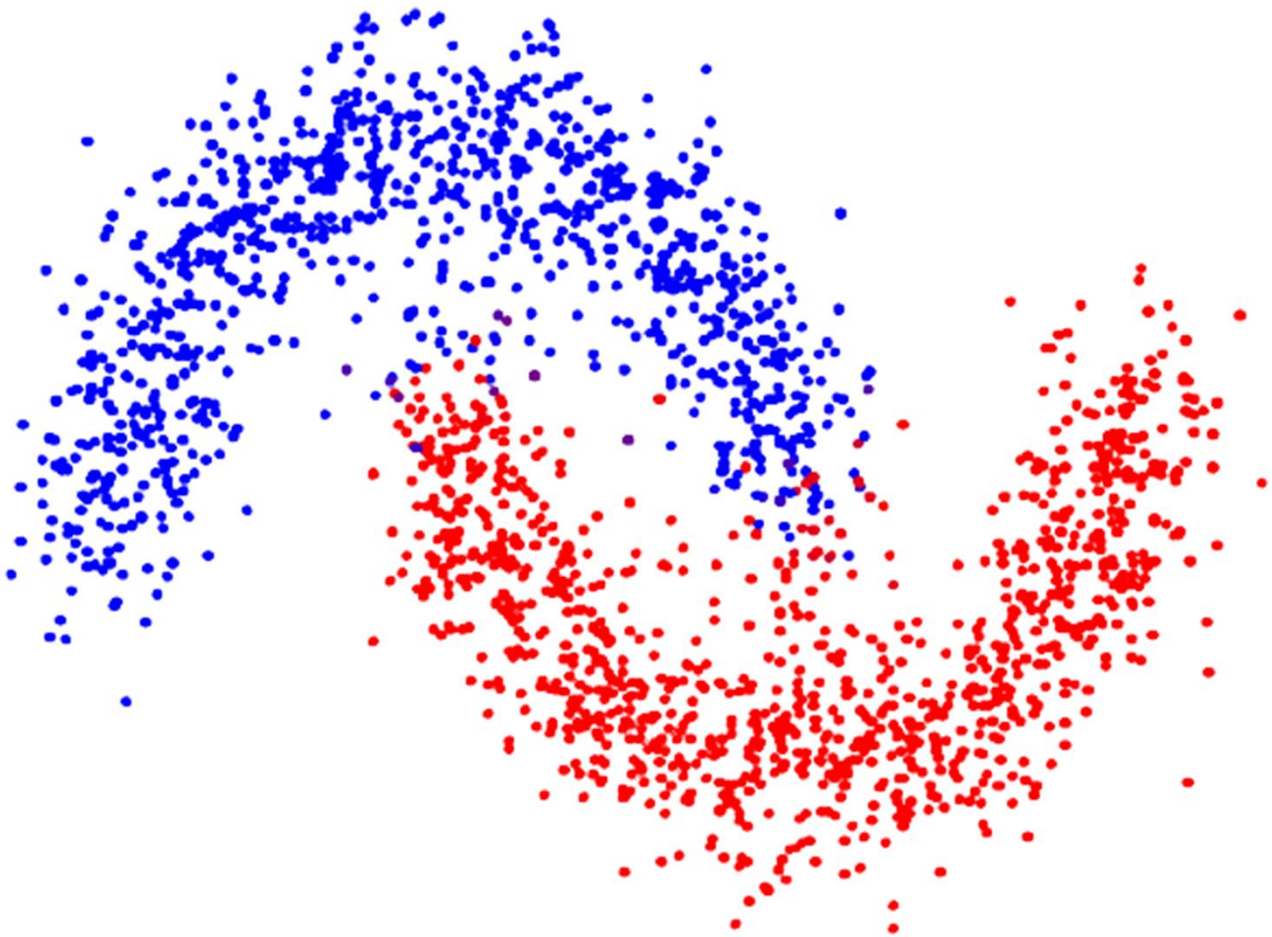
Motivation

- Techniques such as K-means or Gaussian mixtures will not work well because the clusters are neither spherical nor Gaussian.

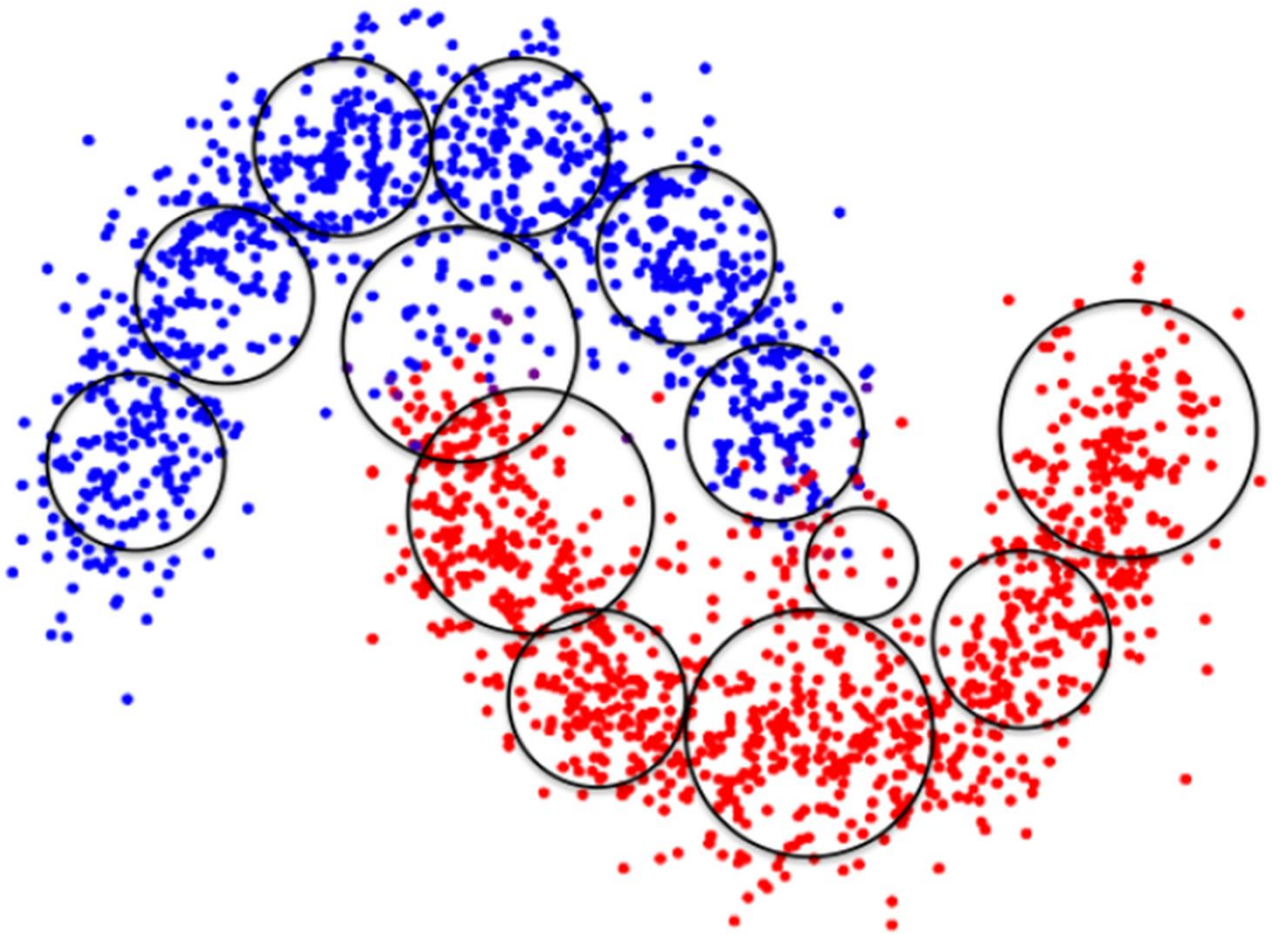
- One needs to apply a non-linear transformation of the data such that “curved" clusters are transformed into “blobs”

- The general idea of spectral clustering is to build an undirected weighted graph and to map the points (the graph's vertices) into the spectral space, spanned by the eigenvectors of the Laplacian matrix.

