Lecture Notes

Fundamentals of Big Data Analytics

Prof. Dr. Rudolf Mathar
Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Theoretische Informationstechnik
Kopernikusstraße 16
52074 Aachen

Version from January 18, 2019

Contents

1 Introduction

What is (big) data analytics? One can simply define it as the discovery of "models" for data to extract information, draw conclusions and make decisions. A "Model" can be one of several things:

- Statistical model which is the underlying distribution from which the data is drawn. **Example:** *given a set of real numbers, each one independently Gaussian distributed, estimate the mean and variance.* The model for data here is Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ where each data is its independent realization.

- Use the data as a training set for algorithms of machine learning, e.g., Bayes nets, support-vector machines, decision trees, etc.
  Example: ([LRU14]) In "Netflix challenge", the goal was to devise an algorithm that predicts the ranking of movies by users.

- Extract the most prominent features of the data and ignore the rest [LRU14, page 4].
  Example: Feature extraction, similarity, PCA

- Summarization of features
  Example: First example is Page rank (Google's web mining), probability that a random walker on the graph meets that page at any given time. Second example is clustering. Points that are close are summarized, e.g, by their clusters.

One should be careful about the effect of big data analytics. In large random data sets, unusual features occur which are the effect of purely random nature of data. This is called *Bonferroni's principle.*

Example ([LRU14, page. 6])**.** Find evil-doers by looking for people who both were in the same hotel on two different days. Here are the assumptions:

- 10^5 hotels

- Everyone goes to a hotel one day in 100

- 10^9 people

- People pick days and hotels at random independently

- Examine hotel records for 1000 days.

Probability that any two people visit a hotel on any given day is equal to $\frac{1}{100} \times \frac{1}{100}$. Probability that they pick the same hotel is $\frac{1}{10^4} \times \frac{1}{10^5} = 10^{-9}$. Probability that two people visit the same hotel on two different days are $10^{-9} \times 10^{-9} = 10^{-18}$.
Cardinality of the event space is: pairs of people $\binom{10^9}{2}$, pairs of days $\binom{10^3}{2}$. Expected number of evil-doing events, using $\binom{n}{2} \approx \frac{n^2}{2}$, is given by:

$$\binom{10^9}{2} \times \binom{10^3}{2} \times 10^{-18} \approx 5.10^{17}.5.10^5.10^{-18} = 25 \times 10^4 = 250000.$$

Below it is shortly discussed how to carry out computation on large data sets, although it will not be he focus of this lecture.

1.1 MapReduce and Hadoop

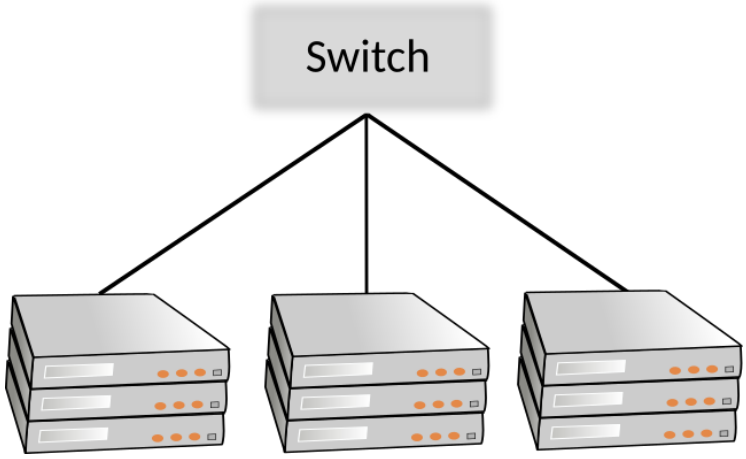


Figure 1.1: Racks of compute nodes

When the computation is to be performed on very large data sets, it is not efficient to fit the whole data in a data-base and perform the computations sequentially. The key idea is to use parallelism from "computing clusters", not a super computer, built of commodity hardware, connected by Ethernet or inexpensive switches.
The software stack consists of distributed file systems (DFS) and MapReduce. In a distributed file system Files are divided into chunks (typically 64 MB) and chunks are replicated, typically 3 times on different racks. There exists a file master mode or name mode with information where to find copies of files. Some of the implementations of DFS are GFS (Google file system), HDFS (Hadoop Distributed File System, Apache) and Cloud Store (open source DFS).
On the other hand MapReduce is the computing paradigm. In MapReduce, the system manages parallel execution and coordination of tasks. Two functions are written by users namely Map and Reduce. The advantage of this system is its robustness to hardware

failures and it is able to handle large datasets. MapReduce is implemented internally by Google.
The architecture of this system is such that compute nodes are stored on racks, each with its own processor and storage device. Many racks are connected by a switch as presented in Figure 1.1. They are connected by some fast network, interconnection by Gigabit Internet. The principles of this system are as follows. First, files must be stored redundantly to protect against failure of nodes. Second, computations must be divided into independent tasks. If one fails it can be restored without affecting others.
We discuss an example of implementation matrix-vector multiplication using MapReduce [LRU14].

