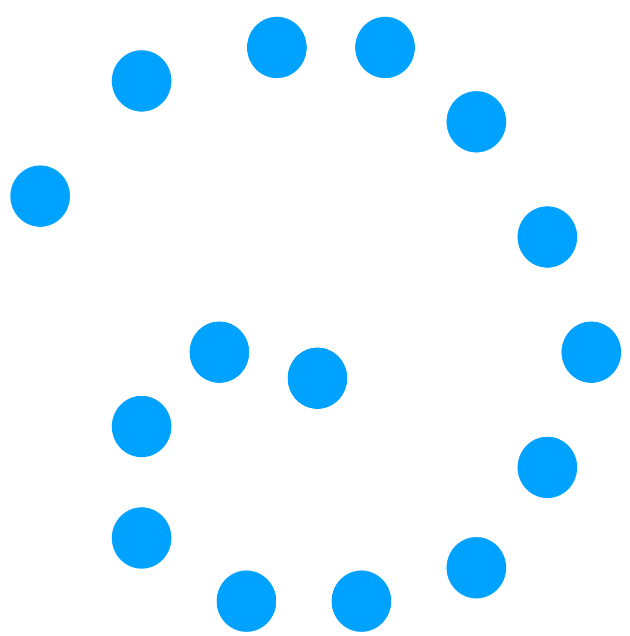
Machine Learning for Data Science (CS4786)
Lecture 9

TSNE + Spectral Embedding

Manifold Based Dimensionality Reduction

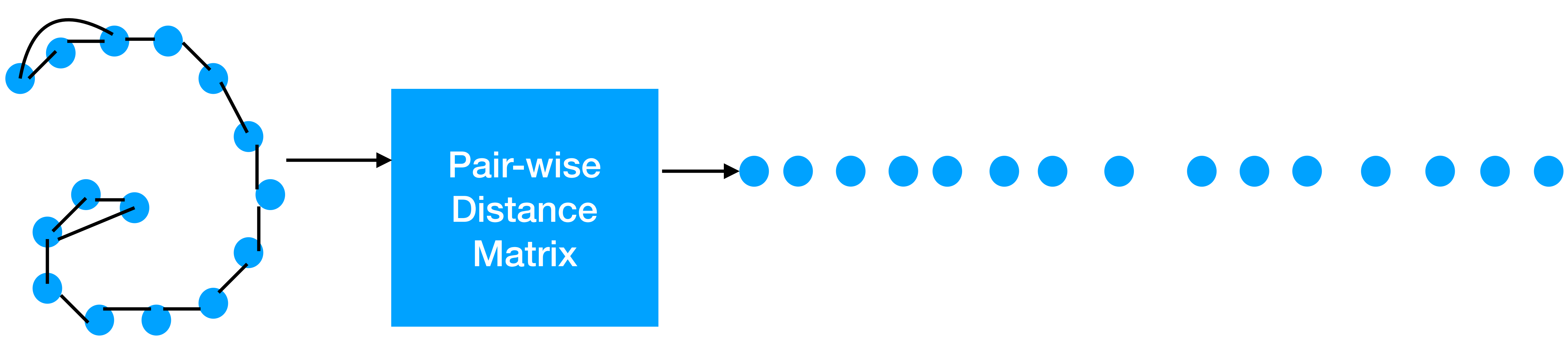
- Key Assumption: Points live on a low dimensional manifold
- Manifold: subspace that looks locally Euclidean
- Given data, can we uncover this manifold?



Can we unfold this?

Method I: Isomap

1. For every point, find its (k-) Nearest Neighbors
2. Form the Nearest Neighbor graph
3. For every pair of points A and B, distance between point A to B is shortest distance between A and B on graph
4. Find points in low dimensional space such that distances between points in this space is equal to distance on graph.



ISOMAP: PITFALLS

1. If we don't take enough nearest neighbors, then graph may not be connected
2. If we connect points too far away, points that should not be connected can get connected
3. There may not be a right number of nearest neighbors we should consider!

Stochastic Neighborhood Embedding

- Use a probabilistic notion of which points are neighbors.

- Close by points are neighbors with high probability, …
  Eg: For point $\mathbf{x}_t$, point $\mathbf{x}_s$ is picked as neighbor with probability

$$p_{t \to s} = \frac{\exp\left(-\frac{\|\mathbf{x}_s - \mathbf{x}_t\|^2}{2\sigma^2}\right)}{\sum_{u \neq t} \exp\left(-\frac{\|\mathbf{x}_u - \mathbf{x}_t\|^2}{2\sigma^2}\right)}$$

  Probability that points s and t are connected $P_{s,t} = P_{t,s} = \frac{p_{t \to s} + p_{s \to t}}{2n}$

- Goal: Find $\mathbf{y}_1, \ldots, \mathbf{y}_n$ with stochastic neighborhood distribution Q such that "P and Q are similar"

  i.e. minimize:

$$\mathrm{KL}(P\|Q) = \sum_{s,t} P_{s,t} \log\left(\frac{P_{s,t}}{Q_{s,t}}\right) = \sum_{s,t} P_{s,t} \log\left(P_{s,t}\right) - \sum_{s,t} P_{s,t} \log\left(Q_{s,t}\right)$$

CHOICE FOR Q

- Just like we defined P, we can define Q for a given $\mathbf{y}_1, \ldots, \mathbf{y}_n$ by

$$q_{t \to s} = \frac{\exp(-\frac{\|\mathbf{y}_s - \mathbf{y}_t\|^2}{2\sigma^2})}{\sum_{u \neq t} \exp(-\frac{\|\mathbf{y}_u - \mathbf{y}_t\|^2}{2\sigma^2})}$$

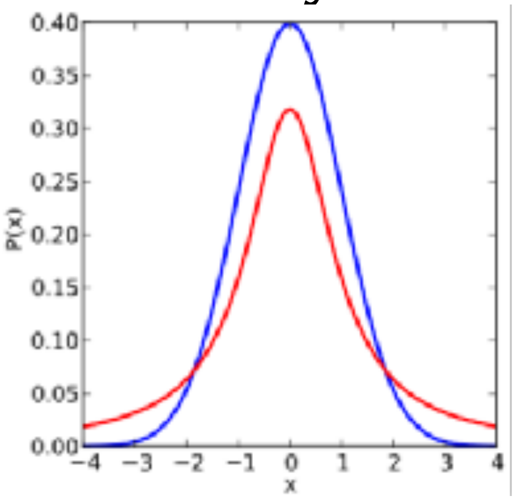
  and then set $Q_{s,t} = \frac{q_{t \to s} + q_{s \to t}}{2n}$

- However we are faced with the crowding problem:
  - In high dimension we have a lot of space, Eg. in d dimension we have $d + 1$ equidistant point
  - For d dimensional gaussians, most points are found at distance $\sqrt{d}$ from mean!
  - If we use gaussians in both high and low dimensional space, all the points are squished in to a small space
  - Too many points crowd the center!

