Data Analysis and Manifold Learning
Lecture 3: Graphs, Graph Matrices, and Graph Embeddings

Radu Horaud
INRIA Grenoble Rhone-Alpes, France
Radu.Horaud@inrialpes.fr
http://perception.inrialpes.fr/

Outline of Lecture 3

- What is spectral graph theory?
- Some graph notation and definitions
- The adjacency matrix
- Laplacian matrices
- Spectral graph embedding

Material for this lecture

- F. R. K. Chung. Spectral Graph Theory. 1997. (Chapter 1)
- M. Belkin and P. Niyogi. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. Neural Computation, 15, 1373–1396 (2003).
- U. von Luxburg. A Tutorial on Spectral Clustering. Statistics and Computing, 17(4), 395–416 (2007). (An excellent paper)
- Software: `http://open-specmatch.gforge.inria.fr/index.php`. Computes, among others, Laplacian embeddings of very large graphs.

Spectral graph theory at a glance

- The *spectral graph theory* studies the properties of graphs via the eigenvalues and eigenvectors of their associated graph matrices: the *adjacency matrix*, the *graph Laplacian* and their variants.
- These matrices have been extremely well studied from an algebraic point of view.
- The Laplacian allows a natural link between discrete representations (graphs), and continuous representations, such as metric spaces and manifolds.
- Laplacian embedding consists in representing the vertices of a graph in the space spanned by the smallest eigenvectors of the Laplacian – *A geodesic distance on the graph becomes a spectral distance in the embedded (metric) space*.

Spectral graph theory and manifold learning

- First we construct a graph from $\boldsymbol{x}_1, \dots \boldsymbol{x}_n \in \mathbb{R}^D$
- Then we compute the d **smallest** eigenvalue-eigenvector pairs of the graph Laplacian
- Finally we represent the data in the $\mathbb{R}^d$ space spanned by the correspodning orthonormal eigenvector basis. The choice of the dimension d of the embedded space is not trivial.
- Paradoxically, d may be **larger** than D in many cases!

Basic graph notations and definitions

We consider *simple graphs* (no multiple edges or loops), $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$:

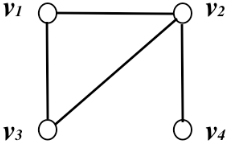
- $\mathcal{V}(\mathcal{G}) = \{v_1, \ldots, v_n\}$ is called the *vertex set* with $n = |\mathcal{V}|$;
- $\mathcal{E}(\mathcal{G}) = \{e_{ij}\}$ is called the *edge set* with $m = |\mathcal{E}|$;
- An edge e_{ij} connects vertices v_i and v_j if they are adjacent or neighbors. One possible notation for adjacency is $v_i \sim v_j$;
- The number of neighbors of a node v is called the *degree* of v and is denoted by $d(v)$, $d(v_i) = \sum_{v_i \sim v_j} e_{ij}$. If all the nodes of a graph have the same degree, the graph is *regular*; The nodes of an *Eulerian* graph have even degree.
- A graph is *complete* if there is an edge between every pair of vertices.

The adjacency matrix of a graph

- For a graph with n vertices, the entries of the $n \times n$ adjacency matrix are defined by:

$$\mathbf{A} := \begin{cases} A_{ij} = 1 & \text{if there is an edge } e_{ij} \\ A_{ij} = 0 & \text{if there is no edge} \\ A_{ii} = 0 & \end{cases}$$

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



Eigenvalues and eigenvectors

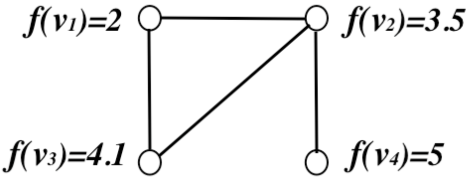
- $\mathbf{A}$ is a real-symmetric matrix: it has n real eigenvalues and its n real eigenvectors form an orthonormal basis.
- Let $\{\lambda_1, \ldots, \lambda_i, \ldots, \lambda_r\}$ be the set of *distinct* eigenvalues.
- The eigenspace S_i contains the eigenvectors associated with λ_i:

$$S_i = \{\boldsymbol{x} \in \mathbb{R}^n | \mathbf{A}\boldsymbol{x} = \lambda_i \boldsymbol{x}\}$$

- For real-symmetric matrices, the algebraic multiplicity is equal to the geometric multiplicity, for all the eigenvalues.
- The dimension of S_i (geometric multiplicity) is equal to the multiplicity of λ_i.
- If $\lambda_i \neq \lambda_j$ then S_i and S_j are mutually orthogonal.

