

Graph Embeddings and Spectral Methods

- ▶ weighted graphs: terminology and representations
- ▶ graph Laplacian and Dirichlet energy
- ▶ spectral graph embedding (scalar and vector)
- ▶ node ordering and graph partitioning

Graphs: Nodes and Edges

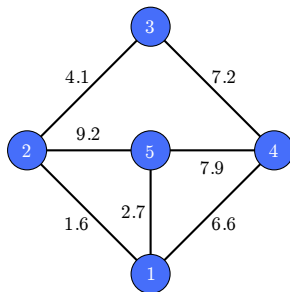
$$\{i, j\}, \quad (i, j) \neq (j, i)$$

- ▶ A graph $G = (V, E)$ consists of:
 - ▶ a set of n *nodes* (or *vertices*) $V = \{1, \dots, n\}$
 - ▶ a set of m *edges* (or *links*) $E = \{\{i, j\} \mid \text{nodes } i, j \text{ are connected}\}$
- ▶ Nodes i and j are *adjacent* (neighbors) if $\{i, j\} \in E$.
 - ▶ Neighbor set of i : $N(i) = \{j \mid \{i, j\} \in E\}$
 - ▶ *Degree* of node i : $d_i = |N(i)|$
- ▶ *Undirected graph*: $\{i, j\}$ is the same as $\{j, i\}$.

Weighted Graphs and Adjacency Matrix

- ▶ In a *weighted graph*, each edge $\{i, j\} \in E$ has a positive *weight* $W_{ij} > 0$.
 - ▶ $W_{ij} = W_{ji}$ (symmetric).
 - ▶ $W_{ij} = 0$ if $\{i, j\} \notin E$.
 - ▶ Diagonal entries $W_{ii} = 0$.
- ▶ The *weighted adjacency matrix* $W \in \mathbb{R}^{n \times n}$ completely specifies the graph.
 - ▶ $2m$ non-zero entries (for symmetric W).
- ▶ *Weighted degree* of node i : $d_i^w = \sum_{\{i, j\} \in E} W_{ij}$.
 - ▶ In matrix notation: $d^w = W\mathbf{1}$.

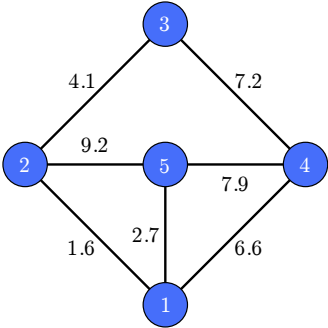
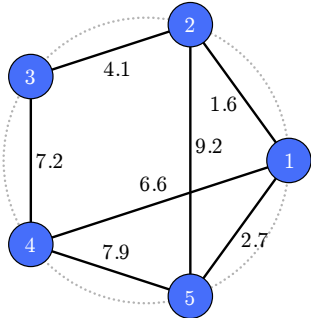
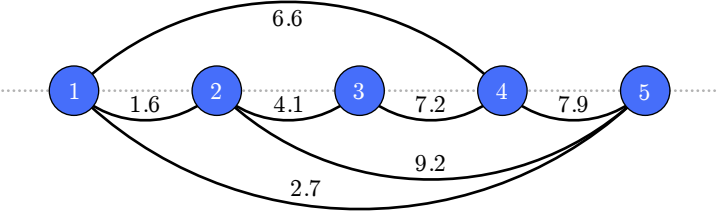
Example weighted graph



$$W = \begin{bmatrix} 0 & 1.6 & 0 & 6.6 & 2.7 \\ 1.6 & 0 & 4.1 & 0 & 9.2 \\ 0 & 4.1 & 0 & 7.2 & 0 \\ 6.6 & 0 & 7.2 & 0 & 7.9 \\ 2.7 & 9.2 & 0 & 7.9 & 0 \end{bmatrix}$$

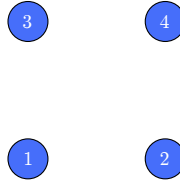
- $n = 5$ nodes, $m = 7$ edges
- Weighted degrees: $d^w = (10.9, 14.9, 11.3, 21.7, 19.8)$

Example graph drawing styles

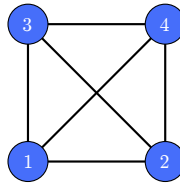


Some named graphs

- An *empty* graph has no edges, *i.e.*, $E = \emptyset$.



- A *full* or *complete* graph has an edge between every pair of nodes.