Point Cloud Representation



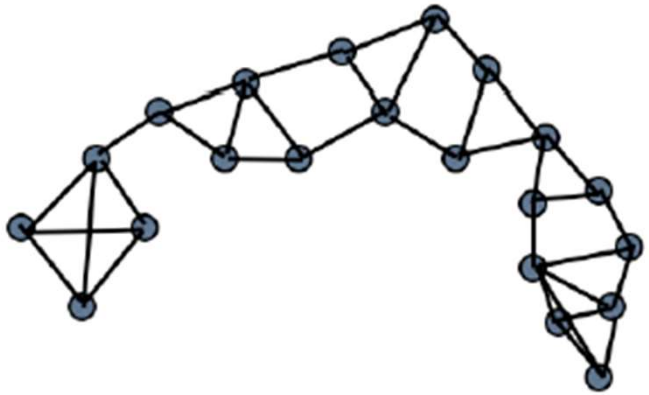
Point Cloud Representation

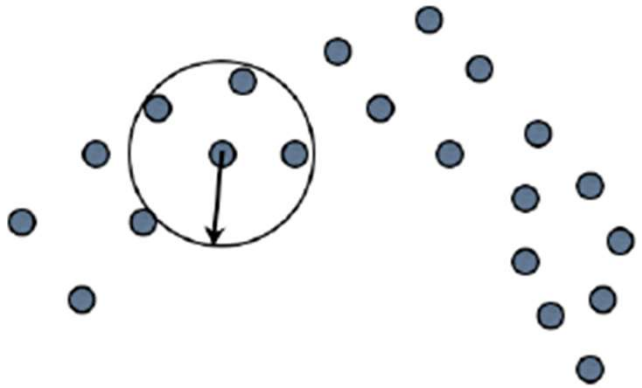


Graph Representation

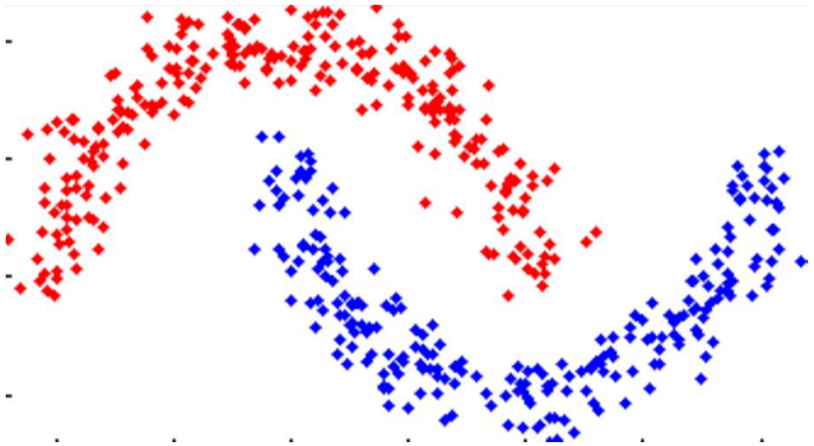
- Graph Construciton
  - K-nearest neighbors
  - ε-neighbourhood
  - Other more sophisticated methods can be found in the literature, i.e., Lee and Verleysen 2007.

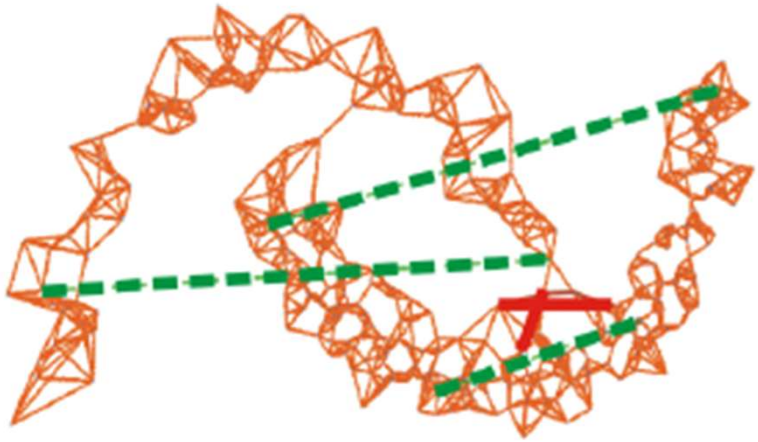
- Thus, each data point becomes a graph node and a set of undirected weighted edges connecting nodes shows the strength of affinity (or similarity) between associated data points.

