ICSA 2023 Short Course

On the memory and speed scalability of spectral clustering

Guangliang Chen

6/11/2023, Ann Arbor, MI

About the instructor

- **Name**: Guangliang Chen
- **Education**:
  - B.S. Math, University of Science & Technology of China, 2003
  - Ph.D. Applied Math, University of Minnesota, 2009
- **Employment history**:
  - 2009-2013: Visiting Assist. Prof., Duke University
  - 2013-2014: Visiting Assist. Prof., Claremont McKenna College

  - 2014-2023: Associate Professor of Statistics, San Josè State University, California
  - 07/2023- : Associate Professor of Statistics and Data Science, Hope College, Holland, MI

- **Research interests**: Machine learning, data science.
- **Recent work**: Scalable spectral clustering, graph convolutional nets
- **Courses recently taught**: Mathematical data visualization[1], Statistical and machine learning classification

[1] `https://www.sjsu.edu/faculty/guangliang.chen/Math250.html`

Course Introduction

Lecture slides:

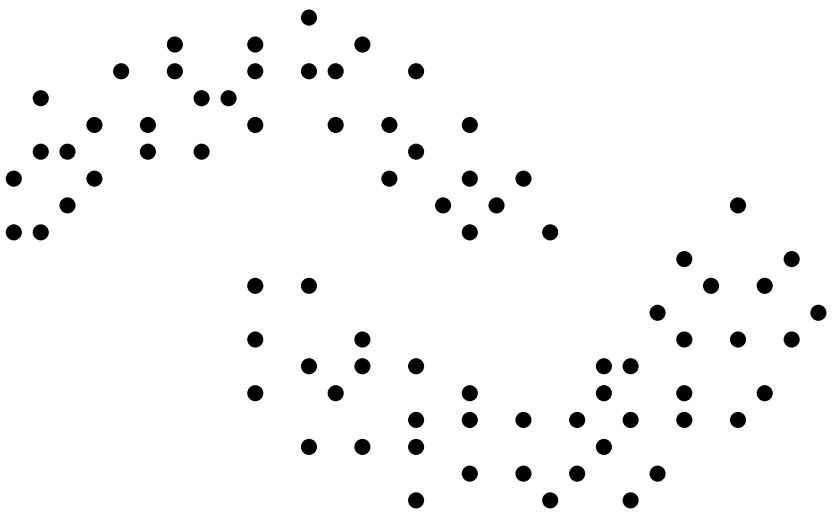
https://www.sjsu.edu/faculty/guangliang.chen/ICSA2023lecture.pdf

Matlab scripts:

1. **Plain spectral clustering**:
   https://www.sjsu.edu/faculty/guangliang.chen/ICSA2023scripts.zip
2. **SSC-cosine**:
   https://github.com/glsjsu/rprr2018

The data clustering problem

Assume a set of objects (e.g., $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^n$) and a similarity measure. We would like to divide the data set into k disjoint subsets (called clusters) such that within every cluster objects are "similar" (at least to their near neighbors), and between clusters objects are "dissimilar".



Remark. Clustering is an *unsupervised* machine learning task, often called *learning without a teacher*.

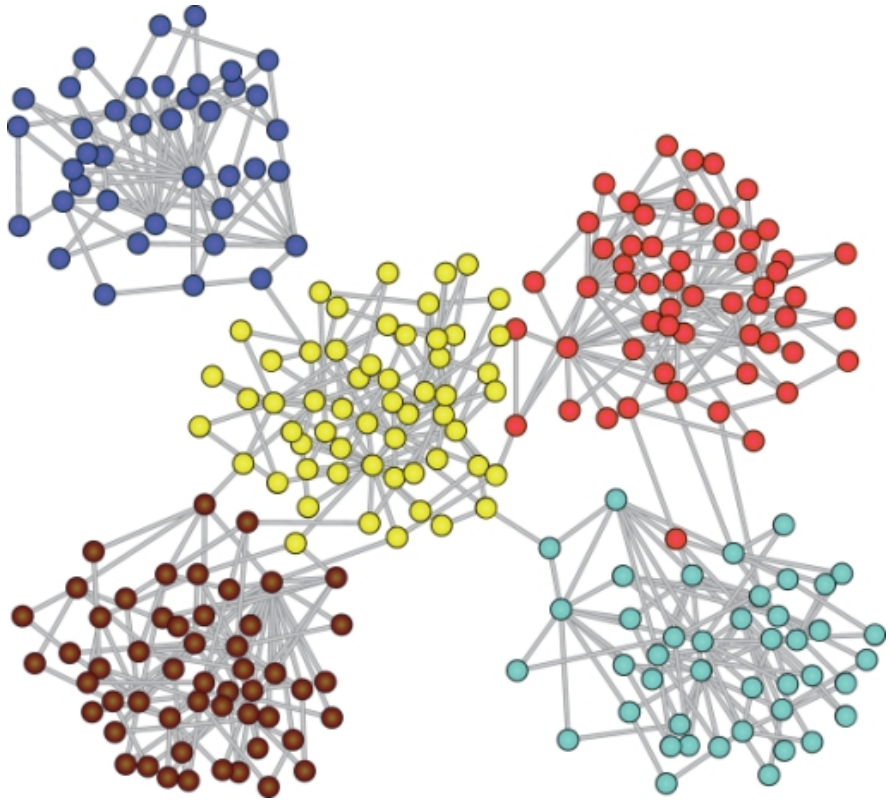
In contrast, classification is supervised (learning with a teacher).

Applications of clustering

Clustering is the process of discovering groups and patterns in data, a very common data mining task.

Examples of its application include

- Document grouping
- Customer segmentation
- Social network partitioning
- Image segmentation



Some necessary components of clustering

- Objects and their attributes
- Number of clusters
- Similarity measure
- Algorithm
- Evaluation criterion
- Interpretation of results

Clustering methods

- Hierarchical clustering
- Centroid-based clustering (e.g., k-means)
- Distribution-based clustering (e.g., mixture of Gaussians)
- Geometry-based clustering (e.g., subspace clustering)
- **Spectral clustering** (for separating non-intersecting manifolds)

What is spectral clustering?

A family of clustering algorithms that utilize the **spectral decomposition of a similarity matrix** constructed on the given data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$:

$$\mathbf{W} = (w_{ij}) \in \mathbb{R}^{n \times n}, \quad w_{ij} = \begin{cases} s(\mathbf{x}_i, \mathbf{x}_j), & \text{if } i \neq j \\ 0, & \text{if } i = j. \end{cases}$$

Here, $s(\cdot, \cdot)$ is a similarity function, such as

- a 0/1-valued **indicator function**,
- the **Gaussian radial basis function (RBF)**, and
- the **cosine similarity**.

References

- J. Shi and J. Malik, “Normalized cuts and image segmentation", IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 888–905, 2000.
- M. Meila and J. Shi, “A random walks view of spectral segmentation”, in Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics, 2001.
- A. Ng, M. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm”, in Advances in Neural Information Processing Systems 14, 2001, pp. 849–856.
- U. von Luxburg, “A tutorial on spectral clustering", Statistics and Computing, vol. 17, no. 4, pp. 395–416, 2007.
- R. Coifman and S. Lafon, “Diffusion maps”, Applied and Computational Harmonic Analysis, vol. 21, no. 1, pp. 5–30, 2006.

Advantages of spectral clustering

- Simple and easy to implement (a few lines of matrix operations)
- Flexible and accurate (can especially handle nonconvex clusters well)
- Rich theory (manifold learning, graph cut, random walk, matrix perturbation theory)
- Implementation through neural networks

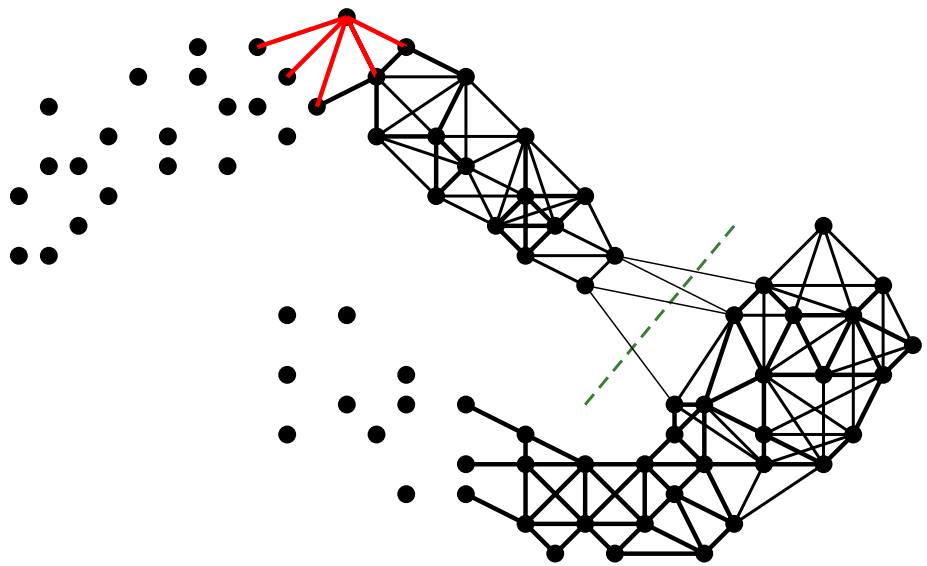
Spectral clustering as a graph cut problem

Given data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, we can represent it by an (undirected) similarity graph $G = (V, E)$.

Each vertex $v_i \in V$ represents a data point $\mathbf{x}_i$.

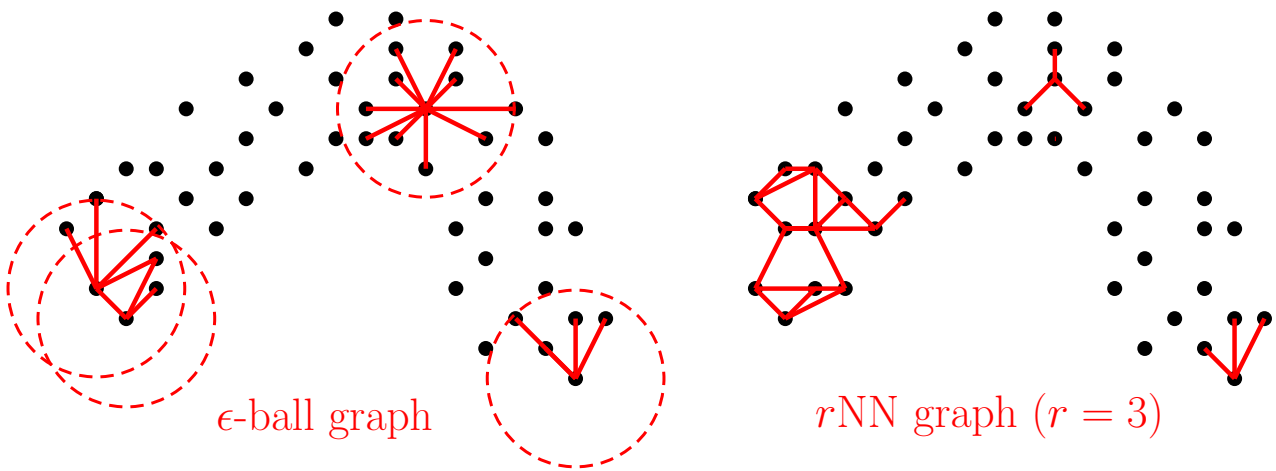
"Nearby" vertices $v_i, v_j \in V$ are connected by an edge, denoted as $e_{ij} \in E$.

Clustering can now be reformulated as a graph cut problem.



How to construct similarity graphs from Euclidean data

- **The ϵ-neighborhood graph**: connect with weight 1 any two points $\mathbf{x}_i, \mathbf{x}_j$ whose distance is less than ϵ
- **The rNN graph**: connect with weight 1 any two points $\mathbf{x}_i, \mathbf{x}_j$ if one is among the r nearest neighbors of the other;



- **The fully connected graph**: connect any two points $\mathbf{x}_i, \mathbf{x}_j$ with weight according to some similarity function s:

$$w_{ij} = s(\mathbf{x}_i, \mathbf{x}_j), \quad \text{for all } i, j = 1, \ldots, n$$

  For example,

  - **Gaussian weights**: $s(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$, where $\sigma > 0$ is a scale parameter whose value is fixed.
  - **Cosine weights**: $s(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \cdot \frac{\mathbf{x}_j}{\|\mathbf{x}_j\|}$, which is often used in documents clustering

It is also possible to mix up the different kinds of weights.

Course overview

The short course aims to introduce spectral clustering as a modern approach to clustering (in the setting of large data):

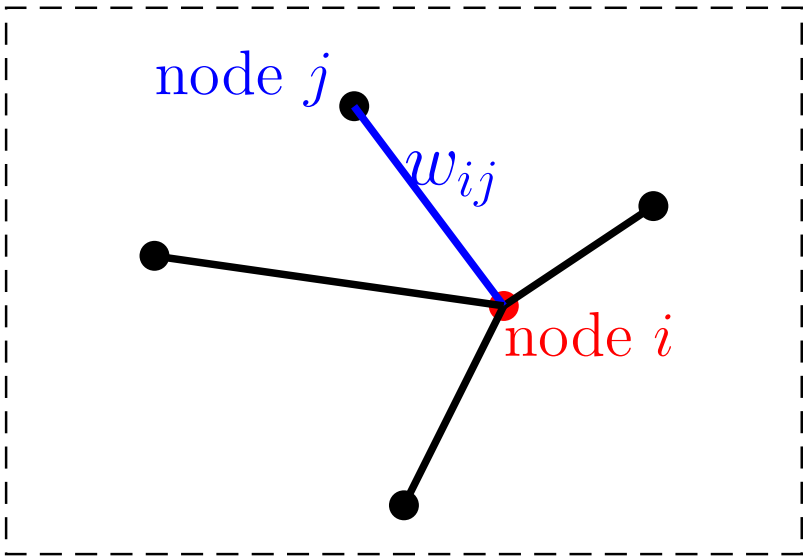
- Part 1: **Spectral graph theory**
- Part 2: **Spectral clustering theory and algorithms**
- Part 3: **Scalability and applications**
- Appendix: Advanced linear algebra

It also has a programming component (Matlab scripts are provided).

Part 1: Spectral Graph Theory

Some graph terminology

Let $G = (V, E, \mathbf{W})$ be an undirected, weighted graph.



The **degree** of a vertex $i \in V$ is defined as

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

It measures the *connectivity* of the vertex in the graph.

The degrees of all vertices of G can be used to form

- a **degree vector**:

$$\mathbf{d} = (d_1, \ldots, d_n)^T \in \mathbb{R}^n,$$

- a diagonal **degree matrix**:

$$\mathbf{D} = \mathrm{diag}(\mathbf{d}) \in \mathbb{R}^{n \times n}.$$

It is obvious that

$$\mathbf{d} = \mathbf{W} \cdot \mathbf{1}.$$

For example, for the undirected graph with the following weight matrix

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 & 0 & 0 \\ 1 & 0 & 0 & 1 & 4 \\ 2 & 0 & 0 & 1 & 2 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 4 & 2 & 1 & 0 \end{pmatrix}$$

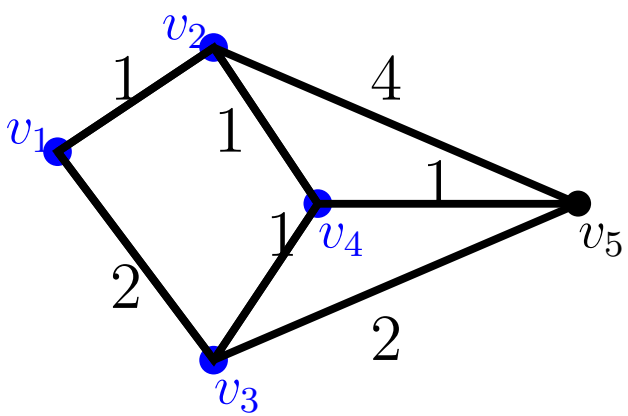
the degree vector and matrix are respectively

$$\mathbf{d} = \mathbf{W1} = \begin{pmatrix} 3 \\ 6 \\ 5 \\ 3 \\ 7 \end{pmatrix}, \qquad \mathbf{D} = \operatorname{diag}(\mathbf{d}) = \begin{pmatrix} 3 & & & & \\ & 6 & & & \\ & & 5 & & \\ & & & 3 & \\ & & & & 7 \end{pmatrix}$$

We define the **indicator vector** $\mathbf{1}_A$ associated to a subset of nodes $A \subseteq V$ by

$$\mathbf{1}_A = (a_1, \ldots, a_n)^T, \quad a_i = 1 \text{ (if } i \in A) \text{ and } a_i = 0 \text{ (if } i \notin A)$$

For example, for the subset of nodes $A = \{1, 2, 3, 4\}$ below (in blue),



the indicator vector is $\mathbf{1}_A = (1, 1, 1, 1, 0)^T$.

A **subgraph** of a given graph $G = (V, E, \mathbf{W})$, induced by a subset of vertices $A \subseteq V$, is another graph, $G_A = (A, E_A, \mathbf{W}_A)$

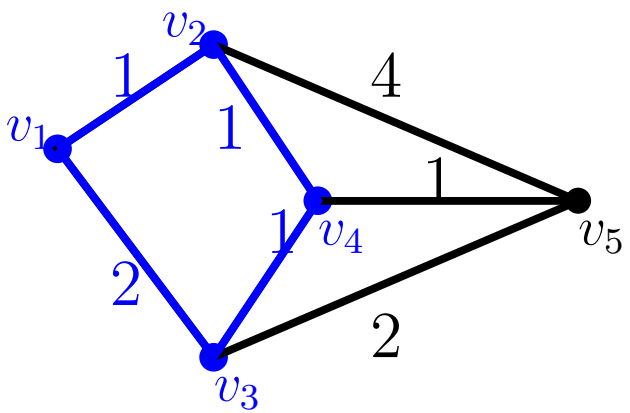
- whose vertex set is A,
- whose edge set E_A contains all of the edges of G that have both endpoints in A, i.e.,

$$E_A = \{\{i, j\} \in E \mid i, j \in A\}$$

- and whose weight matrix is the restriction of $\mathbf{W}$ to A:

$$\mathbf{W}_A = (w_{ij})_{i,j \in A} \in \mathbb{R}^{|A| \times |A|}.$$

For example, the subset of nodes $A = \{1, 2, 3, 4\}$ induces the following subgraph (in blue), with weight matrix:



$$\mathbf{W}_A = \begin{pmatrix} 0 & 1 & 2 & 0 \\ 1 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

There are two ways to measure the "size" of a subgraph $A \subset V$:

$$|A| = \#\text{vertices in } A;$$
$$\text{Vol}(A) = \sum_{i \in A} d_i.$$

The former simply counts the number of vertices in A while the latter measures how strongly the vertices in A are connected to all vertices of G.

For example, for the subgraph (in blue),

$$|A| = 4,$$
$$\text{Vol}(A) = d_1 + d_2 + d_3 + d_4 = 3 + 6 + 5 + 3 = 17.$$

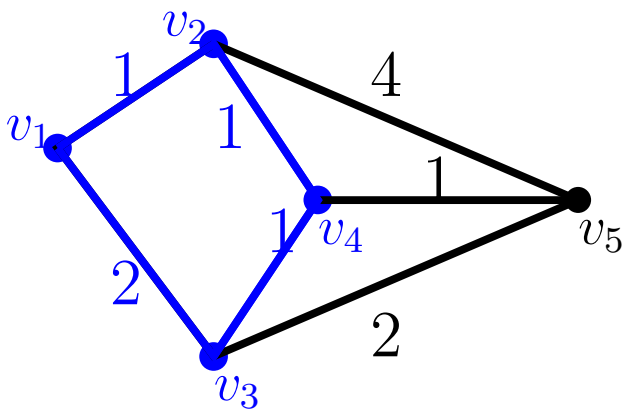
Clearly,

$$\mathrm{Vol}(V) = \mathrm{Vol}(A) + \mathrm{Vol}(\bar{A}).$$

Additionally, the total volume of the entire graph G is equal to the overall sum of the entries of the weight matrix $\mathbf{W}$:

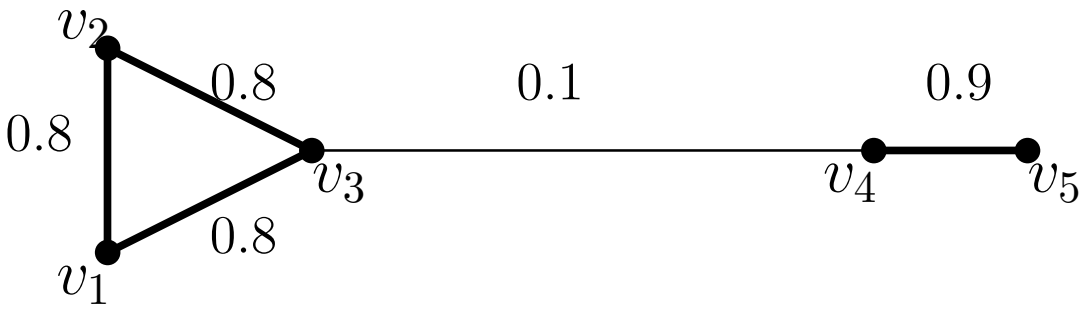
$$\mathrm{Vol}(V) = \sum_{i=1}^{n} d_i = \sum_{i=1}^{n} \left(\sum_{j=1}^{n} w_{ij} \right) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} = \mathbf{1}^T \mathbf{W} \mathbf{1}.$$

A subgraph $A \subset V$ is **connected** if any two vertices in A can be joined by a path such that all intermediate points also lie in A.



The subgraph A is called a **connected component** of V if it is connected and if there are no connections between vertices in A and $\bar{A}$.

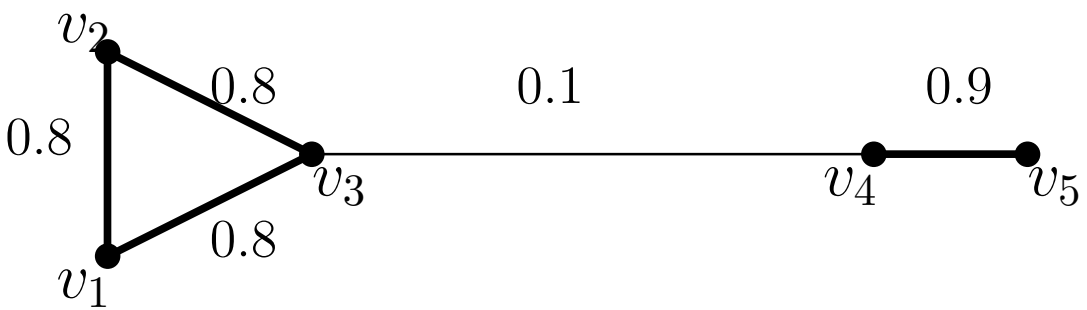
For example, the following graph has one connected component and is said to be **connected**:



In contrast, the following graph has two connected components (and thus is not connected):

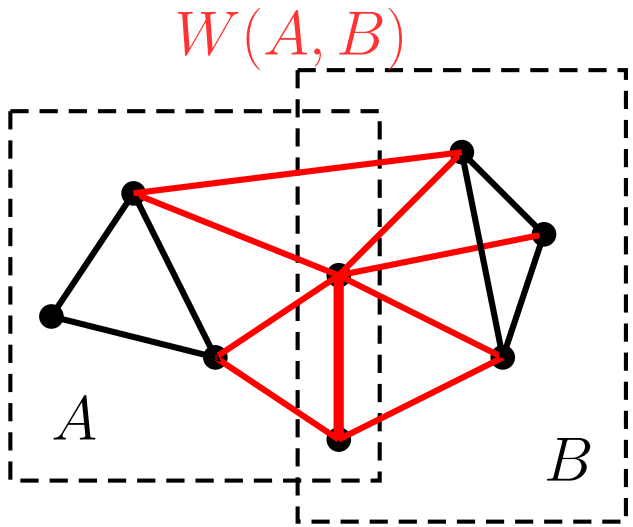
The nonempty (vertex) sets $A_1, \ldots, A_k$ form a **partition** of the graph if $A_i \cap A_j = \emptyset$ for all $i \neq j$ and $A_1 \cup \cdots \cup A_k = V$.