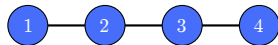


$$\begin{matrix} n \\ \text{nodes} \end{matrix} \rightarrow \frac{n(n-1)}{2} = \binom{n}{2}$$

Some named graphs

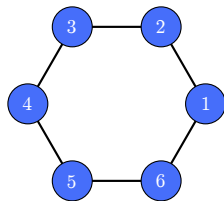
- A *chain* graph P_n is a graph with n nodes arranged in a line. Adjacency matrix is tridiagonal, *e.g.* for unweighted P_4

$$W = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



- A *cycle* graph C_n . Similar to a path, but endpoints are connected.

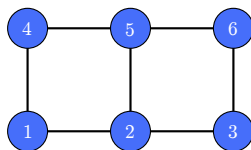
$$W = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$



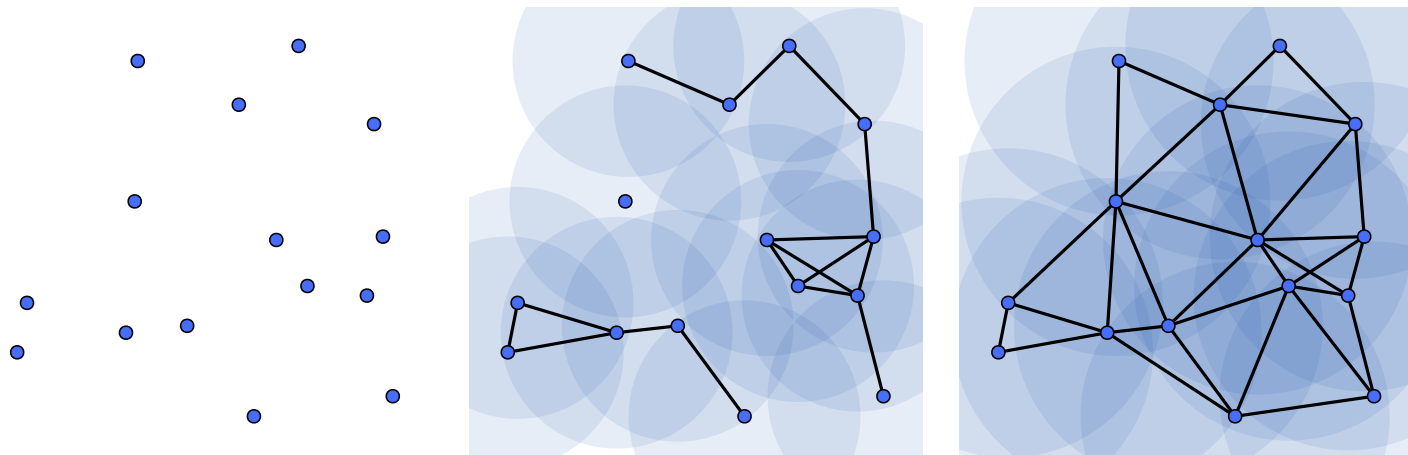
Some named graphs

- A *mesh* or *grid* graph

$$W = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

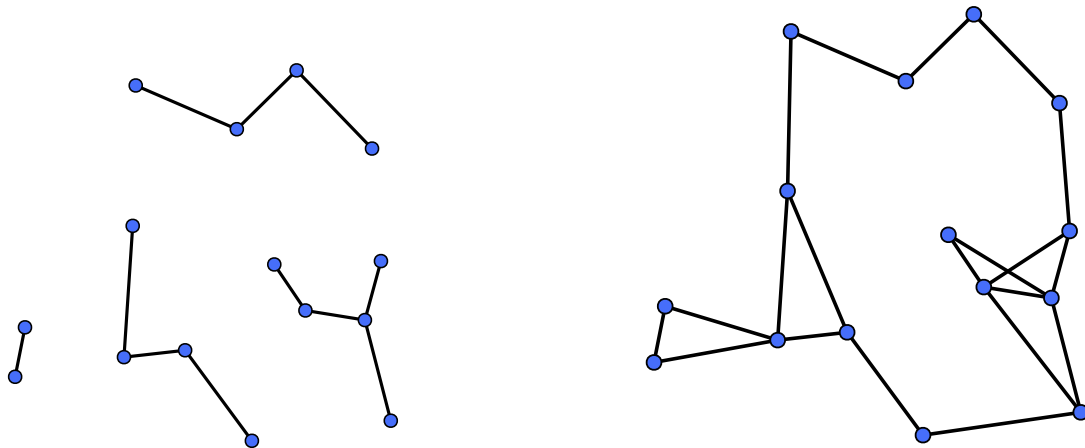


Geometric graphs



- ▶ in some applications, we create a graph from a collection of vectors $x_i \in \mathbb{R}^n$
- ▶ We create an edge whenever $\|x_i - x_j\|_2 \leq d$, where d is less than a critical distance
- ▶ left $d = 0.3$, right $d = 0.4$