Real-valued functions on graphs

- We consider real-valued functions on the set of the graph's vertices, $\boldsymbol{f} : \mathcal{V} \longrightarrow \mathbb{R}$. Such a function assigns a real number to each graph node.
- $\boldsymbol{f}$ is a vector indexed by the graph's vertices, hence $\boldsymbol{f} \in \mathbb{R}^n$.
- Notation: $\boldsymbol{f} = (f(v_1), \dots, f(v_n)) = (f_1, \dots, f_n)$.
- The eigenvectors of the adjacency matrix, $\mathbf{A}\boldsymbol{x} = \lambda\boldsymbol{x}$, can be viewed as *eigenfunctions*.



Matrix $\mathbf{A}$ as an operator and quadratic form

- The adjacency matrix can be viewed as an operator

$$\boldsymbol{g} = \mathbf{A}\boldsymbol{f}; g(i) = \sum_{i \sim j} f(j)$$

- It can also be viewed as a quadratic form:

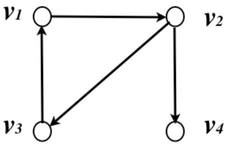
$$\boldsymbol{f}^\top \mathbf{A}\boldsymbol{f} = \sum_{e_{ij}} f(i)f(j)$$

The incidence matrix of a graph

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

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

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



The incidence matrix: A discrete differential operator

- The mapping $\boldsymbol{f} \longrightarrow \bigtriangledown \boldsymbol{f}$ is known as the *co-boundary mapping* of the graph.
- $(\bigtriangledown \boldsymbol{f})(e_{ij}) = f(v_j) - f(v_i)$

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

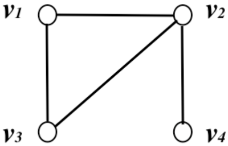
The Laplacian matrix of a graph

- $\mathbf{L} = \nabla^\top \nabla$
- $(\mathbf{L}\boldsymbol{f})(v_i) = \sum_{v_j \sim v_i} (f(v_i) - f(v_j))$
- Connection between the Laplacian and the adjacency matrices:

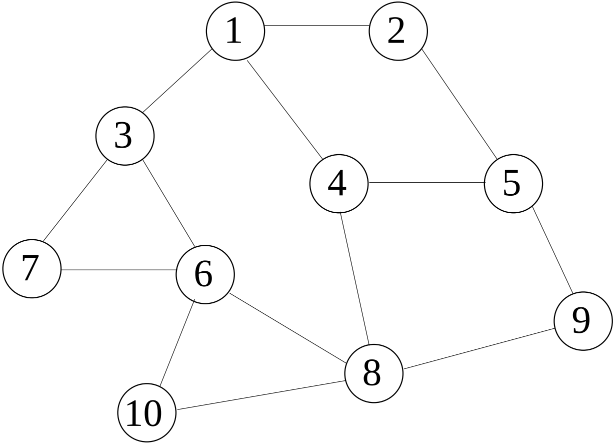
$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

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

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



Example: A graph with 10 nodes



The adjacency matrix

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

The Laplacian matrix

$$\mathbf{L} = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 3 & 0 & 0 & -1 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 3 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 4 & -1 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 & 0 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 4 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 2 \end{bmatrix}$$

The Eigenvalues of this Laplacian

$$\mathbf{\Lambda} = \begin{bmatrix} 0.0000 & 0.7006 & 1.1306 & 1.8151 & 2.4011 \\ 3.0000 & 3.8327 & 4.1722 & 5.2014 & 5.7462 \end{bmatrix}$$