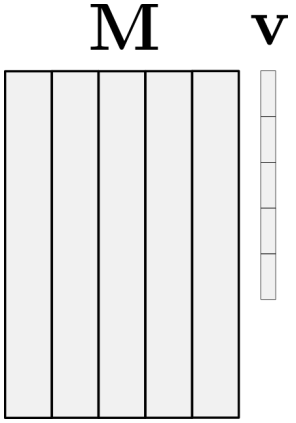


Figure 1.2: Matrix-Vector Multiplication

Example (Matrix-Vector Multiplication by MapReduce)**.** Suppose that the matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ and the vector $\mathbf{v} \in \mathbb{R}^n$ are given and the goal is to compute their multiplication $\mathbf{x} = \mathbf{Mv}$:

$$x_i = \sum_{j=1}^{n} m_{ij} v_j.$$

When n is large, say 10^7 then the direct computation requires the storage of the whole matrix in the storage which might not be efficient. Particularly in practice the matrix $\mathbf{M}$ can be sparse with say 10 or 15 non-zeros per row.
First the matrix and the vector is stored as the pairs $(i, j, m_{ij}$ and the vector is stored as (i, v_i). MapReduce consists of two main functions, Map function and Reduce function. To implement the multiplication using MapReduce, Map function produces a key-value pair to each entries of the matrix and the vector. To the entry m_{ij} the pair $(i, m_{ij}v_j)$ is associated where i is the key and $m_{ij}v_j$ is the pair. Note that it is assumed here that m is small enough to store the vector $\mathbf{v}$ in its entirety in the memory. The Reduce function receives all the key-value pairs, lists all pairs with key i and sum their values to get $(i, \sum_{j=1}^{n} m_{ij}x_j)$ which gives the ith entry of the product.
If the vector $\mathbf{v}$ cannot fit into the memory then the matrix $\mathbf{M}$ is divided into horizontal strips with certain width and the vector $\mathbf{v}$ is divided into vertical stripes with the same size

as the matrix stripes' width. Accordingly the multiplication can be divided into sub-tasks, each feasible using the MapReduce.

Example (Matrix-Matrix Multiplication by MapReduce)**.** Given two matrices $\mathbf{M} \in \mathbb{R}^{n \times m}$ and $\mathbf{N} \in \mathbb{R}^{m \times r}$, the goal is to compute $\mathbf{MN}$. Map function generates the following key-value pairs:

- For each element m_{ij} of $\mathbf{M}$ produce r key-value pairs $((i,k),(\mathbf{M},j,m_{ij}))$ for $k = 1,\ldots,r$.
- For each element n_{jk} of $\mathbf{N}$ produce n key-value pairs $((i,k),(\mathbf{N},j,n_{jk}))$ for $i = 1,\ldots,n$.

The Reduce function computes the multiplication as follows:

- For each key (i,k), find the values with the same j.
- Multiply m_{ij} and n_{jk} to get $m_{ij}n_{jk}$.
- Sum up all $m_{ij}n_{jk}$ over j to get $\sum_{j=1}^{m} m_{ij}n_{jk}$.

2 Prerequisites from Matrix Algebra

In this chapter we review some basic results from matrix algebra.
Real $m \times n$ matrices will be written as:

$$\mathbf{M} = (m_{ij})_{1 \le i \le m, 1 \le j \le n} \in \mathbb{R}^{m \times n}$$

Diagonal matrices are written as $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^{n \times n}$. A Matrix $\mathbf{U} \in \mathbb{R}^{n \times n}$ is called orthogonal if :

$$\mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I}_n$$

where $\mathbf{I}_n$ is the $n \times n$ identity matrix. $\mathcal{O}(n)$ denotes the set of orthogonal $n \times n$ matrices.

Theorem 2.1 (Singular Value Decomposition, SVD)**.** *Given $\mathbf{M} \in \mathbb{R}^{m \times n}$, there exists $\mathbf{U} \in \mathcal{O}(m)$ and $\mathbf{V} \in \mathcal{O}(n)$ and some $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ with non-negative entries in its diagonal and zeros otherwise such that:*

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

The diagonal elements of $\mathbf{\Sigma}$ are called singular values. The columns of $\mathbf{U}$ and $\mathbf{V}$ are called left and right singular vectors of $\mathbf{M}$.