Nearest neighbor graphs

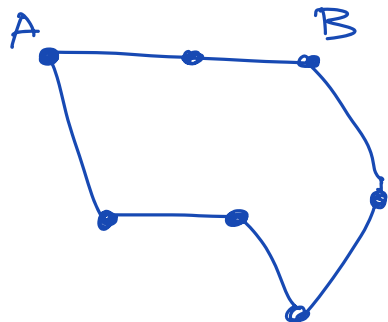
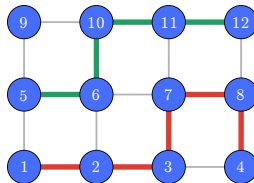


- ▶ we have an edge between i and j if i is among the k -nearest neighbors of j or vice versa.
- ▶ Here k is a parameter that we choose.
- ▶ When $k = 1$ the graph is called the nearest neighbor graph. Left $k = 1$, right $k = 2$

Choosing a method

- ▶ the choice of method (and parameters) depends on the final application
- ▶ all methods share the same basic idea
- ▶ edges of the graph indicate that nodes are 'similar', with higher weight indicating a higher degree of similarity.
- ▶ small $\|x_i - x_j\|_2$ indicates that nodes i and j are similar

Graph Connectivity and Paths



- ▶ A *path* of length L : sequence of nodes $i_1, \dots, i_{L+1}$ where $\{i_j, i_{j+1}\} \in E$.
- ▶ Nodes i and j are *pathwise connected* if a path exists between them.
- ▶ A graph is *connected* if every pair of nodes is pathwise connected.
- ▶ Powers of unweighted adjacency matrix: $(W^L)_{ij}$ is number of paths of length L between i, j .
- ▶ The i, j entry of the matrix

$$Z = I + W + W^2 + \dots + W^{n-1}$$

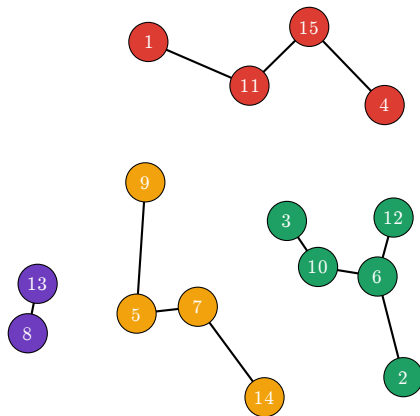
gives the number of paths of length at most $n - 1$ between nodes i and j .

Connected components

- ▶ a graph partitions the nodes into groups of mutually pathwise connected nodes.
- ▶ that is, we have subsets of nodes $G_1, \dots, G_K \subseteq \{1, \dots, n\}$ with

$$G_i \cap G_j = \emptyset \text{ for } i \neq j, \quad G_1 \cup \dots \cup G_K = \{1, \dots, n\}.$$

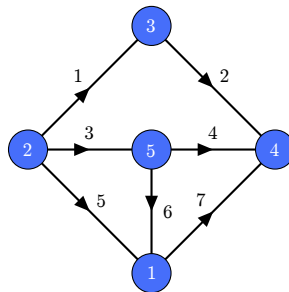
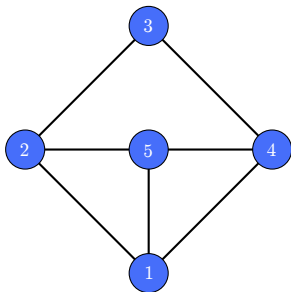
- ▶ Within each group of nodes, every node is pathwise connected to every other one



Incidence matrix (directed graphs)

- For an undirected graph, assign arbitrary directions to edges to form a *network*.
- The $n \times m$ *incidence matrix* A has entries:

$$A_{ij} = \begin{cases} 1 & \text{edge } j \text{ points to node } i \\ -1 & \text{edge } j \text{ points from node } i \\ 0 & \text{otherwise} \end{cases}$$