For example, $A = \{1, 2, 3\}$ and $B = \{4, 5\}$ form a partition of the following graph



For any two subsets $A, B \subset V$ (not necessarily disjoint), we define

$$W(A,B) = \sum_{i \in A} \sum_{j \in B} w_{ij} = \mathbf{1}_A^T \mathbf{W} \mathbf{1}_B.$$



When $B = \bar{A}$, the quantity $W(A, B)$ is called a **cut**:

$$\mathrm{Cut}(A, \bar{A}) = W(A, \bar{A}) = \sum_{i \in A} \sum_{j \in \bar{A}} w_{ij}$$

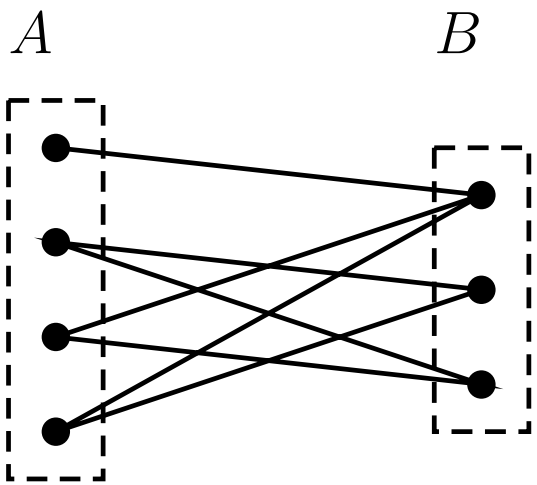
Another special case is when $B = V$,

$$W(A, V) = \sum_{i \in A} \sum_{j=1}^{n} w_{ij} = \sum_{i \in A} d_i = \mathrm{Vol}(A).$$

A graph $G = (V, E, \mathbf{W})$ is called a **bipartite graph** if there is a partition of the nodes $V = A \cup B$ such that there is no edge inside each of the two parts A and B, i.e.,

$$w_{ij} = 0,\ i, j \in A, \quad \text{and } w_{ij} = 0,\ i, j \in B.$$

In other words, all the edges in E are between A and B.



Functions on graphs

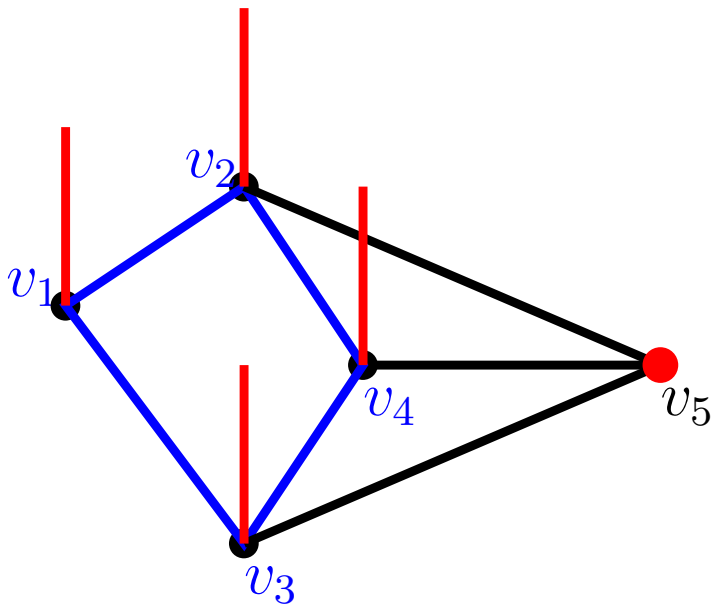
A (univariate) function over the graph G is a map from the vertex set of $G = (V, E)$ to real numbers, i.e.,

$$f : V \longmapsto \mathbb{R}$$

For a finite graph with $n = |V|$ nodes, we can alternatively use a vector $\mathbf{f} \in \mathbb{R}^n$ to represent a function over the graph, that is

$$\mathbf{f} = (f_i) \in \mathbb{R}^n, \quad f_i = f(v_i),\ 1 \leq i \leq n.$$

For any subset $A \subseteq V$, the associated indicator vector $\mathbf{1}_A$ is a function over the graph whose values are 1 over A and 0 over $\bar{A}$.

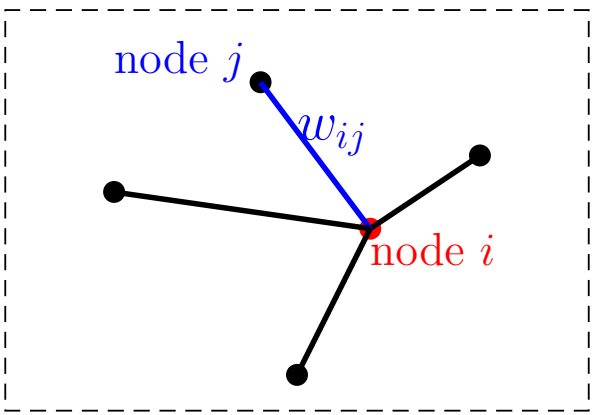


For this reason, we also refer to $\mathbf{1}_A$ as an indicator function.

A probabilistic model on weighted graphs

We can define a stochastic process on any undirected, weighted graph $G = (V, E, \mathbf{W})$. Suppose that there is no isolated vertex.

Imagine that a person initially stands on one of the vertices of G, say node i, and will move, one step a time, from vertex to vertex along the existing edges of the graph.



To choose which edge (incident on node i) to move along, the person will follow the following probabilities at node i:

$$p_{ij} = \frac{w_{ij}}{d_i}, \quad j \in V$$

This is a discrete distribution over the vertex set V because

$$p_{ij} \geq 0, \quad j \in V$$

and

$$\sum_{j \in V} p_{ij} = \frac{1}{d_i} \sum_{j \in V} w_{ij} = \frac{1}{d_i} \cdot d_i = 1.$$

Let X_k for each $k \geq 0$ denote the node where the person is at time k.

Then $\{X_k\}_{k\geq 0}$ is a **Markov chain** with state space $S = V$ because the process is discrete in time and the transitions do not depend on the previously visited nodes but only on the current location (by following the local distribution).

We say that the Markov chain process is induced by the weighted graph. The process is called a **random walk** because of the above interpretation.

The set of all transition probabilities form a **transition matrix**

$$\mathbf{P} = (p_{ij}) \in \mathbb{R}^{n\times n}, \quad n = |V|$$

In matrix notation, we have

$$\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}.$$

The transition matrix $\mathbf{P}$ has the following properties:

- $\mathbf{P}$ is nonnegative ($p_{ij} \geq 0$ for all $i, j \in V$);
- $\mathbf{P}$ is *row-stochastic* (i.e., $\sum_{j \in V} p_{ij} = 1$ for each $i \in V$):

$$\mathbf{P1} = \left(\mathbf{D}^{-1}\mathbf{W}\right)\mathbf{1} = \mathbf{D}^{-1}\left(\mathbf{W1}\right) = \mathbf{D}^{-1}\mathbf{d} = \mathbf{1}.$$

  where the last step is due to $\mathbf{D} = \mathrm{diag}(\mathbf{d})$.

For example, the Markov chain associated to the following undirected graph has a state space of $S = \{1, 2, 3, 4, 5\}$ and the transition matrix is

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 & 0 & 0 \\ 1 & 0 & 0 & 1 & 4 \\ 2 & 0 & 0 & 1 & 2 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 4 & 2 & 1 & 0 \end{pmatrix} \longrightarrow \mathbf{P} = \begin{pmatrix} 0 & \frac{1}{3} & \frac{2}{3} & 0 & 0 \\ \frac{1}{6} & 0 & 0 & \frac{1}{6} & \frac{4}{6} \\ \frac{2}{5} & 0 & 0 & \frac{1}{5} & \frac{2}{5} \\ 0 & \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{4}{7} & \frac{2}{7} & \frac{1}{7} & 0 \end{pmatrix}$$

It is obvious that this transition matrix $\mathbf{P} \in \mathbb{R}^{5\times 5}$ is nonnegative and row-stochastic.

We also mention a symmetric normalized version of the weight matrix $\mathbf{W}$:

$$\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} \in \mathbb{R}^{n\times n}.$$

This matrix has the same size with $\mathbf{P}$, but is symmetric (note that $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$ is not symmetric). However, the two matrices are similar, because

$$\underbrace{\mathbf{D}^{-1}\mathbf{W}}_{\mathbf{P}} = \underbrace{\mathbf{D}^{-1/2}}_{\text{inverse}} \cdot \underbrace{\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}}_{\widetilde{\mathbf{W}}} \cdot \underbrace{\mathbf{D}^{1/2}}_{\text{invertible}}.$$

This implies that the two matrices $\mathbf{P}, \widetilde{\mathbf{W}}$ have the same eigenvalues, which are all real (however, they do not necessarily have the same eigenvectors).

In fact, a vector $\mathbf{v} \in \mathbb{R}^n$ is an eigenvector of $\mathbf{P}$ corresponding to eigenvalue λ if and only if the vector $\mathbf{D}^{1/2}\mathbf{v}$ is an eigenvector of $\widetilde{\mathbf{W}}$ corresponding to the same eigenvalue λ:

$$\underbrace{\mathbf{D}^{-1}\mathbf{W}}_{\mathbf{P}} \cdot \mathbf{v} = \lambda \mathbf{v} \quad \Longleftrightarrow \quad \underbrace{\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}}_{\widetilde{\mathbf{W}}} \cdot \mathbf{D}^{1/2}\mathbf{v} = \lambda \cdot \mathbf{D}^{1/2}\mathbf{v}.$$

Because of the symmetry of $\widetilde{\mathbf{W}}$ and the fact that it is similar to $\mathbf{P}$, the matrix $\widetilde{\mathbf{W}}$ is often used to mathematically study the properties of $\mathbf{P}$ or numerically compute the eigenvalues and eigenvectors of $\mathbf{P}$.

Graph Laplacians

Let $G = (V, E)$ be an undirected, weighted graph with weight matrix $\mathbf{W}$ and degree matrix $\mathbf{D} = \mathrm{diag}(\mathbf{W1})$.

Def 0.1. The Laplacian matrix of the graph, or in short, the **graph Laplacian** is defined as

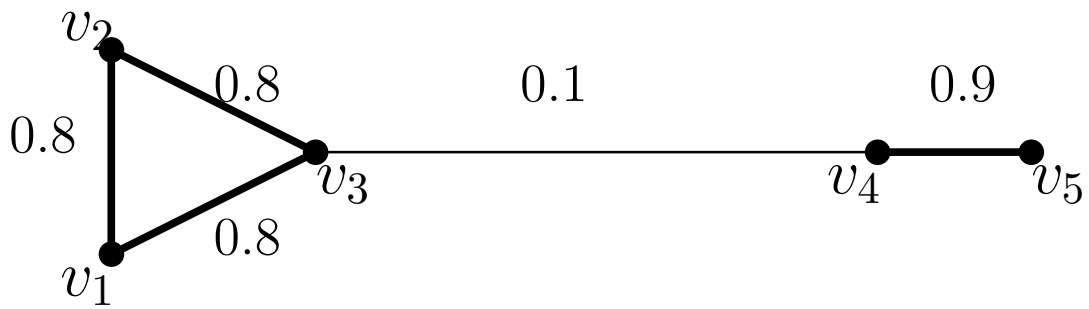
$$\mathbf{L} = \mathbf{D} - \mathbf{W}.$$

Remark. The graph Laplacian $\mathbf{L}$ is a symmetric matrix, with all the rows (and columns) summing to 0:

$$\mathbf{L1} = (\mathbf{D} - \mathbf{W})\mathbf{1} = \mathbf{D1} - \mathbf{W1} = \mathbf{d} - \mathbf{d} = \mathbf{0}.$$

This implies that $\mathbf{L}$ has a eigenvalue 0 with eigenvector $\mathbf{1}$.

Example 0.1.



$$\mathbf{L} = \begin{pmatrix} 1.6 & -0.8 & -0.8 & & \\ -0.8 & 1.6 & -0.8 & & \\ -0.8 & -0.8 & 1.7 & -0.1 & \\ & & -0.1 & 1 & -0.9 \\ & & & -0.9 & 0.9 \end{pmatrix}$$

The graph Laplacian has many nice properties.

Theorem 0.1. Suppose $G = (V, E, \mathbf{W})$ is an undirected, weighted graph with Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$. Then

- For every vector $\mathbf{f} \in \mathbb{R}^n$ we have

$$\mathbf{f}^T \mathbf{L} \mathbf{f} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (f_i - f_j)^2.$$

  Therefore, $\mathbf{L}$ is positive semidefinite and accordingly, its eigenvalues are all nonnegative: $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

- The algebraic multiplicity of the eigenvalue 0 equals the number of connected components in the graph. Furthermore, the corresponding eigenspace is

$$E(0) = \text{span}\{\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}\} \subset \mathbb{R}^n$$

where $A_1, \ldots, A_k$ are all the connected components of G.

Proof. The identity is verified as follows:

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(f_i - f_j)^2 &= \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left(f_i^2 + f_j^2 - 2f_i f_j\right) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_i^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_j^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_i f_j \\
&= \sum_{i=1}^{n}\left(\sum_{j=1}^{n} w_{ij}\right) f_i^2 + \sum_{j=1}^{n}\left(\sum_{i=1}^{n} w_{ij}\right) f_j^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_i f_j \\
&= \sum_{i=1}^{n} d_i f_i^2 + \sum_{j=1}^{n} d_j f_j^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_i f_j \\
&= 2\mathbf{f}^T\mathbf{D}\mathbf{f} - 2\mathbf{f}^T\mathbf{W}\mathbf{f} \\
&= 2\mathbf{f}^T\mathbf{L}\mathbf{f}.
\end{aligned}$$

Since $\mathbf{L} \in S^n(\mathbb{R})$, the eigenvalue $\lambda_1 = 0$ must have identical algebraic and geometric multiplicities, i.e., $a_1 = g_1$. To find the geometric multiplicity g_1, let $\mathbf{v} \neq \mathbf{0} \in \mathbb{R}^n$ be an arbitrary eigenvector corresponding to the eigenvalue 0, i.e., $\mathbf{Lv} = \mathbf{0}$. Then

$$0 = \mathbf{v}^T \underbrace{\mathbf{Lv}}_{\mathbf{0}} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(v_i - v_j)^2.$$

Since every term in the sum is nonnegative, we obtain that

$$w_{ij}(v_i - v_j)^2 = 0, \quad \text{for all } i \neq j$$

For two adjacent nodes i, j such that $w_{ij} \neq 0$, we must have $v_i = v_j$. This shows that $\mathbf{v}$ is constant over each connected component. That is,

$$\mathbf{v} = \sum_{i=1}^{k} c_i \mathbf{1}_{A_i}$$

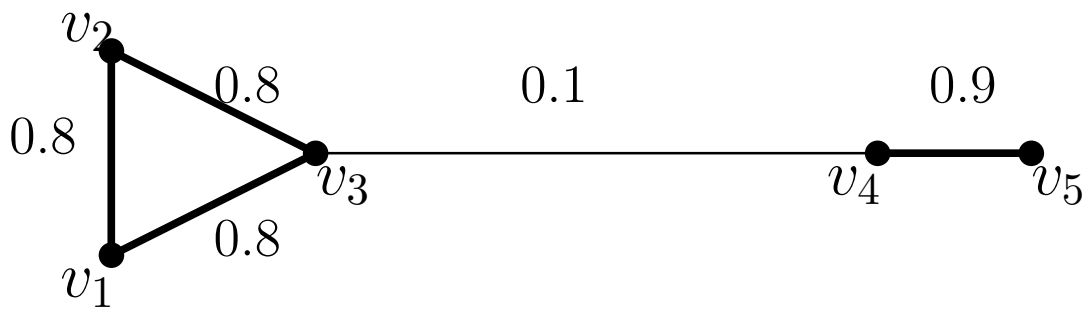
where $c_1, \ldots, c_k \in \mathbb{R}$ are free variables. This shows that the eigenspace of eigenvalue 0 is

$$E(0) = \operatorname{span}\{\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}\} \subset \mathbb{R}^n$$

Since $\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k} \in \mathbb{R}^n$ are orthogonal vectors, we conclude that its geometric (and thus also algebraic) multiplicity is

$$g_1 = \dim(E(0)) = k.$$

Example 0.2.

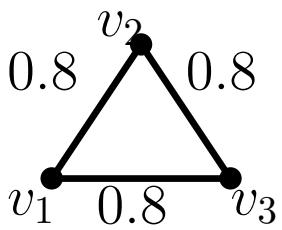


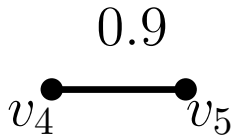
The Laplacian matrix of the above graph has eigenvalues

$$\lambda_1 = 0,\ \lambda_2 = 0.0788,\ \lambda_3 = 1.8465,\ \lambda_4 = 2.4000,\ \lambda_5 = 2.4747.$$

The algebraic multiplicity of the zero eigenvalue is 1, which coincides with the number of connected components in the graph.

Example 0.3. Consider the following modified graph





$$\mathbf{W} = \begin{pmatrix} 0 & .8 & .8 & 0 & 0 \\ .8 & .0 & .8 & 0 & 0 \\ .8 & .8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .9 \\ 0 & 0 & 0 & .9 & 0 \end{pmatrix}$$

It can be shown that

$$\det(\lambda \mathbf{I} - \mathbf{L}) = \lambda(\lambda - 2.4)^2 \cdot \lambda(\lambda - 1.8).$$

Thus, the unnormalized graph Laplacian has a repeated eigenvalue 0, with multiplicity 2 (the number of connected components).

Normalized graph Laplacians

In many cases, we need to deal with a normalized graph Laplacian. We present the definition below.

Suppose $G = (V, E, \mathbf{W})$ is an undirected, weighted graph with a positive definite degree matrix $\mathbf{D}$ and a Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$.

Let

$$\widetilde{\mathbf{L}}_{\text{rw}} = \mathbf{D}^{-1}\mathbf{L}, \qquad \text{and} \quad \widetilde{\mathbf{L}}_{\text{sym}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$$

They are called the **random-walk normalized Laplacian** and **symmetric normalized Laplacian**, respectively.

We make some comments on the two normalized graph Laplacians.

First, we can write

$$\widetilde{\mathbf{L}}_{\text{rw}} = \mathbf{D}^{-1}\underbrace{(\mathbf{D}-\mathbf{W})}_{\mathbf{L}} = \mathbf{I} - \underbrace{\mathbf{D}^{-1}\mathbf{W}}_{\mathbf{P}} = \mathbf{I} - \mathbf{P}$$

$$\widetilde{\mathbf{L}}_{\text{sym}} = \mathbf{D}^{-1/2}\underbrace{(\mathbf{D}-\mathbf{W})}_{\mathbf{L}}\mathbf{D}^{-1/2} = \mathbf{I} - \underbrace{\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}}_{\widetilde{\mathbf{W}}} = \mathbf{I} - \widetilde{\mathbf{W}}$$

This implies that that each pair of matrices, $(\widetilde{\mathbf{L}}_{\text{rw}}, \mathbf{P})$ or $(\widetilde{\mathbf{L}}_{\text{sym}}, \widetilde{\mathbf{W}})$, have the same eigenvectors but reversed eigenvalues:

$$\underbrace{(\mathbf{I}-\mathbf{P})}_{\widetilde{\mathbf{L}}_{\text{rw}}}\mathbf{v} = \lambda\mathbf{v} \quad \Longleftrightarrow \quad \mathbf{P}\mathbf{v} = (1-\lambda)\mathbf{v}.$$

Second, $\widetilde{\mathbf{L}}_{\text{sym}}$ is symmetric while $\widetilde{\mathbf{L}}_{\text{rw}}$ is not, but they are similar:

$$\underbrace{\mathbf{D}^{-1}\mathbf{L}}_{\widetilde{\mathbf{L}}_{\text{rw}}} = \underbrace{\mathbf{D}^{-1/2}}_{\text{inverse}} \cdot \underbrace{\mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}}_{\widetilde{\mathbf{L}}_{\text{sym}}} \cdot \underbrace{\mathbf{D}^{1/2}}_{\text{invertible}}.$$

Thus, both matrices have the same (real) eigenvalues but not the same eigenvectors.

In fact, a vector $\mathbf{v} \in \mathbb{R}^n$ is an eigenvector of $\widetilde{\mathbf{L}}_{\text{rw}}$ corresponding to eigenvalue λ if and only if the vector $\mathbf{D}^{1/2}\mathbf{v}$ is an eigenvector of $\widetilde{\mathbf{L}}_{\text{sym}}$ corresponding to the same eigenvalue λ:

$$\underbrace{\mathbf{D}^{-1}\mathbf{L}}_{\widetilde{\mathbf{L}}_{\text{rw}}} \cdot \mathbf{v} = \lambda \mathbf{v} \quad \Longleftrightarrow \quad \underbrace{\mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}}_{\widetilde{\mathbf{L}}_{\text{sym}}} \cdot \mathbf{D}^{1/2}\mathbf{v} = \lambda \cdot \mathbf{D}^{1/2}\mathbf{v}.$$

The normalized graph Laplacians have similar properties with the unnormalized Laplacian.

Theorem 0.2. For any undirected, weighted graph $G = (V, E, \mathbf{W})$,

1. The row sums of $\widetilde{\mathbf{L}}_{\text{rw}}$ are 0, i.e.,

$$\widetilde{\mathbf{L}}_{\text{rw}}\mathbf{1} = \left(\mathbf{D}^{-1}\mathbf{L}\right)\mathbf{1} = \mathbf{D}^{-1}\left(\mathbf{L}\mathbf{1}\right) = \mathbf{D}^{-1}\mathbf{0} = \mathbf{0}.$$

   This implies that $\widetilde{\mathbf{L}}_{\text{rw}}$ has an eigenvalue 0 with corresponding eigenvector $\mathbf{1}$ and the symmetric normalized graph Laplacian $\widetilde{\mathbf{L}}_{\text{sym}}$ has an eigenvalue 0 with corresponding eigenvector $\mathbf{D}^{1/2}\mathbf{1}$.

2. For any vector $\mathbf{f} \in \mathbb{R}^n$,

$$\mathbf{f}^T \widetilde{\mathbf{L}}_{\text{sym}} \mathbf{f} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2 .$$

   This implies that $\widetilde{\mathbf{L}}_{\text{sym}}$ is positive semidefinite and accordingly, its eigenvalues are all nonnegative: $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

3. The algebraic multiplicity of the eigenvalue 0 of $\widetilde{\mathbf{L}}_{\text{rw}}$ and $\widetilde{\mathbf{L}}_{\text{sym}}$ is equal to the number of connected components of the graph.

Proof. Using $\mathbf{D}^{-1/2}\mathbf{f} = \left(\frac{f_i}{\sqrt{d_i}}\right)$ in place of $\mathbf{f}$ in the identity $\mathbf{f}^T\mathbf{L}\mathbf{f}$, we have

$$\mathbf{f}^T\widetilde{\mathbf{L}}_{\text{sym}}\mathbf{f} = \left(\mathbf{f}^T\mathbf{D}^{-1/2}\right)\mathbf{L}\left(\mathbf{D}^{-1/2}\mathbf{f}\right) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2$$

Let $\mathbf{v} \neq \mathbf{0} \in \mathbb{R}^n$ be an arbitrary eigenvector of $\widetilde{\mathbf{L}}_{\text{sym}}$ corresponding to the eigenvalue 0, i.e., $\widetilde{\mathbf{L}}_{\text{sym}}\mathbf{v} = \mathbf{0}$. Using the definition of the symmetric normalized Laplacian, we have

$$\mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}\mathbf{v} = \mathbf{0} \quad \longrightarrow \quad \mathbf{L}\left(\mathbf{D}^{-1/2}\mathbf{v}\right) = \mathbf{0}$$

This shows that $\mathbf{D}^{-1/2}\mathbf{v}$ must be an eigenvector of $\mathbf{L}$ corresponding to the eigenvalue 0.

