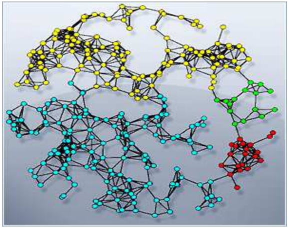
Clustering
Graphs, Spectra and Semidefinite Programming

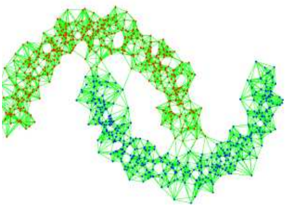
Tony Jebara

April 13, 2015

What is Clustering?

- Split n items into k partitions to minimize some **Cost**
- **Given:** dataset $\{x_1, \ldots, x_n\}$ where $x_i \in \Omega$ and $k \in \mathbb{Z}$
- **Output:** $\mathcal{X}_1, \ldots, \mathcal{X}_k \subseteq \{1, \ldots, n\}$
  such that $\mathcal{X}_i \cap \mathcal{X}_j = \{\}$, $\cup_{i=1}^{k} \mathcal{X}_i = \{1, \ldots, n\}$





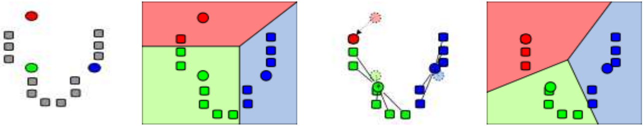
What is Clustering?

- Split n items into k partitions to minimize some **Cost**
- **Given:** dataset $\{x_1, \dots, x_n\}$ where $x_i \in \Omega$ and $k \in \mathbb{Z}$
- **Output:** $\mathcal{X}_1, \dots, \mathcal{X}_k \subseteq \{1, \dots, n\}$
  such that $\mathcal{X}_i \cap \mathcal{X}_j = \{\}$, $\cup_{i=1}^{k} \mathcal{X}_i = \{1, \dots, n\}$
- Additional possible assumptions
  - The x_i are independent identically distributed (iid) from $p(x)$
  - We are given a distance $d(x_i, x_j)$ or kernel $\kappa(x_i, x_j) = K_{ij}$, equivalent since $d(x_i, x_j) \equiv \sqrt{\kappa(x_i, x_i) - 2\kappa(x_i, x_j) + \kappa(x_j, x_j)}$
    e.g.

Linear (Euclidean)	$\kappa(x_i, x_j) = x_i^\top x_j$
Polynomial	$\kappa(x_i, x_j) = (x_i^\top x_j + 1)^p$
Radial Basis Function	$\kappa(x_i, x_j) = \exp(-\|x_i - x_j\|^2/\sigma^2)$
Laplace	$\kappa(x_i, x_j) = \exp(-\|x_i - x_j\|/\sigma)$

. . . but what **Cost** function to use?

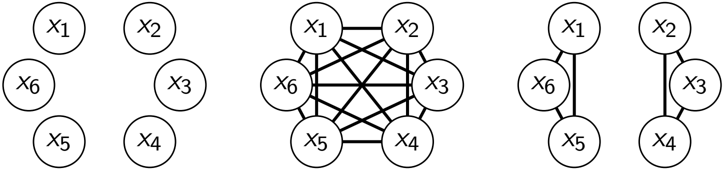
k-means - Lloyd 1957



- k-means minimizes the **Cost**
  $\min_{\mathcal{X}_1,\ldots,\mathcal{X}_k} \sum_{i=1}^{k} \sum_{j\in\mathcal{X}_i} \|x_j - \frac{1}{|\mathcal{X}_i|}\sum_{m\in\mathcal{X}_i} x_m\|^2$
- Kernelized k-means minimizes the following:
  $\sum_{i=1}^{k}\left(\sum_{j\in\mathcal{X}_i} K_{jj} - 2\sum_{j,m\in\mathcal{X}_i}\frac{1}{|\mathcal{X}_i|}K_{jm} + \frac{1}{|\mathcal{X}_i|}\sum_{j,m\in\mathcal{X}_i} K_{jm}\right)$