Matrices of an undirected weighted graph

- We consider *undirected weighted graphs*; Each edge e_{ij} is weighted by $w_{ij} > 0$. We obtain:

$$\mathbf{\Omega} := \begin{cases} \Omega_{ij} = w_{ij} & \text{if there is an edge } e_{ij} \\ \Omega_{ij} = 0 & \text{if there is no edge} \\ \Omega_{ii} = 0 & \end{cases}$$

- The degree matrix: $\mathbf{D} = \sum_{i \sim j} w_{ij}$

The Laplacian on an undirected weighted graph

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

Other adjacency matrices

- The *normalized weighted adjacency matrix*

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

- The *transition* matrix of the Markov process associated with the graph:

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

Several Laplacian matrices

- The *unnormalized Laplacian* which is also referred to as the *combinatorial Laplacian* $\mathbf{L}_C$,
- the *normalized Laplacian* $\mathbf{L}_N$, and
- the *random-walk Laplacian* $\mathbf{L}_R$ also referred to as the *discrete Laplace operator*.

We have:

$$\begin{aligned}
\mathbf{L}_C &= \mathbf{D} - \mathbf{\Omega} \\
\mathbf{L}_N &= \mathbf{D}^{-1/2}\mathbf{L}_C\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{\Omega}_N \\
\mathbf{L}_R &= \mathbf{D}^{-1}\mathbf{L}_C = \mathbf{I} - \mathbf{\Omega}_R
\end{aligned}$$

Relationships between all these matrices

$$
\begin{aligned}
\mathbf{L}_C &= \mathbf{D}^{1/2}\mathbf{L}_N\mathbf{D}^{1/2} = \mathbf{D}\mathbf{L}_R \\
\mathbf{L}_N &= \mathbf{D}^{-1/2}\mathbf{L}_C\mathbf{D}^{-1/2} = \mathbf{D}^{1/2}\mathbf{L}_R\mathbf{D}^{-1/2} \\
\mathbf{L}_R &= \mathbf{D}^{-1/2}\mathbf{L}_N\mathbf{D}^{1/2} = \mathbf{D}^{-1}\mathbf{L}_C
\end{aligned}
$$

Some spectral properties of the Laplacians

Laplacian	Null space	Eigenvalues	Eigenvectors
$\mathbf{L}_C = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$	$\boldsymbol{u}_1 = \mathbf{1}$	$0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_n \leq 2\max_i(d_i)$	$\boldsymbol{u}_{i>1}^\top \mathbf{1} = 0$, $\boldsymbol{u}_i^\top \boldsymbol{u}_j = \delta_{ij}$
$\mathbf{L}_N = \mathbf{W}\mathbf{\Gamma}\mathbf{W}^\top$	$\boldsymbol{w}_1 = \mathbf{D}^{1/2}\mathbf{1}$	$0 = \gamma_1 < \gamma_2 \leq \ldots \leq \gamma_n \leq 2$	$\boldsymbol{w}_{i>1}^\top \mathbf{D}^{1/2}\mathbf{1} = 0$, $\boldsymbol{w}_i^\top \boldsymbol{w}_j = \delta_{ij}$
$\mathbf{L}_R = \mathbf{T}\mathbf{\Gamma}\mathbf{T}^{-1}$ $\mathbf{T} = \mathbf{D}^{-1/2}\mathbf{W}$	$\boldsymbol{t}_1 = \mathbf{1}$	$0 = \gamma_1 < \gamma_2 \leq \ldots \leq \gamma_n \leq 2$	$\boldsymbol{t}_{i>1}^\top \mathbf{D}\mathbf{1} = 0$, $\boldsymbol{t}_i^\top \mathbf{D}\boldsymbol{t}_j = \delta_{ij}$

Spectral properties of adjacency matrices

From the relationship between the normalized Laplacian and adjacency matrix: $\mathbf{L}_N = \mathbf{I} - \mathbf{\Omega}_N$ one can see that their eigenvalues satisfy $\gamma = 1 - \delta$.