Therefore, we obtain that

$$\mathbf{D}^{-1/2}\mathbf{v} = \sum_{i=1}^{k} c_i \, \mathbf{1}_{A_i}, \quad \longrightarrow \quad \mathbf{v} = \sum_{i=1}^{k} c_i \, \mathbf{D}^{1/2}\mathbf{1}_{A_i}$$

where $A_1, \ldots, A_k$ represent all the connected components of G and $c_1, \ldots, c_k \in \mathbb{R}$ are arbitrary constants.

This shows that the eigenspace of $\widetilde{\mathbf{L}}_{\text{sym}}$ corresponding to eigenvalue 0 is

$$E(0) = \text{span}\{\mathbf{D}^{1/2}\mathbf{1}_{A_1}, \ldots, \mathbf{D}^{1/2}\mathbf{1}_{A_k}\} \subset \mathbb{R}^n$$

Since $\mathbf{D}^{1/2}\mathbf{1}_{A_1}, \ldots, \mathbf{D}^{1/2}\mathbf{1}_{A_k} \in \mathbb{R}^n$ are linearly independent vectors, we conclude that the geometric (and algebraic) multiplicity of eigenvalue 0 is

$$g_1 = \dim(E(0)) = k.$$

Example 0.4. Consider the graph that has two connected components:

By direct calculation, the random walk graph Laplacian is

$$\widetilde{\mathbf{L}}_{\mathrm{rw}} = \mathbf{D}^{-1}\mathbf{L} = \begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} & & \\ -\frac{1}{2} & 1 & -\frac{1}{2} & & \\ -\frac{1}{2} & -\frac{1}{2} & 1 & & \\ & & & 1 & -1 \\ & & & -1 & 1 \end{pmatrix}$$

with eigenvalues $\lambda_1 = \lambda_2 = 0, \lambda_3 = \lambda_4 = 1.5, \lambda_5 = 2$.

Now, for the connected graph that has an extra thin edge ($w_{34} = 0.1$), the random walk graph Laplacian is

$$\widetilde{\mathbf{L}}_{\text{rw}} = \begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} & & \\ -\frac{1}{2} & 1 & -\frac{1}{2} & & \\ -\frac{8}{17} & -\frac{8}{17} & 1 & -\frac{1}{17} & \\ & & -\frac{1}{10} & 1 & -\frac{9}{10} \\ & & & -1 & 1 \end{pmatrix}$$

Using software, the eigenvalues of the graph Laplacian are $0 < 0.0693 < 1.4773 < 1.5000 < 1.9534$. We see that the eigenvalue 0 now has a multiplicity of 1. However, the second eigenvalue is quite small, which is due to the weak link added between the two parts.

In the last two examples, the eigenvalues of the normalized Laplacians are all in the range $[0, 2]$. This in fact holds true in general, as stated in the following theorem.

Theorem 0.3. Let $G = (V, E, \mathbf{W})$ be an undirected, weighted, connected graph with $n = |V|$. Denote the eigenvalues of $\widetilde{\mathbf{L}}_{\text{sym}}$ and $\widetilde{\mathbf{L}}_{\text{rw}}$ by

$$0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n.$$

Then the following are true:

- The sum of the eigenvalues is n:

$$\sum_{i=2}^{n} \lambda_i = n.$$

This implies that

$$\lambda_2 \leq \frac{n}{n-1} \leq \lambda_n.$$

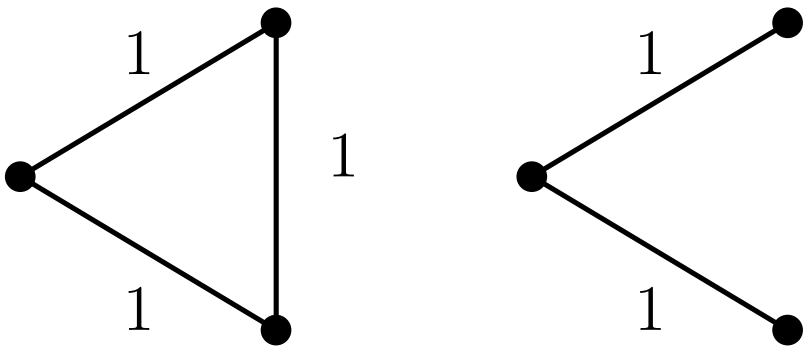
The equalities simultaneously hold true, i.e.,

$$\lambda_2 = \cdots = \lambda_n = \frac{n}{n-1},$$

if and only if G is a fully connected graph with equal weights.

- If G is not a fully connected graph, i.e., there exist a pair of distinct nodes k, ℓ such that $w_{k\ell} = 0$, we must have $\lambda_2 \leq 1$.
- $\lambda_n \leq 2$, with $\lambda_n = 2$ if and only if G is bipartite.

Example 0.5. Consider the following two graphs:



The first is a fully connected graph with equal weights. By the above theorem, the eigenvalues of the normalized graph Laplacians are

$$\lambda_1 = 0,\ \lambda_2 = \lambda_3 = \frac{3}{2}.$$

The second is a bipartite graph. Thus,

$$\lambda_1 = 0,\ \lambda_3 = 2, \qquad \text{and} \quad \lambda_2 \leq 1.$$

The exact value is $\lambda_2 = 1$, calculated as follows:

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \longrightarrow \quad \widetilde{\mathbf{L}}_{\text{rw}} = \begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

$$\longrightarrow \quad \det(\widetilde{\mathbf{L}}_{\text{rw}} - \lambda \mathbf{I}) = -\lambda(1-\lambda)(2-\lambda).$$

Remark. Let $G = (V, E, \mathbf{W})$ be an undirected, weighted graph. Below is what we know so far about the second smallest eigenvalue λ_2 of the normalized Laplacians:

- If G has more than one connected component, $\lambda_2 = 0$;
- If G is connected but not fully connected, $0 < \lambda_2 \leq 1$;
- If G is fully connected with equal weights, $\lambda_2 = \frac{n}{n-1}$ (maximum possible value).

It is a measure of the algebraic connectivity of G as a whole, and is called the **Fiedler value** of the graph (the corresponding eigenvector is called the **Fiedler eigenvector** of the graph).

Remark. Another measure of the algebraic connectivity of a weighted graph $G = (V, E, \mathbf{W})$ is the **conductance** of the graph, also called **Cheeger's constant**:

$$h_G = \min_{A \subset V} h_G(A) = \min_{A \subset V} \frac{\mathrm{Cut}(A, \bar{A})}{\min(\mathrm{Vol}(A), \mathrm{Vol}(\bar{A}))}.$$

The following theorem indicates that it is closely related to the second smallest eigenvalue λ_2 of the normalized graph Laplacians of G.

Theorem 0.4 (Cheeger's Inequality).

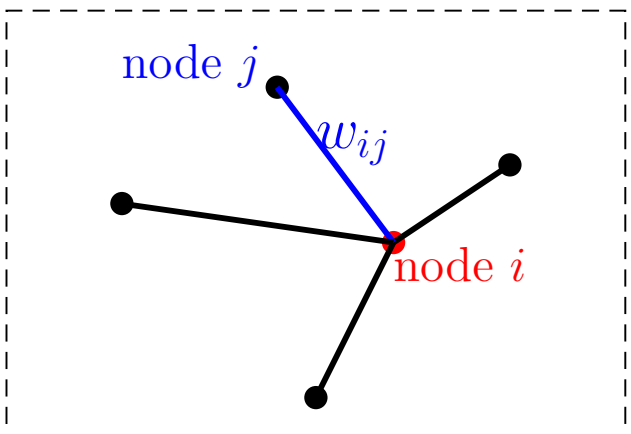
$$\frac{h_G^2}{2} \leq \lambda_2 \leq 2h_G$$

Interpretation of the graph Laplacians

Let $\mathbf{f}$ be a function defined on a graph $G = (V, E, \mathbf{W})$.

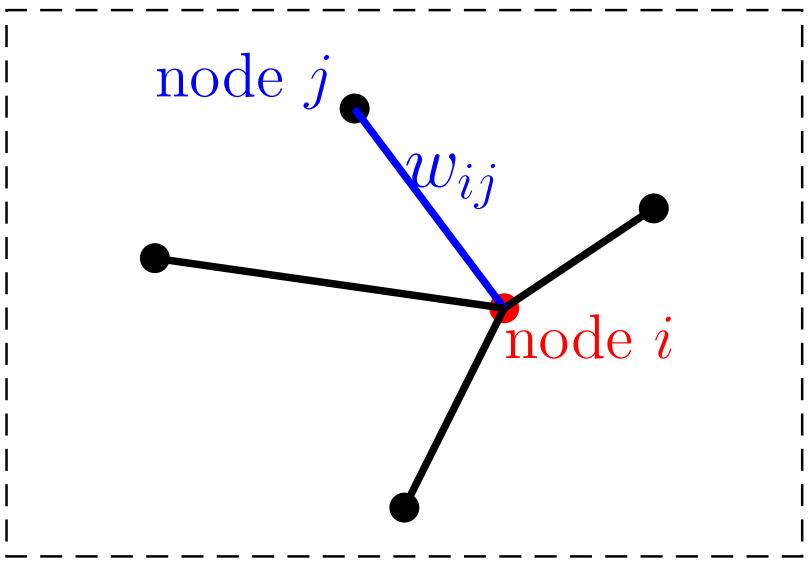
- $\mathbf{L} = \mathbf{D} - \mathbf{W}$: **net flow operator**

$$(\mathbf{Lf})_i = d_i f_i - \sum_{j \in V} w_{ij} f_j = \sum_{j \in V} w_{ij}(f_i - f_j), \quad i \in V$$



- $\widetilde{\mathbf{L}}_{\mathrm{rw}} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{P}$: **divergence operator**

$$\left(\widetilde{\mathbf{L}}_{\mathrm{rw}}\mathbf{f}\right)_i = f_i - \sum_{j\in V} p_{ij} f_j = \frac{1}{d_i}\sum_{j\in V} w_{ij}(f_i - f_j), \quad i \in V$$



- $\mathbf{S}_\gamma := \mathbf{I} - \gamma\widetilde{\mathbf{L}}_{\text{rw}} = (1-\gamma)\mathbf{I} + \gamma\mathbf{P}$: **Laplacian smoothing operator**, where $0 < \gamma < 1$ controls the strength of smoothing:

$$\mathbf{S}_\gamma \mathbf{f} = \mathbf{f} - \gamma\widetilde{\mathbf{L}}_{\text{rw}}\mathbf{f} = (1-\gamma)\mathbf{f} + \gamma\mathbf{P}\mathbf{f}$$

It can be used to construct graph convolutional nets:

$$\mathbf{F}^{(i)} = \sigma\left(\mathbf{S}_\gamma \mathbf{F}^{(i-1)}\mathbf{W}^{(i)}\right), \quad i \geq 1$$

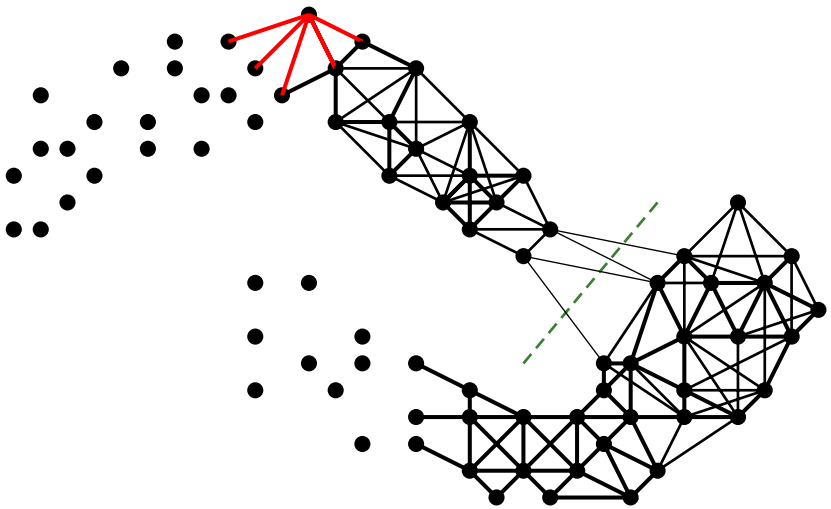
where

- $\mathbf{F}^{(i-1)}$: feature matrix of the graph nodes at layer $i-1$
- $\mathbf{W}^{(i)}$: matrix of weights between layers $i-1$ and i
- σ: activation function

Part 2: Spectral Clustering

SC via a graph cut point of view

$\mathbf{W}$ (as a weight matrix) defines a weighted graph on the given data.



Therefore, clustering $=$ finding an optimal cut (under some criterion).

Some graph terminology:

– **Degree matrix**: $\mathbf{D} = \mathrm{diag}(\mathbf{W1})$ with $\mathbf{D}_{ii} = \sum_j w_{ij}$.

– **Graph Laplacian**: $\mathbf{L} = \mathbf{D} - \mathbf{W}$ and its normalized version:

$$\mathbf{L}_{\mathrm{rw}} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \underbrace{\mathbf{D}^{-1}\mathbf{W}}_{\mathbf{P}\ \text{(row stochastic)}}$$

Remark. $\mathbf{P}$ defines a random walk on the graph.

The Normalized Cut (NCut) algorithm

Shi and Malik (2001) first considered the case of $k = 2$ clusters and proposed to perform spectral clustering by solving:

$$\min_{\substack{A\cup B=V\\ A\cap B=\emptyset}} \mathrm{Ncut}(A,B) = \mathrm{cut}(A,B)\left(\frac{1}{\mathrm{Vol}(A)} + \frac{1}{\mathrm{Vol}(B)}\right)$$

Remark. To minimize the above normalized cut,

- $\mathrm{cut}(A,B)$ needs to be small: little loss of edge weights
- both $\mathrm{Vol}(A)$ and $\mathrm{Vol}(B)$ need to be large: balanced clusters

Remark. Other measures of the quality of a cut include

- Cheeger constant

$$\mathrm{CheegerCut}(A, B) = \frac{\mathrm{cut}(A, B)}{\min(\mathrm{Vol}(A), \mathrm{Vol}(B))}$$

- Min-max cut

$$\mathrm{MinMaxCut}(A, B) = \mathrm{cut}(A, B)\left(\frac{1}{W(A, A)} + \frac{1}{W(B, B)}\right)$$

- Ratio cut

$$\mathrm{RatioCut}(A, B) = \mathrm{cut}(A, B)\left(\frac{1}{|A|} + \frac{1}{|B|}\right)$$

It turns out that the normalized cut can be conveniently expressed in terms of the graph Laplacian matrix.

Theorem 0.5. Given a similarity graph $G = \{V, E, \mathbf{W}\}$ with $n = |V|$ and a partition $A \cup B = V$, we have

$$\mathrm{NCut}(A, B) = \frac{\mathbf{x}^T \mathbf{L} \mathbf{x}}{\mathbf{x}^T \mathbf{D} \mathbf{x}},$$

where $\mathbf{x}$ is an indicator vector for the bipartition:

$$\mathbf{x} = (x_i) \in \mathbb{R}^n, \quad x_i = \begin{cases} \frac{1}{\mathrm{Vol}(A)}, & i \in A \\ \frac{-1}{\mathrm{Vol}(B)}, & i \in B \end{cases}$$

Proof. To simplify notation, let

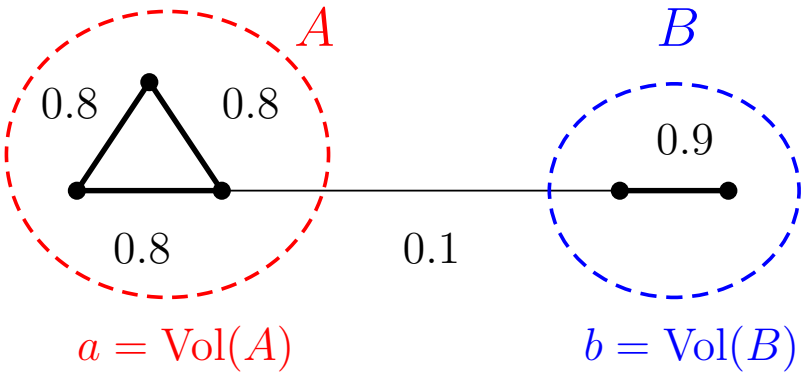
$$a = \mathrm{Vol}(A),\ b = \mathrm{Vol}(B),$$

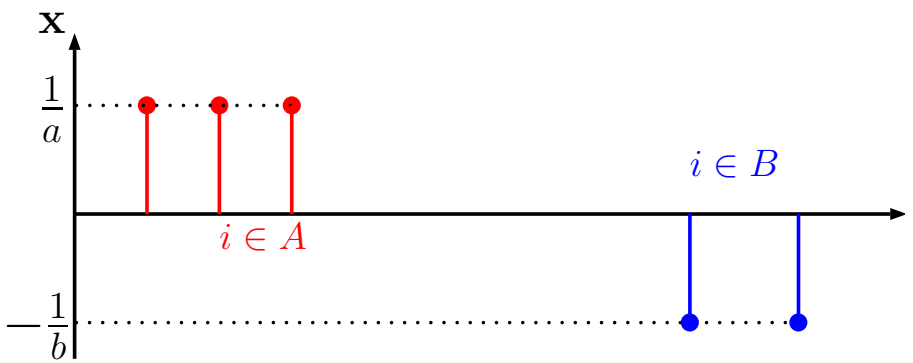
so that $\mathbf{x} = (x_i) \in \mathbb{R}^n$ with

$$x_i = \begin{cases} \frac{1}{a}, & i \in A \\ -\frac{1}{b}, & i \in B \end{cases}$$

It can be regarded as an indicator vector for the bipartition:

$$\mathbf{x} = \frac{1}{a}\mathbf{1}_A - \frac{1}{b}\mathbf{1}_B$$





We have

$$\begin{aligned}
\mathbf{x}^T\mathbf{L}\mathbf{x} &= \frac{1}{2}\sum_i\sum_j w_{ij}(x_i - x_j)^2 \\
&= \sum_{i\in A}\sum_{j\in B} w_{ij}\left(\frac{1}{a}+\frac{1}{b}\right)^2 \\
&= \mathrm{Cut}(A,B)\left(\frac{1}{a}+\frac{1}{b}\right)^2 \\
\mathbf{x}^T\mathbf{D}\mathbf{x} &= \sum_i d_i x_i^2 \\
&= \sum_{i\in A}\frac{1}{a^2}d_i + \sum_{j\in B}\frac{1}{b^2}d_i \\
&= \frac{1}{a^2}\mathrm{Vol}(A) + \frac{1}{b^2}\mathrm{Vol}(B) = \frac{1}{a}+\frac{1}{b}
\end{aligned}$$

It follows that

$$\frac{\mathbf{x}^T\mathbf{L}\mathbf{x}}{\mathbf{x}^T\mathbf{D}\mathbf{x}} = \mathrm{Cut}(A,B)\left(\frac{1}{a}+\frac{1}{b}\right) = \mathrm{NCut}(A,B)$$

Remark. The vector $\mathbf{x}$ also satisfies a hidden constraint

$$\begin{aligned}\mathbf{x}^T\mathbf{D}\mathbf{1} &= \sum_{i=1}^{n} x_i d_i \\ &= \frac{1}{a}\sum_{i\in A} d_i - \frac{1}{b}\sum_{i\in B} d_i \\ &= \frac{1}{a}\mathrm{Vol}(A) - \frac{1}{b}\mathrm{Vol}(B) \\ &= 0.\end{aligned}$$

Therefore, minimizing the NCut over all bipartitions of the graph is equivalent to minimizing the generalized Rayleigh quotient over all indicator vectors $\mathbf{x}$:

$$\min_{\substack{A\cup B=V\\ A\cap B=\emptyset}} \mathrm{NCut}(A,B) \quad \Longleftrightarrow \quad \min_{\substack{\mathbf{x} \text{ is binary}\\ \mathbf{x}^T\mathbf{D}\mathbf{1}=0}} \frac{\mathbf{x}^T\mathbf{L}\mathbf{x}}{\mathbf{x}^T\mathbf{D}\mathbf{x}}$$

This problem is discrete (and hard to solve), and in order to solve it efficiently, we relax it by eliminating the binary requirement:

$$\min_{\substack{\mathbf{x}\in\mathbb{R}^n\\ \mathbf{x}^T\mathbf{D}\mathbf{1}=0}} \frac{\mathbf{x}^T\mathbf{L}\mathbf{x}}{\mathbf{x}^T\mathbf{D}\mathbf{x}}.$$

The vectors $\mathbf{x}$ in the new, expanded domain are expected to be a smoothed-out version of the discrete $\mathbf{x}$ and still contain the cluster information.

Remark. The relaxed NCut problem is equivalent to a locality-preserving manifold learning algorithm, called **Laplacian Eigenmaps**:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_i\sum_j w_{ij}(x_i - x_j)^2$$

subject to

$$\mathbf{x}^T\mathbf{D}\mathbf{1} = 0, \quad \mathbf{x}^T\mathbf{D}\mathbf{x} = 1$$

To minimize the objective function, any two data points that are close in original space ($w_{ij} \approx 1$) must remain close in the embedding domain ($x_i \approx x_j$).

The first constraint $0 = \mathbf{x}^T\mathbf{D}\mathbf{1} = \sum d_i x_i$ removes translation invariance, while the second ($1 = \mathbf{x}^T\mathbf{D}\mathbf{x} = \sum d_i x_i^2$) fixes the scale of embedding.

The relaxed NCut problem is a restricted generalized Rayleigh quotient problem:

- Without the constraint $\mathbf{x}^T\mathbf{D}\mathbf{1} = 0$, the minimizer would be the smallest generalized eigenvector of $(\mathbf{L}, \mathbf{D})$, i.e., $\mathbf{1}$:

$$\mathbf{L}\mathbf{1} = 0 \cdot \mathbf{D}\mathbf{1}$$

  and the minimum value is $\lambda_1 = 0$.

- The constraint $\mathbf{x}^T\mathbf{D}\mathbf{1} = 0$ effectively excludes the trivial solution and thus the new minimizer would be the second smallest generalized eigenvector of $(\mathbf{L}, \mathbf{D})$:

$$\mathbf{L}\,\mathbf{x} = \lambda_2 \mathbf{D}\mathbf{x} \qquad (\lambda_2 > 0)$$

In sum, we need to find the second smallest generalized eigenvector of $(\mathbf{L}, \mathbf{D})$ and use it to partition the data into two clusters .

We already know that it coincides with the second smallest eigenvector of $\widetilde{\mathbf{L}}_{\text{rw}} = \mathbf{D}^{-1}\mathbf{L}$.

In fact, it is also the second largest eigenvector of $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$, because λ is an eigenvalue of $\widetilde{\mathbf{L}}_{\text{rw}}$ if and only if $1 - \lambda$ is an eigenvalue of $\mathbf{P}$ with the same corresponding eigenvector:

$$\underbrace{(\mathbf{I} - \mathbf{P})}_{\widetilde{\mathbf{L}}_{\text{rw}}}\mathbf{x} = \lambda\mathbf{x} \quad \Longleftrightarrow \quad \mathbf{P}\mathbf{x} = (1 - \lambda)\mathbf{x}$$

We will use the matrix $\mathbf{P}$ to efficiently compute $\mathbf{x}$.

Remark. The RatioCut criterion leads to the following relaxed problem:

$$\min_{\substack{\mathbf{x}\in\mathbb{R}^n:\\ \mathbf{x}^T\mathbf{1}=0}} \frac{\mathbf{x}^T\mathbf{L}\mathbf{x}}{\mathbf{x}^T\mathbf{x}}.$$

The solution is obviously given by the second smallest eigenvector of the unnormalized graph Laplacian $\mathbf{L}$:

$$\mathbf{L}\mathbf{x} = \lambda_2\mathbf{x}.$$

The resulting algorithm is equivalent to that of Ncut when the clusters have comparable sizes and is worse otherwise.

Algorithm 1 Normalized Cut for 2-way clustering (by Shi and Malik)

Input: Data $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^d$, and scale parameter σ

Output: A partition of $X = C_1 \cup C_2$

Steps:

1: Calculate the weight matrix

$$\mathbf{W} = (w_{ij}) \in \mathbb{R}^{n \times n}, \quad w_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$$

1: Compute the degree matrix $\mathbf{D} = \mathrm{diag}(\mathbf{W1})$, and use it to normalize $\mathbf{W}$ to get $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$.