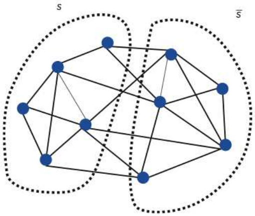
GREEDY KERNEL KMEANS (Dhillon et al. 04):
1. Initialize $\mathcal{X}_1,\ldots,\mathcal{X}_k$ randomly
2. Set $z_j = \arg\min_i \frac{1}{|\mathcal{X}_i|^2}\sum_{l,m\in\mathcal{X}_i}(K_{lm} - 2K_{jm})$
3. Set $\mathcal{X}_i = \{j : z_j = i\}$
4. If not converged goto 2

Clustering as Graph Partition



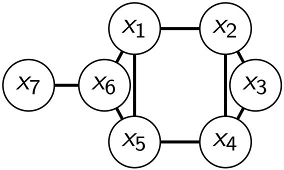
- Try clustering as graph partition problem (Shi & Malik 2000)
- Make $\{x_1, \ldots, x_n\}$ an undirected graph $G = (V, E)$ of vertices $V = \{1, \ldots, n\}$ and edges $E = \{(i, j) : i < j \in \{1, \ldots, n\}\}$
- Get adjacency $W \in \mathbb{R}^{n \times n}$ via $W_{ij} = \kappa(x_i, x_j)$ and $W_{ii} = 0$
- Clustering $\equiv$ cutting graph into vertex subsets $S_1, \ldots, S_k$
- Define a weight over two *sets* as $W(A, B) = \sum_{i \in A, j \in B} W_{ij}$
- We want cuts with big intra-cluster weight $W(S_i, S_i)$ and small inter-cluster weight $W(S_i, S_j)$

Problem with k-Means



- Let's only consider $k = 2$ and recover $S = V_1$ and $\bar{S} = V_2$
- Use $W_{ij} = \kappa(x_i, x_j)$ and assume $W_{ii} = const = 0$.
- The k-means **Cost** becomes $\min_S -\frac{W(S,S)}{|S|} - \frac{W(\bar{S},\bar{S})}{|\bar{S}|}$
- Problem: k-means ignores $W(S, \bar{S})$, the amount of *cutting*
- Let's consider alternative **Cost** functions...

Unbalanced Graph Partition



- One cost function that considers cutting is
  min-cut $\quad \min_S W(S, \bar{S})$
- This can be solved optimally in polynomial time!
- Problem: it can give trivially small partitions, in the above it just disconnects x_7 from the graph...
- We need both $|S|$ and $|\bar{S}|$ to be balanced!

Balanced Graph Partition Cost Functions

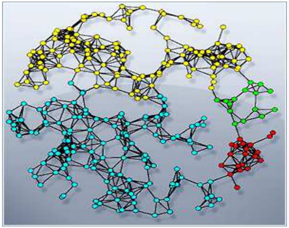
- There are many balanced graph partition cost functions

$$
\begin{aligned}
\text{k-means} &\quad \min_S \phi(S) = -\frac{W(S,S)}{|S|} - \frac{W(\bar{S},\bar{S})}{|\bar{S}|} \\
\text{sparse cut} &\quad \min_S \phi(S) = \frac{W(S,\bar{S})}{|S||\bar{S}|/n} \\
\text{ratio cut} &\quad \min_S \phi(S) = \frac{W(S,\bar{S})}{|S|} + \frac{W(S,\bar{S})}{|\bar{S}|} \\
\text{expansion} &\quad \min_S \phi(S) = \frac{W(S,\bar{S})}{\min(|S|,|\bar{S}|)} \\
\text{normalized cut} &\quad \min_S \phi(S) = \frac{W(S,\bar{S})}{W(S,S)} + \frac{W(S,\bar{S})}{W(\bar{S},\bar{S})}
\end{aligned}
$$

- All are NP-hard to *solve* (Ambuhl et al. 2007) or *approximate* within a constant factor (Konstantin & Harald '04). Can't find $\hat{S}$ such that $\phi(\hat{S}) \leq O(1) \min_S \phi(S)$ in polynomial time!
- Need efficient algorithms where factor grows slowly with n
$O(n) \geq O(\sqrt{n}) \geq O(\sqrt{\log n}) \geq O(\log \log n) \geq O(1)$