Adjacency matrix	Eigenvalues	Eigenvectors
$\mathbf{\Omega}_N = \mathbf{W}\mathbf{\Delta}\mathbf{W}^\top$, $\mathbf{\Delta} = \mathbf{I} - \mathbf{\Gamma}$	$-1 \leq \delta_n \leq \ldots \leq \delta_2 < \delta_1 = 1$	$\boldsymbol{w}_i^\top \boldsymbol{w}_j = \delta_{ij}$
$\mathbf{\Omega}_R = \mathbf{T}\mathbf{\Delta}\mathbf{T}^{-1}$	$-1 \leq \delta_n \leq \ldots \leq \delta_2 < \delta_1 = 1$	$\boldsymbol{t}_i^\top \mathbf{D} \boldsymbol{t}_j = \delta_{ij}$

The Laplacian of a graph with one connected component

- $\mathbf{L}\boldsymbol{u} = \lambda \boldsymbol{u}$.
- $\mathbf{L}\mathbf{1} = \mathbf{0}$, $\lambda_1 = 0$ is the smallest eigenvalue.
- The *one* vector: $\mathbf{1} = (1 \dots 1)^\top$.
- $0 = \boldsymbol{u}^\top \mathbf{L} \boldsymbol{u} = \sum_{i,j=1}^{n} w_{ij}(u(i) - u(j))^2$.
- If any two vertices are connected by a path, then $\boldsymbol{u} = (u(1), \dots, u(n))$ 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.

A graph with $k > 1$ connected components

- 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.

The eigenspace of $\lambda_1 = 0$ with multiplicity k

- The eigenspace corresponding to $\lambda_1 = \ldots = \lambda_k = 0$ is spanned by the k mutually orthogonal vectors:

$$\begin{gathered} \boldsymbol{u}_1 = \mathbf{1}_{L_1} \\ \ldots \\ \boldsymbol{u}_k = \mathbf{1}_{L_k} \end{gathered}$$

- with $\mathbf{1}_{L_i} = (000011111 0000)^\top \in \mathbb{R}^n$
- These vectors are the *indicator vectors* of the graph's connected components.
- Notice that $\mathbf{1}_{L_1} + \ldots + \mathbf{1}_{L_k} = \mathbf{1}$

The Fiedler vector of the graph Laplacian

- The first non-null eigenvalue λ_{k+1} is called the Fiedler value.
- The corresponding eigenvector $\boldsymbol{u}_{k+1}$ is called the Fiedler vector.
- The multiplicity of the Fiedler eigenvalue depends on the graph's structure and it is difficult to analyse.
- The Fiedler value is the *algebraic connectivity of a graph*, the further from 0, the more connected.
- The Fiedler vector has been extensively used for *spectral bi-partioning*
- Theoretical results are summarized in Spielman & Teng 2007: `http://cs-www.cs.yale.edu/homes/spielman/`

Eigenvectors of the Laplacian of connected graphs

- $\boldsymbol{u}_1 = \mathbf{1}, \mathbf{L}\mathbf{1} = \mathbf{0}$.
- $\boldsymbol{u}_2$ is the *the Fiedler vector* with multiplicity 1.
- The eigenvectors form an orthonormal basis: $\boldsymbol{u}_i^\top \boldsymbol{u}_j = \delta_{ij}$.
- For any eigenvector $\boldsymbol{u}_i = (\boldsymbol{u}_i(v_1) \dots \boldsymbol{u}_i(v_n))^\top$, $2 \leq i \leq n$:

$$\boldsymbol{u}_i^\top \mathbf{1} = 0$$

- Hence the components of $\boldsymbol{u}_i$, $2 \leq i \leq n$ satisfy:

$$\sum_{j=1}^{n} \boldsymbol{u}_i(v_j) = 0$$

- Each component is bounded by:

$$-1 < \boldsymbol{u}_i(v_j) < 1$$

Laplacian embedding: Mapping a graph on a line

- Map a weighted graph onto a line such that connected nodes stay as close as possible, i.e., minimize $\sum_{i,j=1}^{n} w_{ij}(f(v_i) - f(v_j))^2$, or:

$$\arg\min_{\boldsymbol{f}} \boldsymbol{f}^\top \mathbf{L} \boldsymbol{f} \text{ with: } \boldsymbol{f}^\top \boldsymbol{f} = 1 \text{ and } \boldsymbol{f}^\top \mathbf{1} = 0$$