2: Find the second largest eigenvector $\mathbf{v}_2$ of $\mathbf{P}$.

3: Assign labels based on the sign of the coordinates of $\mathbf{v}_2$.

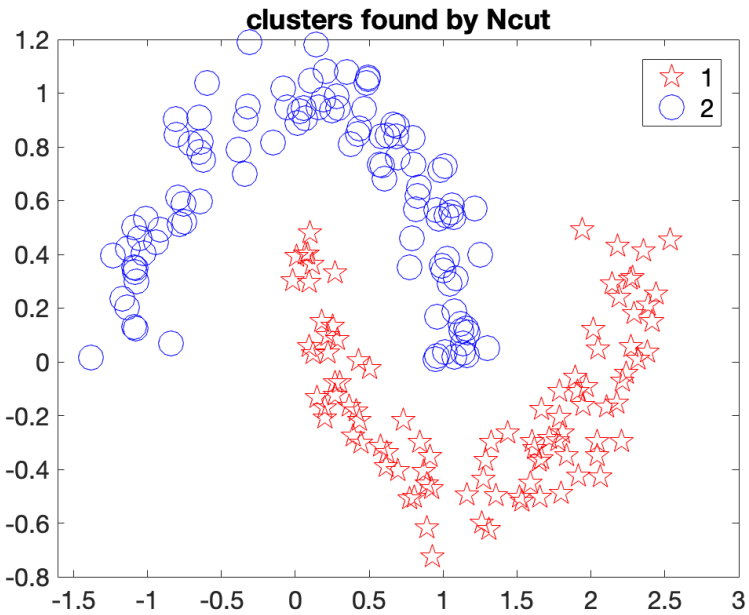
How to choose σ

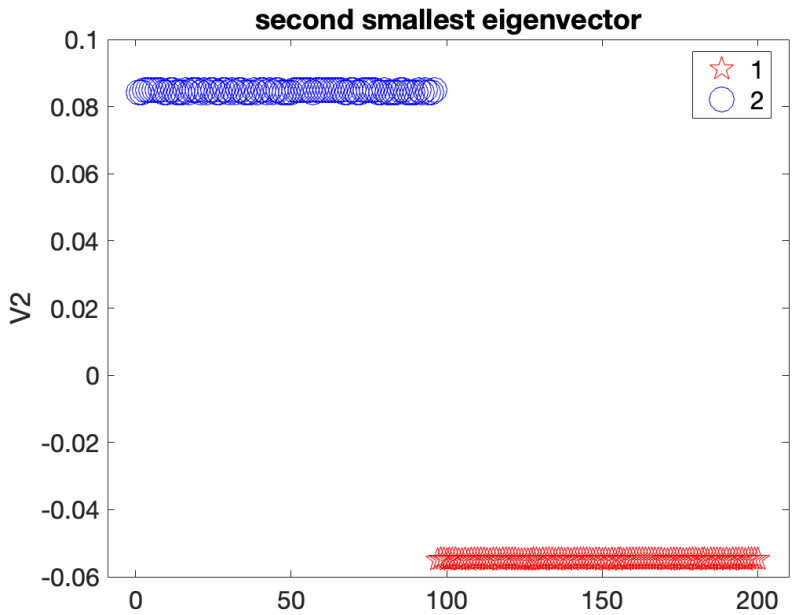
The parameter σ can be set directly as the average distance of the data points to their respective rNNs in the data:

$$\sigma = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{x}_i^{(r\text{nn})}\|.$$

For fast speed, use a subset of 30 to 50 randomly selected points to calculate σ. Additionally, $r = \mathcal{O}(\log(n))$ and typically, r is 6 to 10.

Computer demonstration





(For NCut, $\mathbf{v}_1 = \mathbf{1}$ is discarded because it is a trivial eigenvector of $\mathbf{P}$)

What if $k > 2$?

Use the subsequent eigenvectors (besides $\mathbf{v}_2$) of $\mathbf{P}$:

$$\mathbf{V} = [\mathbf{v}_2, \mathbf{v}_3, \ldots, \mathbf{v}_k] \in \mathbb{R}^{n \times (k-1)}$$

since they represent suboptimal 2-way partitions.

Now regard the rows of $\mathbf{V}$ as an embedding of the original data in $\mathbf{X}$,

$$\mathbf{X}(i,:) \in \mathbb{R}^d \quad \longrightarrow \quad \mathbf{V}(i,:) \in \mathbb{R}^{k-1}, \quad i = 1, \ldots, n$$

and apply k-means to group the row vectors of $\mathbf{V}$ into k clusters.

Algorithm 2 Normalized Cut (by Shi and Malik)

Input: Data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, #clusters k, scale parameter σ

Output: A partition $C_1, \ldots, C_k$

1: Construct a weighted graph by assigning weights

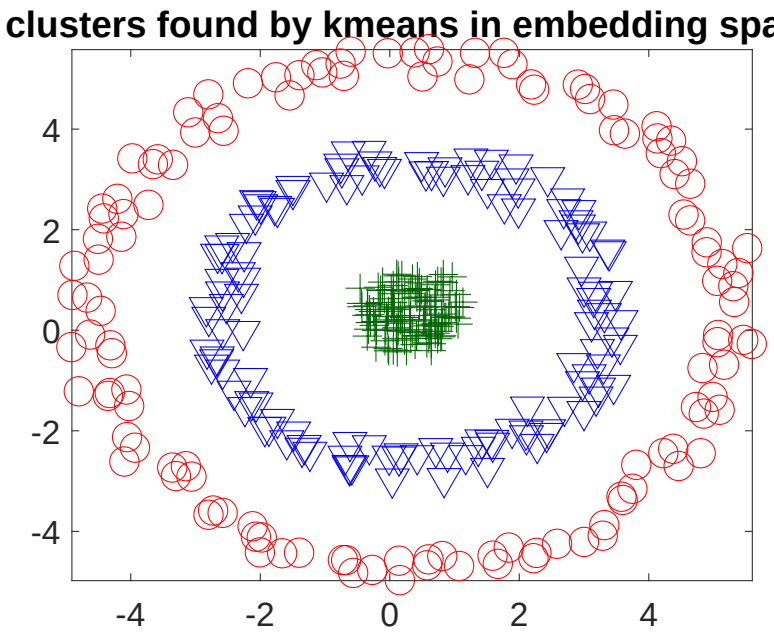
$$\mathbf{W} = (w_{ij}), \quad w_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$$

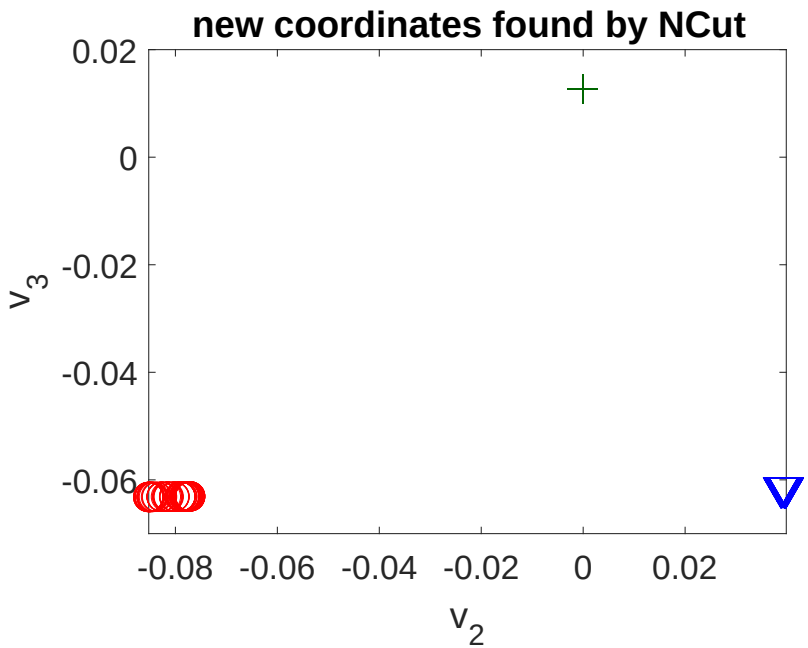
2: Find the degree matrix $\mathbf{D} = \mathrm{diag}(\mathbf{W1})$ and use it to normalize $\mathbf{W}$ to get $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$.

3: Find the 2nd to kth largest eigenvectors $\mathbf{V} = [\mathbf{v}_2 \ldots \mathbf{v}_k]$ of $\mathbf{P}$.

4: Apply k-means to group the rows of $\mathbf{V}$ into k clusters.

Computer demonstration





(For NCut, $\mathbf{v}_1 = \mathbf{1}$ is discarded because it is a trivial eigenvector of $\mathbf{P}$)

The random walk perspective

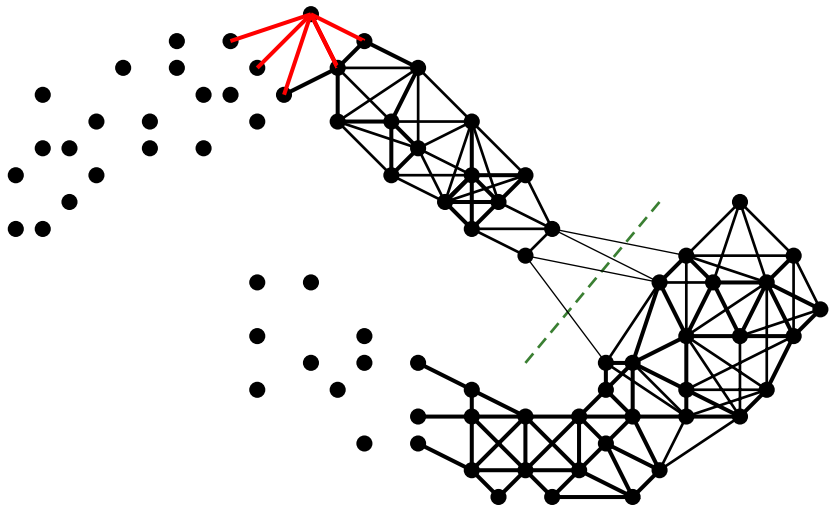
The matrix

$$\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$$

is row-stochastic and thus defines a *random walk* on the graph.

It can be shown that if the graph is connected but non-bipartite, and the random walk has converged to the stationary distribution,

$$\mathrm{Ncut}(A, \bar{A}) = P(\bar{A} \mid A) + P(A \mid \bar{A}).$$



If the random walk is moved forward (for a small number of t steps), then it can generate diffusion coordinates: $\mathbf{U}^{(t)} = \mathbf{U}\mathbf{\Lambda}^t$.

The matrix perturbation perspective

Ng, Jordan and Weiss (2001) proposed a spectral clustering algorithm from a **matrix perturbation** point of view.

Their algorithm uses the bottom eigenvectors of the symmetric graph Laplacian

$$\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \widetilde{\mathbf{W}},$$

which are the top eigenvectors of the $\widetilde{\mathbf{W}}$ matrix, to embed the data for clustering.

The matrix $\widetilde{\mathbf{W}}$ can be regarded as a perturbed version of a block-diagonal matrix with each block corresponding to a distinct cluster.

Algorithm 3 Spectral clustering by Ng, Jordan and Weiss (2001)

Input: Data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, #clusters k, and scale parameter σ

Output: Clusters $C_1, \ldots, C_k$

1: Construct the weight matrix $\mathbf{W} = (w_{ij})$ by

$$w_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}, \quad i \neq j \quad (0 \text{ otherwise})$$

2: Compute $\mathbf{D} = \operatorname{diag}(\mathbf{W1})$ and use it to obtain $\widetilde{\mathbf{W}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$

3: Find the top k eigenvectors of $\widetilde{\mathbf{W}}$ to form an eigenvectors matrix $\mathbf{U} = [\mathbf{u}_1 \ldots \mathbf{u}_k] \in \mathbb{R}^{n \times k}$

4: Renormalize the rows of $\mathbf{U}$ to have unit length and apply k-means to group them into k clusters

Application to image segmentation

Given an image $\mathbf{A} \in \mathbb{R}^{m \times n}$, the goal of image segmentation is to group pixels based on the content:

- Input data: mn pixels;
- Clusters tend to be local;
- Pixels are considered similar only if they are close in both distance and intensity value.

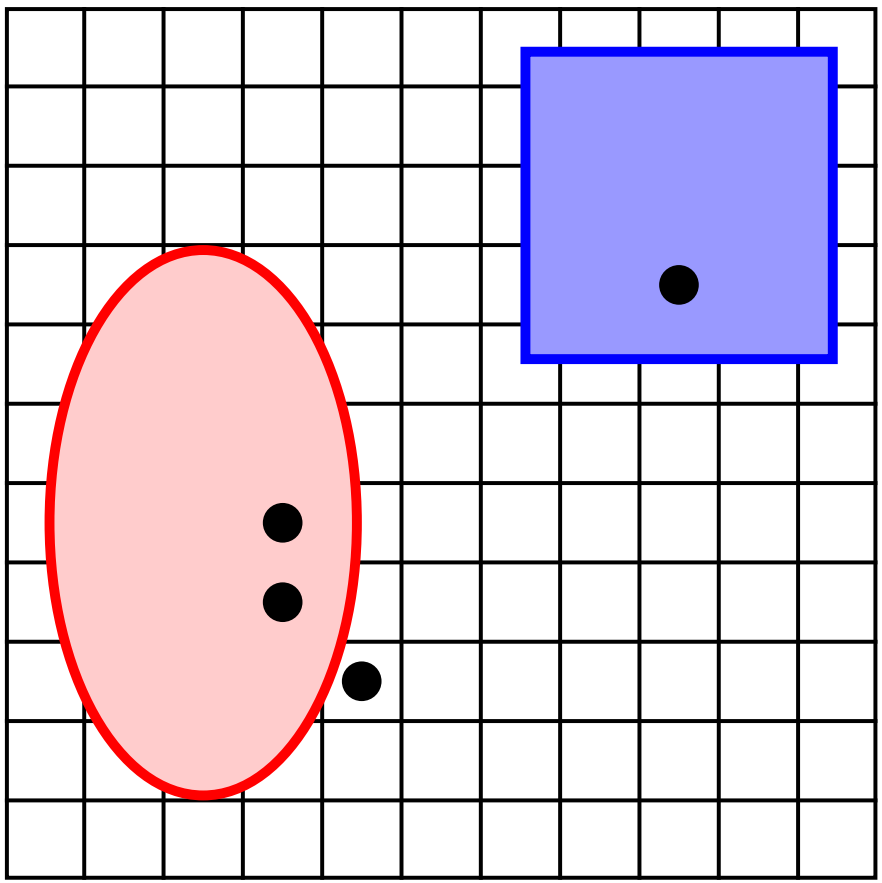


Image segmentation by spectral clustering

In principle, one only needs to compute a similarity matrix between all pixels and feed it to spectral clustering algorithms such as Ncut and NJW:

$$\mathbf{W} = (w_{(i,j),(i',j')}) \in \mathbb{R}^{mn \times mn},$$

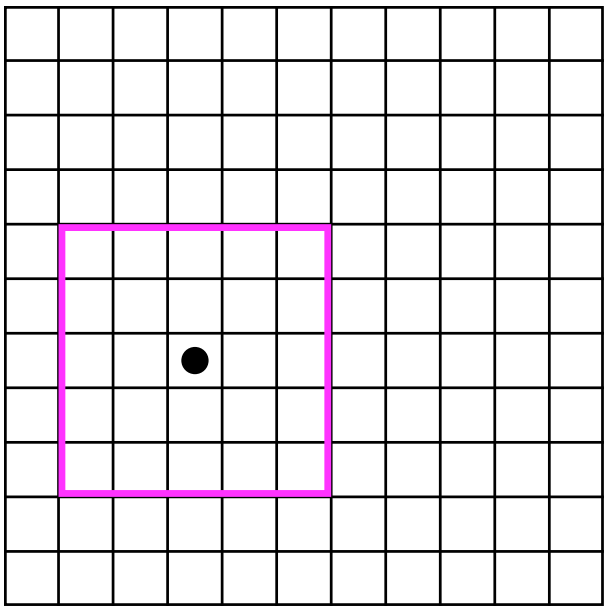
where

$$w_{(i,j),(i',j')} = \exp\left(-\frac{(i-i')^2 + (j-j')^2}{2\sigma_P^2}\right) \exp\left(-\frac{(a_{ij} - a_{i'j'})^2}{2\sigma_I^2}\right)$$

However, computationally the large size of $\mathbf{W}$ may be an issue.

One way to mitigate the computational burden is to restrict the calculation of $w_{(i,j),(i',j')}$, for each fixed pixel (i,j), to only nearby pixels (i',j').

This will yield a sparse $\mathbf{W}$ which is computationally much more efficient.



Part 3: Scalability of spectral clustering

- Speed scalability
- Memory scalability

Computational challenges

Spectral clustering has achieved superior results in many applications (such as image segmentation, documents clustering, social network partitioning), but requires significant computational power due to the matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$:

- Extensive memory requirement: $\mathcal{O}(n^2)$
- High computational cost:
  - Construction of $\mathbf{W}$: $\mathcal{O}(n^2 d)$
  - Spectral decomposition of $\mathbf{W}$: $\mathcal{O}(n^3)$

Consequently, there has been an urgent need to develop **fast**, **approximate** spectral clustering algorithms that are **scalable to large data**.

Addressing the speed scalability (previous work)

Nyström approximation methods: Use a subset of rows and columns of $\mathbf{W}$ to estimate the eigenvectors of the full matrix.

Column sampling methods: Use only a subset of the columns (but keep all rows) to estimate the eigenvectors of $\mathbf{W}$, e.g., cSPEC (Wang et al., PAKDD'09)

Data reduction methods: Reduce input data to a small set of data representatives and perform spectral clustering directly on the reduced set, e.g., KASP (Yan, Huang, and Jordan, KDD'09)

Sparse representation methods: Find a dictionary to sparsely represent the given data, and perform spectral clustering with the sparse feature vectors, e.g., LSC (Cai and Chen, IEEE Trans. Cybernetics, 2015)

Our two-step treatment

1. Scalable spectral clustering with cosine similarity:

   $$\mathbf{W} = \mathbf{X}\mathbf{X}^T - \mathbf{I} \quad (\text{suppose } \mathbf{X} \in \mathbb{R}^{n\times d} \text{ has } L_2\text{-normalized rows})$$

   We assume "some sort of low dimensional structure" for the data matrix $\mathbf{X}$ (e.g., documents, or small images) and show in those cases that spectral clustering can be performed based on "very efficient operations" on $\mathbf{X}$.

2. Scalable spectral clustering with general similarity:

   We present a landmark-based technique to transform this problem to the case of cosine similarity.

Scalable spectral clustering with cosine similarity

We assume

- Data set $\mathbf{X} \in \mathbb{R}^{n \times d}$ with L_2-normalized rows
- **$\mathbf{X}$ is row-sparse ($s \ll d$), or of a moderate dimension ($d \ll n$)**:
  - Documents data
  - Small images (e.g., MNIST handwritten digits: $70,000 \times 784$)
  - Data obtained through PCA
- Cosine similarity is appropriate: $\mathbf{W} = \mathbf{X}\mathbf{X}^T - \mathbf{I}$

We show that:

One can perform the following three kinds of spectral clustering

NJW ($\widetilde{\mathbf{U}}$), NCut ($\mathbf{U} = \mathbf{D}^{-\frac{1}{2}}\widetilde{\mathbf{U}}$), and DM$^{(t)}$ ($\mathbf{U}^{(t)} = \mathbf{D}^{-\frac{1}{2}}\widetilde{\mathbf{U}}\mathbf{\Lambda}^t$),

using only the following **efficient** operations on the data matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$:

- **Elementwise operations**: $\mathcal{O}(ns)$ or $\mathcal{O}(nd)$ complexity
- **Matrix-vector multiplication**: $\mathcal{O}(ns)$ or $\mathcal{O}(nd)$ complexity
- **Rank-k SVD**: $\mathcal{O}(nsk)$ or $\mathcal{O}(ndk)$ complexity

The math

We focus on the NJW algorithm for exposition of ideas.

First, we can calculate the **degree** matrix directly from $\mathbf{X}$:

$$\mathbf{D} = \text{diag}(\mathbf{W1}) = \text{diag}((\mathbf{X}\mathbf{X}^T - \mathbf{I})\mathbf{1}) = \text{diag}(\mathbf{X}(\mathbf{X}^T\mathbf{1}) - \mathbf{1}).$$

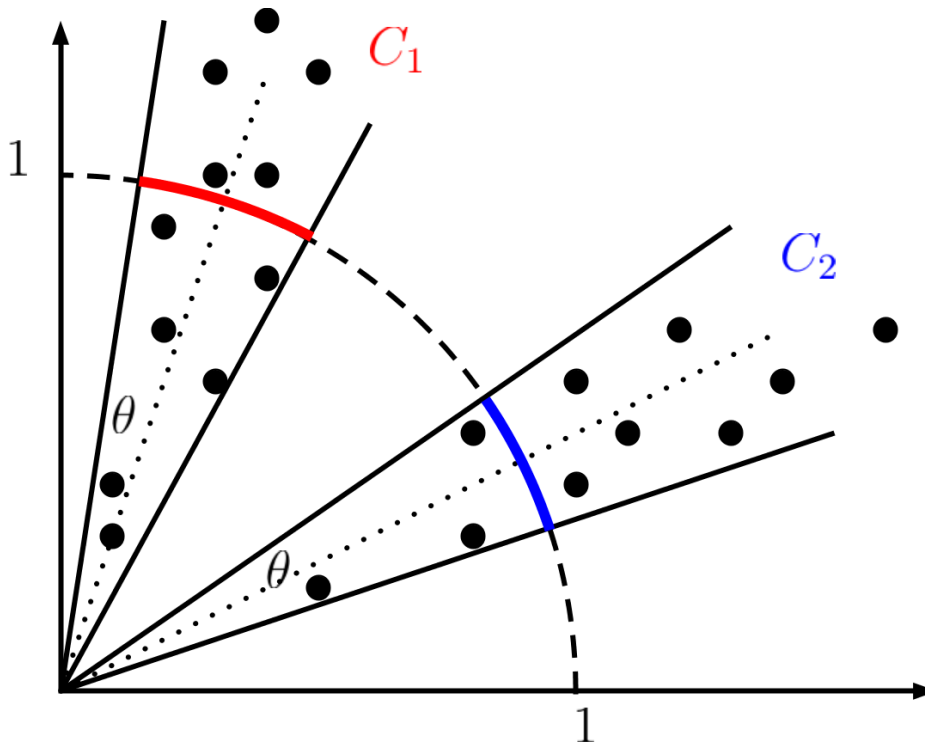
Next, we write

$$\widetilde{\mathbf{W}} = \mathbf{D}^{-\frac{1}{2}}(\underbrace{\mathbf{X}\mathbf{X}^T - \mathbf{I}}_{\mathbf{W}})\mathbf{D}^{-\frac{1}{2}} = (\underbrace{\mathbf{D}^{-\frac{1}{2}}\mathbf{X}}_{:=\widetilde{\mathbf{X}}})(\underbrace{\mathbf{X}^T\mathbf{D}^{-\frac{1}{2}}}_{\widetilde{\mathbf{X}}^T}) - \mathbf{D}^{-1}$$

Note that $\widetilde{\mathbf{X}}$ has the same size $(n \times d)$ and zero entries as $\mathbf{X}$.