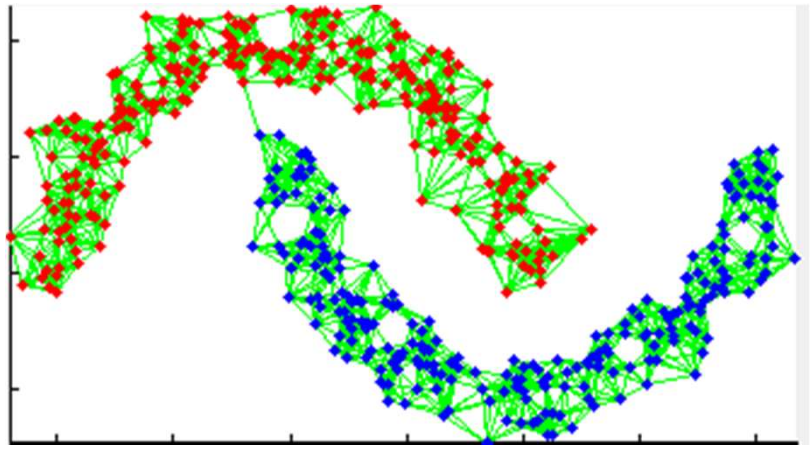




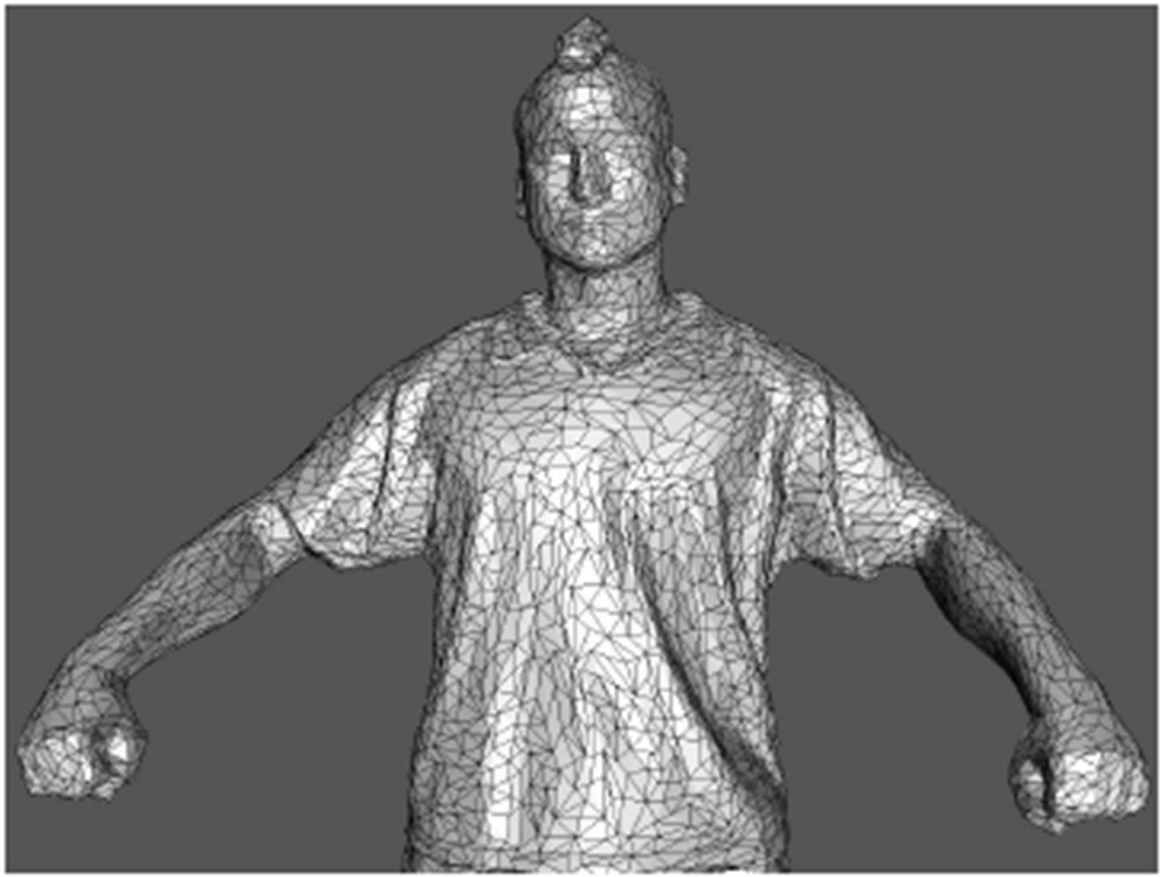
Graph Representation







Graph Representation



Clustering as Graph Partitioning

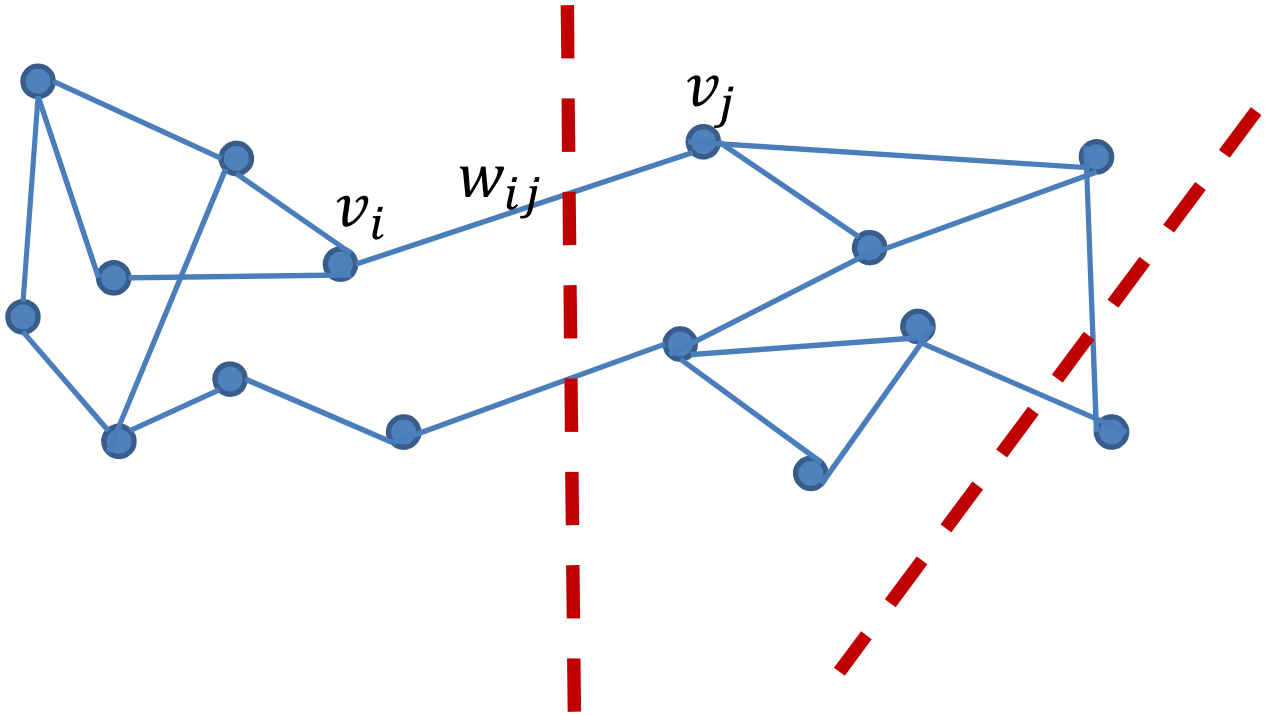
- We want to find a partition of the graph such that the edges across groups have very low weights, while the edges within a group have high weights.
- **The mincut problem**:
  - Edges across groups have very low weight, and
  - Edges within a group have high weight.
  - Choose a partition of the graph into k groups that minimizes the following criterion:

$$\text{cut}(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

$$\text{Mincut}(C_1, \ldots, C_k) = \frac{1}{2} \sum_{i=1}^{k} \text{cut}(C_i, \overline{C_i},)$$

Example: 2-way Graph Partition

- Let's represent the graph as $\boldsymbol{\mathcal{G}} = \{\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{\Omega}\}$ where $\boldsymbol{V} = \{v_1, \ldots, v_n\}$ be the set of nodes corresponding to data points $\mathbf{X} = \{x_1, \ldots, x_n\}$, $\boldsymbol{E}$ being the set of edges and $\boldsymbol{\Omega}$ being the set of respective edge weights.



Ration Cut and Normalized Cut

- Ratio cut (Hagen & Kahng 1992) minimizes:

$$\text{RatioCut}(C_1, \dots, C_1) = \frac{1}{2}\sum_{i=1}^{k} \frac{\text{cut}(C_i, \overline{C}_i)}{|C_i|}$$

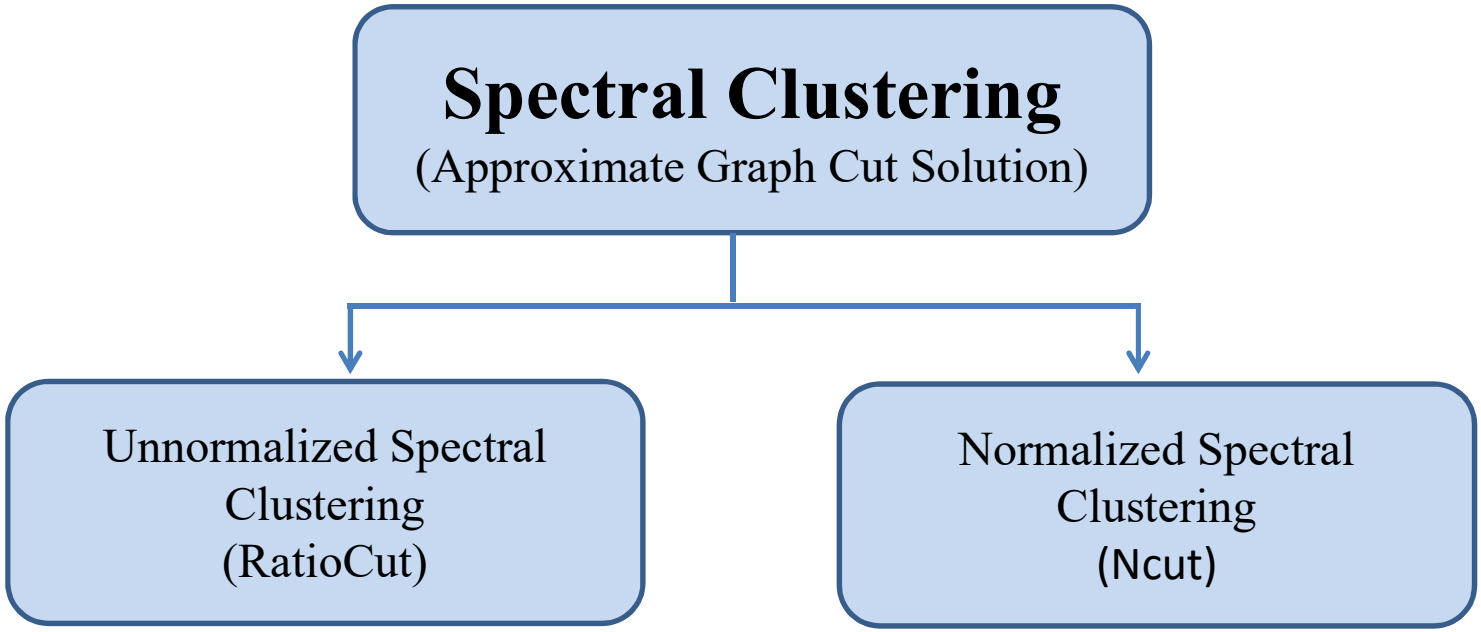
- Normalized cut (Shi & Malik 2000) minimizes:

$$\text{NCut}(C_1, \dots, C_1) = \frac{1}{2}\sum_{i=1}^{k} \frac{\text{cut}(C_i, \overline{C_i,})}{\text{vol}(C_i)}$$

<< Both Ratio cut and Normalized cut minimizations are NP-hard problems >>

Spectral Clustering

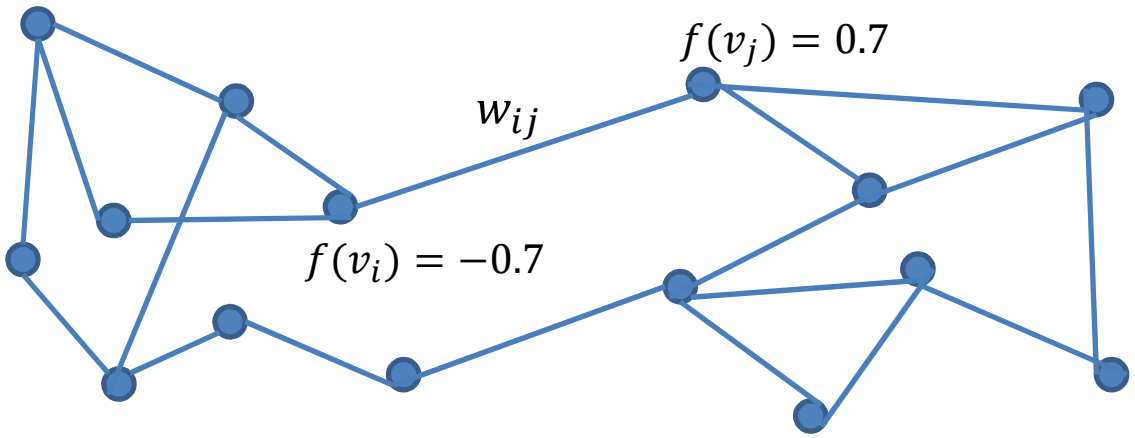
- Spectral clustering is a way to solve relaxed versions of these problems.