Remark 1. If $m < n$, say, SVD may be written as :

$$\exists \mathbf{U} \in \mathbb{R}^{m \times n}, \mathbf{U}\mathbf{U}^T = \mathbf{I}_m, \exists \mathbf{V} \in \mathcal{O}(n); \exists \mathbf{\Sigma} \in \mathbb{R}^{n \times n} \text{ diagonal, such that: } \mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

Theorem 2.2 (Spectral Decomposition)**.** *Given $\mathbf{M} \in \mathbb{R}^{n \times n}$ symmetric, there exists $\mathbf{V} \in \mathcal{O}(n)$, $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_n)$ such that:*

$$\mathbf{M} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^T.$$

$\mathbf{v}_i$'s are eigenvectors of $\mathbf{M}$ with the eigenvalues λ_i.

- If for the symmetric matrix $\mathbf{M}$, $\lambda_i > 0$, $i = 1, \ldots, n$, then $\mathbf{M}$ is called positive definite (p.d.) and writes as $\mathbf{M} \succ 0$.
  If for the symmetric matrix $\mathbf{M}$, $\lambda_i \ge 0$, $i = 1, \ldots, n$, then $\mathbf{M}$ is called non-negative definite (n.n.d.) and writes as $\mathbf{M} \succeq 0$.
- If $\mathbf{M}$ is non-negative definite, then it has a Cholesky decomposition

  $$\mathbf{M} = \mathbf{V}\mathbf{\Lambda}^{\frac{1}{2}}(\mathbf{V}\mathbf{\Lambda}^{\frac{1}{2}})^T,$$

  where $\mathbf{\Lambda}^{\frac{1}{2}} = \text{diag}(\lambda_1^{\frac{1}{2}}, \ldots, \lambda_n^{\frac{1}{2}})$.

- $\mathbf{M} \succeq 0 \iff \mathbf{x}^T\mathbf{M}\mathbf{x} \geq 0 \quad \forall \mathbf{x} \in \mathbb{R}^n$
- $\mathbf{M} \succ 0 \iff \mathbf{x}^T\mathbf{M}\mathbf{x} > 0 \quad \forall \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{x} \neq 0.$

Definition 2.3. (a) Given $\mathbf{M} = (m_{ij}) \in \mathbb{R}^n$, $\mathrm{tr}(\mathbf{M}) = \sum_{i=1}^n m_{ii}$ is called the trace of $\mathbf{M}$.

(b) Given $\mathbf{M} \in \mathbb{R}^{n\times n}$, $\|\mathbf{M}\|_F = \sqrt{\sum_{i,j} m_{ij}^2} = \sqrt{\mathrm{tr}(\mathbf{M}^T\mathbf{M})}$ is called the Frobenius norm.

(c) Given $\mathbf{M} \in \mathbb{R}^{n\times n}$, $\mathbf{M}$ symmetric, $\|\mathbf{M}\|_S = \max\limits_{1\leq i\leq n} |\lambda_i|$ is called the spectral norm.

- It holds that $\mathrm{tr}(\mathbf{A}\mathbf{B}) = \mathrm{tr}(\mathbf{B}\mathbf{A})$, $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times m}$.
- $\mathrm{tr}(\mathbf{M}) = \sum_{i=1}^n \lambda_i(\mathbf{M})$, $\det(\mathbf{M}) = \prod_{i=1}^n \lambda_i(\mathbf{M})$.

  A simple proof of this statement uses the spectral decomposition of $\mathbf{M}$, for symmetric $\mathbf{M}$. First using the invariance property of trace under matrix commutation, we have

$$\begin{aligned}\mathrm{tr}(\mathbf{M}) &= \mathrm{tr}(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T)\\ &= \mathrm{tr}(\mathbf{\Lambda}\mathbf{V}^T\mathbf{V})\\ &\overset{(a)}{=} \mathrm{tr}(\mathbf{\Lambda}\mathbf{I}_n)\\ &= \mathrm{tr}(\mathbf{\Lambda}) = \sum_{i=1}^n \lambda_i(\mathbf{M}),\end{aligned}$$

  where (a) follows from the fact that $\mathbf{V}$ is an orthogonal matrix. Similarly, the spectral decomposition of symmetric matrix $\mathbf{M}$ can be used to prove the respective statement for the determinant.

$$\begin{aligned}\det(\mathbf{M}) &= \det(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T)\\ &= \det(\mathbf{\Lambda})\det(\mathbf{V}^T)\det(\mathbf{V})\\ &\overset{(b)}{=} \det(\mathbf{\Lambda}) = \prod_{i=1}^n \lambda_i(\mathbf{M}),\end{aligned}$$

  where (b) follows from the fact that the determinant of an orthogonal matrix is either $+1$ or -1.