We just obtained that $\boxed{\widetilde{\mathbf{W}} = \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T - \mathbf{D}^{-1}}$. Some first observations:

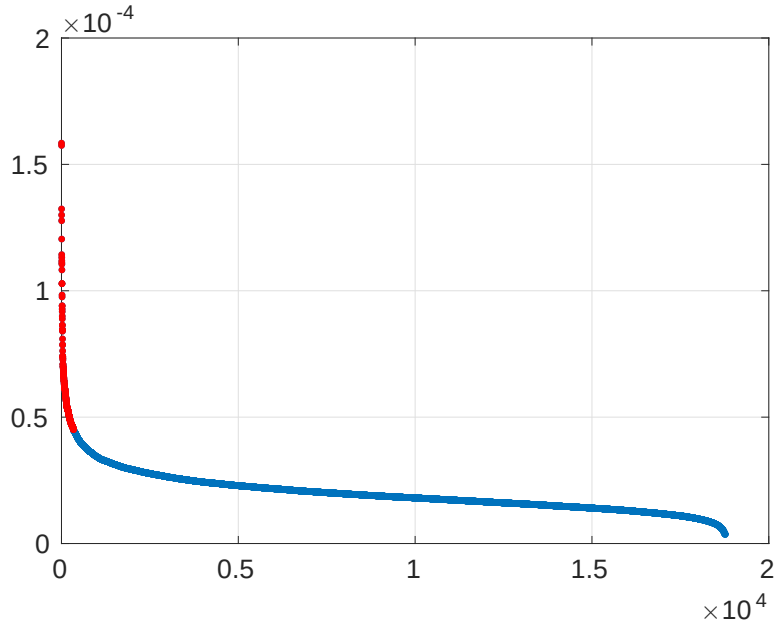
- $\widetilde{\mathbf{W}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$ is similar to a row-stochastic matrix $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$: Therefore, the largest eigenvalue of $\widetilde{\mathbf{W}} = \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T - \mathbf{D}^{-1}$ is 1; the next $k-1$ eigenvalues of $\widetilde{\mathbf{W}}$ are expected to be close to 1. Meanwhile, there should be a sizable drop at the $(k+1)$-th eigenvalue.
- $\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T$ is symmetric and positive semidefinite, and $\mathbf{D}^{-1}$ can be viewed as a perturbation to it.
- If $\mathbf{D}$ has a constant diagonal, then $\widetilde{\mathbf{W}}$ have the same eigenvectors with $\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T$ (but not the same eigenvalues), and discarding $\mathbf{D}^{-1}$ won't change the eigenvectors in such a case. $\longleftarrow$ We will thus try to **equalize the diagonals of $\mathbf{D}$** as much as possible.



If each cluster has $\mathcal{O}(n)$ points and is tight (i.e., within an angle of θ around the axis), then it can be shown that

$$d_i \geq \mathcal{O}(n \cos^2 \theta), \quad 1 \leq i \leq n$$

Sorted diagonals of $\mathbf{D}^{-1}$ computed from a real data set (20newsgroups)



Strategy: Remove a fraction α of points with smallest degrees (which tend to be outliers!), to satisfy the condition $\mathbf{D}^{-1} \approx \frac{1}{\beta}\mathbf{I}$, for some (large) β. This will allow us to approximate the eigenvectors of $\widetilde{\mathbf{W}}$ by the left singular vectors of $\widetilde{\mathbf{X}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{X}$, which are very efficient to compute!

Alg. 1: SSC-cosine

Input:

- Data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ (sparse or of moderate dimension, with L_2-normalized rows)
- Clustering method (NJW, NCut or DM$^{(t)}$)
- #clusters k
- %outliers α

Output: Clusters $C_1, \ldots, C_k$ and a set of outliers C_0

Steps:

1. Compute the degree matrix $\mathbf{D} = \mathrm{diag}(\mathbf{X}(\mathbf{X}^T\mathbf{1}) - \mathbf{1})$ and remove a fraction α of points (with lowest degrees) as outliers.

2. Find $\widetilde{\mathbf{X}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{X}$ and its top k left singular vectors $\widetilde{\mathbf{U}}$ (NJW). Convert $\widetilde{\mathbf{U}}$ to $\mathbf{U} = \mathbf{D}^{-\frac{1}{2}}\widetilde{\mathbf{U}}$ (for NCut) or to $\mathbf{U}^{(t)} = \mathbf{D}^{-\frac{1}{2}}\widetilde{\mathbf{U}}\mathbf{\Lambda}^t$ (for DM$^{(t)}$).

3. Normalize the rows of the singular vectors matrix to have unit length and apply $k$means to find k clusters $C_1, \ldots, C_k$.

Remark. We can easily assign the "outliers" (C_0) to the clusters $C_1, \ldots, C_k$.
$\longleftarrow$ This is a classification problem.

Complexity analysis of Alg. 1

Data	memory	time
Sparse	$\mathcal{O}(ns)$	$\mathcal{O}(nsk + nk^2)$
moderate dim	$\mathcal{O}(nd)$	$\mathcal{O}(ndk + nk^2)$

Applications: document and image clustering

We use the following data:

- **The 20newsgroups data**: 18,774 documents (partitioned into 20 newsgroups), 61,118 words (including stopwords)
- **Handwritten digits data**: USPS $(9,298 \times 256)$, PENDIGITS $(10,992 \times 16)$, MNIST $(70,000 \times 784)$, each having 10 classes

Performance metrics: clustering accuracy and CPU time.

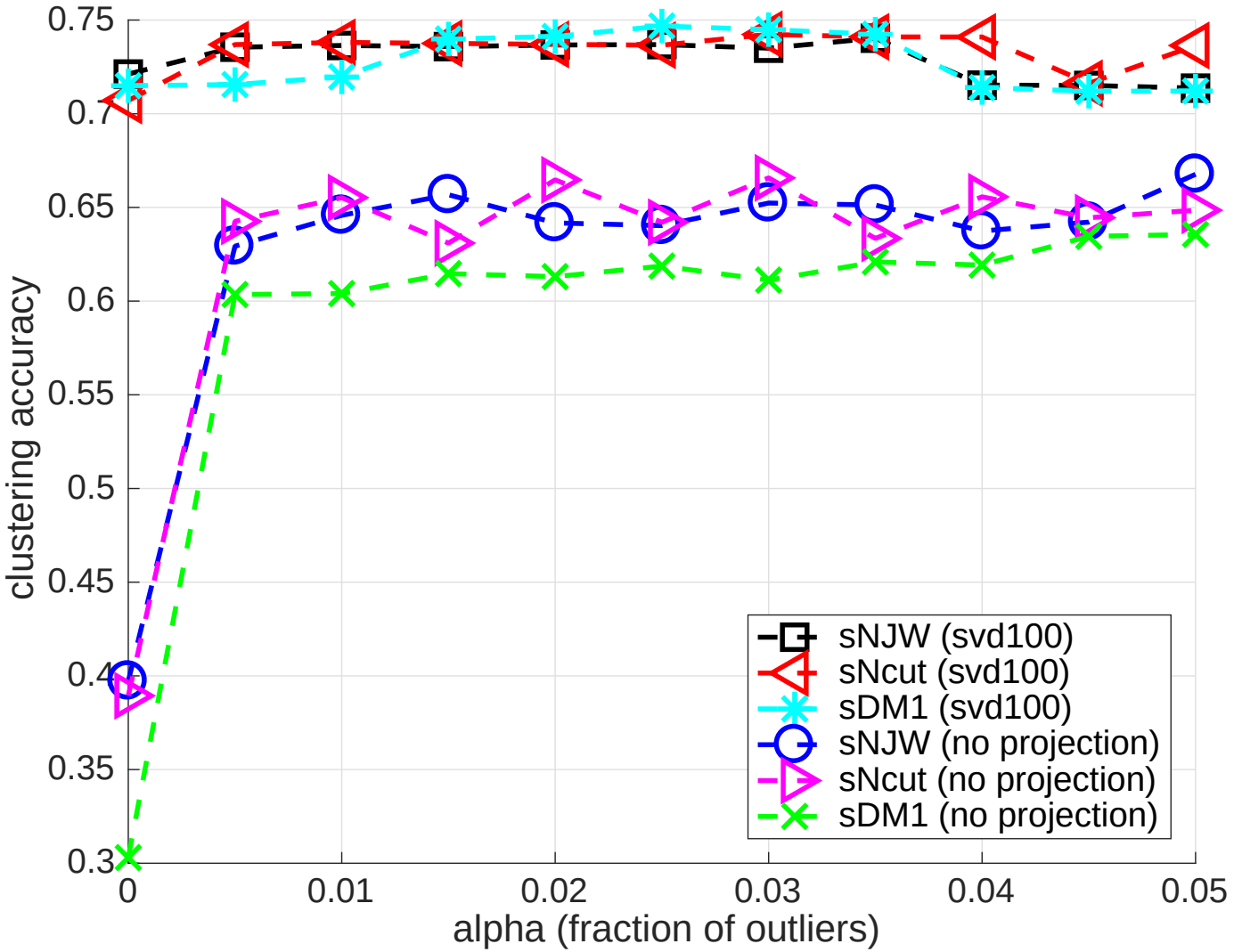
Experiments were conducted in MATLAB 2017a on a compute server with 48 GB of RAM and 2 CPU's with 12 total cores.

The 20newsgroups data: results

Table 1: Clustering accuracy (and CPU time in seconds) on the 20 newsgroups data as well as its SVD projection (into 100 dimensions).

Projection	Method	NJW	NCut	DM$^{(1)}$
None	plain	64.1% (113)	63.8% (108)	60.3% (111)
	scalable	**64.4**% (46.4)	63.1% (43.5)	60.7% (48.8)
SVD (100)	plain	72.7% (128)	72.8% (127)	72.6% (132)
	scalable	73.6% (17.4)	73.8% (17.8)	**74.1**% (18.0)

Parameter setting: $\alpha = 0.01$ (for all algorithms).


0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
clustering accuracy
0
0.01
0.02
0.03
0.04
0.05
alpha (fraction of outliers)
sNJW (svd100)
sNcut (svd100)
sDM1 (svd100)
sNJW (no projection)
sNcut (no projection)
sDM1 (no projection)

The image data: results

Table 2: Clustering accuracy (and CPU time in seconds)

Data	Method	NJW		NCut		$DM^{(1)}$	
usps	plain	67.6%	(23.7)	**67.7%**	(22.3)	57.9%	(24.9)
	scalable	67.4%	(6.4)	**67.7%**	(6.3)	57.7%	(9.2)
pendigits	plain	73.5%	(27.0)	**73.6%**	(30.7)	63.9%	(26.5)
	scalable	**73.6%**	(6.5)	73.3%	(7.6)	63.9%	(4.9)
mnist	plain	(out of memory)					
	scalable	**52.6%**	(55.6)	52.5%	(54.6)	50.4%	(67.9)

Scalable spectral clustering with general similarity

We focus on the Gaussian similarity (but our technique to be presented applies to any kind of similarity):

$$\mathbf{W} = (w_{ij}), \quad w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

In this case, $\mathbf{W}$ is not in product form, so we cannot use the previous procedure directly.

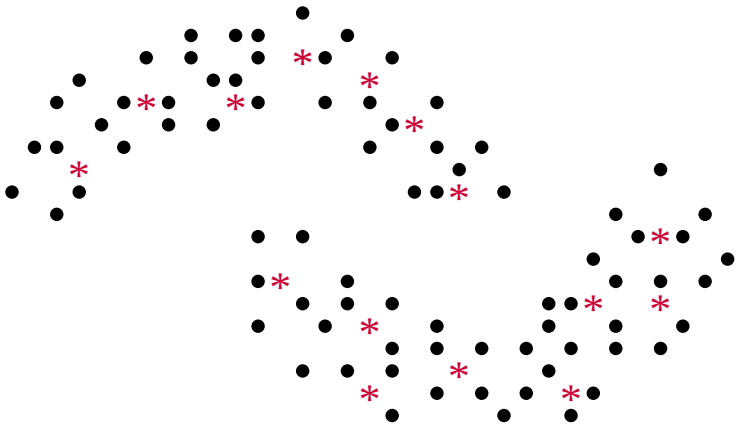
We present a landmark-based technique to convert this problem back to the case of cosine similarity.

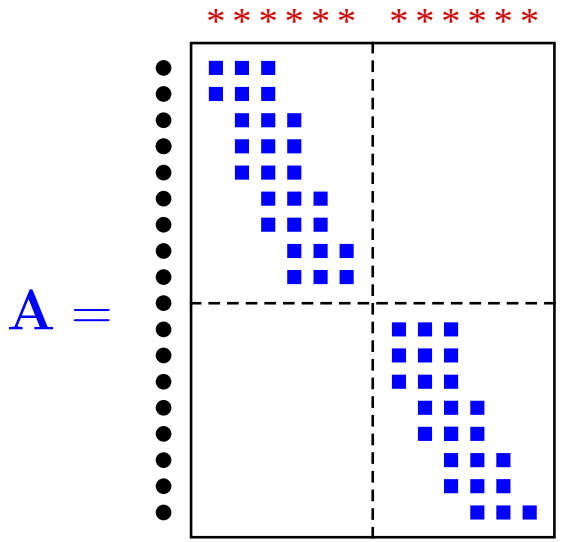
Strategy: Select $\ell \ll n$ landmark points $\mathbf{c}_1, \ldots, \mathbf{c}_\ell \in \mathbb{R}^d$ and compute the similarities between each $\mathbf{x}_i$ and the r closest landmarks $\mathbf{c}_j$, yielding a row-sparse matrix $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{n \times \ell}$ with nonzero entries

$$a_{ij} = \exp\left(-\|\mathbf{x}_i - \mathbf{c}_j\|^2 / 2\sigma^2\right)$$

We then regard $\mathbf{A}$ as a "document-term" matrix in 2 different ways:

- **data documents**
- landmark documents





Next, just apply Alg. 1 (SSC-cosine) with the **rows (data documents)** or columns (landmark documents) of the matrix $\mathbf{A}$ as input data (after IDF normalization):

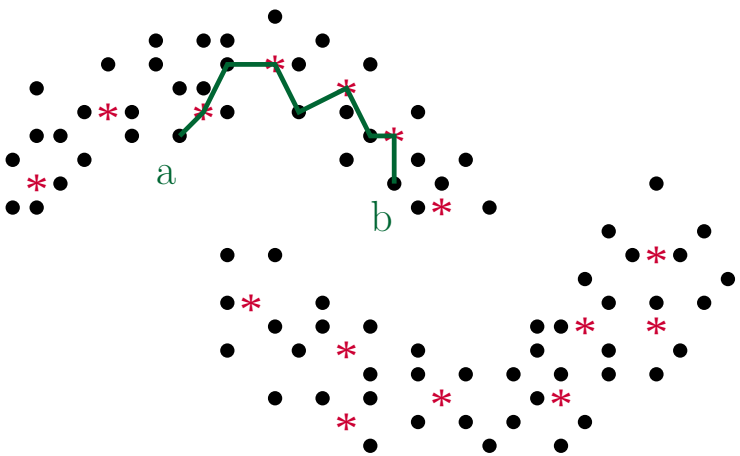
- **Data documents clustering**: ✓
- Landmark documents clustering: Still need to extend the labels to the original data ⟵ This is easily achieved by weighted s-NN classification!

Remark. Second method is much faster (less data for $k$means to cluster, nearest neighbors and weights have already computed).

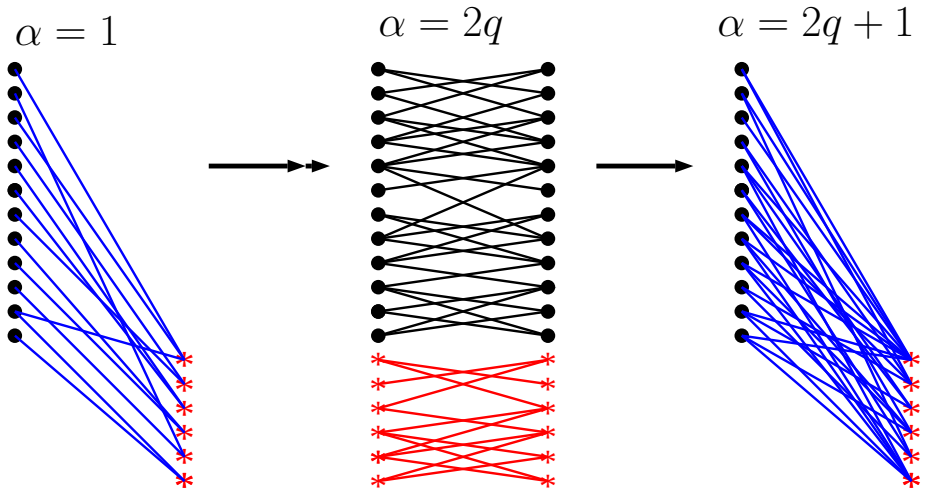
Other relevant work: The bipartite graph model

Besides the documents model just presented for landmark-based clustering with general similarity, we have also developed a bipartite graph model.

(1) We construct a bipartite graph by using **the given data** and a small landmark set as its two parts.



(2) We run a **diffusion process** on the bipartite graph to gather global information about the data set.



Addressing the memory scalability

We consider first the special case of cosine similarity: $\mathbf{W} = \mathbf{X}\mathbf{X}^T - \mathbf{I}$, where $\mathbf{X} \in \mathbb{R}^{n\times d}$ represents an extremely large data set that cannot be fully loaded to computer memory (thus, the scalable algorithm introduced earlier cannot be applied).

We suppose we are able to access a small random sample $\mathbf{X}_s \in \mathbb{R}^{s\times d}$ of size $s \ll n$. We will use the sample to estimate the top k singular values $\widetilde{\mathbf{\Sigma}}$ and right singular vectors $\widetilde{\mathbf{V}}$ of $\widetilde{\mathbf{X}} = \mathbf{D}^{-1/2}\mathbf{X}$. Note that $\widetilde{\mathbf{U}} = \widetilde{\mathbf{X}}\widetilde{\mathbf{V}}\widetilde{\mathbf{\Sigma}}^{-1}$.

We then embed the sample as $\mathbf{X}_s \longmapsto \widetilde{\mathbf{X}}_s\widetilde{\mathbf{V}}\widetilde{\mathbf{\Sigma}}^{-1}$, to group $\mathbf{X}_s$ into k clusters and extend clustering to the rest of the data as they come in.

Derivation

Since the right singular vectors of $\widetilde{\mathbf{X}}$ are the eigenvectors of $\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}}$, we obtain an expression for this product matrix as follows:

$$\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}} = (\mathbf{D}^{-1/2}\mathbf{X})^T(\mathbf{D}^{-1/2}\mathbf{X}) = \mathbf{X}^T\mathbf{D}^{-1}\mathbf{X} = \sum_{i=1}^{n}\frac{1}{d_i}\mathbf{x}_i\mathbf{x}_i^T$$

Let the restrictions of $\mathbf{D}$ and $\widetilde{\mathbf{X}}$ to the sample $\mathbf{X}_s$ be

$$\mathbf{D}_s = \mathbf{D}(1:s, 1:s), \quad \widetilde{\mathbf{X}}_s = \mathbf{D}_s^{-\frac{1}{2}}\mathbf{X}_s$$

Then

$$\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}} \approx \frac{n}{s}\sum_{i=1}^{s}\frac{1}{d_i}\mathbf{x}_i\mathbf{x}_i^T = \frac{n}{s}\mathbf{X}_s\mathbf{D}_s^{-1}\mathbf{X}_s = \frac{n}{s}\widetilde{\mathbf{X}}_s^T\widetilde{\mathbf{X}}_s \quad \longrightarrow \widetilde{\mathbf{V}}, \widetilde{\mathbf{\Sigma}}$$

It remains to estimate $\mathbf{D}_s$ (which depends on the full data set) in order to compute $\widetilde{\mathbf{X}}_s = \mathbf{D}_s^{-\frac{1}{2}} \mathbf{X}_s$.

The exact degrees of all the data points are

$$\mathbf{d} = \underbrace{(\mathbf{X}\mathbf{X}^T - \mathbf{I})}_{\mathbf{W}} \mathbf{1}_n = \mathbf{X}\mathbf{X}^T \mathbf{1}_n - \mathbf{1}_n = \mathbf{X} \cdot \sum_{i=1}^{n} \mathbf{x}_i - \mathbf{1}_n$$

The degrees of the points in the sample are

$$\mathbf{d}_s = \mathbf{X}_s \cdot \sum_{i=1}^{n} \mathbf{x}_i - \mathbf{1}_s \approx \mathbf{X}_s \cdot \frac{n}{s} \sum_{i=1}^{s} \mathbf{x}_i - \mathbf{1}_s = \frac{n}{s} \mathbf{X}_s (\mathbf{X}_s^T \mathbf{1}_s) - \mathbf{1}_s.$$

We summarize the procedure here: Given the random sample $\mathbf{X}_s$, we first estimate the degrees of the points as follows

$$\mathbf{D}_s \approx \operatorname{diag}\left(\frac{n}{s}\mathbf{X}_s\left(\mathbf{X}_s^T\mathbf{1}_s\right) - \mathbf{1}_s\right)$$

and then use it to normalize $\mathbf{X}_s$ to obtain that

$$\widetilde{\mathbf{X}}_s = \mathbf{D}_s^{-1/2}\mathbf{X}_s$$

Next, perform rank-k SVD on $\widetilde{\mathbf{X}}_s$ to get $\widetilde{\mathbf{U}}_s, \widetilde{\mathbf{\Sigma}}_s$, and $\widetilde{\mathbf{V}}_s^T$. It follows that

$$\widetilde{\mathbf{V}} \approx \widetilde{\mathbf{V}}_s, \quad \widetilde{\mathbf{\Sigma}} \approx \sqrt{\frac{n}{s}}\widetilde{\mathbf{\Sigma}}_s.$$

Finally, perform k-means clustering on the rows of $\widetilde{\mathbf{X}}_s\widetilde{\mathbf{V}}\widetilde{\mathbf{\Sigma}}^{-1}$.

To explain how to extend clustering to the rest of the data (as they are gradually loaded into computer memory), we consider an arbitrary new point $\mathbf{x}_{s+1} \in \mathbb{R}^d$. Its degree is estimated similarly to the sample:

$$d_{s+1} = \mathbf{x}_{s+1}^T \cdot \sum_{i=1}^{n} \mathbf{x}_i - 1 \approx \mathbf{x}_{s+1}^T \cdot \frac{n}{s} \sum_{i=1}^{s} \mathbf{x}_i - 1 = \frac{n}{s} \mathbf{x}_{s+1}^T (\mathbf{X}_s^T \mathbf{1}_s) - 1.$$

Following the embedding of the initial sample, i.e.,

$$\mathbf{X}_s \quad \longmapsto \quad \widetilde{\mathbf{X}}_s \widetilde{\mathbf{V}} \widetilde{\mathbf{\Sigma}}^{-1},$$

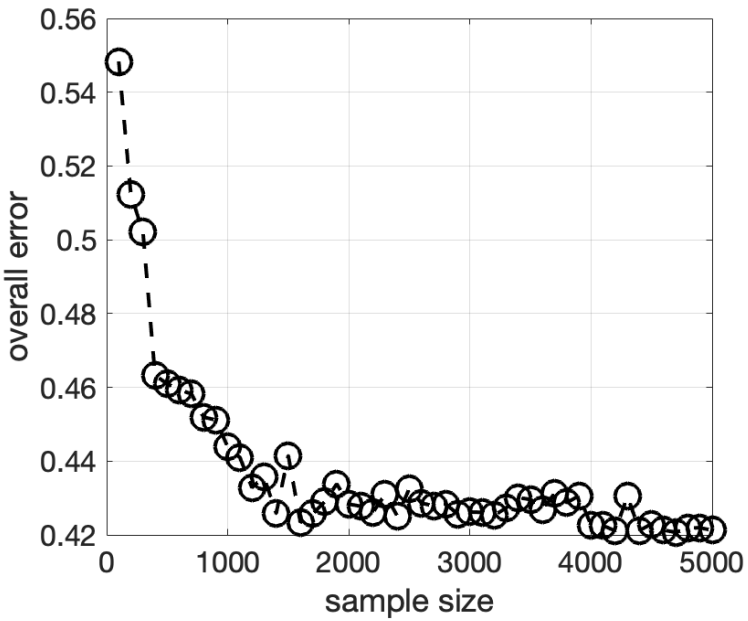
we embed $\mathbf{x}_{s+1}$ into the $\widetilde{\mathbf{U}}_s$ space as follows:

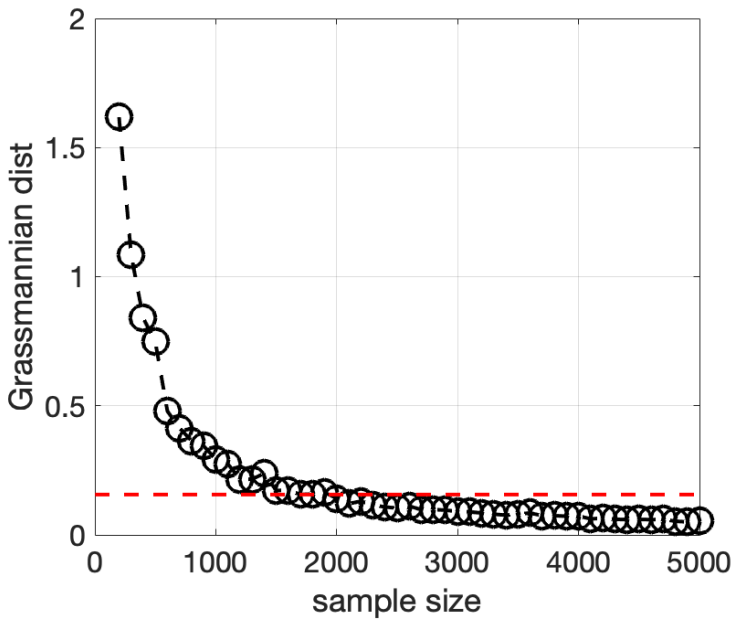
$$\mathbf{x}_{s+1} \quad \longmapsto \quad \left(d_{s+1}^{-1/2} \mathbf{x}_{s+1}^T \right) \widetilde{\mathbf{V}} \widetilde{\mathbf{\Sigma}}^{-1}$$

and then assign it to the cluster with the closest centroid.