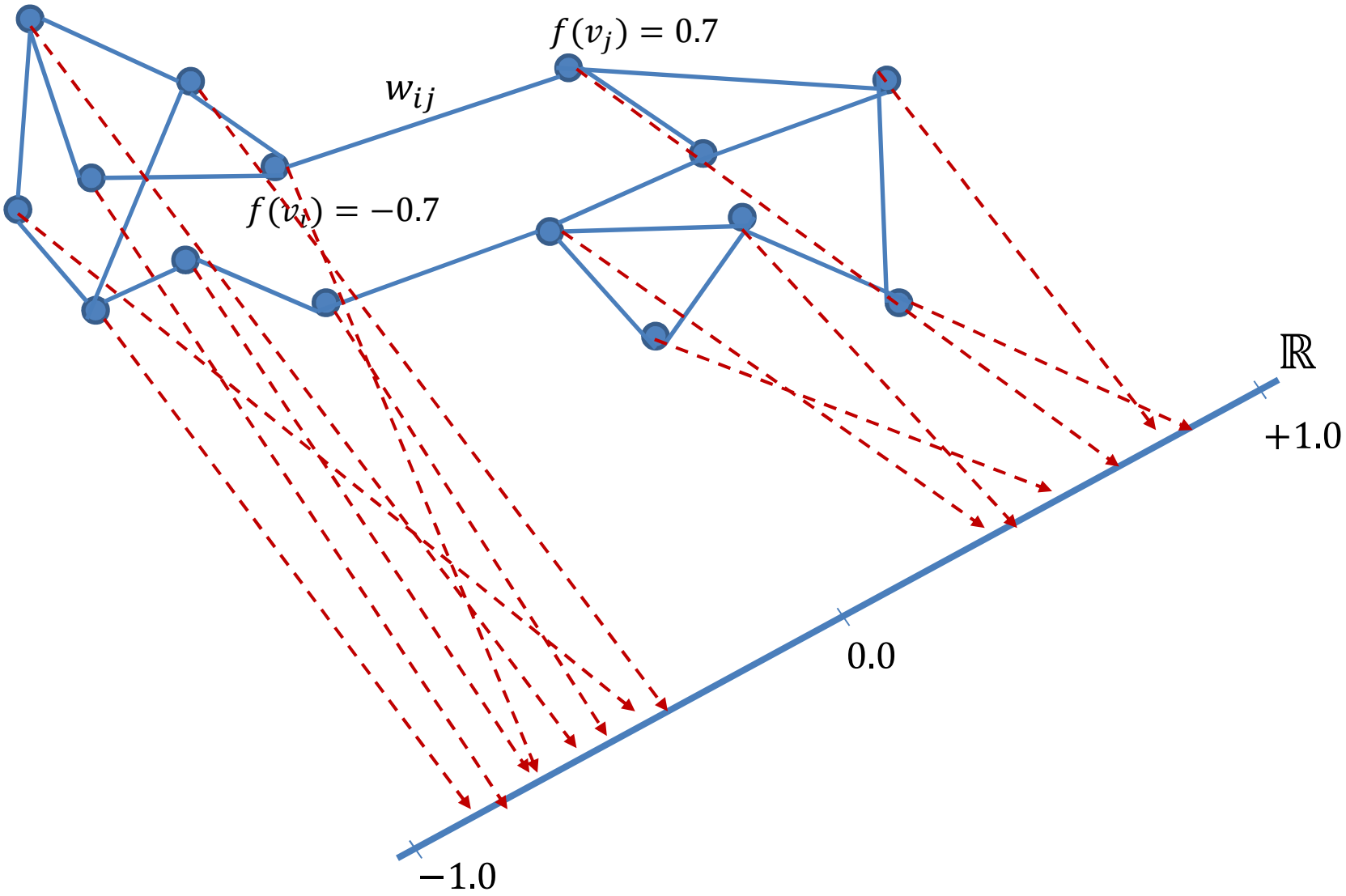
Relaxed Version (2-way Partitioning)

- Let's define discrete function over graph nodes such that:

$$\boldsymbol{f} : \boldsymbol{V} \rightarrow \mathbb{R}, \qquad \boldsymbol{f} = [f(v_1), \ldots, f(v_n)]^T$$



Graph Projection for Partitioning



Relaxed Version (2-way Partitioning)

- Let's define discrete function over graph nodes such that:

$$\boldsymbol{f} : \boldsymbol{V} \rightarrow \mathbb{R}, \qquad \boldsymbol{f} = [f(v_1), \ldots, f(v_n)]^T$$

- Let's restrict the function values such that :

$$f(v_i) = \begin{cases} \sqrt{|\bar{C}|/|C|} & \text{if } v_i \in C, \\ -\sqrt{|C|/|\bar{C}|} & \text{if } v_i \in \bar{C} \end{cases}$$

$$|\boldsymbol{V}| = |C| + |\bar{C}| \qquad \text{Mincut}(A,B) = \frac{1}{2}\sum_{i\in A, j\in B} w_{ij} \qquad f(v_i) = \begin{cases} \sqrt{|\bar{C}|/|C|} & \text{if } v_i \in C, \\ -\sqrt{|C|/|\bar{C}|} & \text{if } v_i \in \bar{C} \end{cases}$$

$$|\boldsymbol{V}|.\,\text{RatioCut}(C,\bar{C}) = (|C| + |\bar{C}|).\left(\frac{\text{Mincut}(C,\bar{C})}{|C|} + \frac{\text{Mincut}(\bar{C},C)}{|\bar{C}|}\right)$$

$$= \text{Mincut}(C,\bar{C}).\left(\frac{(|C| + |\bar{C}|)}{|C|} + \frac{(|C| + |\bar{C}|)}{|\bar{C}|}\right)$$

$$= \text{Mincut}(C,\bar{C}).\left(\frac{|\bar{C}|}{|C|} + \frac{|C|}{|\bar{C}|} + 2\right)$$

$$= \frac{1}{2}\sum_{i\in C, j\in \bar{C}} w_{ij}\left(\frac{|\bar{C}|}{|C|} + \frac{|C|}{|\bar{C}|} + 2\right) + \frac{1}{2}\sum_{i\in \bar{C}, j\in C} w_{ij}\left(\frac{|\bar{C}|}{|C|} + \frac{|C|}{|\bar{C}|} + 2\right)$$

$$= \frac{1}{2}\sum_{i\in C, j\in \bar{C}} w_{ij}\left(\sqrt{\frac{|\bar{C}|}{|C|}} + \sqrt{\frac{|C|}{|\bar{C}|}}\right)^2 + \frac{1}{2}\sum_{i\in \bar{C}, j\in C} w_{ij}\left(-\sqrt{\frac{|\bar{C}|}{|C|}} - \sqrt{\frac{|C|}{|\bar{C}|}}\right)^2$$

$$= \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}\big(f(v_i) - f(v_j)\big)^2$$

Quadratic Form

Minimization of Quadratic form

- Minimization:

$$\min \text{RatioCut}(C, \bar{C}) \approx \min |\boldsymbol{V}|.\,\text{RatioCut}(C, \bar{C}) = \min \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} \big(f(v_i) - f(v_j)\big)^2$$

- Subject to:
  - Orthogonal to a unit vector constraint i.e., $\boldsymbol{f} \perp \mathbb{1}$.

$$\sum_i^n f(v_i) = \sum_{i \in C} \sqrt{|\bar{C}|/|C|} - \sum_{i \in \bar{C}} \sqrt{|C|/|\bar{C}|} = |C|. \sqrt{|\bar{C}|/|C|} - |\bar{C}|. \sqrt{|C|/|\bar{C}|} = 0$$

  - Fixed norm constraint .