Balanced Graph Partition Cost Functions beyond k=2

- We can extend beyond 2-way cuts to multi-way cuts
- For example, normalized cut for $k = 2$ is $\min_{S_1,S_2} \phi(S_1, S_2) = \frac{W(S_1,S_2)}{W(S_1,S_1)} + \frac{W(S_1,S_2)}{W(S_2,S_2)}$
- Multi-way normalized cut for $k > 2$ is simply $\min_{S_1,S_2,\ldots,S_k} \sum_{i=1}^{k} \sum_{j=i+1}^{k} \phi(S_i, S_j)$



Equivalence of Cost Functions

Lemma

The cost functions satisfy
$expansion(S) < ratio\ cut(S) \leq 2 \times expansion(S)$

Lemma

The minima of the cost functions satisfy
$\min_S expansion(S) \leq \min_S sparse\ cut(S) \leq 2 \times \min_S expansions(S)$

Lemma

For b-regular graphs, $W \in \mathbb{B}^{n\times n}$, $\sum_i W_{ij} = b$, $W_{ii} = 0$, $W_{ij} = W_{ji}$
we have $normalized\ cut(S) = ratio\ cut(S)/b$

- So, let's focus on sparse cut $\phi^* = \min_s \phi(S) = \min_S \frac{W(S,\bar{S})}{|S||\bar{S}|/n}$ and consider spectral heuristics for minimizing it

Spectral Cut - Donath & Hoffman 1973

SpectralCut: Input regular adjacency matrix W. Output cut $\hat{S}$

1. Compute the 2nd eigenvector $\mathbf{v} \in \mathbb{R}^n$ of W
2. For $i = 1, \ldots, n$ create partition $\hat{S}_i = \{j : \mathbf{v}_j \leq \mathbf{v}_i\}$
3. Output $\hat{S} = \hat{S}_i$ with smallest sparse cut $i = \arg\min_i \phi(\hat{S}_i)$

Theorem (Alon & Milman 1985, Chung 1997)

Given a b-regular graph, SpectralCut *provides a cut* $\hat{S}$ *that achieves a sparse cut value* $\phi(\hat{S}) \leq \sqrt{8b\phi^*}$

Corollary

Given a b-regular graph, SpectralCut *provides a cut* $\hat{S}$ *that achieves a sparse cut value* $\phi(\hat{S}) \leq O(\sqrt{n})\phi^*$

Spectral Cut - Donath & Hoffman 1973

Proof. (Alon & Milman 1985, Chung 1997).

Clearly, $W\mathbf{1} = b\mathbf{1}$ so $\lambda_1 = b$ and $\lambda_2 = \max_{\mathbf{x}\in\mathbb{R}^n, \mathbf{x}\perp\mathbf{1}} \frac{\mathbf{x}^T W \mathbf{x}}{\mathbf{x}^T\mathbf{x}}$.

It is easy to show that $b - \lambda_2 \leq \phi^*$ by relaxing the minimization

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{\sum_{ij} W_{ij}(\mathbf{x}_i - \mathbf{x}_j)^2}{\frac{1}{n}\sum_{ij}(\mathbf{x}_i - \mathbf{x}_j)^2} \leq \min_{\mathbf{x}\in\{-1,1\}^n} \frac{\sum_{ij} W_{ij}(\mathbf{x}_i - \mathbf{x}_j)^2}{\frac{1}{n}\sum_{ij}(\mathbf{x}_i - \mathbf{x}_j)^2}.$$

Define $\hat{\mathbf{v}} \propto \mathbf{v}$ the 2^{nd} eigenvector such that $\max_i \hat{\mathbf{v}}_i - \min_i \hat{\mathbf{v}}_i = 1$.

Select cut S by picking t uniformly in $t \in [\min_i \hat{\mathbf{v}}_i, \max_i \hat{\mathbf{v}}_i]$.

Probability edge (i,j) is in cut $(S, \bar{S})$ is proportional to $|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}j|$.

Note $\mathrm{E}_t[W(S,\bar{S})] = \sum_{ij} W_{ij} \frac{|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j|}{2}$ and $\mathrm{E}_t[|S||\bar{S}|] = \sum_{ij} \frac{|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j|}{2}$.