Method II: t-SNE

- Instead for Q we use, student t distribution which is heavy tailed:

$$q_{t\to s} = \frac{\left(1 + \|\mathbf{y}_s - \mathbf{y}_t\|^2\right)^{-1}}{\sum_{u\neq t}\left(1 + \|\mathbf{y}_u - \mathbf{y}_t\|^2\right)^{-1}}$$



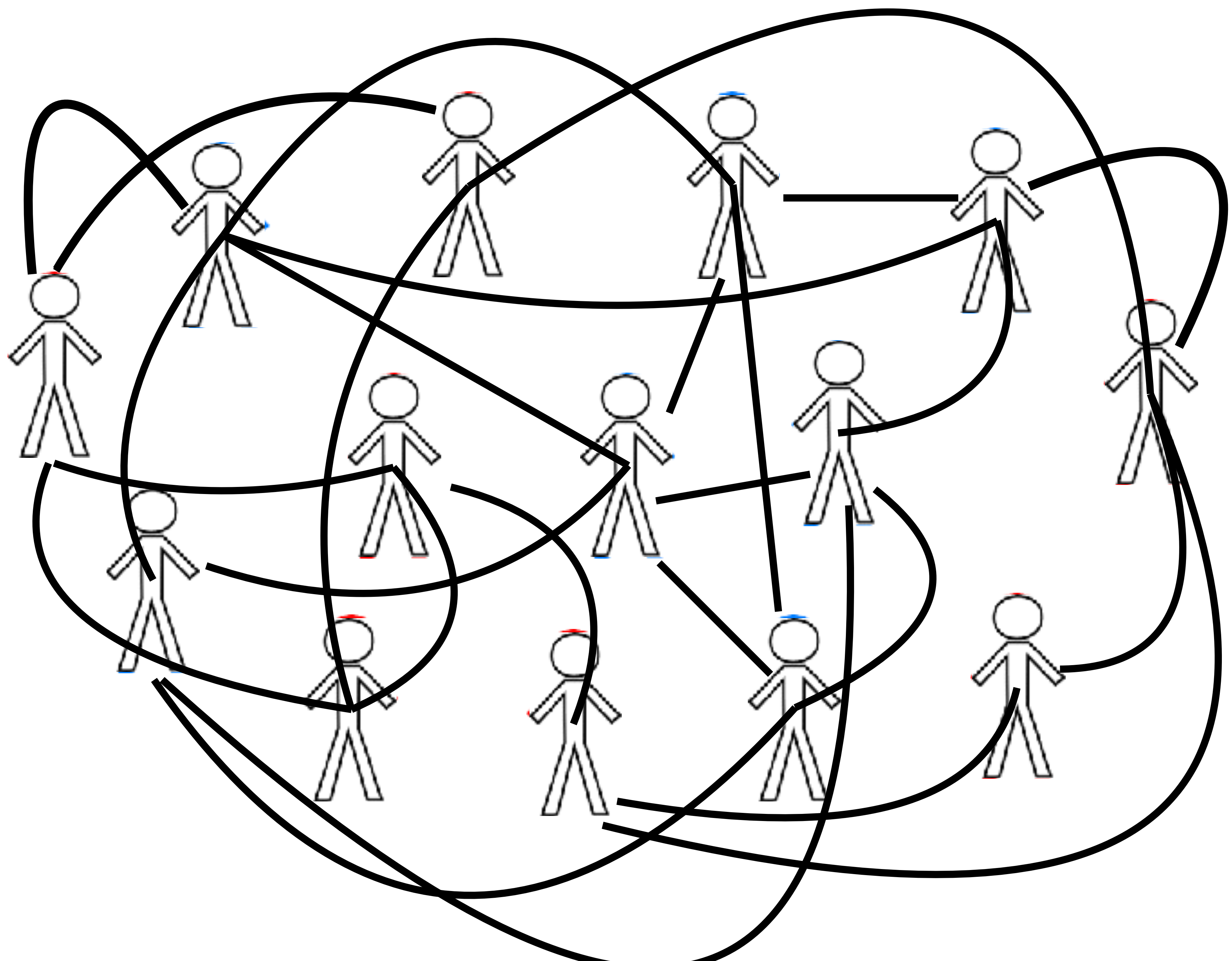
  and then set $Q_{s,t} = \frac{q_{t\to s} + q_{s\to t}}{2n}$

- It can be verified that

$$\nabla_{\mathbf{y}_t} \mathrm{KL}(P\|Q) = 4\sum_{s=1}^{n}(P_{s,t} - Q_{s,t})(\mathbf{y}_t - \mathbf{y}_s)\left(1 + \|\mathbf{y}_s - \mathbf{y}_t\|^2\right)^{-1}$$

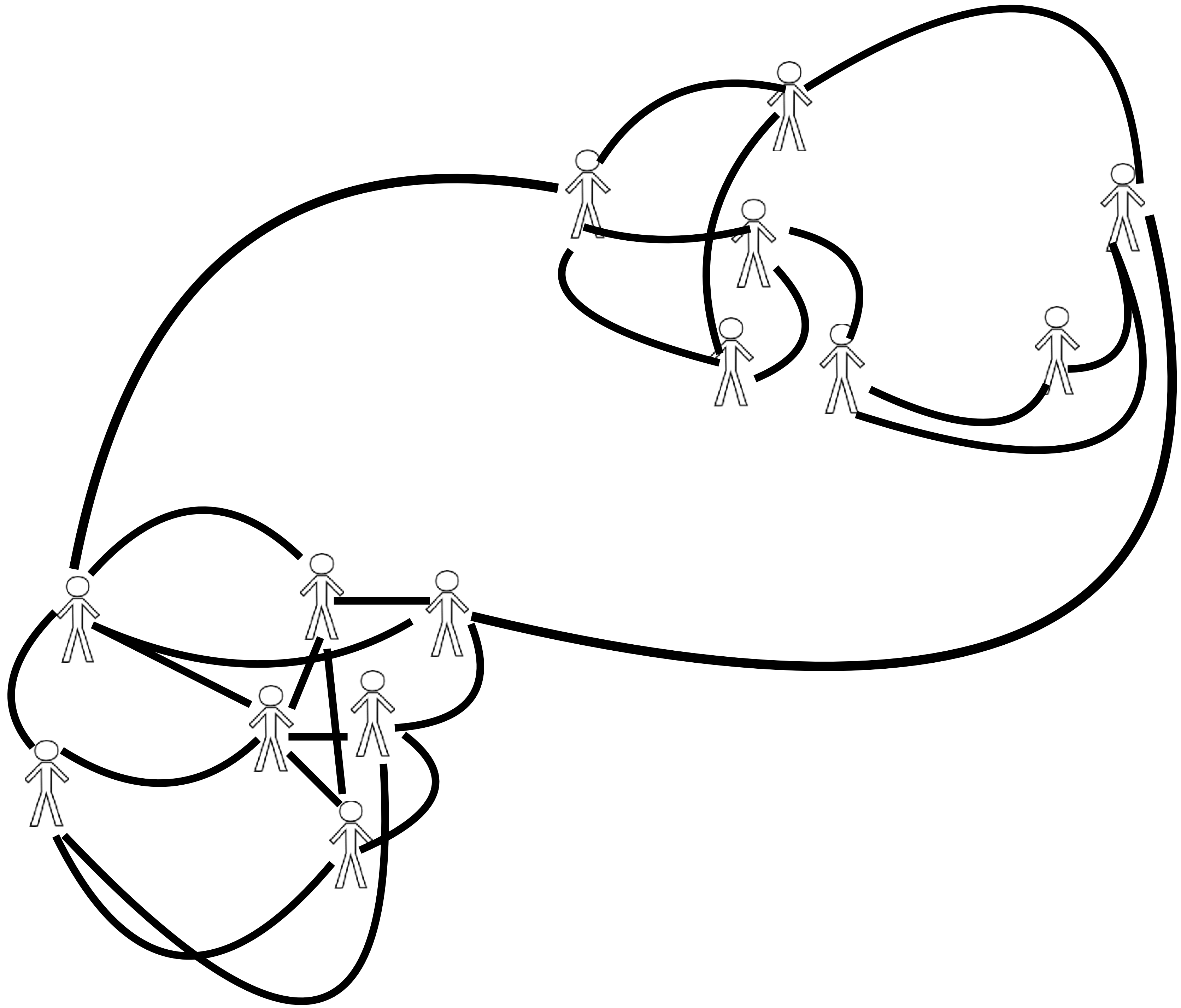
- Algorithm: Find $\mathbf{y}_1, \ldots, \mathbf{y}_n$ by performing gradient descent