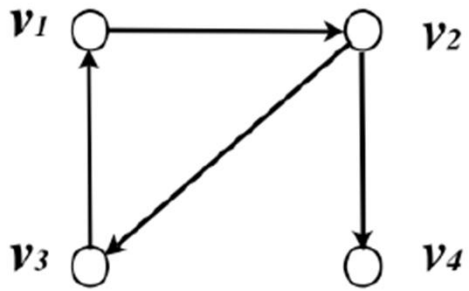
$$\|\boldsymbol{f}\| = \sum_i^n f(v_i)^2 = |C|.{}^{|\bar{C}|}/_{|C|} + |\bar{C}|.{}^{|C|}/_{|\bar{C}|} = |\bar{C}| + |C| = n$$

Graph Incidence Matrix

- Let each edge in the graph have an arbitrary but fixed orientation;
- The **incidence matrix** of a graph is a $|\boldsymbol{E}| \times |\boldsymbol{V}|$ $(m \times n)$ matrix defined as follows:

$$\nabla := \begin{cases} \nabla_{ev} = -1 & \text{if } v \text{ is the initial vertex of edge } e \\ \nabla_{ev} = 1 & \text{if } v \text{ is the terminal vertex of edge } e \\ \nabla_{ev} = 0 & \text{if } v \text{ is not in } e \end{cases}$$

$$\nabla = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & +1 \end{bmatrix}$$



Graph Incidence Matrix as Discrete Differential Operator

$$(\bigtriangledown \boldsymbol{f})(e_{ij}) = f(v_j) - f(v_i)$$

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & +1 \end{bmatrix} \begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \end{pmatrix} = \begin{pmatrix} f(2) - f(1) \\ f(1) - f(3) \\ f(3) - f(2) \\ f(4) - f(2) \end{pmatrix}$$

Graph Laplacian Matrix

$\mathbf{L} = \bigtriangledown^{\top} \bigtriangledown$

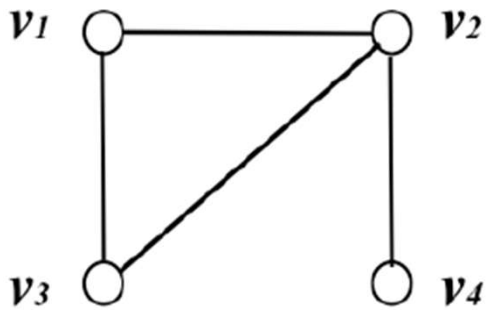
$(\mathbf{L}\boldsymbol{f})(v_i) = \sum_{v_j \sim v_i} (f(v_i) - f(v_j))$

Connection between the Laplacian and the adjacency matrices:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

The degree matrix: $\mathbf{D} := D_{ii} = d(v_i)$.

$$\mathbf{L} = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$



Graph Laplacian Matrix

$\mathbf{L} = \mathbf{D} - \mathbf{\Omega}$

The Laplacian as an operator:

$$(\mathbf{L}\boldsymbol{f})(v_i) = \sum_{v_j \sim v_i} w_{ij}(f(v_i) - f(v_j))$$

As a quadratic form:

$$\boldsymbol{f}^\top \mathbf{L} \boldsymbol{f} = \frac{1}{2} \sum_{e_{ij}} w_{ij}(f(v_i) - f(v_j))^2$$

$\mathbf{L}$ is symmetric and positive semi-definite

$\mathbf{L}$ has n non-negative, real-valued eigenvalues:
$0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.

Relaxation for Graph Partitioning

- Both graph Laplacian and RatioCut have the same quadratic form for an undirected weighted graph.

$$\min_{C \subset V} \text{RatioCut}(C, \bar{C}) \approx \min_{C \subset V} \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} \left(f(v_i) - f(v_j) \right)^2$$

Subject to Constraints $\boldsymbol{f} \perp \mathbb{1}, \quad \|\boldsymbol{f}\| = n$

$$\approx \min_{\boldsymbol{f} \in \mathbb{R}^n} \boldsymbol{f}^T \mathbf{L} \boldsymbol{f}$$ Subject to Constraints $\boldsymbol{f} \perp \mathbb{1}, \quad \|\boldsymbol{f}\| = n$

- Solution to this least square optimization problem is given by the eigen-values/vectors (spectra) of Laplacian matrix*.

*Rayleigh-Ritz theorem

Graph Laplacian Spectra for Single Connected Component Graph

$\mathbf{L}\boldsymbol{u} = \lambda \boldsymbol{u}$.

$\mathbf{L}\mathbf{1} = \mathbf{0}$, $\lambda_1 = 0$ is the smallest eigenvalue.

The *one* vector: $\mathbf{1} = (1 \dots 1)^\top$.

$0 = \boldsymbol{u}^\top \mathbf{L} \boldsymbol{u} = \sum_{i,j=1}^{n} w_{ij}(u(v_i) - u(v_j))^2$.

If any two vertices are connected by a path, then $\boldsymbol{u} = (u(v_1), \dots, u(v_n))$ needs to be constant at all vertices such that the quadratic form vanishes. Therefore, a graph with one connected component has the constant vector $\boldsymbol{u}_1 = \mathbf{1}$ as the only eigenvector with eigenvalue 0.

Relaxation for Graph Partitioning

- The second non-zero eigenvalue and corresponding eigenvector (also known as Fiedler vector) provides a solution for relaxed RatioCut ($k = 2$).

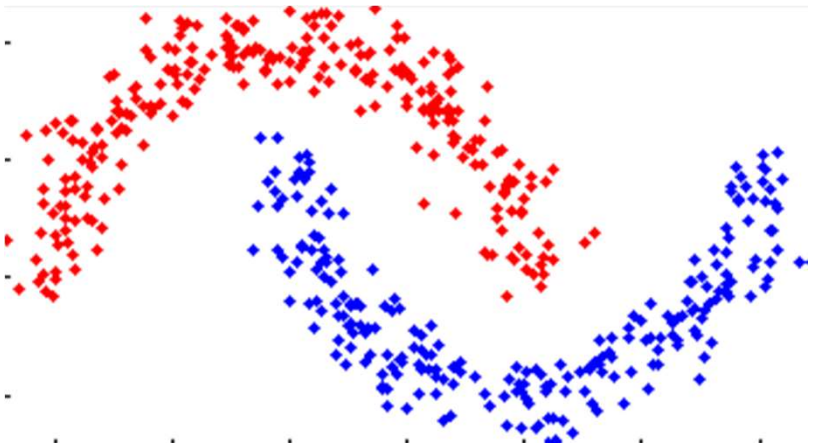
$$\begin{cases} v_i \in C & \text{if} \quad f(v_i) \geq 0 \\ v_i \in \bar{C} & \text{if} \quad f(v_i) < 0 \end{cases}$$

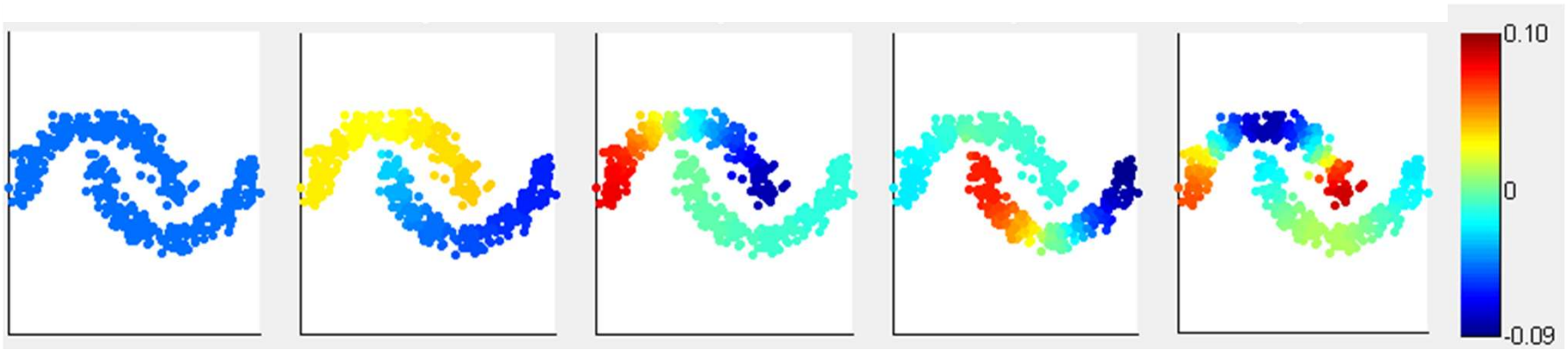
- Similarly, relaxed graph partitioning can be obtained for Normalized Cut with normalized graph Laplacian (J. Shi and J. Malik 2000).

$$\mathbf{L}_N = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{\Omega}\mathbf{D}^{-1/2}$$