Sampling t achieves $\frac{W(S,\bar{S})}{\frac{1}{n}|S||\bar{S}|} \leq \frac{\sum_{ij} W_{ij}|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j|}{\sum_{ij}|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j|}$.

Min over $\mathbf{v}$ gives $\phi_{sc} = \min_{\mathbf{v}\in\mathbb{R}^n} \frac{\sum_{ij} W_{ij}|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j|}{\frac{1}{n}\sum_{ij}|\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j|} = \min_{\mathbf{v}\in\mathbb{R}^n} \frac{\sum_{ij} W_{ij}|\mathbf{v}_i - \mathbf{v}_j|}{\frac{1}{n}\sum_{ij}|\mathbf{v}_i - \mathbf{v}_j|}$

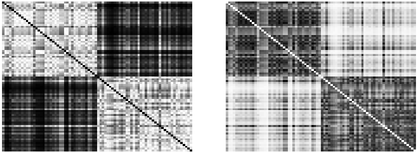
A few more steps yield $\phi_{sc} \leq \sqrt{8b(b - \lambda_2)}$. □

Spectral Cut - Shi & Malik 2000

- A continuous relaxation of Normalized Cut
- Use eigenvectors of the Laplacian to find partition

SHIMALIKCUT: Input adjacency matrix W. Output cut $\hat{S}$

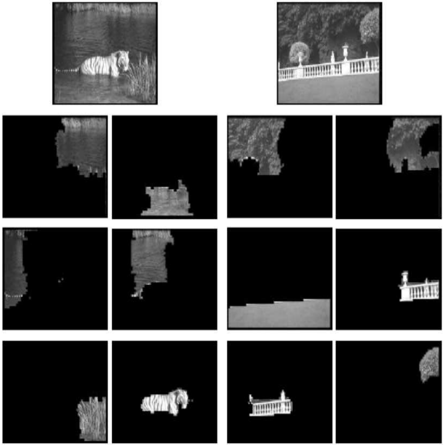
1. Define diagonal $\Delta \in \mathbb{R}^{n \times n}$ as $\Delta_{ii} = \sum_j W_{ij}$
2. Get Laplacian $L = I - \Delta^{-1/2} W \Delta^{-1/2}$
3. Compute second smallest 2nd eigenvector $\mathbf{v} \in \mathbb{R}^n$ of L
4. Create partition $\hat{S} = \{j : \mathbf{v}_j \leq median(\mathbf{v})\}$



W L

Spectral Cut - Shi & Malik 2000

Results of eigenvectors on $(D - W)y = \lambda Dy$

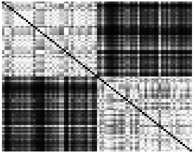


Spectral Cut - Ng, Jordan & Weiss 2001

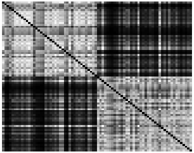
- A slight normalization procedure is applied to SHIMALIKCUT
- Helps improve eigenvector stability

NJWCUT: Input adjacency matrix W. Output cut $\hat{S}$

1. Define diagonal $\Delta \in \mathbb{R}^{n \times n}$ as $\Delta_{ii} = \sum_j W_{ij}$
2. Get normalized Laplacian $\mathcal{L} = \Delta^{-1/2} W \Delta^{-1/2}$
3. Obtain $\mathbf{v}, \mathbf{w}$ as largest eigenvectors of $\mathcal{L}$ and form $X = [\mathbf{v}\ \mathbf{w}]$
4. Form $Y \in \mathbb{R}^{n \times 2}$ as $Y_{ij} = X_{ij} / \sqrt{X_{i1}^2 + X_{i2}^2}$
5. Taking each row of Y as a point in $\mathbb{R}^2$, obtain $\hat{S}$ via k-means