Motivating Example



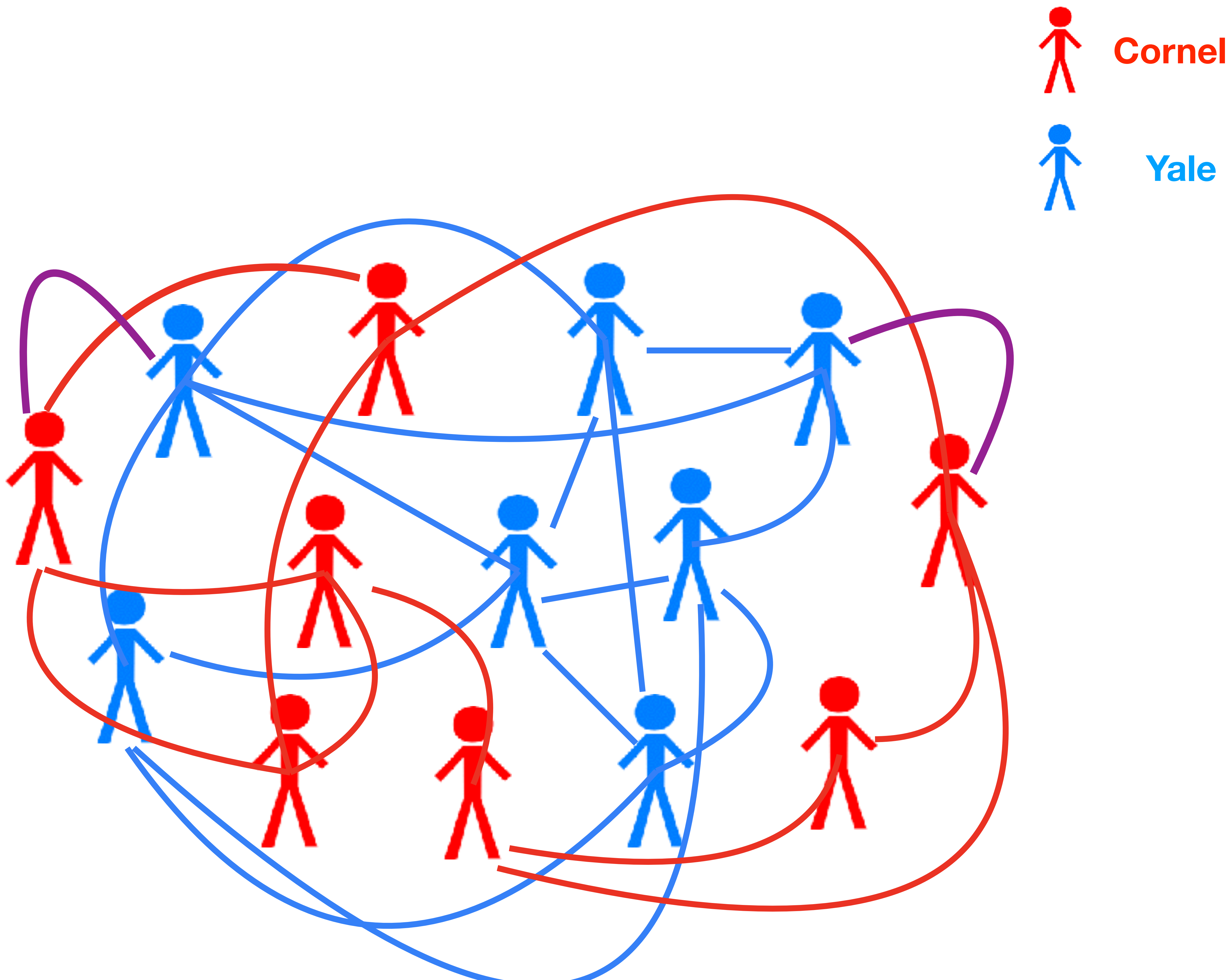
What can you say from this network?