- This analysis can be generalized to higher eigenvectors when $k > 2$

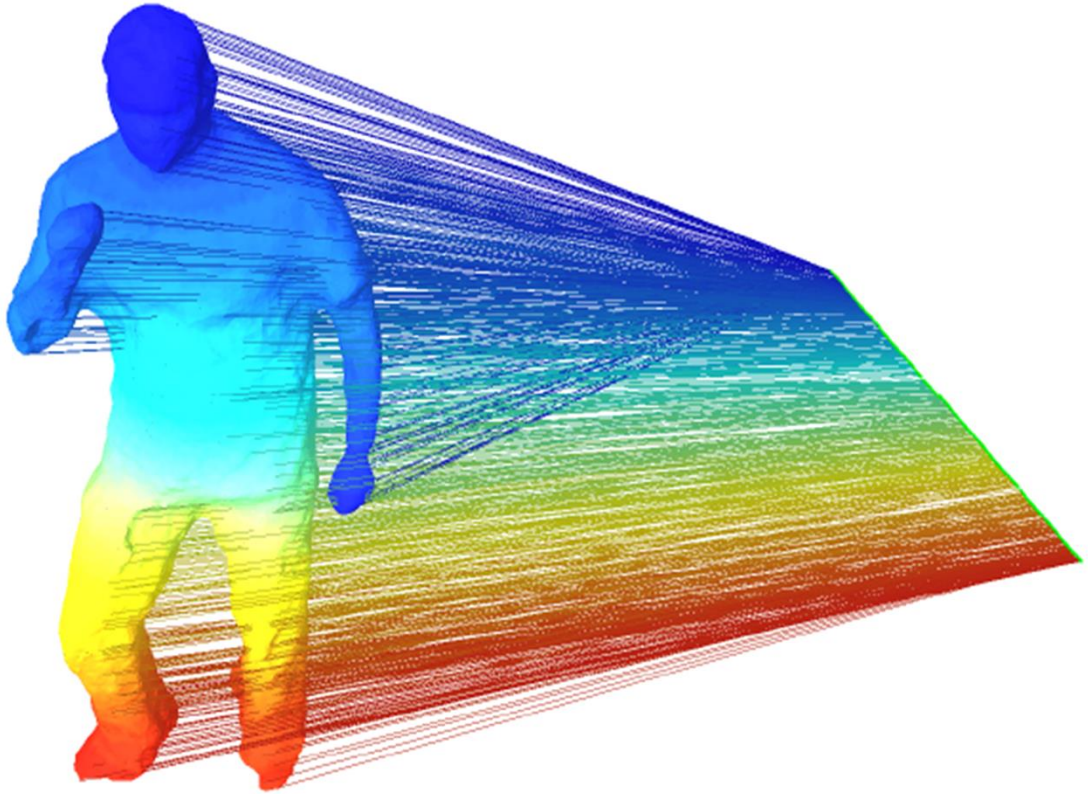
Visualization of Laplacian Eigenvector



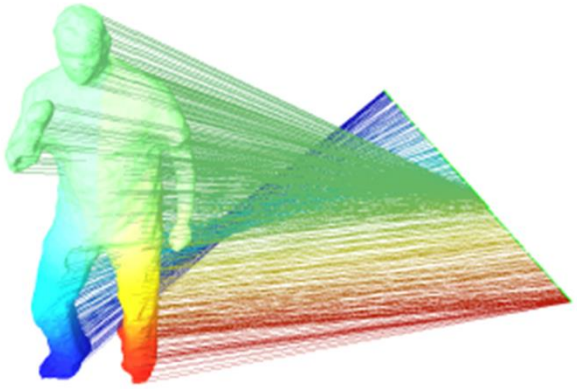


u_1 u_2 u_3 u_4 u_5

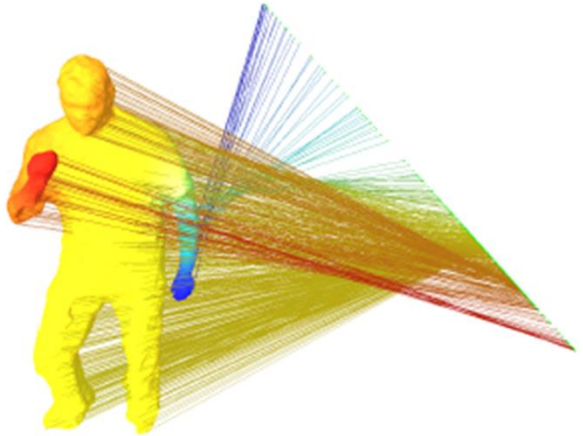
Visualization of Laplacian Eigenvector



u_2



u_3



u_4

Graph Laplacian Spectra for Multiple Connected Component Graph

Each connected component has an associated Laplacian.
Therefore, we can write matrix $\mathbf{L}$ as a *block diagonal matrix*:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_1 & & \\ & \ddots & \\ & & \mathbf{L}_k \end{bmatrix}$$

The spectrum of $\mathbf{L}$ is given by the union of the spectra of $\mathbf{L}_i$.

Each block corresponds to a connected component, hence each matrix $\mathbf{L}_i$ has an eigenvalue 0 with multiplicity 1.

The spectrum of $\mathbf{L}$ is given by the union of the spectra of $\mathbf{L}_i$.

The eigenvalue $\lambda_1 = 0$ has multiplicity k.

Graph Spectral Embedding

Compute the eigendecomposition $\mathbf{L}_C = \mathbf{D} - \mathbf{\Omega} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$.

Select the k smallest non-null eigenvalues $\lambda_2 \leq \ldots \leq \lambda_{k+1}$

$\lambda_{k+2} - \lambda_{k+1} =$ **eigengap**.

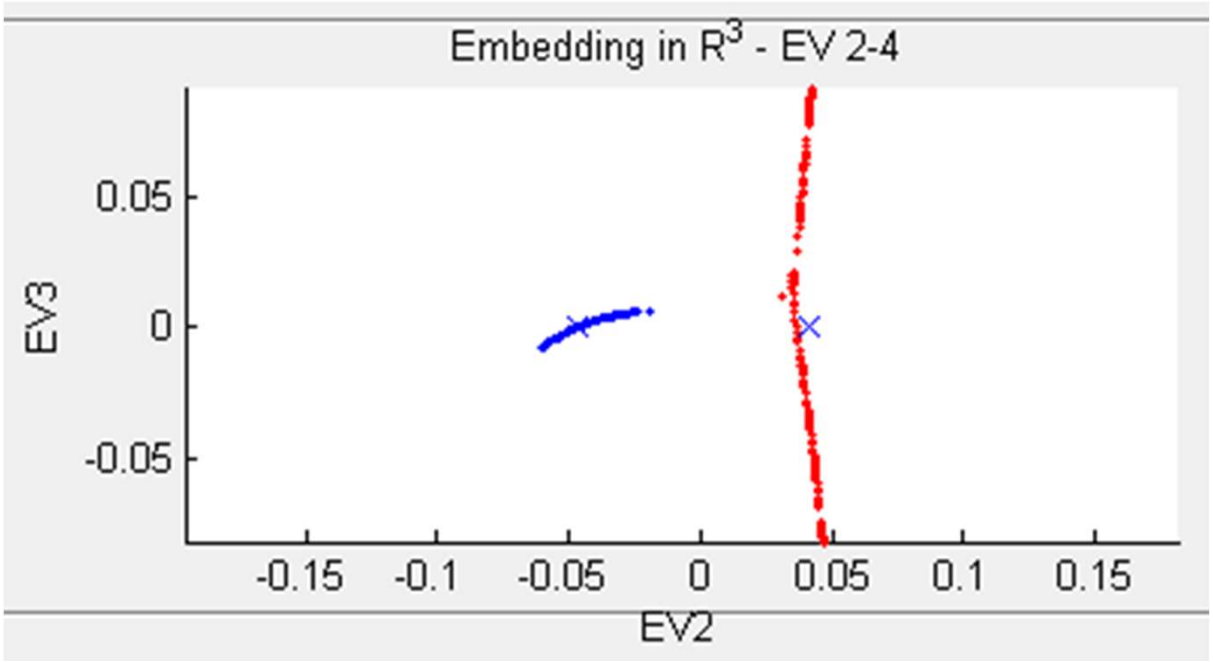
We obtain the $n \times k$ column-orthogonal matrix $\widetilde{\mathbf{U}} = [\boldsymbol{u}_2 \ldots \boldsymbol{u}_{k+1}]$:

$$\widetilde{\mathbf{U}} = \begin{bmatrix} \boldsymbol{u}_2(v_1) & \ldots & \boldsymbol{u}_{k+1}(v_1) \\ \vdots & & \vdots \\ \boldsymbol{u}_2(v_n) & \ldots & \boldsymbol{u}_{k+1}(v_n) \end{bmatrix}$$