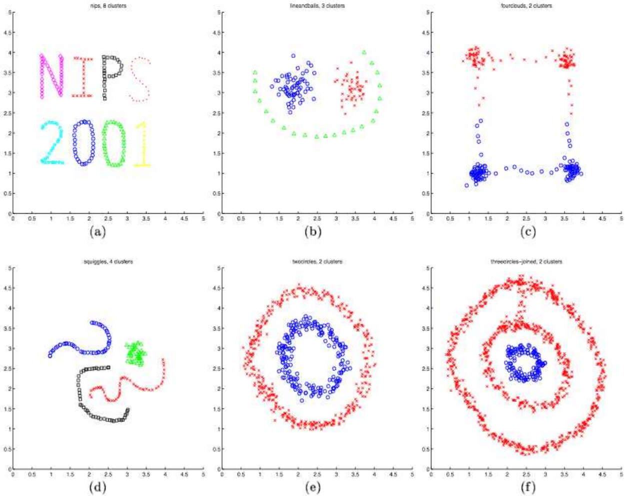


W

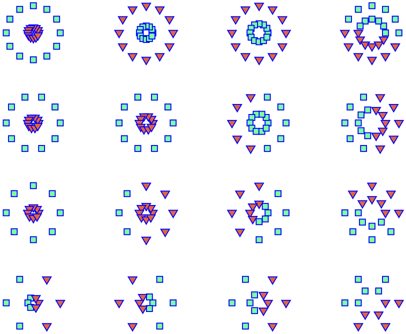


$\mathcal{L}$

Spectral Cut - Ng, Jordan & Weiss 2001

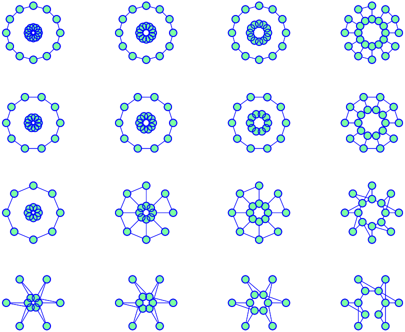


Irregularity Problems with Spectral Methods



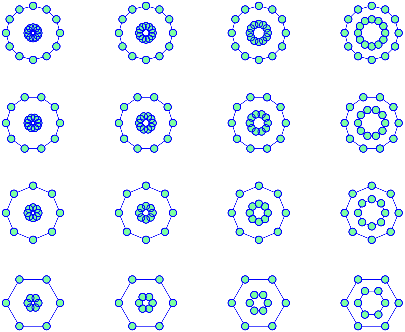
- Problems even if multiple values of σ used in RBF kernel.
- The previous spectral methods fail for some situations.
- Suboptimality of spectral methods if the graph is irregular.

Irregularity Problems with Spectral Methods



- Try pruning the graph with k-nearest neighbors.
- Get popularity problem as interior points over-selected.
- Still end up with irregular graph due to greediness.

Irregularity Problems with Spectral Methods



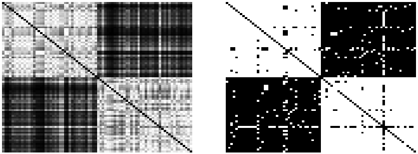
- Prune graph with b-matching, gives perfectly regular graph.
- Minimizes distance while creating exactly b edges per node.
- Max-product takes $O(n^3)$ (Huang & Jebara 2007)

B-Matched Spectral Cut - Jebara & Shchogolev 2006

- First run b-matching on the points to get a regular graph
- Then use NJWCUT on the graph to get a partition

BMATCHCUT: Input kernel matrix K. Output cut $\hat{S}$

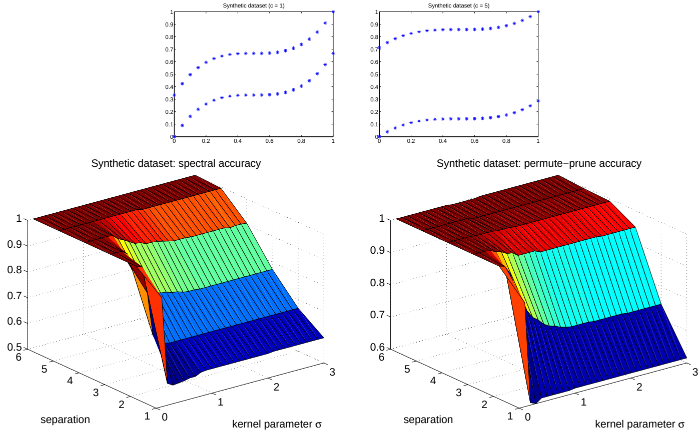
1. Compute distance matrix $D \in \mathbb{R}^{n \times n}$ as $D_{ij} = \sqrt{K_{ii} - 2K_{ij} + K_{jj}}$
2. Set $b = \lfloor n/2 \rfloor$
3. $A = \arg\min_{A \in \mathbb{B}^{n \times n}} \sum_{ij} A_{ij} D_{ij} \;\; s.t. \sum_i A_{ij} = b, A_{ij} = A_{ji}, A_{ii} = 0$
4. Run NJWCUT on A