Motivating Example

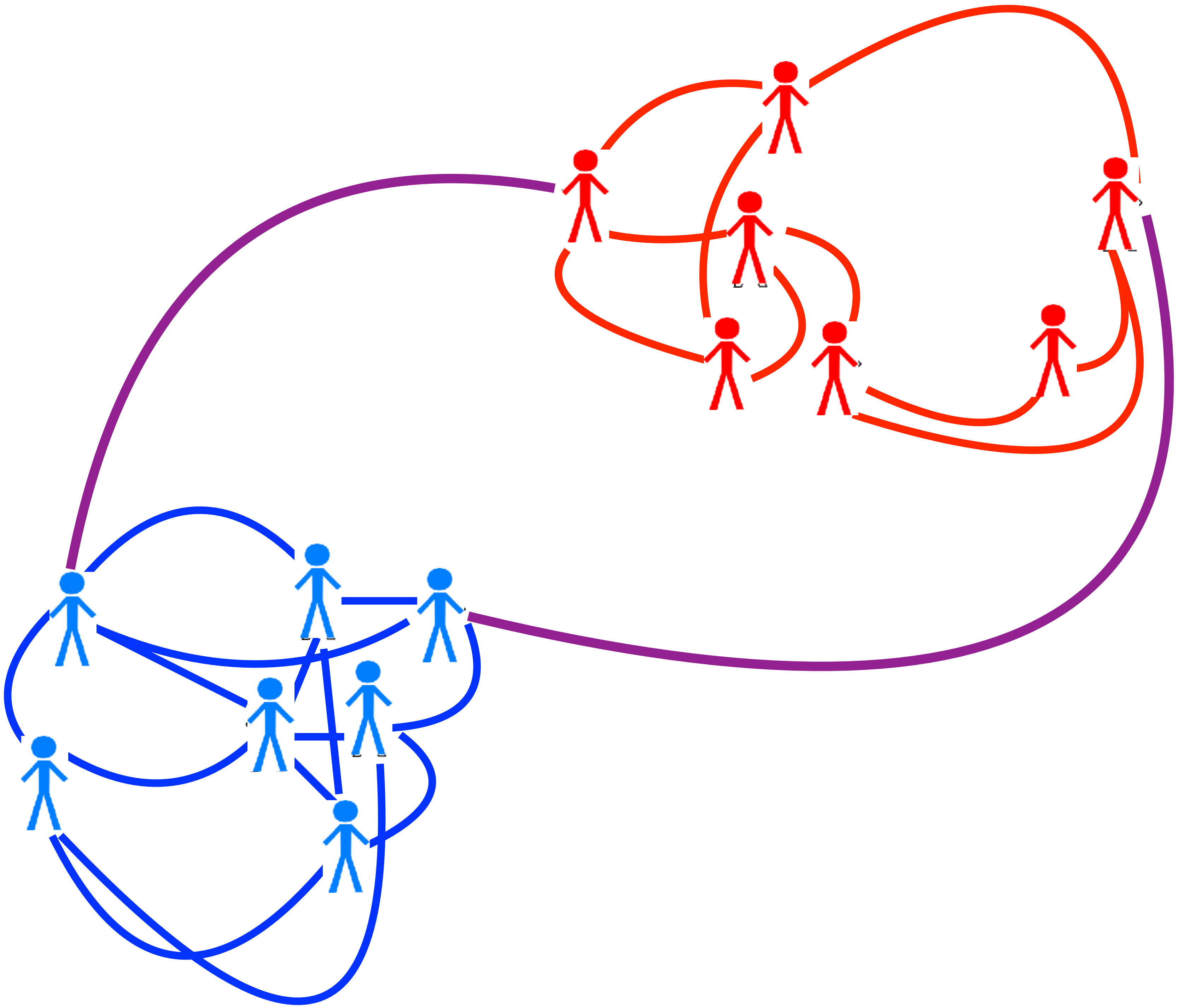


How about now?

Motivating Example



Motivating Example



Graph Embedding

- GOAL: Place vertices (users) of the graph in appropriate locations (in a K dimensional space)
- Distances between vertices (users) should be representative of some desired properties of the graph
  - Eg. Cornell folks are together, all Yale folks are together

How do we do this?

- If I gave you a proposed location how would you evaluate it for instance?
- What are the desirable properties?