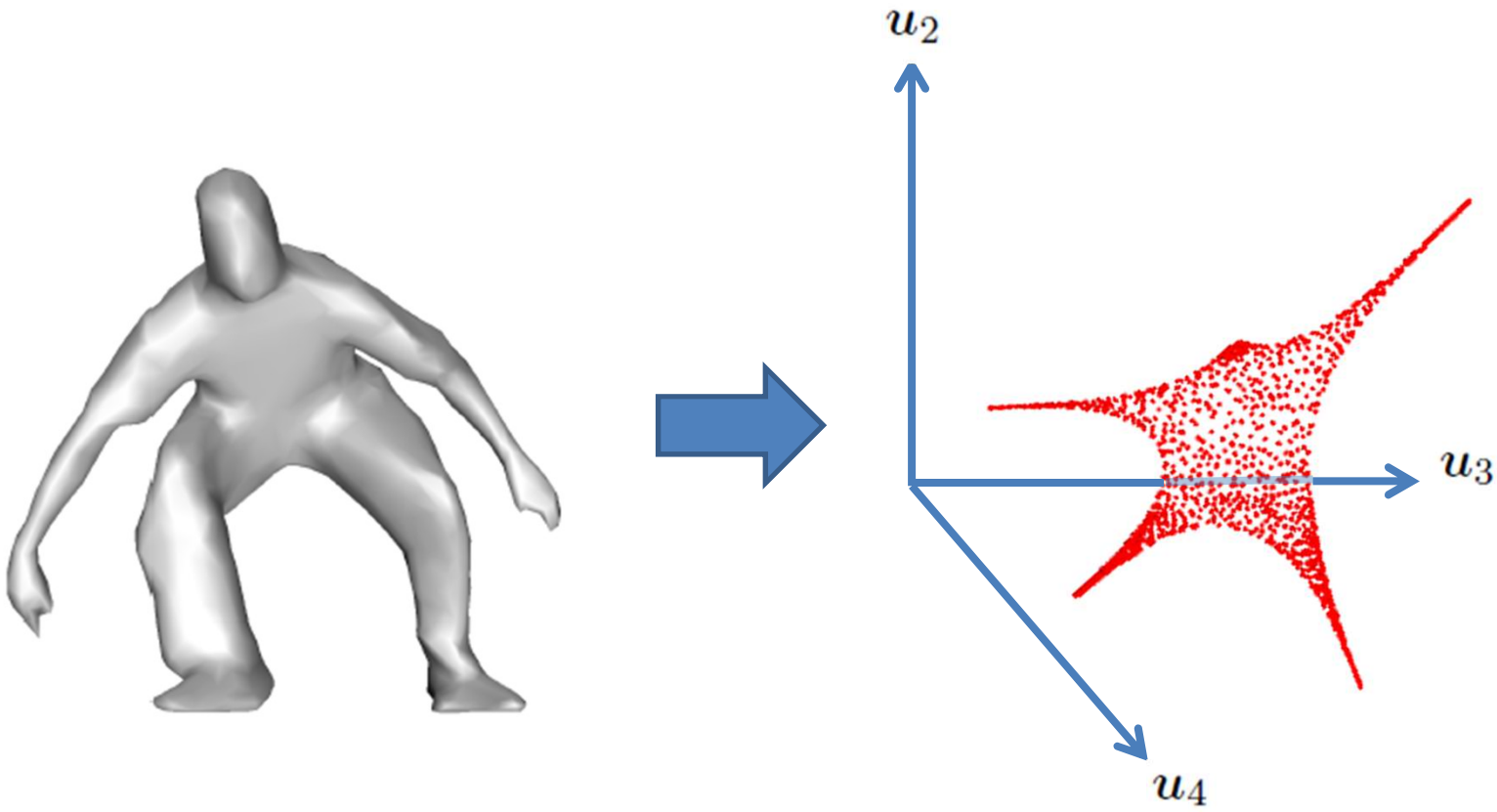
Embedding: The i-row of this matrix correspond to the representation of vertex v_I in the $\mathbb{R}^k$ basis spanned by the orthonormal vector basis $\boldsymbol{u}_2, \ldots, \boldsymbol{u}_{k+1}$.

Therefore: $\mathbf{Y} = [\boldsymbol{y}_1 \ldots \boldsymbol{y}_i \ldots \boldsymbol{y}_n] = \widetilde{\mathbf{U}}^\top$

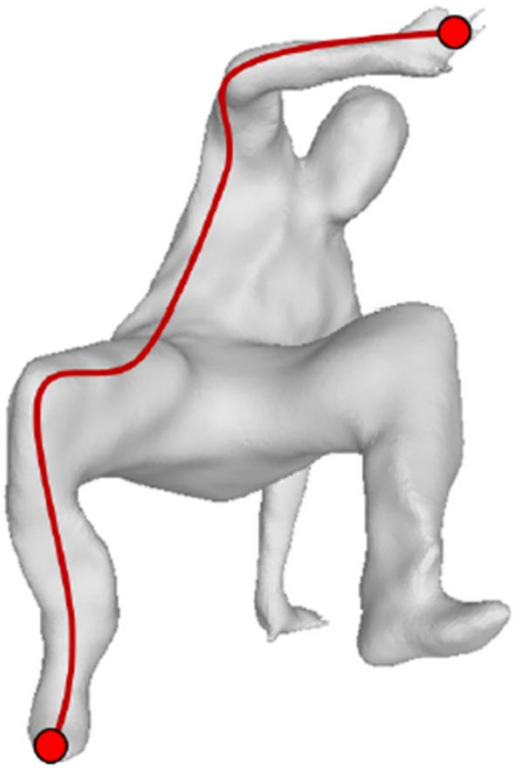
Laplacian Embedding Visualization

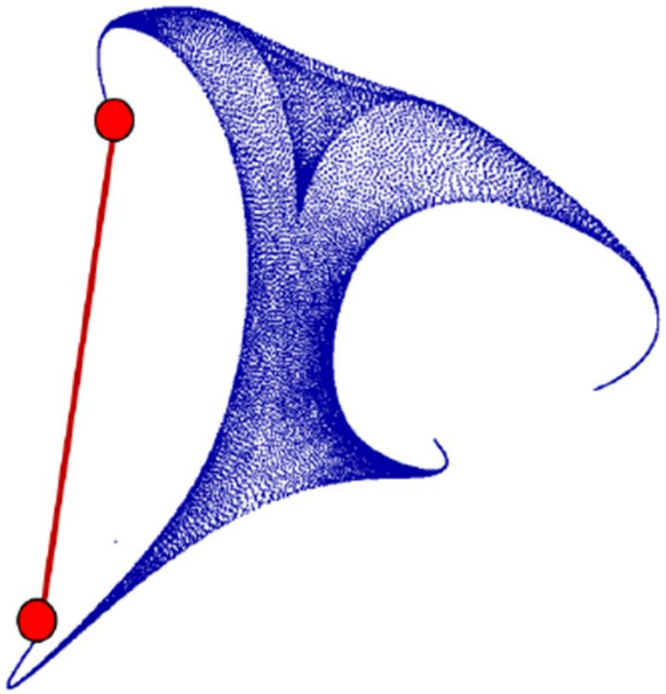


Laplacian Embedding Visualization



A Distance Preserving Mapping





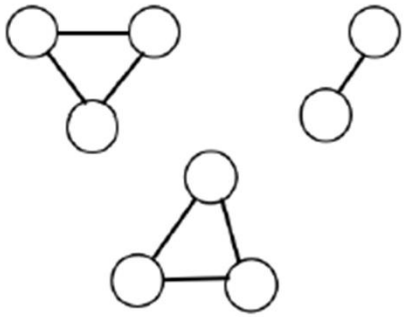
$$d_{geodesic}(x_i, x_j) \approx \|y_i - y_j\|$$

Un-normalized Spectral Clustering

- Input: Laplacian matrix ($\mathbf{L}$) and the number (k) of clusters to compute.
- Output: Cluster $C_1, \dots, C_k$

1. Compute eigen-decomposition of $\mathbf{L}$ matrix : $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\boldsymbol{T}}$
   where, $\mathbf{U} = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_n]$ and $\boldsymbol{\Lambda} = \mathrm{diag}([\lambda_1, \dots, \lambda_n])$
2. Define k-dimensional graph embedding using the eigenvectors associated with k smallest eigenvalues: $\mathbf{Y} = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_{\boldsymbol{k}}]^T$
3. Cluster the columns y_i for $i = 1, \dots, n$ into k clusters using the K-means clustering algorithm.

Spectral Clustering: Ideal Case

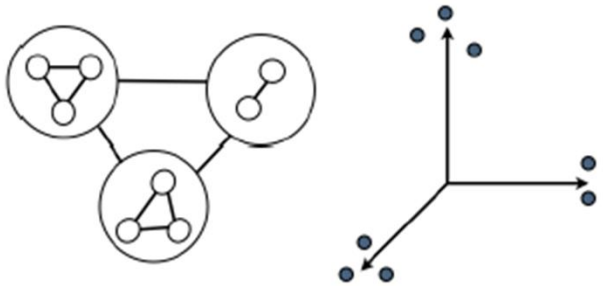


- $\lambda_1 = \lambda_2 = \lambda_3 = 0$
- $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$ form an orthonormal basis.
- The connected components collapse to $(100), (010), (001)$.
- Clustering is trivial in this case.

$$\mathbf{U} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Spectral Clustering: Perturbed Case