- The solution is the eigenvector associated with the smallest nonzero eigenvalue of the eigenvalue problem: $\mathbf{L}\boldsymbol{f} = \lambda \boldsymbol{f}$, namely the Fiedler vector $\boldsymbol{u}_2$.
- Practical computation of the eigenpair $\lambda_2, \boldsymbol{u}_2)$: the shifted inverse power method (see lecture 2).

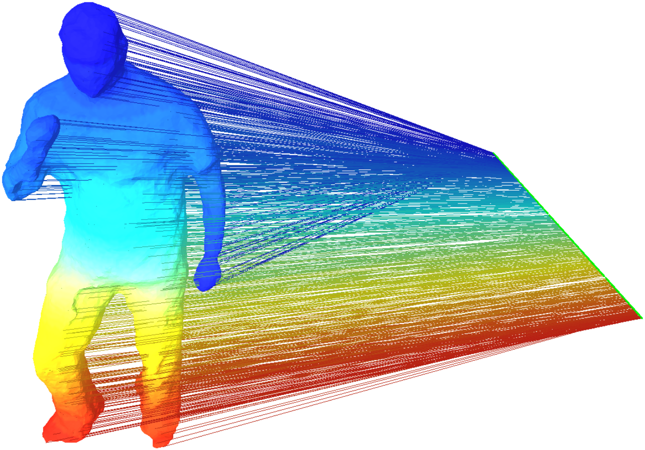
The shifted inverse power method (from Lecture 2)

- Let's consider the matrix $\mathbf{B} = \mathbf{A} - \alpha\mathbf{I}$ as well as an eigenpair $\mathbf{A}\boldsymbol{u} = \lambda\boldsymbol{u}$.
- $(\lambda - \alpha, \boldsymbol{u})$ becomes an eigenpair of $\mathbf{B}$, indeed:

$$\mathbf{B}\boldsymbol{u} = (\mathbf{A} - \alpha\mathbf{I})\boldsymbol{u} = (\lambda - \alpha)\boldsymbol{u}$$

  and hence $\mathbf{B}$ is a **real symmetric** matrix with eigenpairs $(\lambda_1 - \alpha, \boldsymbol{u}_1), \dots (\lambda_i - \alpha, \boldsymbol{u}_i), \dots (\lambda_D - \alpha, \boldsymbol{u}_D)$
- If $\alpha > 0$ is choosen such that $|\lambda_j - \alpha| \ll |\lambda_i - \alpha| \; \forall i \neq j$ then $\lambda_j - \alpha$ becomes the smallest (in magnitude) eivenvalue.
- The inverse power method (in conjuction with the *LU decomposition* of $\mathbf{B}$) can be used to estimate the eigenpair $(\lambda_j - \alpha, \boldsymbol{u}_j)$.

Example of mapping a graph on the Fiedler vector



Laplacian embedding

- Embed the graph in a k-dimensional Euclidean space. The embedding is given by the $n \times k$ matrix $\mathbf{F} = [\boldsymbol{f}_1 \boldsymbol{f}_2 \dots \boldsymbol{f}_k]$ where the i-th row of this matrix – $\boldsymbol{f}^{(i)}$ – corresponds to the Euclidean coordinates of the i-th graph node v_i.
- We need to minimize (Belkin & Niyogi '03):

$$\arg \min_{\boldsymbol{f}_1 \dots \boldsymbol{f}_k} \sum_{i,j=1}^{n} w_{ij} \|\boldsymbol{f}^{(i)} - \boldsymbol{f}^{(j)}\|^2 \text{ with: } \mathbf{F}^\top \mathbf{F} = \mathbf{I}.$$

- The solution is provided by the matrix of eigenvectors corresponding to the k lowest nonzero eigenvalues of the eigenvalue problem $\mathbf{L}\boldsymbol{f} = \lambda \boldsymbol{f}$.