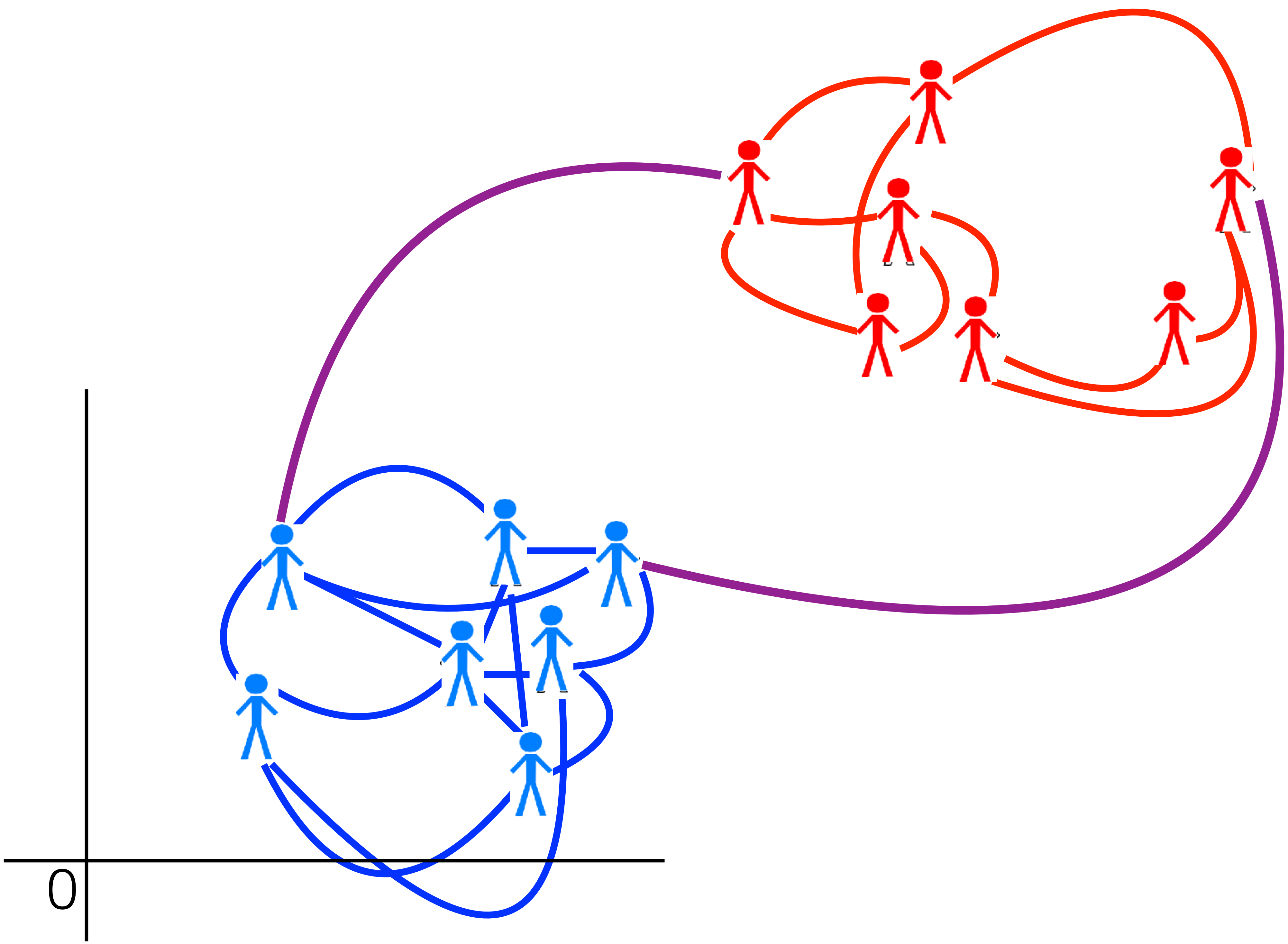
Key Principle

- Keep your friends close
- Spread the vertices (users) around

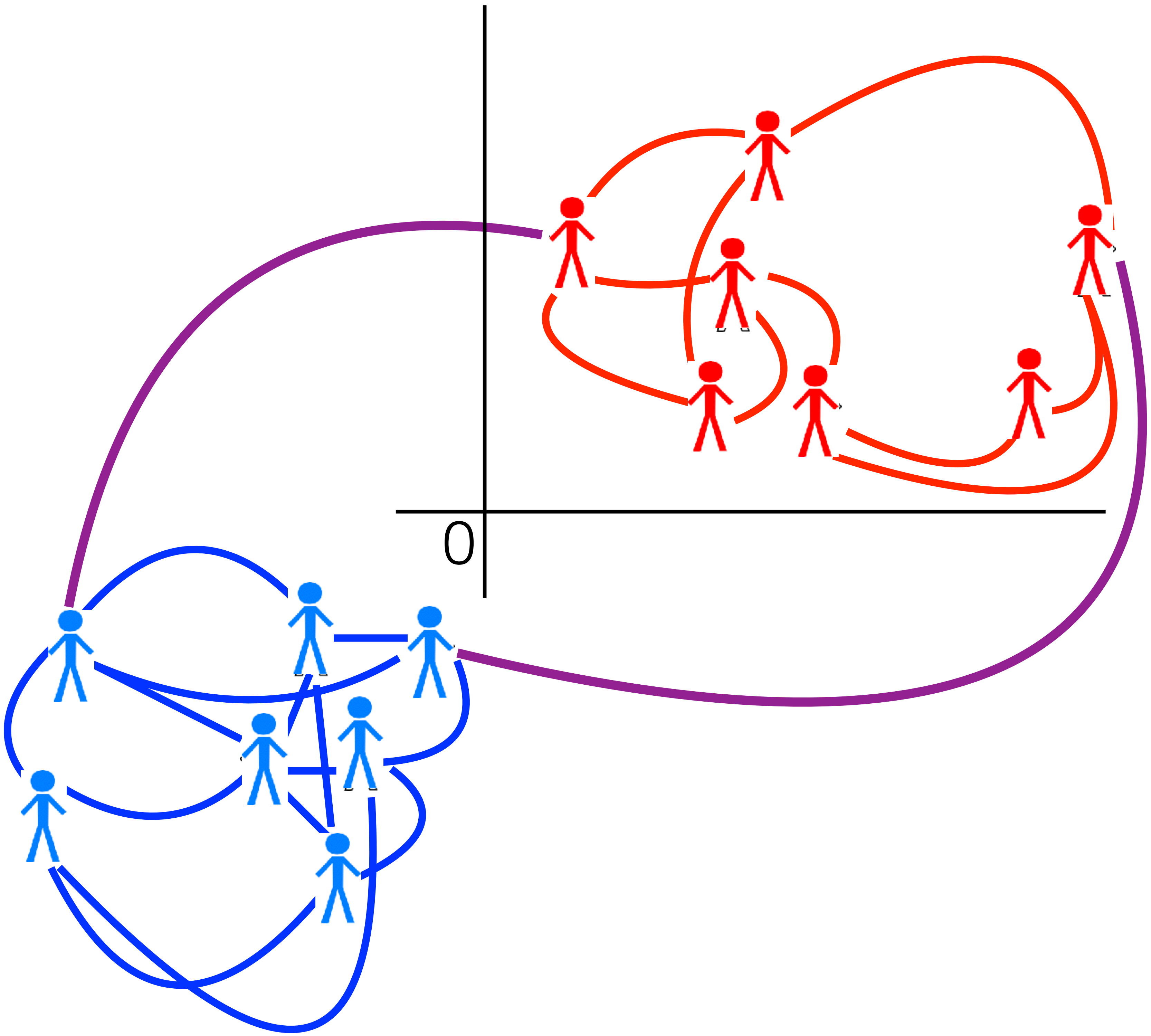
Thought Experiment

- For each user i we specify embedding (location) y_i
- How do we find good locations $y_1, \ldots, y_n$?
- What are good properties?

Centering locations



Centering locations



Key Principle

- **Points are centered at 0**

Key Principle

Make total distance between friends small:

$$\mathrm{Obj}(y_1, \ldots, y_n) = \sum_{(i,j) \in E} \mathrm{dist}^2(y_i, y_j)$$

Key Principle

- Points are centered at 0
- **Keep your Friends close**
  (sum of distances between linked nodes should be small)

Key Principle

If all $y's$ are at same location then friends are all close

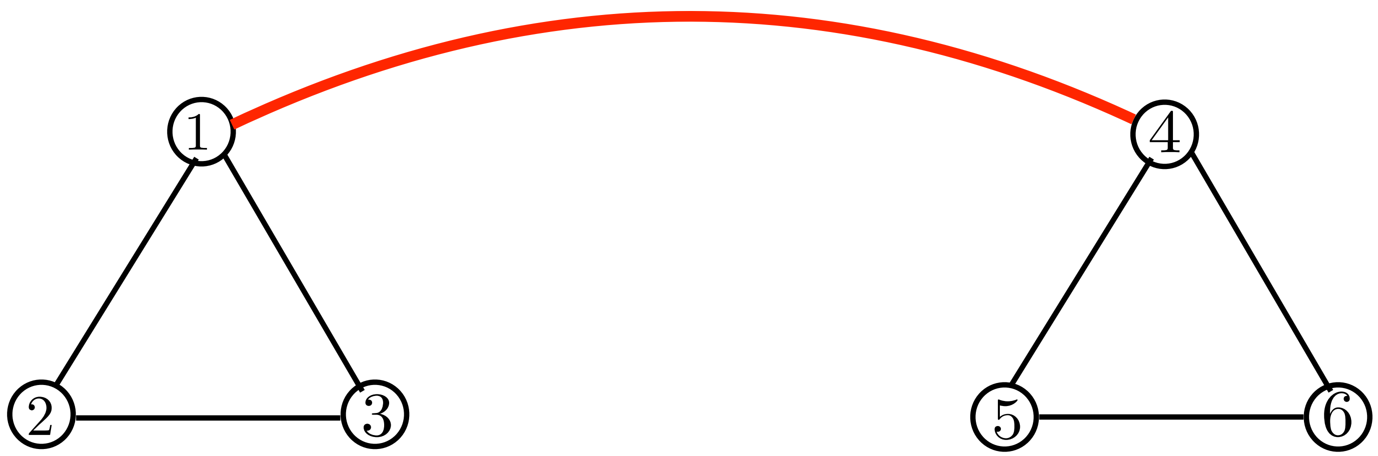
Spread around the points!

Make $\mathrm{Var}(y_1, \ldots, y_n)$ large.

Key Principle

- Points are centered at 0
- Keep your Friends close
  (sum of distances between linked nodes should be small)
- **Variance or spread amongst the nodes should be large**

Examples



Spectral Embedding

- Lets start with one dimensional projection
- Single number y_i for each node i
- Lets review the three desired properties

KEY PRINCIPLE

- **Points are centered at 0**
- Keep your Friends close
- Variance or spread amongst the nodes should be large