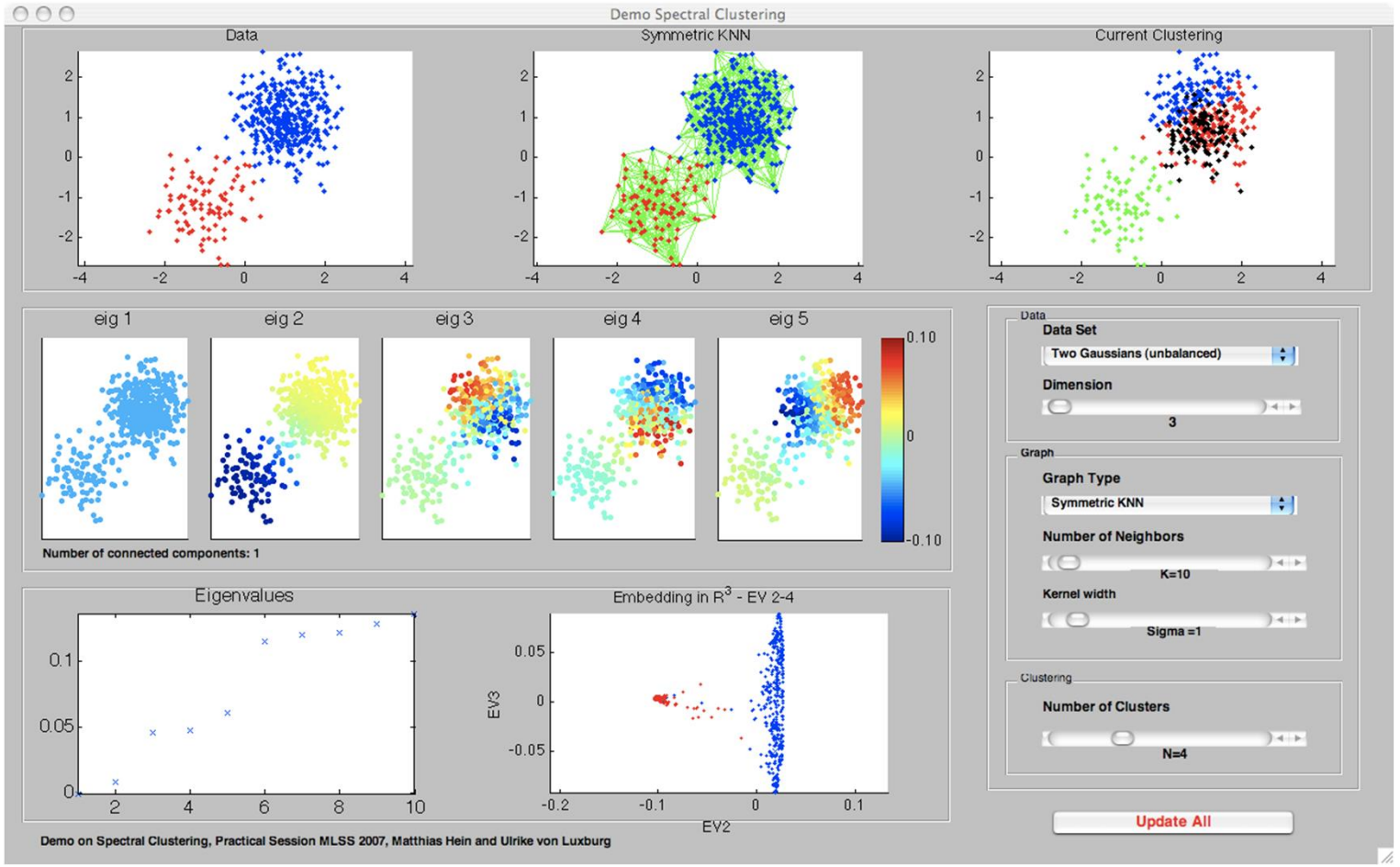


- See (von Luxburg '07) for a detailed analysis.
- The connected components are no longer *disconnected*, but they are only connected by few edges with low weight.

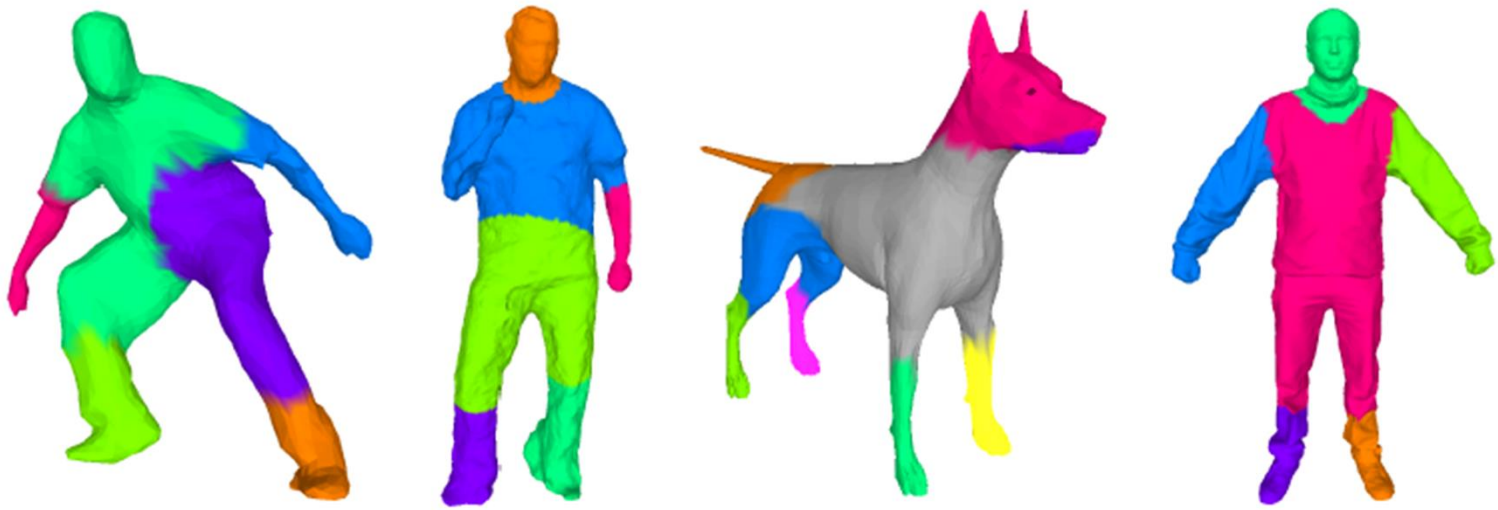
- The Laplacian is a perturbed version of the ideal case.
- Choosing the first k nonzero eigenvalues is easier the larger the eigengap between λ_{k+1} and λ_{k+2}.
- The fact that the first k eigenvectors of the perturbed case are approximately piecewise constant depends on $|\lambda_{k+2} - \lambda_{k+1}|$.
- Choosing k is a crucial issue.

Demo

- http://www.ml.uni-saarland.de/code/GraphDemo/DemoSpectralClustering.htm



Shape Segmentation Results



Practical Aspects

- What is the best graph construction ?
  - Choice of graph induction
  - Choice of weights (similarity function)
- Choice of Laplacian
  - Un-normalized (Combinatorial)
  - Normalized
- Eigen-decomposition
  - Scalability
  - Accuracy
- How to decide k ?
  - Eigengap analysis
  - Choose different k for embedding size and k-means

Kernel Kmeans

- Minimize $J = \sum_{i=1}^{N} \sum_{j=1}^{K} a_{ij} \left\| \varphi(\mathbf{x}_i) - \widetilde{\boldsymbol{\mu}_j} \right\|^2$

  such that $a_{ij} \epsilon \{0,1\}$ and $\sum_{j=1}^{K} a_{ij} = 1$

$$\widetilde{\boldsymbol{\mu}_j} = \frac{\sum_{i=1}^{N} a_{ij} \varphi(\mathbf{x}_i)}{\sum_{i=1}^{N} a_{ij}}$$

- We can rewrite the criterion function as:

  Minimize $J = \text{trace}(G) - \text{trace}(AGA^T)$

  Or, Maximize $\text{trace}(AGA^T)$

where, G is an $N \times N$ kernel matrix and A is the optimal normalized cluster membership matrix

Related but not discussed here

- Random walk Laplacian
  - Transition Matrix
  - Commute Time Distance
  - Page Rank Algorithm (Google Search)
- Manifold Learning
  - Dimensionality Reduction Techniques
  - Non-linear Kernel Extensions
  - Relationship to Geometric Laplacian
- Spectral Clustering Variants
  - Constraint Spectral Clustering
- Nodal Domain and Sets of Eigenvectors

References

- **Hagen & Kahng 1992** New Spectral Methods for Ratio Cut Partitioning and Clustering", IEEE Trans. on CAD 11(9), September 1992, pp. 1074-1085.
- **Chung 1997** Spectral Graph Theory. 1997. (Chapter 1)
- **Shi & Malik 2000** Normalized Cuts and Image Segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 22(8): 888-905
- **Belkin & Niyogi 2003** Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. Neural Computation, 15, 1373-1396 .
- **Luxburg 2007** A Tutorial on Spectral Clustering. Statistics and Computing, 17(4), 395-416 (An excellent paper)
- **Lee & Verleysen 2007** Nonlinear Dimensionality Reduction, Springer.

Tutorial Slides

- **Chris Ding** : A Tutorial on Spectral Clustering
- **Zitao Liu** : Spectral Clustering
- **Eyal David** : Spectral Clustering
- **Radu Horaud** : Lecture notes on Manifold Learning for Signal and Visual Processing