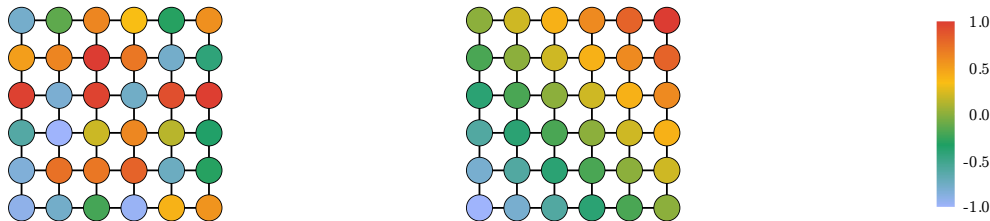
Node potentials and Dirichlet energy

$$x^T A x = \sum A_{ij} x_i x_j$$

- ▶ Assign a scalar value x_i to each node i : *node potentials* $x \in \mathbb{R}^n$.
- ▶ *Dirichlet energy* $D(x)$ measures smoothness of x across the graph.

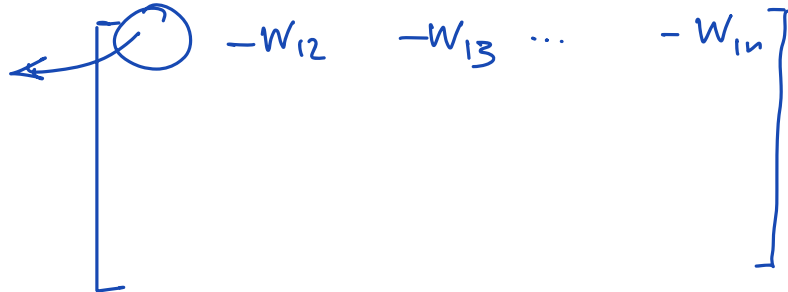
$$D(x) = \frac{1}{2} \sum_{i,j=1}^n W_{ij} (x_i - x_j)^2$$

- ▶ Small $D(x)$ means $x_i \approx x_j$ for adjacent nodes (especially with large W_{ij}).



The Graph Laplacian Matrix

$$W_{12} + \dots + W_{1n}$$



- ▶ The Dirichlet energy is a quadratic form

$$D(x) = x^T Lx$$

- ▶ L is the *Laplacian matrix*: $L = D - W$.
 - ▶ $D = \text{diag}(d^w)$ is the diagonal matrix of weighted degrees.
 - ▶ Entries: $L_{ii} = d_i^w$, $L_{ij} = -W_{ij}$ for $i \neq j$.
- ▶ L is symmetric and positive semi-definite (PSD).
 - ▶ Not positive definite (PD), since $L\mathbf{1} = 0$ (hence $D(\mathbf{1}) = 0$).

Laplacian matrix

let's establish $L = D - W$

$$\begin{aligned}\mathcal{D}(x) &= (1/2) \sum_{i,j=1}^n W_{ij} (x_i - x_j)^2 \\ &= (1/2) \sum_{i,j=1}^n W_{ij} (x_i^2 - 2x_i x_j + x_j^2) \\ &= (1/2) \sum_{i=1}^n \left(\sum_{j=1}^n W_{ij} \right) x_i^2 - \sum_{i,j=1}^n W_{ij} x_i x_j + (1/2) \sum_{j=1}^n \left(\sum_{i=1}^n W_{ij} \right) x_j^2 \\ &= x^T D x - x^T W x.\end{aligned}$$

we use that fact that the first and last sums are both equal to $\sum_{i=1}^n d_i x_i^2$.

Laplacian from Incidence Matrix

- ▶ let w be the vector of edge weights.
- ▶ the Laplacian can also be defined using the incidence matrix A

$$L = A \text{diag}(w) A^T$$

- ▶ this definition is independent of the arbitrary edge orientations chosen for A .
- ▶ multiplication by A^T gives a vector of potential differences
- ▶ when $w = 1$ then L is the *Gram matrix* of A^T