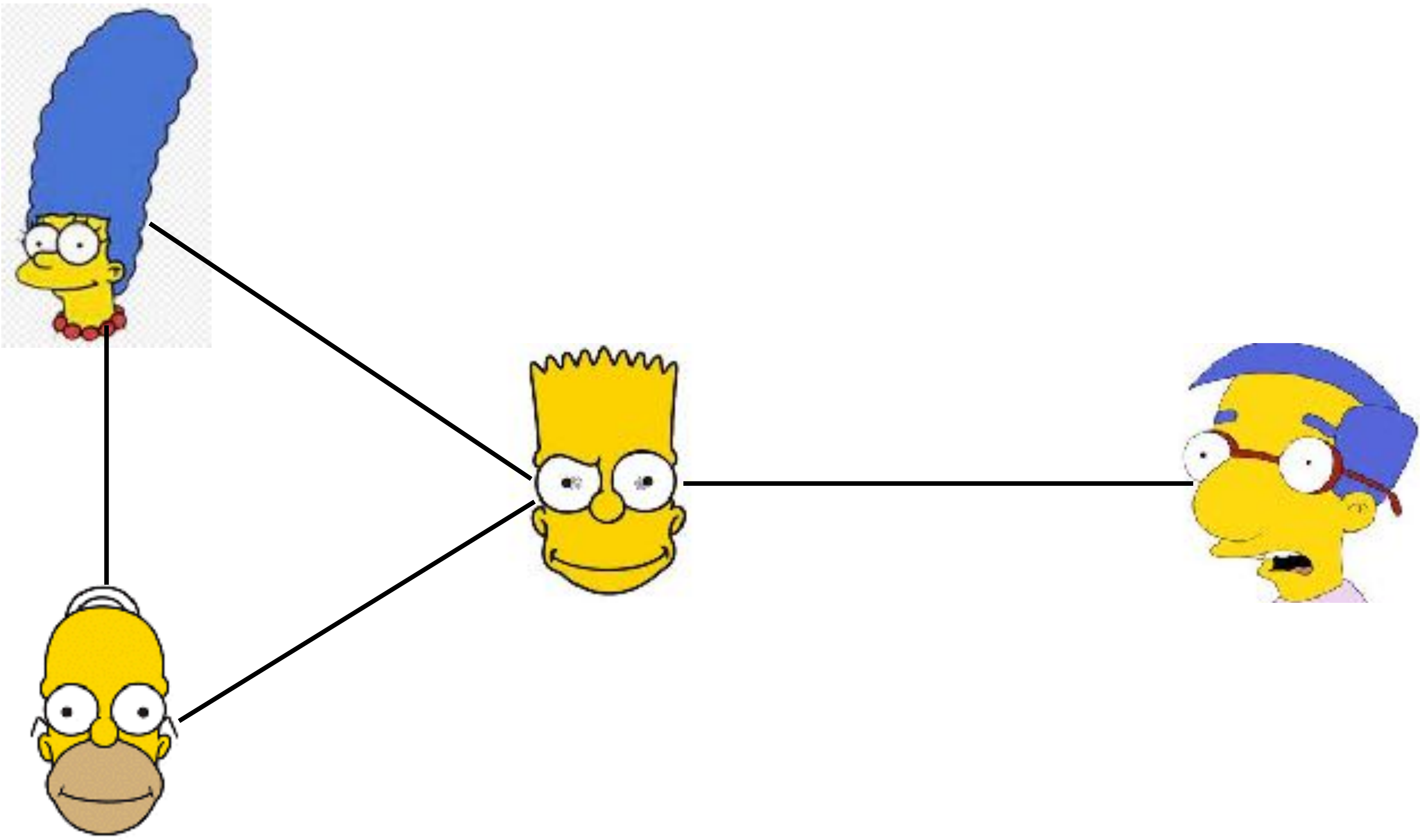
$$\frac{1}{n}\sum_{t=1}^{n} y_t = 0$$

$$y^\top \mathbf{1} = 0$$

Key Principle

- Points are centered at 0 $y^\top \mathbf{1} = 0$
- **Keep your Friends close**
- Variance or spread should be large

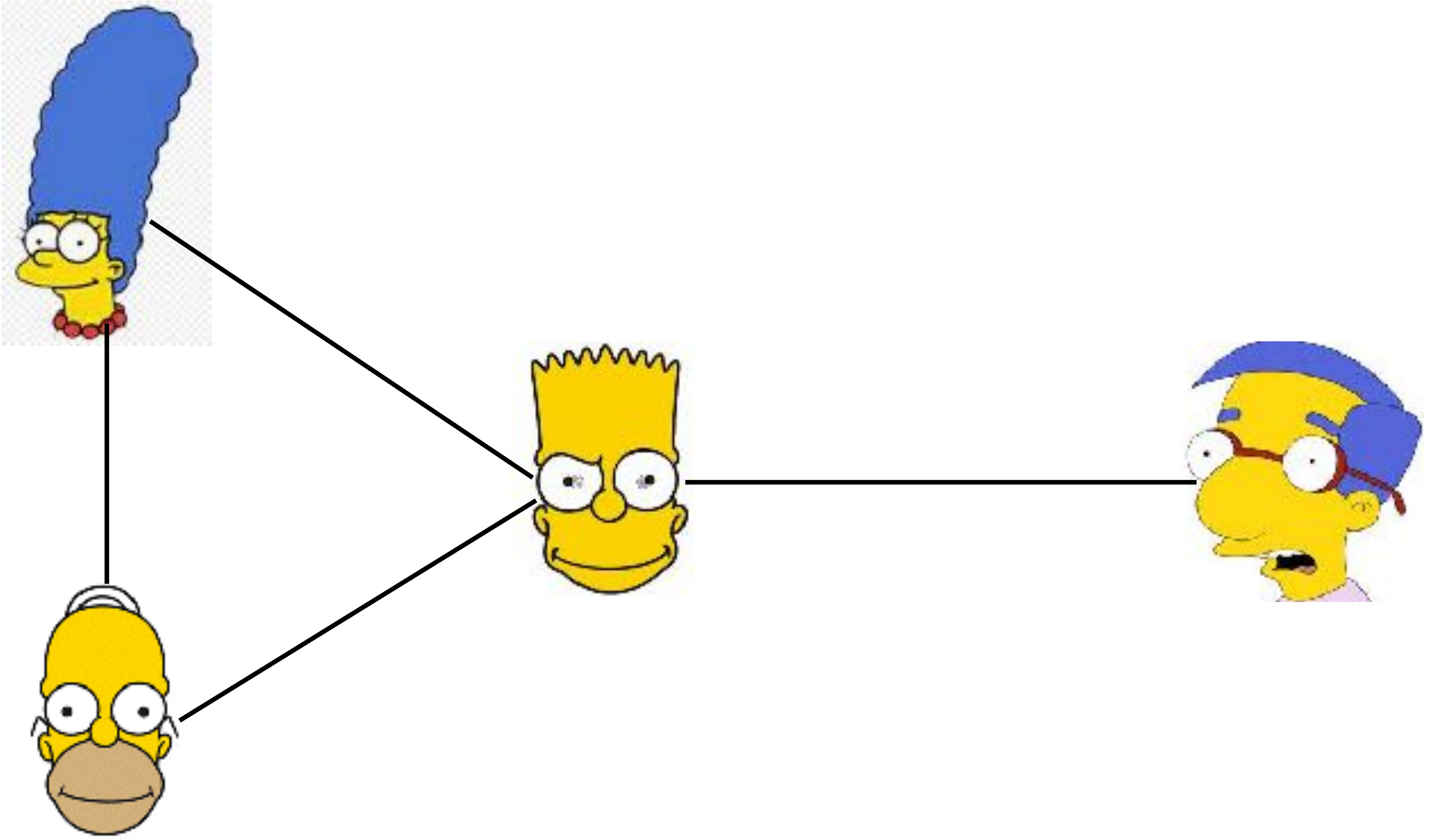
Representing the Graph



A =

				
	0	1	1	0
	1	0	1	0
	1	1	0	1
	0	0	1	0

Representing the Graph



D =

				
	2	0	0	0
	0	2	0	0
	0	0	3	0
	0	0	0	1

Why the Laplacian?