W A

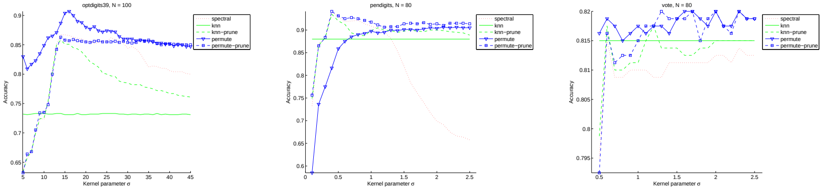
B-Matched Spectral Cut - Jebara & Shchogolev 2006

- Cluster two S curves varying separation and σ in RBF kernel
- Compare NJWCUT to BMATCHCUT



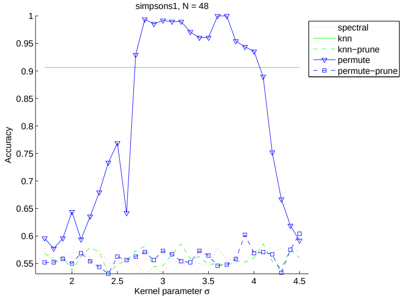
B-Matched Spectral Cut - Jebara & Shchogolev 2006

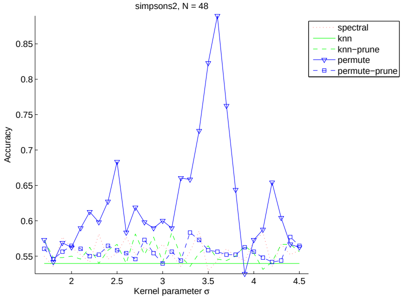
- UCI experiments varying σ in RBF kernel
- Compare NJWCUT to BMATCHCUT to KNNCUT.



B-Matched Spectral Cut - Jebara & Shchogolev 2006

- Video clustering experiments varying σ in RBF kernel
- Compare NJWCut to BMatchCut to KNNCut.





Equivalence of Spectral Algorithms for $O(\sqrt{n})$

Lemma

For regular graphs, $\textsc{bMatchCut} = \textsc{NJWCut}$.

Lemma

For regular graphs, $\phi_{\textsc{NJW}} \geq \phi_{\textsc{Spectral}}$ *and* $\phi_{\textsc{ShiMalik}} \geq \phi_{\textsc{Spectral}}$.

Proof.

$\Delta = bI$ so eigenvectors of L, W and $\mathcal{L}$ are the same.
Top eigenvector of $\mathcal{L}$ is constant so NJW normalization is same.
SpectralCut tries all thresholds so more thorough rounding. □

Theorem

Thus, all these spectral algorithms achieve a factor of $O(\sqrt{n})$.

Graph Partition Beyond $O(\sqrt{n})$

- Linear programming obtains $O(\log n)$ (Leighton & Rao 1999)
- Best guarantee is $O(\sqrt{\log n})$ (Arora, Rao & Vazirani 2004)
- Solve the following semidefinite programming (SDP)

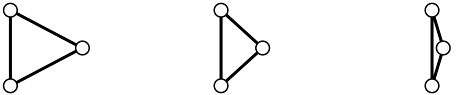
$$\min_{Y} \sum_{i \neq j} W_{ij} \|y_i - y_j\|^2$$

$$s.t. \|y_i - y_j\|^2 + \|y_j - y_k\|^2 \geq \|y_i - y_k\|^2, \sum_{i<j} \|y_i - y_j\|^2 = 1$$