How to choose the sample size s

Technique: Incrementally sample more and more data ($s_1 < s_2 < \cdots$) and monitor the convergence of $\{\widetilde{\mathbf{V}}_{s_i}\}_{i\geq 1}$ under the Grassmaniann distance, $\|\widetilde{\mathbf{V}}_{s_{i+1}}\widetilde{\mathbf{V}}_{s_{i+1}}^T - \widetilde{\mathbf{V}}_{s_i}\widetilde{\mathbf{V}}_{s_i}^T\|_F$ (see results below on MNIST)





Thank you for your attention!

Contact: cheng@hope.edu, guangliang.chen@sjsu.edu

My papers:

1. "Memory-efficient spectral clustering with cosine similarity through batch-based learning", R. Li and G. Chen (submitted)
2. "A general framework for scalable spectral clustering based on document models", G. Chen (Pattern Recognition Letters, June 2019)
3. "Scalable spectral clustering with cosine similarity", G. Chen (ICPR 2018)
4. "Large-scale spectral clustering using diffusion coordinates on landmark based bipartite graphs", K. Pham and G. Chen (TextGraphs 2018)

Questions?

Appendix: Advanced linear algebra

Symmetric matrices

A square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is said to be **symmetric** if it coincides with its transpose, i.e., $\mathbf{A}^T = \mathbf{A}$.

We denote the set of symmetric matrices of size $n \times n$ by

$$S^n(\mathbb{R}) = \{\mathbf{A} \in \mathbb{R}^{n \times n} \mid \mathbf{A}^T = \mathbf{A}\}$$

Symmetric matrices have many nice properties. For example,

- all their eigenvalues are real
- eigenvectors corresponding to distinct eigenvalues must be orthogonal to each other.

Furthermore, symmetric matrices are *orthogonally diagonalizable*, i.e., diagonalizable via orthogonal matrices.

Theorem 0.6. Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a symmetric matrix. Then there exist a diagonal matrix $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^{n\times n}$ and an orthogonal matrix $\mathbf{Q} = [\mathbf{q}_1 \ldots \mathbf{q}_n] \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T = \sum_{i=1}^{n} \lambda_i \mathbf{q}_i \mathbf{q}_i^T$$

Remark. The above factorization represents the **eigendecomposition** or **spectral decomposition** of $\mathbf{A}$: The λ_i's represent the eigenvalues of $\mathbf{A}$ while the $\mathbf{q}_i$'s are the associated eigenvectors (with unit norm and orthogonal to each other).

Positive (semi)definite matrices

A *symmetric* matrix $\mathbf{A} \in S^n(\mathbb{R})$ is said to be **positive semidefinite (PSD)** if $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$.

For a PSD matrix $\mathbf{A} \in S^n(\mathbb{R})$, if the equality holds true only for $\mathbf{x} = \mathbf{0}$ (i.e., $\mathbf{x}^T\mathbf{A}\mathbf{x} > 0$ for all $\mathbf{x} \neq \mathbf{0}$), then $\mathbf{A}$ is further said to be **positive definite (PD)**.

We denote by $S^n_{0+}(\mathbb{R})$ and $S^n_{+}(\mathbb{R})$ the sets of positive semidefinite and of positive definite matrices of size $n \times n$, respectively.

Note that we must have

$$S^n_{+}(\mathbb{R}) \subset S^n_{0+}(\mathbb{R}) \subset S^n(\mathbb{R}) \subset \mathbb{R}^{n\times n}.$$

Spectral decomposition of PSD matrices (in reduced form)

For a PSD matrix $\mathbf{A} \in S_{0+}^n(\mathbb{R})$,

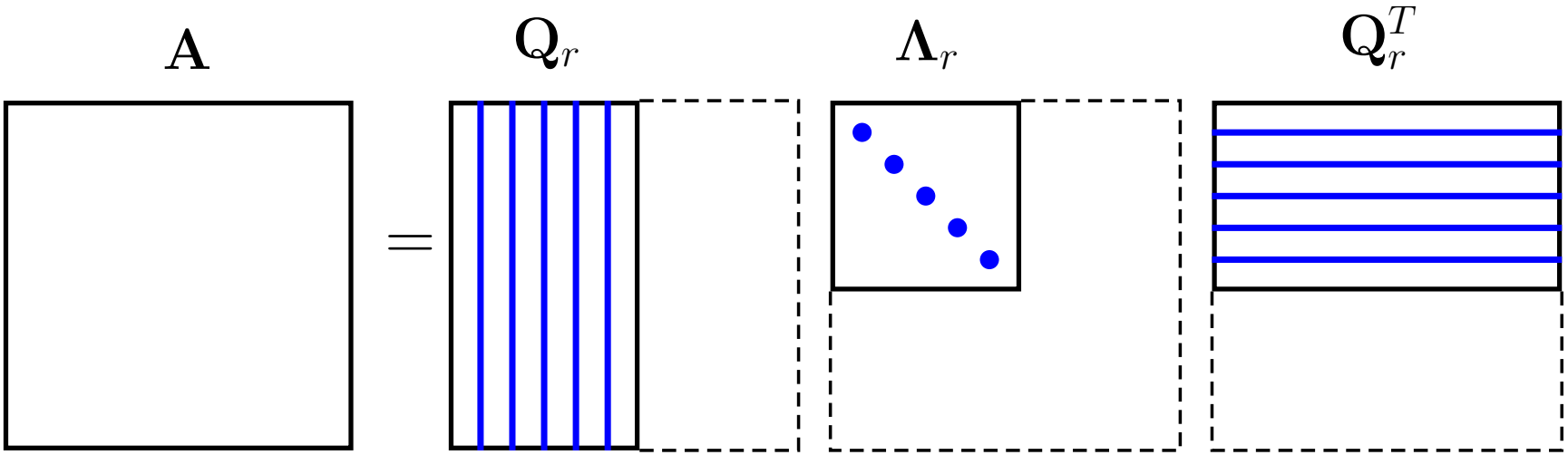
$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > \lambda_{r+1} = \cdots = \lambda_n = 0, \quad r = \operatorname{rank}(\mathbf{A}).$$

Correspondingly, we may obtain

$$\mathbf{A} = \sum_{i=1}^{r} \lambda_i \mathbf{q}_i \mathbf{q}_i^T = \begin{bmatrix} \mathbf{q}_1 & \dots & \mathbf{q}_r \end{bmatrix} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_r \end{pmatrix} \begin{bmatrix} \mathbf{q}_1^T \\ \vdots \\ \mathbf{q}_r \end{bmatrix} = \mathbf{Q}_r \mathbf{\Lambda}_r \mathbf{Q}_r^T$$

where $\mathbf{Q}_r = [\mathbf{q}_1 \ \dots \ \mathbf{q}_r] \in \mathbb{R}^{n \times r}$ is a tall matrix with orthonormal columns, and $\mathbf{\Lambda}_r = \operatorname{diag}(\lambda_1, \dots, \lambda_r) \in \mathbb{R}^{r \times r}$.

Illustration of the reduced form:



The reduced form is more efficient than the full decomposition!

Matrix square roots (for PSD matrices)

An interesting aspect of positive semidefinite matrices is that they have square roots (which are also matrices), just like nonnegative numbers have square roots (which are still numbers).

Def 0.2. Let $\mathbf{A} \in S_{0+}^n(\mathbb{R})$. The **square root** of $\mathbf{A}$ is defined as the matrix $\mathbf{R} \in S_{0+}^n(\mathbb{R})$ such that $\mathbf{R}^2 = \mathbf{A}$. We denote it by $\mathbf{R} = \mathbf{A}^{1/2}$.

Remark. Note that if $\mathbf{A} \in S_{+}^n(\mathbb{R})$, then $\mathbf{R} = \mathbf{A}^{1/2} \in S_{+}^n(\mathbb{R})$. In such a case, we can further define the **reciprocal square root** of $\mathbf{A}$ as

$$\mathbf{A}^{-1/2} = (\mathbf{A}^{1/2})^{-1} \in S_{+}^n(\mathbb{R}).$$

Special case: If $\mathbf{A} \in S_{0+}^n(\mathbb{R})$ happens to be diagonal, i.e.,

$$\mathbf{A} = \operatorname{diag}(a_1, \ldots, a_n), \qquad \text{where} \quad a_1, \ldots, a_n \geq 0,$$

then there is an easy way to find its square root. Define

$$\mathbf{R} = \operatorname{diag}\left(a_1^{1/2}, \ldots, a_n^{1/2}\right) \in S_{0+}^n(\mathbb{R}).$$

Clearly, $\mathbf{R}^2 = \mathbf{A}$. This shows that $\mathbf{R}$ is indeed a square root of $\mathbf{\Lambda}$.

Remark. Without the positive semidefiniteness requirement in the definition of matrix square roots, $\mathbf{R}$ won't be unique as we can arbitrarily modify the signs of the diagonals $a_i^{1/2}$ without violating the equality condition.

Theorem 0.7. Let $\mathbf{A} \in S_{0+}^n(\mathbb{R})$ with spectral decomposition $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, where $\mathbf{Q} \in \mathbb{R}^{n\times n}$ is an orthogonal matrix and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ with $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$. Then $\mathbf{A}$ has a unique matrix square root

$$\mathbf{R} = \mathbf{Q}\mathbf{\Lambda}^{1/2}\mathbf{Q}^T.$$

Proof. First, such defined matrix $\mathbf{R}$ is PSD. By direct calculation,

$$\mathbf{R}^2 = (\mathbf{Q}\mathbf{\Lambda}^{1/2}\mathbf{Q}^T)(\mathbf{Q}\mathbf{\Lambda}^{1/2}\mathbf{Q}^T) = \mathbf{Q}\underbrace{\mathbf{\Lambda}^{1/2}\mathbf{\Lambda}^{1/2}}_{\mathbf{\Lambda}}\mathbf{Q}^T = \mathbf{A}.$$

We omit the proof of the uniqueness part.

Remark. For any $\mathbf{A} \in S_+^n(\mathbb{R})$ with eigendecomposition $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$,

$$\mathbf{A}^{-1/2} = \mathbf{Q}\mathbf{\Lambda}^{-1/2}\mathbf{Q}^T, \quad \mathbf{\Lambda}^{-1/2} = \mathrm{diag}\left(\lambda_1^{-1/2}, \ldots, \lambda_n^{-1/2}\right).$$

The generalized eigenvalue problem

Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ be two square matrices of the same size. We say that $\lambda \in \mathbb{R}$ is a **generalized eigenvalue** of $(\mathbf{A}, \mathbf{B})$ if there exists a nonzero vector $\mathbf{v} \in \mathbb{R}^n$ such that

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{B}\mathbf{v}.$$

The vector $\mathbf{v}$ is called a **generalized eigenvector** of $(\mathbf{A}, \mathbf{B})$ corresponding to λ.

Remark. In the above definition, if we let $\mathbf{B} = \mathbf{I}$, then the generalized eigenvalues of $(\mathbf{A}, \mathbf{B})$ would reduce to the ordinary eigenvalues of $\mathbf{A}$:

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{v}.$$

Now, let us rewrite the definition of generalized eigenvalues as

$$(\mathbf{A} - \lambda \mathbf{B})\mathbf{v} = 0.$$

Note that there exists a nonzero solution $\mathbf{v}$ if and only if $\mathbf{A} - \lambda \mathbf{B}$ is singular. Thus, λ is a generalized eigenvalue of $(\mathbf{A}, \mathbf{B})$ if and only if

$$\det(\mathbf{A} - \lambda \mathbf{B}) = 0.$$

Let $p_{\mathbf{A},\mathbf{B}}(\lambda) = \det(\mathbf{A} - \lambda \mathbf{B})$, the **characteristic polynomial** of $(\mathbf{A}, \mathbf{B})$.

Interestingly, $p_{\mathbf{A},\mathbf{B}}(\lambda)$ is also a polynomial in λ, but it can have an arbitrary order between 0 and n, as we show next.

Example 0.6. Let

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

To find the generalized eigenvalues of $(\mathbf{A}, \mathbf{B})$, compute

$$\det(\mathbf{A} - \lambda\mathbf{B}) = \begin{vmatrix} 1-\lambda & 2-\lambda \\ 2-\lambda & 4-\lambda \end{vmatrix} = (1-\lambda)(4-\lambda) - (2-\lambda)^2 = -\lambda.$$

Thus, $(\mathbf{A}, \mathbf{B})$ has only one generalized eigenvalue of $\lambda = 0$, with corresponding generalized eigenvectors

$$\mathbf{0} = (\mathbf{A} - 0 \cdot \mathbf{B})\mathbf{v} = \mathbf{A}\mathbf{v} \qquad \longrightarrow \qquad \mathbf{v} = k \begin{pmatrix} -2 \\ 1 \end{pmatrix}, \ k \in \mathbb{R}.$$

Example 0.7. Let

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 1 & 2 \\ 3 & 6 \end{pmatrix}.$$

To find the generalized eigenvalues of $(\mathbf{A}, \mathbf{B})$, compute

$$\det(\mathbf{A}-\lambda\mathbf{B}) = \begin{vmatrix} 1-\lambda & 2-2\lambda \\ 2-3\lambda & 4-6\lambda \end{vmatrix} = (1-\lambda)(4-6\lambda)-(2-2\lambda)(2-3\lambda) = 0.$$

Thus, any scalar λ is a generalized eigenvalue of $(\mathbf{A}, \mathbf{B})$. This pair of matrices has infinitely many generalized eigenvalues!

Generalized symmetric-definite eigenvalue problems

Let $\mathbf{A} \in S^n(\mathbb{R})$ and $\mathbf{B} \in S^n_+(\mathbb{R})$. The generalized eigenvalue problem

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{B}\mathbf{v}$$

is called a **generalized symmetric-definite eigenvalue problem**. Such problems have a lot of applications.

Remark. The generalized eigenpairs of $(\mathbf{A}, \mathbf{B})$ for $\mathbf{A} \in S^n(\mathbb{R}), \mathbf{B} \in S^n_+(\mathbb{R})$ are actually the eigenpairs of $\mathbf{B}^{-1}\mathbf{A}$:

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{B}\mathbf{v} \quad \Longleftrightarrow \quad \left(\mathbf{B}^{-1}\mathbf{A}\right)\mathbf{v} = \lambda \mathbf{v}$$

However, $\mathbf{B}^{-1}\mathbf{A}$ is not symmetric in general.

The following theorem indicates that generalized symmetric-definite eigenvalue problems are predictable in terms of the number of (real) generalized eigenvalues.

Theorem 0.8. Let $\mathbf{A} \in S^n(\mathbb{R})$ and $\mathbf{B} \in S^n_+(\mathbb{R})$. The generalized symmetric-definite generalized eigenvalue problem $\mathbf{Av} = \lambda\mathbf{Bv}$ has n generalized eigenvalues $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$ with linearly independent generalized eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{R}^n$ satisfying

$$\mathbf{v}_i^T \mathbf{B} \mathbf{v}_j = \delta_{ij}, \quad \text{for all } 1 \leq i \neq j \leq n.$$

Proof of the theorem. Since $\mathbf{B} \in S_+^n(\mathbb{R})$, we can rewrite

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{B}\mathbf{v} \quad \Longrightarrow \quad \mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2} \cdot \mathbf{B}^{1/2}\mathbf{v} = \lambda \cdot \mathbf{B}^{1/2}\mathbf{v}$$

Letting

$$\widetilde{\mathbf{A}} = \mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2}, \qquad \text{and} \quad \widetilde{\mathbf{v}} = \mathbf{B}^{1/2}\mathbf{v}$$

we further obtain that

$$\widetilde{\mathbf{A}}\widetilde{\mathbf{v}} = \lambda \widetilde{\mathbf{v}}$$

Since $\widetilde{\mathbf{A}} \in S^n(\mathbb{R})$, there are n eigenpairs $(\lambda_i, \widetilde{\mathbf{v}}_i), 1 \leq i \leq n$, with

$$\delta_{ij} = \widetilde{\mathbf{v}}_i^T \widetilde{\mathbf{v}}_j = \left(\mathbf{B}^{1/2}\mathbf{v}_i\right)^T \mathbf{B}^{1/2}\mathbf{v}_j = \mathbf{v}_i^T \mathbf{B}\mathbf{v}_j.$$

Consequently, $(\mathbf{A}, \mathbf{B})$ has n generalized eigenvalues λ_i with associated generalized eigenvectors $\mathbf{v}_i = \mathbf{B}^{-1/2}\widetilde{\mathbf{v}}_i$.

The ordinary Rayleigh quotients

Rayleigh quotients are encountered in many statistical and machine learning problems. It is thus necessary to study it systematically.

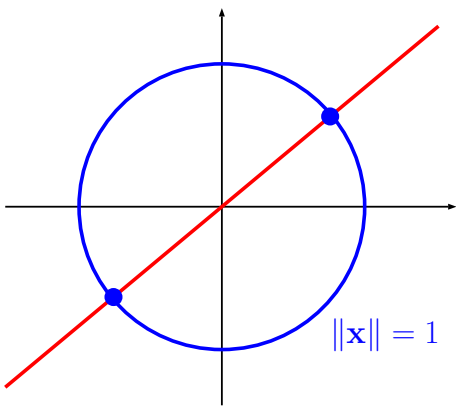
Def 0.3. The Rayleigh quotient for a given symmetric matrix $\mathbf{A} \in S^n(\mathbb{R})$ is a multivariate function $f : \mathbb{R}^n - \{\mathbf{0}\} \longmapsto \mathbb{R}$ defined by

$$f(\mathbf{x}) = \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}}, \quad \mathbf{x} \neq \mathbf{0}.$$

Remark. Rayleigh quotients are always scaling invariant. That is, given any $\mathbf{A} \in S^n(\mathbb{R})$ and $\mathbf{x} \neq \mathbf{0} \in \mathbb{R}^n$,

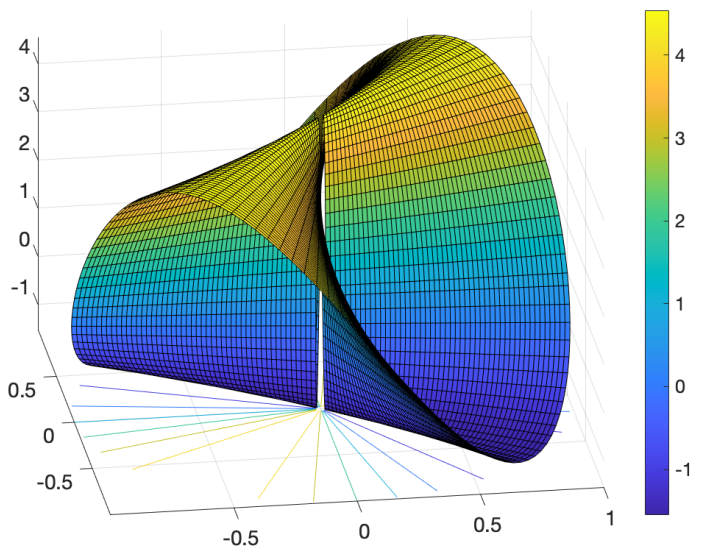
$$f(k\mathbf{x}) = \frac{(k\mathbf{x})^T \mathbf{A}(k\mathbf{x})}{(k\mathbf{x})^T(k\mathbf{x})} = \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = f(\mathbf{x}), \qquad \text{for all } k \neq 0$$

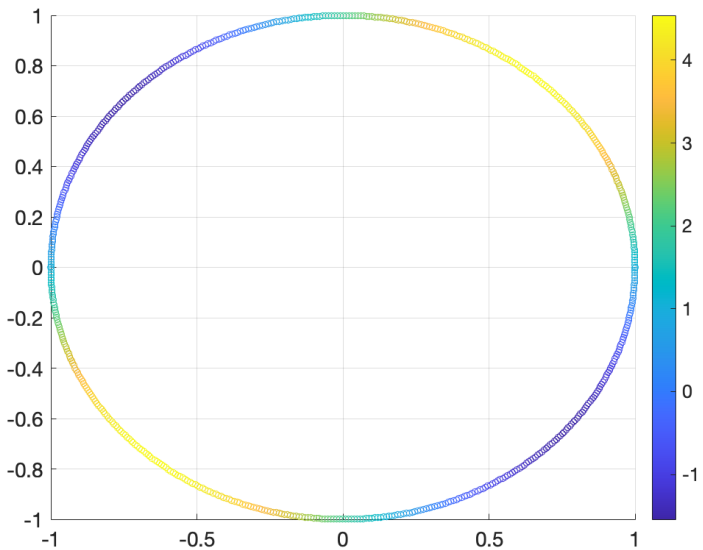
In particular, $f(-\mathbf{x}) = f(\mathbf{x})$, which indicates that the graph of f is always symmetric about the origin.



Example 0.8. The Rayleigh quotient for $\mathbf{A} = \begin{pmatrix} 1 & 3 \\ 3 & 2 \end{pmatrix} \in S^2(\mathbb{R})$ is

$$f(\mathbf{x}) = \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} = \frac{x_1^2 + 2x_2^2 + 6x_1 x_2}{x_1^2 + x_2^2}, \quad \mathbf{x} \neq \mathbf{0}.$$

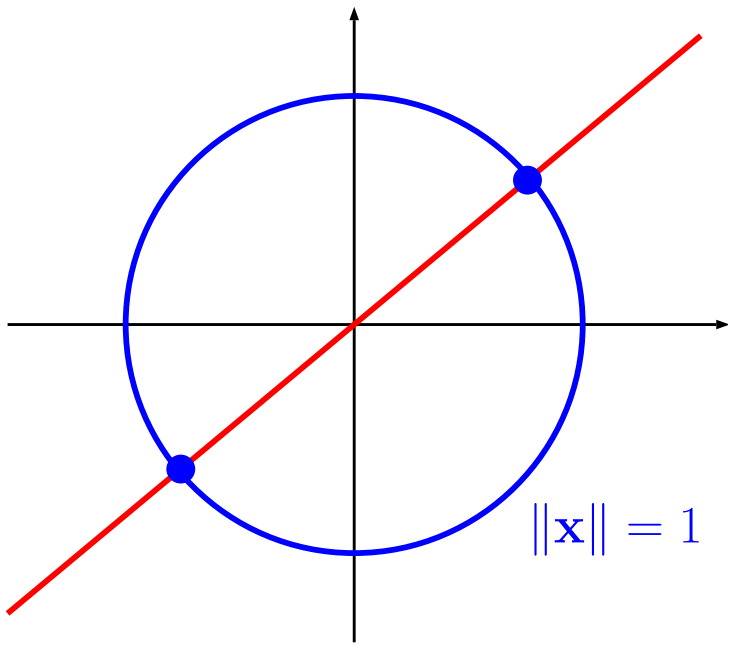




Optimization of Rayleigh quotients

Problem. Given $\mathbf{A} \in S^n(\mathbb{R})$, find the maximum (or minimum) of the associated Rayleigh quotient

$$\max_{\mathbf{x} \neq \mathbf{0} \in \mathbb{R}^n} \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \leftarrow \text{scaling invariant}$$



Equivalent formulations:

$$\max_{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| = 1} \mathbf{x}^T \mathbf{A} \mathbf{x}$$

$$\max_{\mathbf{x} \in \mathbb{R}^n} \mathbf{x}^T \mathbf{A} \mathbf{x} \quad \text{subject to } \|\mathbf{x}\|^2 = 1 \quad \longleftarrow \text{ Quadratic form over unit circle}$$

Theorem 0.9. For any given symmetric matrix $\mathbf{A} \in S^n(\mathbb{R})$, let its largest and smallest eigenvalues be λ_1 and λ_n, with associated eigenvectors $\mathbf{v}_1, \mathbf{v}_n \in \mathbb{R}^n$, respectively. Then

$$\max_{\mathbf{x}\in\mathbb{R}^n:\,\mathbf{x}\neq\mathbf{0}} \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \lambda_1, \quad @\ \mathbf{x} = k\mathbf{v}_1,\ \forall k \neq 0$$

$$\min_{\mathbf{x}\in\mathbb{R}^n:\,\mathbf{x}\neq\mathbf{0}} \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \lambda_n, \quad @\ \mathbf{x} = k\mathbf{v}_n,\ \forall k \neq 0$$

Remark. We often focus on the unit-norm eigenvectors as maximizer and minimizers.