$$
\begin{aligned}
\mathrm{Obj}(y_1, \ldots, y_n) &= \sum_{(i,j) \in \text{ Friends}} (y_i - y_j)^2 \\
&= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} (y_i - y_j)^2 \\
&= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} \left(y_i^2 + y_j^2 - 2 y_i y_j \right) \\
&= \frac{1}{2} \left(\sum_{i=1}^{n} \left(\sum_{j=1}^{n} A_{i,j} \right) y_i^2 + \sum_{j=1}^{n} \left(\sum_{i=1}^{n} A_{i,j} \right) y_j^2 - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} y_i y_j \right) \\
&= \frac{1}{2} \left(\sum_{i=1}^{n} D_{i,i} y_i^2 + \sum_{j=1}^{n} D_{j,j} y_j^2 - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} y_i y_j \right) \\
&= \sum_{i=1}^{n} D_{i,i} \; y_i^2 - \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} y_i y_j \\
&= \left(y^\top D y - y^\top A y \right) \\
&= y^\top (D - A) y = y^\top L y
\end{aligned}
$$

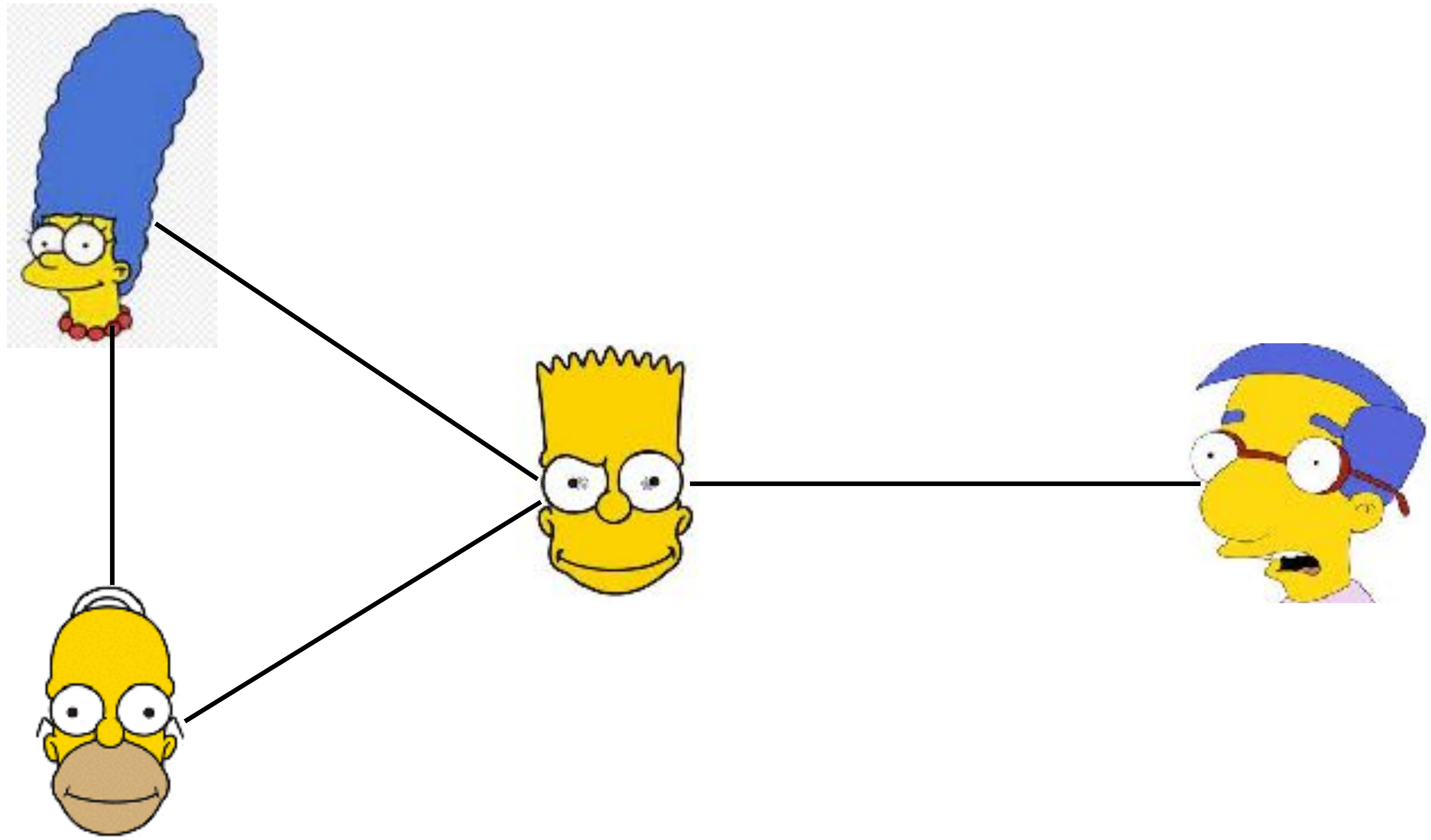
The Laplacian Matrix

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

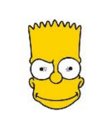
	2	0	0	0
	0	2	0	0
	0	0	3	0
	0	0	0	1

$-$

	0	1	1	0
	1	0	1	0
	1	1	0	1
	0	0	1	0



L =

				
	2	-1	-1	0
	-1	2	-1	0
	-1	-1	3	-1
	0	0	-1	1

Key Principle

- Points are centered at 0 $y^\top \mathbf{1} = 0$
- **Keep your Friends close** $\text{minimize} \;\; y^\top L y$
- Variance or spread should be large

Key Principle

- Points are centered at 0 $\quad y^\top \mathbf{1} = 0$
- Keep your Friends close $\quad \text{minimize} \;\; y^\top L y$
- **Variance or spread should be large**

Maximize Variance

$$\begin{aligned}\mathrm{Var}(y_1, \ldots, y_n) &= \frac{1}{n} \sum_{t=1}^{n} (y_t - \mathrm{mean}(y))^2 \\ &= \frac{1}{n} \sum_{t=1}^{n} y_t^2 = \frac{1}{n} \|y\|_2^2\end{aligned}$$

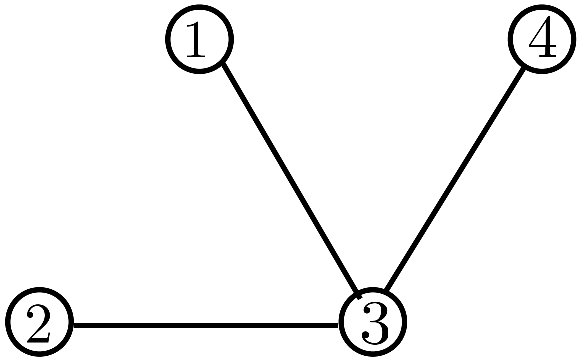
Key Principle

- Points are centered at 0 $y^\top \mathbf{1} = 0$
- Keep your Friends close $\text{minimize} \;\; y^\top L y$
- Variance or spread should be large $\text{Maximize} \; \frac{1}{n}\|y\|_2^2$

$$\text{Minimize} \; \frac{y^\top L y}{\|y\|_2^2} \quad \text{s.t. } y \perp \mathbf{1}$$

$$\text{Minimize} \; y^\top L y \quad \text{s.t. } \|y\|_2^2 = 1 \quad y \perp \mathbf{1}$$