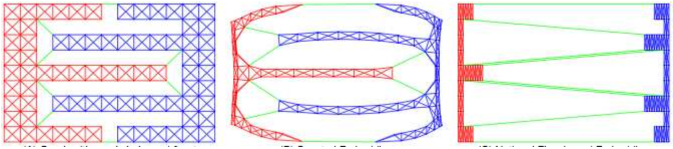
- This semidefinite program finds an embedding of the graph
- Each $y_i \in \mathbb{R}^n$ is the coordinate of vertex i
- SDP ensures connected points with large W_{ij} are close by
- The constraint $\sum_{i<j} \|y_i - y_j\|^2 = 1$ fixes size of embedding
- Uses ℓ_2^2 constraints $\|y_i - y_j\|^2 + \|y_j - y_k\|^2 \geq \|y_i - y_k\|^2$

SDP Graph Partition with $O(\sqrt{\log n})$

- What is an ℓ_2^2 embedding?
- All triples satisfy $\|y_i - y_j\|^2 + \|y_j - y_k\|^2 \geq \|y_i - y_k\|^2$
- In d dimensions, there can only be 2^d such points
- Any triangle of points cannot subtend an obtuse angle



$\theta < 90$ $\theta = 90$ $\theta > 90$ ×



Graph with 8-cut Spectral Embedding ARV Embedding

SDP Graph Partition with $O(\sqrt{\log n})$

ARVEMBED: Input adjacency matrix W. Output $\{y_1, \ldots, y_n\}$.
$\beta = \min_{y_1,\ldots,y_n} \sum_{ij} W_{ij} \|y_i - y_j\|^2$
s.t. $\|y_i - y_j\|^2 + \|y_j - y_k\|^2 \geq \|y_i - y_k\|^2$, $\sum_{i<j} \|y_i - y_j\|^2 = 1$.

ARVCUT: Input embedding $\{y_1, \ldots, y_n\}$. Output cut $\hat{S}$.

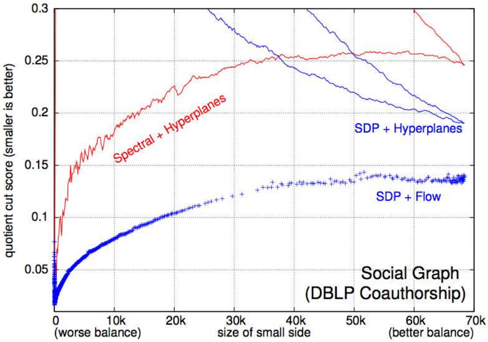
1. Sample $\vec{u} \in \mathbb{R}^d$ from a zero mean, identity covariance Gaussian.
2. Find $m = \frac{1}{n} \sum_i y_i^\top \vec{u}$ and $v = \frac{1}{n} \sum_i (y_i^\top \vec{u} - m)^2$.
3. Let $P = \{i : y_i^\top \vec{u} \geq m + \sqrt{v}\}$ and $N = \{i : y_i^\top \vec{u} \leq m - \sqrt{v}\}$.
4. Discard pairs $y \in P$ and $\tilde{y} \in N$ such that $\|y - \tilde{y}\|^2 \leq 1/\sqrt{\log(n)}$.
5. Choose random $0 \leq r \leq 1/\sqrt{\log(n)}$
6. Output $\hat{S} = \{i : \|y_i - \hat{y}\|^2 \leq r\}$ for some $\hat{y} \in P$.

Theorem (Arora et al. 2004)

Given a graph with n vertices, algorithm ARVEMBED *followed by* ARVCUT *produces a cut $\hat{S}$ satisfying $\phi(\hat{S}) \leq O(\sqrt{\log(n)})\phi^*$*

SDP Graph Partition with $O(\sqrt{\log n})$

- ARV's semidefinite program requires $O(n^{4.5})$ time
- SDP-LR version improves social network partition (Lang 2006)
- Otherwise, still too slow for many problems



Conclusions

- Clustering can be studied as graph partition
- Most interesting cost functions are NP-hard
- Spectral methods work well but only have $O(\sqrt{n})$ guarantees
- Spectral methods can do better if input graph is regular
- Can find closest regular graph quickly via b-matching
- Semidefinite methods get $O(\sqrt{\log n})$ guarantees
- Via ℓ_2^2 property, get a better graph embedding