Remark. This theorem can be proved in two different ways: (1) linear algebra approach (2) method of Lagrange multipliers.

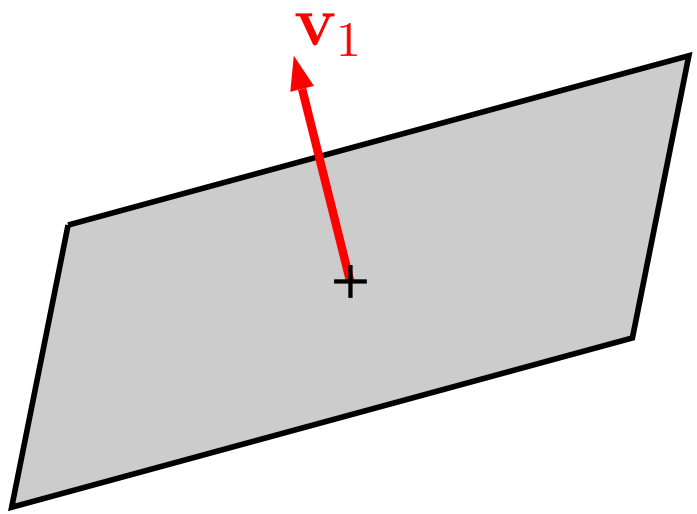
Restricted Rayleigh quotients

Sometimes, we may choose to "exclude" the top (bottom) few eigenvectors from the optimization domain when maximizing (minimizing) a Rayleigh quotient:

$$\max_{\substack{\mathbf{x}\neq\mathbf{0}\\ \mathbf{v}_1^T\mathbf{x}=0}} \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}}$$

$$\max_{\substack{\mathbf{x}\neq\mathbf{0}\\ \mathbf{v}_1^T\mathbf{x}=\mathbf{v}_2^T\mathbf{x}=0}} \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}}$$

In such cases, the effective domain is the orthogonal complement of the excluded eigenvector(s).



Theorem 0.10 (Rayleigh-Ritz). Given a symmetric matrix $\mathbf{A} \in S^n(\mathbb{R})$, let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ be its eigenvalues and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n \in \mathbb{R}^n$ a collection of corresponding eigenvectors (in unit norm). We have

$$\max_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{v}_1^T\mathbf{x}=0}} \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \lambda_2 \quad (\text{when } \mathbf{x} = \pm\mathbf{v}_2)$$

$$\max_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{v}_1^T\mathbf{x}=\mathbf{v}_2^T\mathbf{x}=0}} \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \lambda_3 \quad (\text{when } \mathbf{x} = \pm\mathbf{v}_3)$$

so on and so forth.

The generalized Rayleigh quotients

Def 0.4. For a fixed symmetric matrix $\mathbf{A} \in S^n(\mathbb{R})$ and a positive definite matrix $\mathbf{B} \in S^n_+(\mathbb{R})$ of the same size, a **generalized Rayleigh quotient** corresponding to them is a function $f : \mathbb{R}^n - \{\mathbf{0}\} \longmapsto \mathbb{R}$ defined by

$$f(\mathbf{x}) = \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{B} \mathbf{x}}.$$

Note that if $\mathbf{B} = \mathbf{I}$, then the generalized Rayleigh quotient reduces to an ordinary Rayleigh quotient.

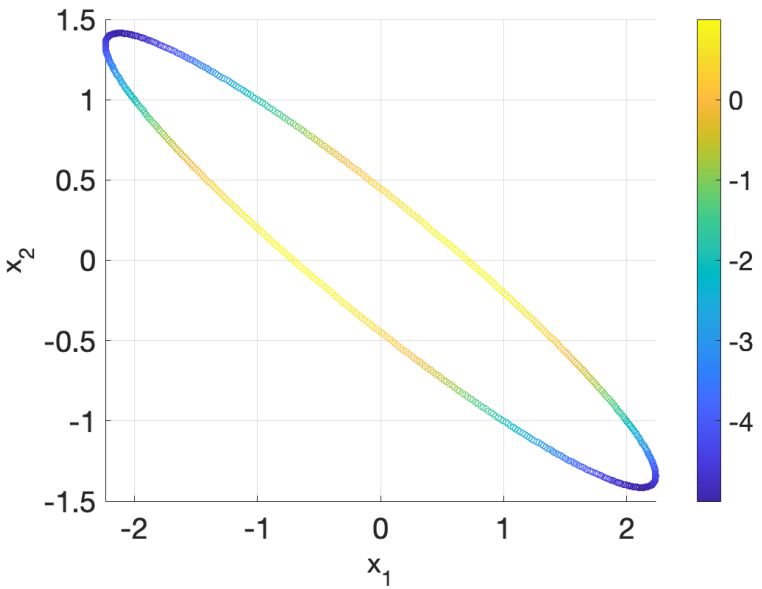
This is also a function defined over $\mathbb{R}^n$ with the origin excluded, having the same scaling invariant property as ordinary Rayleigh quotients: Given $\mathbf{A} \in S^n(\mathbb{R}), \mathbf{B} \in S^n_+(\mathbb{R})$ and $\mathbf{x} \neq \mathbf{0}$, we have

$$f(k\mathbf{x}) = \frac{(k\mathbf{x})^T \mathbf{A}(k\mathbf{x})}{(k\mathbf{x})^T \mathbf{B}(k\mathbf{x})} = \frac{\mathbf{x}^T \mathbf{A}\mathbf{x}}{\mathbf{x}^T \mathbf{B}\mathbf{x}} = f(\mathbf{x}), \qquad \text{for all } k \neq 0.$$

Similarly, the graph of f is also symmetric about the origin because $f(-\mathbf{x}) = f(\mathbf{x})$ for all $\mathbf{x} \neq \mathbf{0}$.

Example 0.9. Given $\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 3 & 2 \end{pmatrix} \in S^2(\mathbb{R})$ and $\mathbf{B} = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix} \in S^2_+(\mathbb{R})$, we have the following generalized Rayleigh quotient:

$$f(\mathbf{x}) = \frac{\mathbf{x}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{B}\mathbf{x}} = \frac{2x_1^2 + 2x_2^2 + 6x_1x_2}{2x_1^2 + 5x_2^2 + 6x_1x_2}, \quad \mathbf{x} \neq \mathbf{0} \in \mathbb{R}^2.$$



Theorem 0.11. For any two matrices $\mathbf{A} \in S^n(\mathbb{R})$ and $\mathbf{B} \in S^n_+(\mathbb{R})$, let the largest and smallest generalized eigenvalues of $(\mathbf{A}, \mathbf{B})$ be λ_1 and λ_n, with corresponding unit-norm generalized eigenvectors $\mathbf{v}_1, \mathbf{v}_n \in \mathbb{R}^n$, respectively. Then

$$\max_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{B} \mathbf{x}} = \lambda_1, \quad @\ \mathbf{x} = \pm \mathbf{v}_1$$

$$\min_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{B} \mathbf{x}} = \lambda_n, \quad @\ \mathbf{x} = \pm \mathbf{v}_n$$

As for the ordinary Rayleigh quotient, there is a restricted version of the generalized Rayleigh quotient.

Theorem 0.12. Let $\mathbf{A} \in S^n(\mathbb{R})$ and $\mathbf{B} \in S^n_+(\mathbb{R})$ be two fixed matrices with generalized eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n$ and eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{R}^n$, that is, $\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{B}\mathbf{v}_i$ for each $i = 1, \ldots, n$. We have

$$\max_{\substack{\mathbf{x} \neq \mathbf{0} \\ \mathbf{v}_1^T \mathbf{B}\mathbf{x} = 0}} \frac{\mathbf{x}^T \mathbf{A}\mathbf{x}}{\mathbf{x}^T \mathbf{B}\mathbf{x}} = \lambda_2 \quad (\text{when } \mathbf{x} = \pm \mathbf{v}_2)$$

$$\min_{\substack{\mathbf{x} \neq \mathbf{0} \\ \mathbf{v}_1^T \mathbf{B}\mathbf{x} = \mathbf{v}_2^T \mathbf{B}\mathbf{x} = 0}} \frac{\mathbf{x}^T \mathbf{A}\mathbf{x}}{\mathbf{x}^T \mathbf{B}\mathbf{x}} = \lambda_3 \quad (\text{when } \mathbf{x} = \pm \mathbf{v}_3)$$

and so on.

The singular value decomposition (SVD) for general matrices

The following is a fundamental result in linear algebra, and also a very important computing tool in many applications.

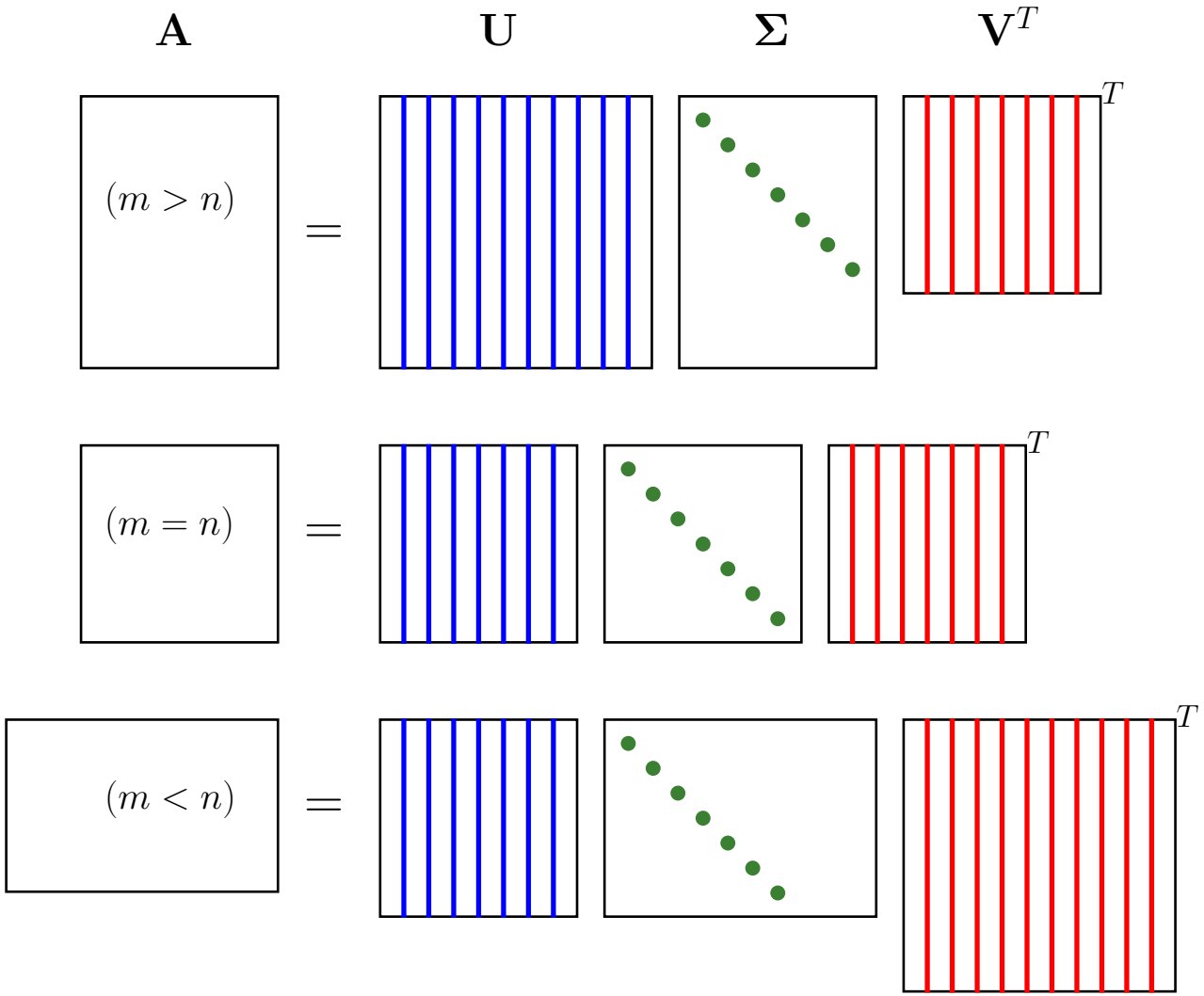
Theorem: For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, there exist two orthogonal matrices $\mathbf{U} \in \mathbb{R}^{m \times m}, \mathbf{V} \in \mathbb{R}^{n \times n}$ and a nonnegative, diagonal matrix $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ such that

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

Moreover, the number of positive diagonals of $\mathbf{\Sigma}$ equals the rank of $\mathbf{A}$.

Remark. This factorization is called the SVD of $\mathbf{A}$:

- The diagonals of $\mathbf{\Sigma}$ are called the singular values of $\mathbf{A}$
- The columns of $\mathbf{U}$ are called the left singular vectors of $\mathbf{A}$.
- The columns of $\mathbf{V}$ are called the right singular vectors of $\mathbf{A}$.


A
U
Σ
V^T
(m > n)
=
T
(m = n)
=
T
(m < n)
=
T

Example 0.10. It can be directly verified that

$$\underbrace{\begin{pmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}}_{\mathbf{A}} = \underbrace{\begin{pmatrix} \frac{2}{\sqrt{6}} & 0 & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{3}} \end{pmatrix}}_{\mathbf{U}} \cdot \underbrace{\begin{pmatrix} \sqrt{3} & \\ & 1 \end{pmatrix}}_{\mathbf{\Sigma}} \cdot \underbrace{\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}^T}_{\mathbf{V}^T} .$$

In the above equation, $\mathbf{U}, \mathbf{V}$ are orthogonal matrices and $\mathbf{\Sigma}$ is a diagonal matrix. Therefore, the above factorization represents a singular value decomposition of $\mathbf{A}$.

Moreover, $\mathrm{rank}(\mathbf{A}) = 2$, and there are precisely 2 positive entries in the diagonal of $\mathbf{\Sigma}$.

Connection to symmetric matrices

From the SVD of $\mathbf{A}$ we obtain that

$$\mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \cdot \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T = \mathbf{U}\left(\mathbf{\Sigma}\mathbf{\Sigma}^T\right)\mathbf{U}^T$$
$$\mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T \cdot \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\left(\mathbf{\Sigma}^T\mathbf{\Sigma}\right)\mathbf{V}^T$$

This shows that

- $\mathbf{U}$ is the eigenvectors matrix of $\mathbf{A}\mathbf{A}^T$;
- $\mathbf{V}$ is the eigenvectors matrix of $\mathbf{A}^T\mathbf{A}$;
- The nonzero eigenvalues of $\mathbf{A}\mathbf{A}^T, \mathbf{A}^T\mathbf{A}$ (which must be the same) are equal to the squared singular values of $\mathbf{A}$.

Example 0.11. For the matrix $\mathbf{A}$ in the preceding example, we have

$$\mathbf{A}\mathbf{A}^T = \underbrace{\begin{pmatrix} \frac{2}{\sqrt{6}} & 0 & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{3}} \end{pmatrix}}_{\mathbf{U}} \cdot \underbrace{\begin{pmatrix} 3 & & \\ & 1 & \\ & & 0 \end{pmatrix}}_{\mathbf{\Sigma}\mathbf{\Sigma}^T} \cdot \underbrace{\begin{pmatrix} \frac{2}{\sqrt{6}} & 0 & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{3}} \end{pmatrix}^T}_{\mathbf{U}^T}$$

$$\mathbf{A}^T\mathbf{A} = \underbrace{\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}}_{\mathbf{V}} \cdot \underbrace{\begin{pmatrix} 3 & \\ & 1 \end{pmatrix}}_{\mathbf{\Sigma}^T\mathbf{\Sigma}} \cdot \underbrace{\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}^T}_{\mathbf{V}^T}$$

How to prove the SVD theorem

Given any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the SVD can be thought of as solving a matrix equation for three unknown matrices (each with certain constraint):

$$\mathbf{A} = \underbrace{\mathbf{U}}_{\text{orthogonal}} \cdot \underbrace{\mathbf{\Sigma}}_{\text{diagonal}} \cdot \underbrace{\mathbf{V}^T}_{\text{orthogonal}} .$$

Suppose such solutions exist. From

$$\mathbf{A}^T\mathbf{A} = \mathbf{V}\left(\mathbf{\Sigma}^T\mathbf{\Sigma}\right)\mathbf{V}^T$$

we can find $\mathbf{V}$ and $\mathbf{\Sigma}$, which contain the eigenvectors and square roots of eigenvalues of $\mathbf{A}^T\mathbf{A}$, respectively.

After we have found both $\mathbf{V}$ and $\mathbf{\Sigma}$, rewrite the matrix equation as

$$\mathbf{A}_{m\times n}\mathbf{V}_{n\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n},$$

or in columns,

$$\mathbf{A}[\mathbf{v}_1 \dots \mathbf{v}_r\ \mathbf{v}_{r+1} \dots \mathbf{v}_n] = [\mathbf{u}_1 \dots \mathbf{u}_r\ \mathbf{u}_{r+1} \dots \mathbf{u}_m] \begin{bmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_r & \\ & & & \end{bmatrix}.$$

By comparing columns, we obtain

$$\mathbf{A}\mathbf{v}_i = \begin{cases} \sigma_i \mathbf{u}_i, & 1 \le i \le r \text{ (\#nonzero singular values)} \\ \mathbf{0}, & r < i \le n \end{cases}$$

This tells us how to find the first r columns of matrix $\mathbf{U} \in \mathbb{R}^{m \times m}$:

$$\mathbf{u}_i = \frac{1}{\sigma_i} \mathbf{A}\mathbf{v}_i \quad \text{for all } 1 \leq i \leq r.$$

The remaining columns of $\mathbf{U}$ will be found by completing an orthonormal basis for $\mathbb{R}^m$, starting with $\{\mathbf{u}_1, \ldots, \mathbf{u}_r\}$:

$$\mathbf{u}_i^T \mathbf{x} = 0, \quad i = 1, \ldots, r$$
$$\|\mathbf{x}\| = 1$$

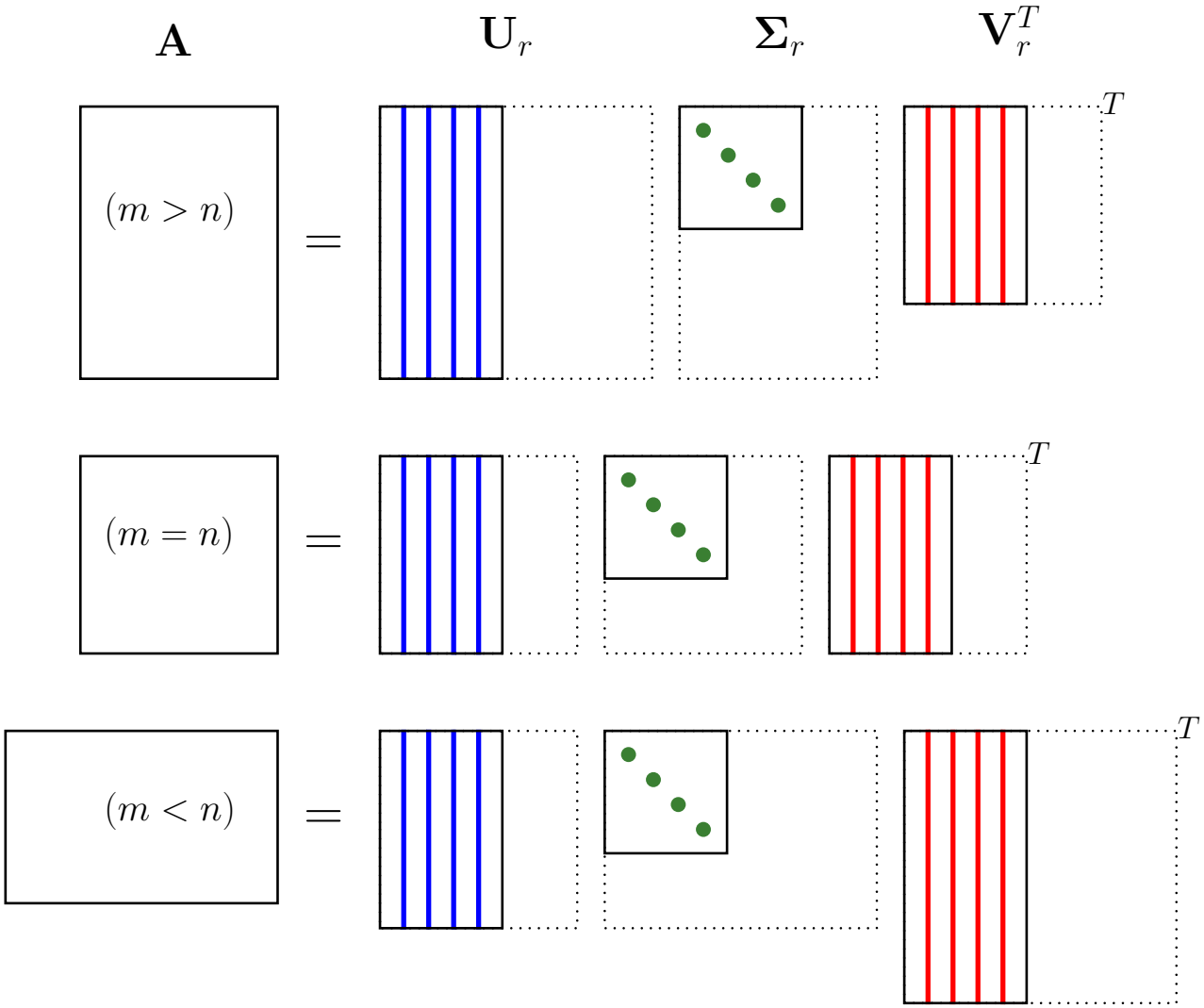
Different versions of SVD

- **Full SVD**: $\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\mathbf{V}^T_{n\times n}$
- **Compact SVD**: Suppose $\mathrm{rank}(\mathbf{A}) = r$. Define

$$\mathbf{U}_r = [\mathbf{u}_1, \ldots, \mathbf{u}_r] \in \mathbb{R}^{m\times r}$$
$$\mathbf{V}_r = [\mathbf{v}_1, \ldots, \mathbf{v}_r] \in \mathbb{R}^{n\times r}$$
$$\mathbf{\Sigma}_r = \mathrm{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$$

Then

$$\mathbf{A} = \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^T.$$


A
$\mathbf{U}_r$
$\mathbf{\Sigma}_r$
$\mathbf{V}_r^T$
$(m > n)$
=
T
$(m = n)$
=
T
$(m < n)$
=
T

- **Rank-1 decomposition**:

$$\mathbf{A} = [\mathbf{u}_1, \ldots, \mathbf{u}_r] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_r^T \end{bmatrix} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T .$$

This has the interpretation that $\mathbf{A}$ is a weighted sum of rank-one matrices, as for a square, symmetric matrix.