Nullspace of the Laplacian and connectivity

$$D(x) = \frac{1}{2} \sum W_{ij} (x_i - x_j)^2 = x^T L x$$

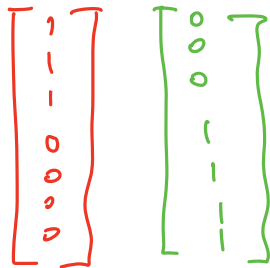
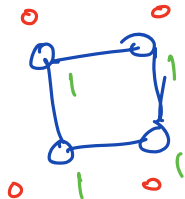
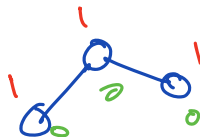
- ▶ If $Lx = 0$, then $D(x) = 0$, implying $x_i = x_j$ for all adjacent nodes.
- ▶ By extension, $x_i = x_j$ for all pathwise connected nodes.
- ▶ If the graph is connected:

$$\mathcal{N}(L) = \text{span}(\mathbf{1})$$

- ▶ $\dim \mathcal{N}(L) = 1$. Smallest eigenvalue $\lambda_1 = 0$, with eigenvector $\mathbf{1}$.
- ▶ If the graph is disconnected with K components:

$$\dim \mathcal{N}(L) = K$$

- ▶ Number of zero eigenvalues of L equals number of connected components.



Scalar spectral embedding problem

$$\lambda_{\max} \leftarrow \max x^T A x$$

- ▶ Assign node potentials $x \in \mathbb{R}^n$ (an *embedding*) to minimize Dirichlet energy.
- ▶ *Smoothness*: adjacent nodes should have similar values (small $|x_i - x_j|$).
- ▶ Constraints needed to prevent trivial or arbitrary solutions:
 - ▶ *Centering*: $\mathbf{1}^T x = 0$ (mean of potentials is zero, due to shift invariance of $D(x)$).
 - ▶ *Scaling*: $\|x\|_2^2 = n$ (RMS value of potentials is one, prevents $x = 0$).

- ▶ we would like to solve

$$\begin{array}{ll} \text{minimize} & x^T L x \\ \text{subject to} & \mathbf{1}^T x = 0 \\ & \|x\|_2 = \sqrt{n} \end{array}$$

Eigendecomposition of the Laplacian

- ▶ eigenvalues of L are $0 = \lambda_1 < \lambda_2 \leq \dots \leq \lambda_n$.
- ▶ $\lambda_1 = 0$ has eigenvector $\mathbf{1}$.
- ▶ The second smallest eigenvalue λ_2 is the *Fiedler eigenvalue* (λ^F).
- ▶ Its associated eigenvector v^F (or v_2) is the *Fiedler eigenvector*.
- ▶ $\mathbf{1}^T v^F = 0$ and $\|v^F\|_2 = 1$.

Scalar spectral embedding solution

- recall the optimization problem, where A is symmetric and $\lambda_1 < \lambda_2$

$$\begin{array}{ll}\text{minimize} & x^T A x \\ \text{subject to} & v_1^T x = 0 \\ & \|x\|_2 = 1,\end{array}$$

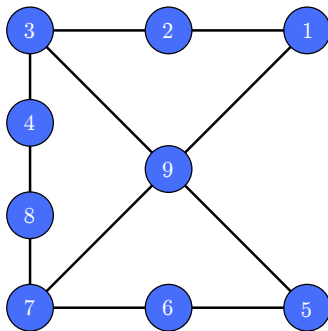
where v_1 is an eigenvector associated with the smallest eigenvalue λ_1

- the normalized second smallest eigenvector v_2 is a solution
- for scalar spectral embedding, the solution is

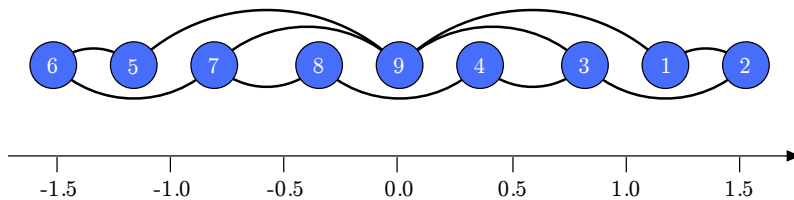
$$x^{\text{se}} = \sqrt{n} v^F$$

- x^{se} is the *spectral embedding* for the weighted graph.
- The optimal value of $D(x)$ is $n\lambda^F$. Small λ^F indicates a graph is 'almost disconnected'.