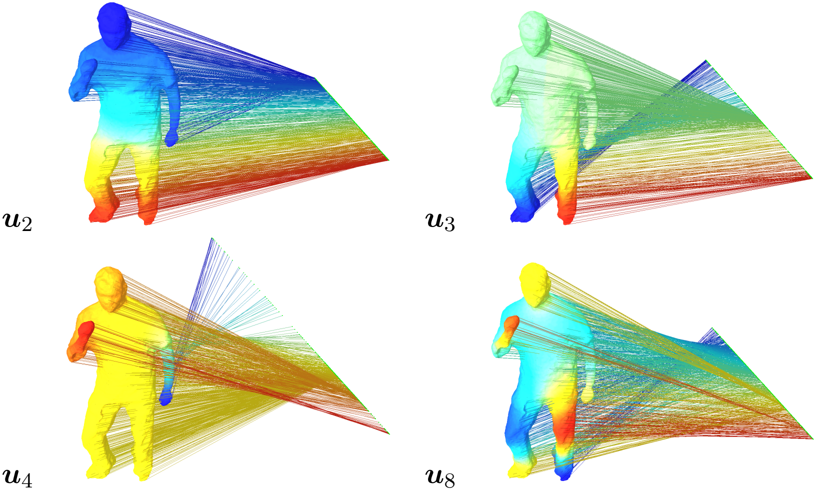
Spectral embedding using the *unnormalized* Laplacian

- Compute the eigendecomposition $\mathbf{L} = \mathbf{D} - \mathbf{\Omega}$.
- 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 $n \times k$ matrix $\mathbf{U} = [\boldsymbol{u}_2 \ldots \boldsymbol{u}_{k+1}]$:

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

- $\boldsymbol{u}_i^\top \boldsymbol{u}_j = \delta_{ij}$ (orthonormal vectors), hence $\mathbf{U}^\top \mathbf{U} = \mathbf{I}_k$.
- Column i $(2 \leq i \leq k+1)$ of this matrix is a mapping on the eigenvector $\boldsymbol{u}_i$.

Examples of one-dimensional mappings



Euclidean L-embedding of the graph's vertices

- (Euclidean) $\mathbf{L}$-embedding of a graph:

$$\mathbf{X} = \mathbf{\Lambda}_k^{-\frac{1}{2}} \mathbf{U}^\top = [\boldsymbol{x}_1 \ \dots \ \boldsymbol{x}_j \ \dots \ \boldsymbol{x}_n]$$

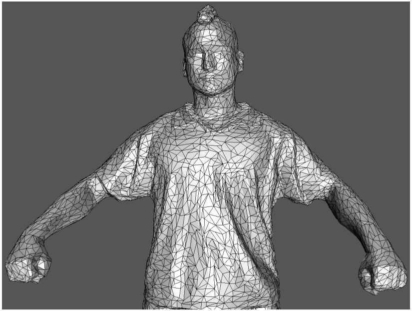
The coordinates of a vertex v_j are:

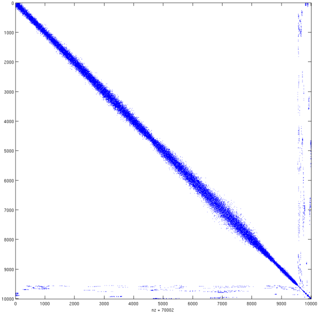
$$\boldsymbol{x}_j = \begin{pmatrix} \frac{\boldsymbol{u}_2(v_j)}{\sqrt{\lambda_2}} \\ \vdots \\ \frac{\boldsymbol{u}_{k+1}(v_j)}{\sqrt{\lambda_{k+1}}} \end{pmatrix}$$

- A formal justification of using this will be provided later.

The Laplacian of a mesh

A mesh may be viewed as a graph: $n = 10,000$ vertices, $m = 35,000$ edges. ARPACK finds the smallest 100 eigenpairs in 46 seconds.





Example: Shape embedding