Note that $-\mathbf{u}_i, -\mathbf{v}_i$ are also corresponding singular vectors to σ_i:

$$\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T = \sum_{i=1}^{r} \sigma_i (-\mathbf{u}_i)(-\mathbf{v}_i)^T .$$

This shows that the SVD of a matrix is not unique.

- **Truncated SVD**: For any integer $1 \leq K \leq r$, we define the truncated SVD of $\mathbf{A}$ with K terms as

$$\mathbf{A} \approx \sum_{i=1}^{K} \sigma_i \mathbf{u}_i \mathbf{v}_i^T = \mathbf{A}_K$$

where the singular values are assumed to be sorted from large to small (so $\sigma_1, \ldots, \sigma_K$ represent the largest K singular values).

Note that $\mathbf{A}_K$ has a rank of K and it can be regarded as a low-rank approximation to $\mathbf{A}$. It is memory efficient and often adequate for computing tasks.

Matrix norm

A matrix norm is a norm on $\mathbb{R}^{m\times n}$ as a vector space (consisting of all matrices of the fixed size).

More specifically, a matrix norm is a function

$$\|\cdot\| : \mathbb{R}^{m\times n} \to \mathbb{R}$$

that satisfies the following three conditions:

- $\|\mathbf{A}\| \geq 0$ for all $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\|\mathbf{A}\| = 0$ if and only if $\mathbf{A} = \mathbf{O}$
- $\|k\mathbf{A}\| = |k| \cdot \|\mathbf{A}\|$ for any scalar $k \in \mathbb{R}$ and matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$
- $\|\mathbf{A} + \mathbf{B}\| \leq \|\mathbf{A}\| + \|\mathbf{B}\|$ for any two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$

Note that multiplication is also defined between matrices (with compatible sizes).

We say that a matrix norm $\|\cdot\|$ is **sub-multiplicative** if for any two matrices $\mathbf{A} \in \mathbb{R}^{m\times n}, \mathbf{B} \in \mathbb{R}^{n\times p}$,

$$\|\mathbf{AB}\| \leq \|\mathbf{A}\| \cdot \|\mathbf{B}\|.$$

Some textbooks include the sub-multiplicative requirement in the definition for matrix norms.

The norms introduced here are all sub-multiplicative anyway.

The Frobenius norm

Def 0.5. The Frobenius norm of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is defined as

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2}$$

It is equivalent to the Euclidean 2-norm on vectorized matrices (i.e., $\mathbb{R}^{mn}$):

$$\|\mathbf{A}\|_F = \|\mathbf{A}(:)\|_2$$

Thus, it must satisfy all the three conditions of a norm.

Example 0.12. Let

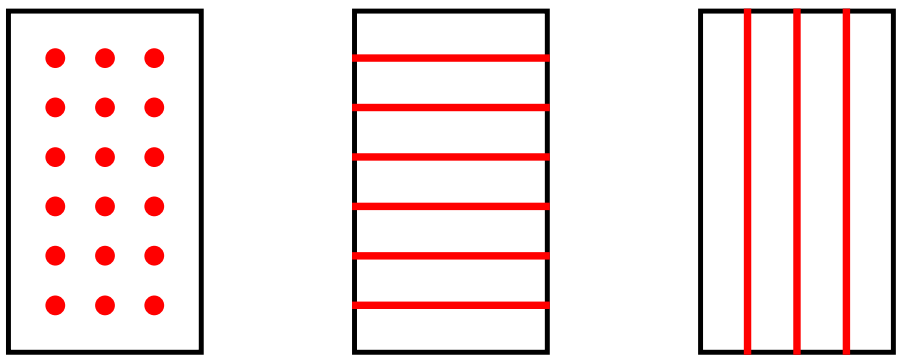
$$\mathbf{A} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

By direct calculation,

$$\|\mathbf{A}\|_F = \sqrt{1^2 + (-1)^2 + 0^2 + 1^2 + 1^2 + 0^2} = 2.$$

Remark. For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$,

$$\|\mathbf{A}\|_F^2 = \sum_{i=1}^{m} \|A_i\|_2^2 = \sum_{j=1}^{n} \|\mathbf{a}_j\|_2^2$$



Proposition 0.13. For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$,

$$\|\mathbf{A}\|_F^2 = \operatorname{trace}(\mathbf{A}\mathbf{A}^T) = \operatorname{trace}(\mathbf{A}^T\mathbf{A})$$

Proof.

$$\operatorname{trace}(\mathbf{A}\mathbf{A}^T) = \sum_{i=1}^{m} A_i \cdot A_i^T = \sum_{i=1}^{m} \|A_i\|_2^2 = \|\mathbf{A}\|_F^2.$$

The other equality can be proved similarly, or instead using the matrix trace property:

$$\operatorname{trace}(\mathbf{A}\mathbf{B}) = \operatorname{trace}(\mathbf{B}\mathbf{A})$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times m}$. □

Theorem 0.14. Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be any matrix. Suppose its (nonzero) singular values are $\sigma_1 \geq \cdots \geq \sigma_r > 0$, where $r = \mathrm{rank}(\mathbf{A})$. Then

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{r} \sigma_i^2}$$

Proof. Consider the matrix $\mathbf{A}^T\mathbf{A}$. Its nonzero eigenvalues are $\lambda_i = \sigma_i^2$. According to the theorem on the preceding slide,

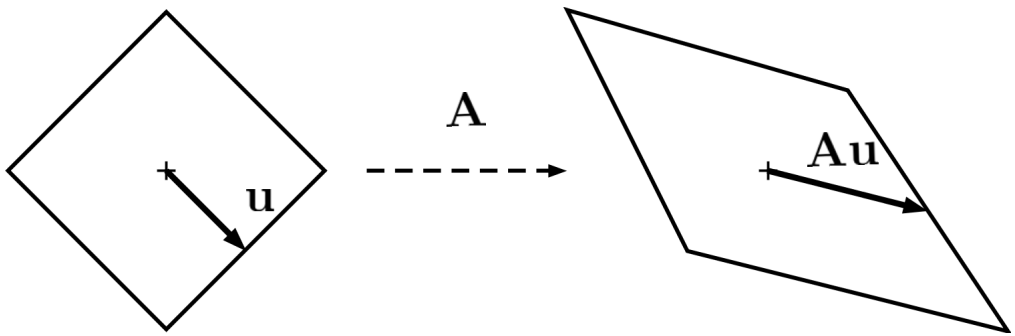
$$\|\mathbf{A}\|_F^2 = \mathrm{trace}(\mathbf{A}^T\mathbf{A}) = \sum_{i=1}^{r} \lambda_i = \sum_{i=1}^{r} \sigma_i^2.$$

The matrix operator norm

A second matrix norm is the operator norm, which is induced by a vector norm on Euclidean spaces.

Theorem 0.15. For any vector norm $\|\cdot\|$ on Euclidean spaces, the following is a matrix norm on $\mathbb{R}^{m\times n}$:

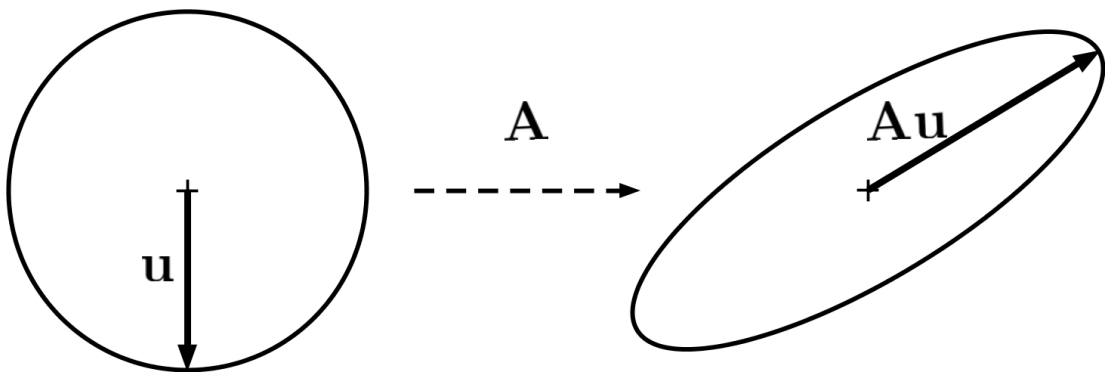
$$\|\mathbf{A}\| \stackrel{\text{def}}{=} \max_{\mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{Ax}\|}{\|\mathbf{x}\|} = \max_{\mathbf{u}\in\mathbb{R}^n:\|\mathbf{u}\|=1} \|\mathbf{Au}\|$$



When the Euclidean norm (i.e., 2-norm) is used, the induced matrix operator norm is called the spectral norm.

Def 0.6. The **spectral norm** of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is defined as

$$\|\mathbf{A}\|_2 = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_2} = \max_{\mathbf{u} \in \mathbb{R}^n : \|\mathbf{u}\|_2 = 1} \|\mathbf{A}\mathbf{u}\|_2$$



Theorem 0.16. Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be any matrix whose singular values (from large to small) are $\sigma_1 \geq \sigma_2 \geq \cdots$. Then

$$\|\mathbf{A}\|_2 = \sigma_1.$$

Proof. Consider the matrix $\mathbf{A}^T\mathbf{A}$ which is a positive semidefinite matrix with largest eigenvalue $\lambda_1 = \sigma_1^2$. We have

$$\|\mathbf{A}\|_2^2 = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \lambda_1 = \sigma_1^2,$$

where we used the Rayleigh quotient theorem. The maximizer is the largest right singular vector $\mathbf{v}_1$ of $\mathbf{A}$ (corresponding to σ_1). □

Low-rank approximation of matrices

Problem. For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ and integer $k \geq 1$, find the rank-k matrix $\mathbf{B}$ that is the closest to $\mathbf{A}$ (under the Frobenius, or spectral norm):

$$\min_{\mathbf{B} \in \mathbb{R}^{m \times n} \,:\, \mathrm{rank}(\mathbf{B}) = k} \|\mathbf{A} - \mathbf{B}\|$$

Remark. This problem arises in a number of tasks, e.g.,

- Data compression (and noise reduction)
- Matrix completion (and recommender systems)
- Orthogonal least squares fitting

Theorem 0.17 (Eckart–Young–Mirsky). Given $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $1 \leq k \leq r = \operatorname{rank}(\mathbf{A})$, let $\mathbf{A}_k$ be the truncated SVD of $\mathbf{A}$ with the largest k terms: $\mathbf{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$. Then $\mathbf{A}_k$ is the best rank-k approximation to $\mathbf{A}$ in terms of both the Frobenius and spectral norms:

$$\min_{\mathbf{B}\,:\,\operatorname{rank}(\mathbf{B})=k} \|\mathbf{A}-\mathbf{B}\|_F = \|\mathbf{A}-\mathbf{A}_k\|_F = \sqrt{\sum_{i>k} \sigma_i^2}$$

$$\min_{\mathbf{B}\,:\,\operatorname{rank}(\mathbf{B})=k} \|\mathbf{A}-\mathbf{B}\|_2 = \|\mathbf{A}-\mathbf{A}_k\|_2 = \sigma_{k+1}.$$

Remark. The theorem still holds true if the equality constraint $\operatorname{rank}(\mathbf{B}) = k$ is relaxed to the inequality constraint $\operatorname{rank}(\mathbf{B}) \leq k$ (which will also include all the lower-rank matrices).

Pseudoinverse

Def 0.7. Let $\mathbf{A} \in \mathbb{R}^{m \times n}$. We call the matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$ the **Moore–Penrose pseudoinverse** of $\mathbf{A}$ if it satisfies all four conditions below:

(1) $\mathbf{ABA} = \mathbf{A}$ $\longleftarrow$ $\mathbf{B}$ is a generalized inverse of $\mathbf{A}$

(2) $\mathbf{BAB} = \mathbf{B}$ $\longleftarrow$ $\mathbf{A}$ is a generalized inverse of $\mathbf{B}$

(3) $(\mathbf{AB})^T = \mathbf{AB}$ $\longleftarrow$ $\mathbf{AB}$ is symmetric

(4) $(\mathbf{BA})^T = \mathbf{BA}$ $\longleftarrow$ $\mathbf{BA}$ is symmetric

Remark.

- If $\mathbf{B}$ only satisfies (1), it is known as a generalized inverse of $\mathbf{A}$; if $\mathbf{B}$ only satisfies (1) and (2), it is called a **reflexive generalized inverse**.
- For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the pseudoinverse always exists and is unique. We denote the pseudoinverse of $\mathbf{A}$ as $\mathbf{A}^\dagger$.
- The symmetric form of the definition implies $\mathbf{B} = \mathbf{A}^\dagger$ and $\mathbf{A} = \mathbf{B}^\dagger$, and thus, $\mathbf{A} = (\mathbf{A}^\dagger)^\dagger$.
- If $\mathbf{A}$ is invertible, then $\mathbf{A}^\dagger = \mathbf{A}^{-1}$.

Example 0.13. Let $\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$. Verify that $\mathbf{A}^\dagger = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix}$.

By direct calculation,

$$\mathbf{A}\mathbf{A}^\dagger = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} \qquad \text{(symmetric)}$$

$$\mathbf{A}^\dagger\mathbf{A} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \qquad \text{(symmetric)}$$

$$\mathbf{A}\mathbf{A}^\dagger\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} = \mathbf{A}$$

$$\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \end{bmatrix} = \mathbf{A}^\dagger$$

Example 0.14. It can be directly verified in the same way that the pseudoinverse of the following "diagonal" matrix

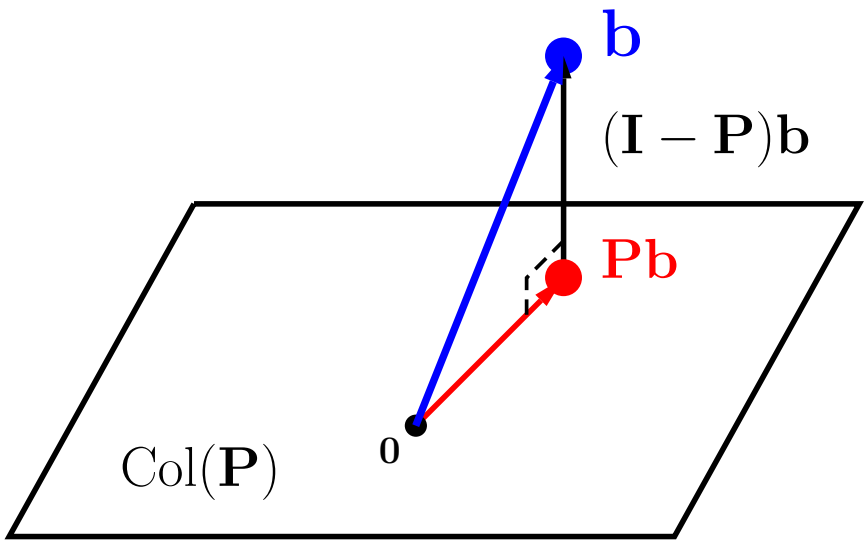
$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \end{bmatrix}$$

is

$$\mathbf{A}^{\dagger} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Orthogonal projection matrices

Def 0.8. A square matrix $\mathbf{P}$ is called a **orthogonal projection matrix** if it is both symmetric and idempotent, i.e., $\mathbf{P} = \mathbf{P}^T$ and $\mathbf{P} = \mathbf{P}^2$.



Remark. For any orthogonal projection matrix $\mathbf{P}$, the matrix $\mathbf{I} - \mathbf{P}$ is also an orthogonal projection matrix:

First, $\mathbf{I} - \mathbf{P}$ is also symmetric.

Second,

$$(\mathbf{I} - \mathbf{P})^2 = \mathbf{I} - 2\mathbf{P} + \mathbf{P}^2 = \mathbf{I} - \mathbf{P}.$$

Let $\mathbf{P} \in \mathbb{R}^{n \times n}$ be any orthogonal projection matrix. It must project any vector $\mathbf{b} \in \mathbb{R}^n$ onto its column space, i.e., $\mathbf{Pb} \in \mathrm{Col}(\mathbf{P})$.

This leads to the following decomposition of $\mathbf{b}$:

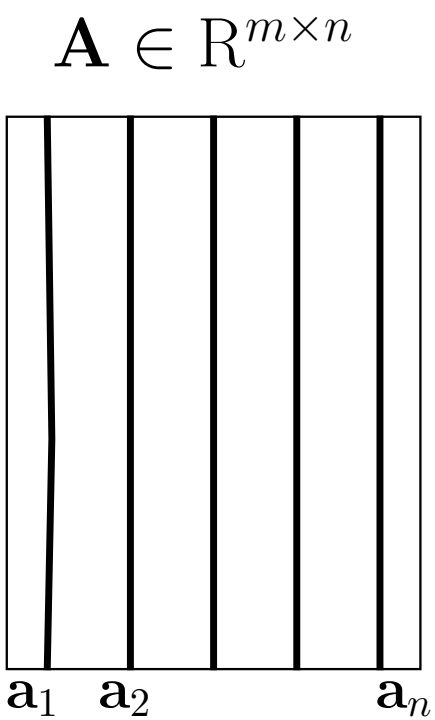
$$\mathbf{b} = \underbrace{\mathbf{Pb}}_{\in \mathrm{Col}(\mathbf{P})} + \underbrace{(\mathbf{I}-\mathbf{P})\mathbf{b}}_{\in \mathrm{Col}(\mathbf{I}-\mathbf{P})}.$$

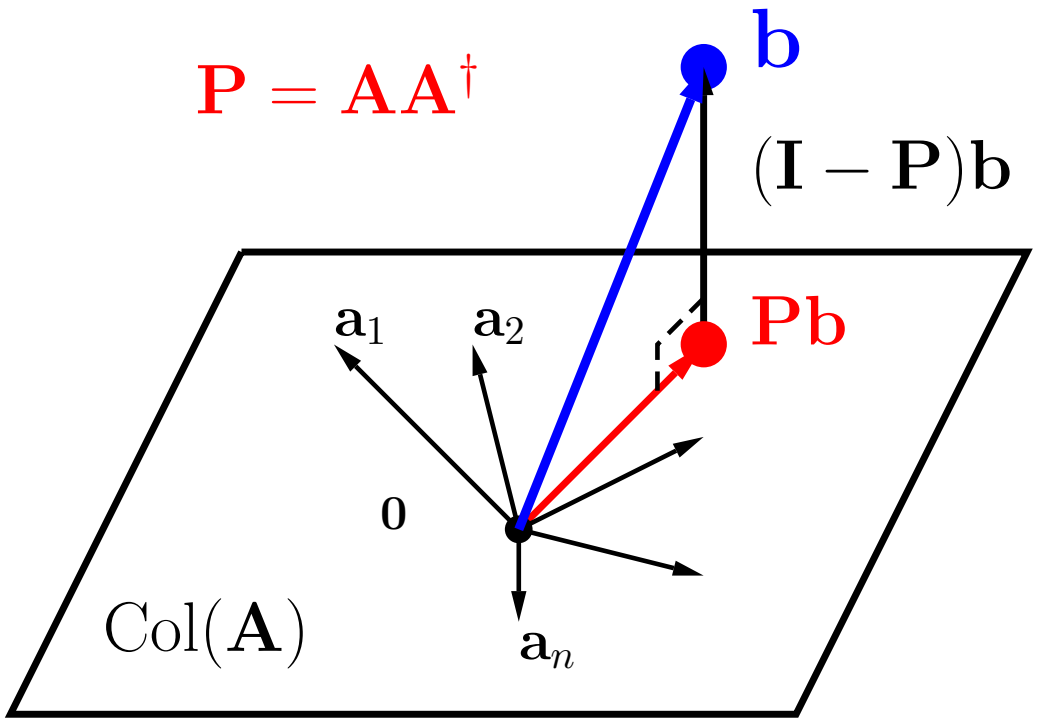
Since $\mathbf{P} = \mathbf{P}^T$ by definition, we have

$$(\mathbf{Pb})^T(\mathbf{I}-\mathbf{P})\mathbf{b} = \mathbf{b}^T\mathbf{P}(\mathbf{I}-\mathbf{P})\mathbf{b} = \mathbf{b}^T(\mathbf{P}-\mathbf{P}^2)\mathbf{b} = 0.$$

This shows that the two components, i.e., $\mathbf{Pb}$ and $(\mathbf{I}-\mathbf{P})\mathbf{b}$, are orthogonal to each other.

Theorem 0.18. For any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{A}\mathbf{A}^\dagger$ is an orthogonal projection matrix (onto the column space of $\mathbf{A}$).





Finding matrix pseudoinverse in general

Theorem 0.19. Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be any matrix. Suppose its full SVD is $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Then the pseudoinverse of $\mathbf{A}$ is

$$\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T.$$

Proof We verify the four conditions directly:

$$\mathbf{A}\mathbf{A}^\dagger\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \cdot \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T \cdot \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^\dagger\mathbf{\Sigma}\mathbf{V}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{A}$$

$$\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T \cdot \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \cdot \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{\Sigma}\mathbf{\Sigma}^\dagger\mathbf{U}^T = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T = \mathbf{A}^\dagger$$

$$\mathbf{A}\mathbf{A}^\dagger = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \cdot \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^\dagger\mathbf{U}^T \qquad \text{(symmetric)}$$

$$\mathbf{A}^\dagger\mathbf{A} = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T \cdot \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{\Sigma}\mathbf{V}^T \qquad \text{(symmetric)}$$

Remark. The previous formula for $\mathbf{A}^\dagger$ is also in full SVD form:

$$\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T$$

It can be simplified into the compact SVD form

$$\mathbf{A}^\dagger = \mathbf{V}_r\mathbf{\Sigma}_r^{-1}\mathbf{U}_r^T$$

Thus, it suffices to find the compact SVD of $\mathbf{A}$ and use it to find $\mathbf{A}^\dagger$.

This simplified formula is computationally more efficient, as it avoids computing the redundant left/right singular vectors.

Finding matrix pseudoinverse in a special case

Theorem 0.20. Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be any tall matrix with full column rank (i.e., $\text{rank}(\mathbf{A}) = n \leq m$). Then the pseudoinverse of $\mathbf{A}$ is

$$\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T.$$

Remark. The theorem implies that for any tall matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with full column rank (i.e., $\text{rank}(\mathbf{A}) = n \leq m$), the following is an **orthogonal projection** matrix (onto the column space of $\mathbf{A}$):

$$\mathbf{A}\mathbf{A}^\dagger = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T.$$

Proof. It suffices to verify the four conditions for being a pseudoinverse:

$$\begin{aligned}
\mathbf{A}\mathbf{A}^\dagger\mathbf{A} &= \mathbf{A}\cdot(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\cdot\mathbf{A} = \mathbf{A} \\
\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger &= (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\cdot\mathbf{A}\cdot(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T = \mathbf{A}^\dagger \\
\mathbf{A}\mathbf{A}^\dagger &= \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T && (\text{symmetric}) \\
\mathbf{A}^\dagger\mathbf{A} &= (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\cdot\mathbf{A} = \mathbf{I}_n && (\text{symmetric})
\end{aligned}$$

Therefore, $\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is the pseudoinverse of $\mathbf{A}$. □

Remark. Let $\mathbf{U} \in \mathbb{R}^{m \times n}$ be a tall matrix with orthonormal columns (e.g., an orthonormal basis matrix). Then it has full column rank, and

$$\mathbf{U}^T\mathbf{U} = \begin{bmatrix} \mathbf{u}_1^T \\ \vdots \\ \mathbf{u}_n^T \end{bmatrix} [\mathbf{u}_1 \dots \mathbf{u}_n] = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix} = \mathbf{I}_n$$

It follows that

- $\mathbf{U}^\dagger = (\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T = \mathbf{U}^T$ (pseudoinverse), and
- $\mathbf{U}\mathbf{U}^\dagger = \mathbf{U}\mathbf{U}^T$ (orthogonal projection matrix).

Further learning

See *Math 250 Mathematical Data Visualization* at San José State:

`https://www.sjsu.edu/faculty/guangliang.chen/Math250.html`