Example: scalar spectral embedding



The scalar spectral embedding is:

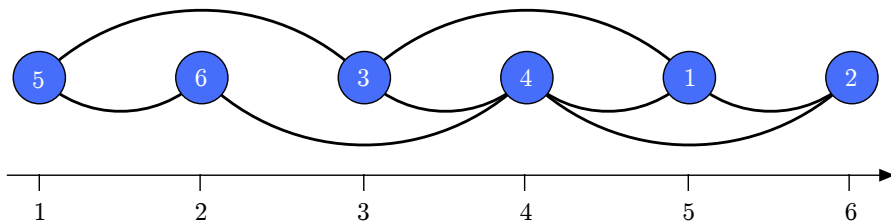


Applications: node ordering

- ▶ *Node ordering problem*: Assign integers $1, \dots, n$ to nodes as new indices σ .
- ▶ Goal: edges connect nodes with nearby indices, especially for high weights.

$$\begin{array}{ll} \text{minimize} & \sigma^\top L \sigma \\ \text{subject to} & \sigma \in S_n \end{array}$$

- ▶ Exact solution is computationally hard ($n!$ permutations).
- ▶ *Approximate solution (spectral ordering)*: Sort entries of x^{se} .
- ▶ *Polishing*: Local search (swapping adjacent indices) to improve Dirichlet energy.

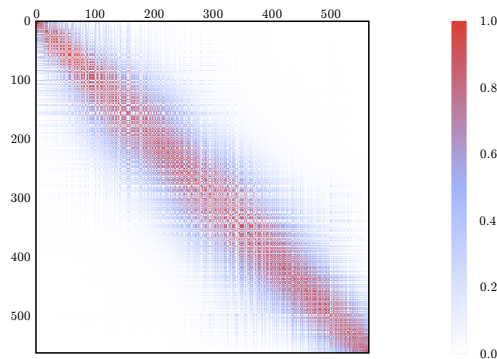
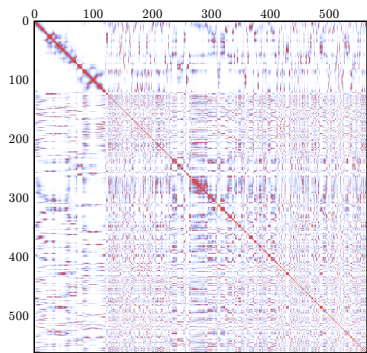


Applications: Permuting a correlation matrix

- ▶ given a correlation matrix C , create a graph, either connect based on a threshold, or use

$$W_{ij} = \left(\frac{C_{ij} + 1}{2} \right)^\gamma$$

- ▶ spectral ordering ensures that highly-correlated items are near each other
- ▶ application: LU or Cholesky solvers have reduced fill-in



Applications: Graph bisection

- ▶ *Node bisection problem*: Partition nodes into two groups A, B .
- ▶ Encode partition: $x_i = -1$ if $i \in A$, $x_i = 1$ if $i \in B$.
- ▶ *Balanced bisection*: $|A| = |B| = n/2 \implies \mathbf{1}^\top x = 0$.
- ▶ *Cut weight*: sum of weights of edges crossing the cut.

$$\begin{array}{ll}\text{minimize} & x^\top Lx \\ \text{subject to} & x_i \in \{-1, 1\}, \text{ for } i = 1, \dots, n \\ & \mathbf{1}^\top x = 0\end{array}$$

- ▶ Exact solution is NP-hard.
- ▶ *Approximate solution (spectral bisection)*: Take x^{se} , assign nodes with smallest $n/2$ values to A , rest to B .

Vector Graph Embedding

- ▶ Assign a k -dimensional vector $x_i \in \mathbb{R}^k$ to each node i .
- ▶ Form an $n \times k$ matrix X , where rows are $x_i^\top$.
- ▶ Objective is to minimize the sum of squared differences between adjacent node vectors:

$$D(X) = \frac{1}{2} \sum_{i,j=1}^n W_{ij} \|x_i - x_j\|_2^2$$

- ▶ This can be expressed in terms of the Laplacian: $D(X) = \text{trace}(X^\top LX)$.
- ▶ Constraints:
 - ▶ *Centering*: $X^\top \mathbf{1} = 0$ (mean of each component is zero).
 - ▶ *Standardization*: $\frac{1}{n} X^\top X = I$ (components are orthogonal and scaled).

Vector spectral embedding solution

- ▶ The columns of the optimal X are proportional to the k eigenvectors of L corresponding to the k smallest non-zero eigenvalues $(v_2, \dots, v_{k+1})$.

$$\begin{bmatrix} x_1^\top \\ \vdots \\ x_n^\top \end{bmatrix} \propto [v_2 \ \dots \ v_{k+1}]$$

- ▶ *Graph drawing*: $k = 2$ allows nodes to be plotted in a 2D plane based on their embedding vectors.