Examples



- Fact: For a connected graph, exactly one, the smallest of eigenvalues is 0, corresponding eigenvector is $(1, 1, \ldots, 1)^\top/\sqrt{n}$

KEY PRINCIPLE

- Points are centered at 0 $y^\top \mathbf{1} = 0$
- Keep your Friends close $\text{minimize} \;\; y^\top L y$
- Variance or spread should be large $\text{Maximize} \; \frac{1}{n}\|y\|_2^2$

$$\text{Minimize } y^\top L y \quad \text{s.t. } \|y\|_2^2 = 1 \quad y \perp \mathbf{1}$$

$$y = \text{ Second smallest eigenvector of } L$$

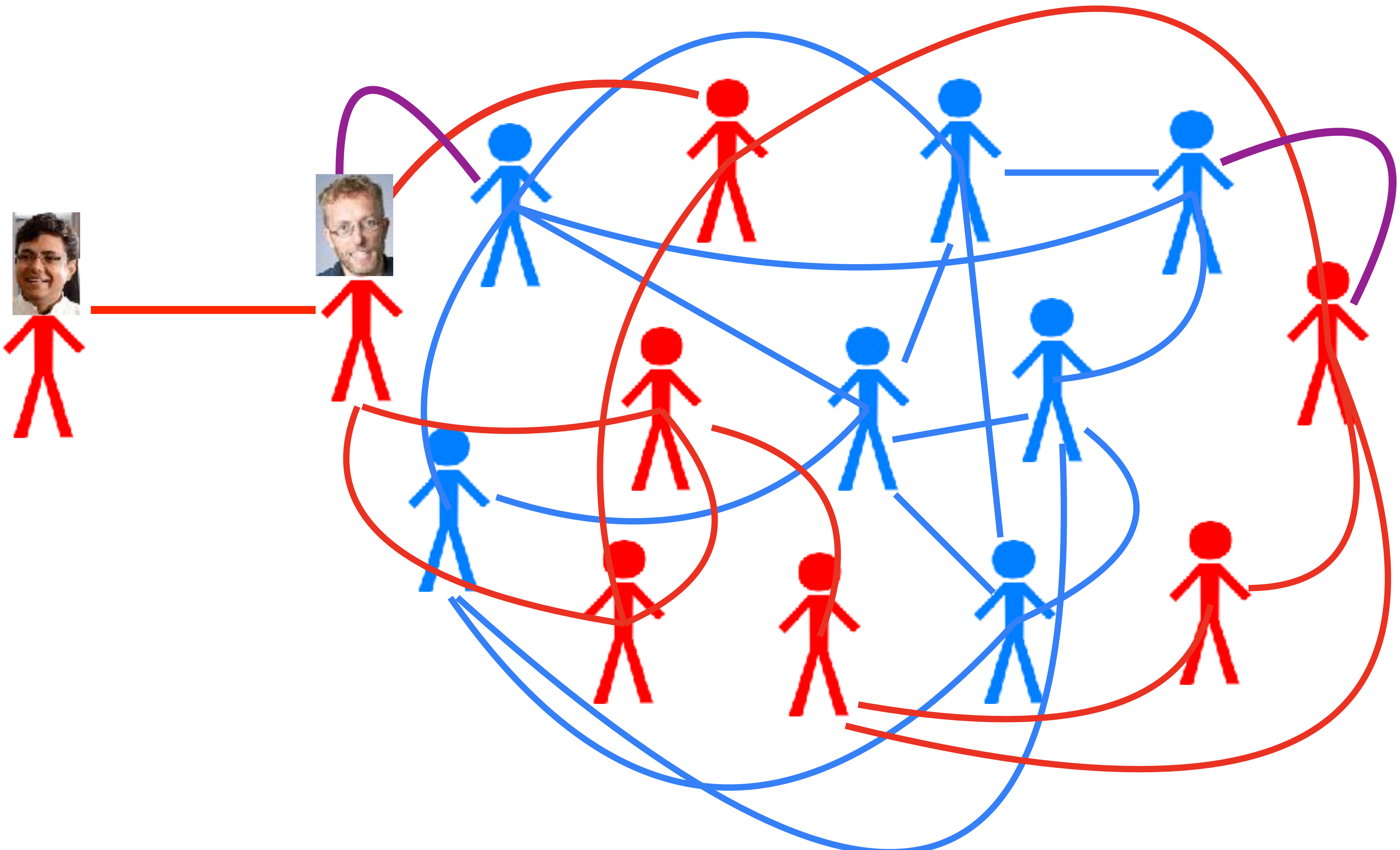
Spectral Embedding

- For $K > 1$ dimensional embedding
- First dimension is the second smallest eigenvector
- Second dimension is the third smallest eigenvector and so on ...
- (Unnormalized) Spectral clustering: compute $2 : K + 1$ smallest eigen vectors
- Set Y_i to be the i'th row

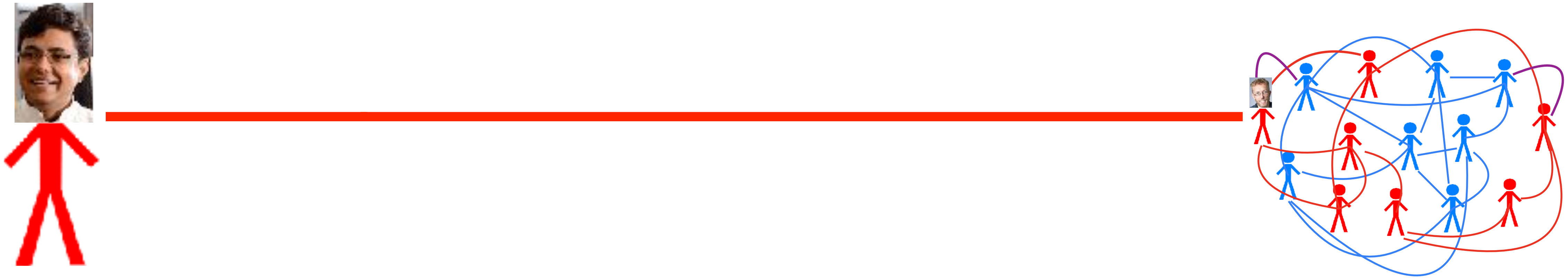
Spectral Clustering Algorithm (Unnormalized)

1. Given matrix A calculate diagonal matrix D s.t. $D_{i,i} = \sum_{j=1}^{n} A_{i,j}$
2. Calculate the Laplacian matrix $L = D - A$
3. Find eigen vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ of L (ascending order of eigenvalues)
4. Pick the K eigenvectors with smallest eigenvalues to get $\mathbf{y}_1, \ldots, \mathbf{y}_n \in \mathbb{R}^K$
5. Use K-means clustering algorithm on $\mathbf{y}_1, \ldots, \mathbf{y}_n$

Trouble Makers



Trouble Makers



- Variance is high
- Almost all connected nodes have same (small value)

Normalized Spectral Embedding

- Nodes linked to each other are close to each other
- Variance or spread should be large
  - But variance under what distribution?
  - Higher degree nodes are more important!
  - Lets try distribution given by p_i \propto D_{i,i}