Theorem 2.4 (Ky Fan, 1950 ([Fan50]))**.** *Let $\mathbf{M} \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\lambda_1(\mathbf{M}) \geq \cdots \geq \lambda_n(\mathbf{M})$ and let $k \leq n$. We have:*

$$\max_{\mathbf{V}\in\mathbb{R}^{n\times k}, \mathbf{V}^T\mathbf{V}=\mathbf{I}_k} \mathrm{tr}(\mathbf{V}^T\mathbf{M}\mathbf{V}) = \sum_{i=1}^k \lambda_i(\mathbf{M}).$$

and

$$\min_{\mathbf{V}\in\mathbb{R}^{n\times k}, \mathbf{V}^T\mathbf{V}=\mathbf{I}_k} \mathrm{tr}(\mathbf{V}^T\mathbf{M}\mathbf{V}) = \sum_{i=1}^k \lambda_{n-i+1}(\mathbf{M}).$$

Proof. First of all, see that $\mathbf{V} = [\mathbf{v}_1 \dots \mathbf{v}_k]$ where $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$ are orthonormal vectors. If the span of orthonormal vectors $\{\mathbf{w}_1, \dots, \mathbf{w}_k\}$ is same as the span of $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$, then for $\mathbf{W} = [\mathbf{w}_1 \dots \mathbf{w}_k]$, there is a unitary matrix $\mathbf{A} \in \mathbb{R}^{k \times k}$, which is a basis changing matrix, such that $\mathbf{V} = \mathbf{WA}$. We have:

$$\text{tr}(\mathbf{V}^T\mathbf{MV}) = \text{tr}(\mathbf{A}^T\mathbf{W}^T\mathbf{MWA}) = \text{tr}(\mathbf{W}^T\mathbf{MWAA}^T) = \text{tr}(\mathbf{W}^T\mathbf{MW}).$$

Also note that since $\text{tr}(\mathbf{V}^T\mathbf{MV}) = \sum_{i=1}^{k} \mathbf{v}_i^T\mathbf{Mv}_i$:

$$\sum_{i=1}^{k} \mathbf{v}_i^T\mathbf{Mv}_i = \sum_{i=1}^{k} \mathbf{w}_i^T\mathbf{Mw}_i.$$

The proof follows an iterative procedure. Suppose that $\mathbf{u}_1, \dots, \mathbf{u}_n$ are eigenvectors of $\mathbf{M}$ corresponding to $\lambda_1(\mathbf{M}) \geq \dots \geq \lambda_n(\mathbf{M})$ and $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_n]$. For $k = 1$, every vector $\mathbf{v}$ can be written as $\mathbf{v} = a_1\mathbf{u}_1 + \dots + a_n\mathbf{u}_n = \mathbf{Ua}$, where $\mathbf{a} = [a_1 \dots a_n]^T$. We have:

$$\max_{\mathbf{v} \in \mathbb{R}^n, \mathbf{v}^T\mathbf{v}=1} \mathbf{v}^T\mathbf{Mv} = \max_{\mathbf{a} \in \mathbb{R}^n, \mathbf{a}^T\mathbf{a}=1} \mathbf{a}^T\mathbf{U}^T\mathbf{MUa} = \max_{\mathbf{a} \in \mathbb{R}^n, \mathbf{a}^T\mathbf{a}=1} \sum_{i=1}^{n} \lambda_i(\mathbf{M}) a_i^2.$$

Therefore for $k = 1$, $\max\limits_{\mathbf{v} \in \mathbb{R}^n, \mathbf{v}^T\mathbf{v}=1} \mathbf{v}^T\mathbf{Mv} = \lambda_1(\mathbf{M})$.

For $k = 2$, see that for each orthonormal vector $\{\mathbf{v}_1, \mathbf{v}_2\}$, one can find two orthonormal vectors $\{\mathbf{v}_1^*, \mathbf{v}_2^*\}$ with the same span so that $\mathbf{v}_2^*$ is inside the span of $\mathbf{u}_2, \dots, \mathbf{u}_n$. First of all, it can be seen that:

$$\sum_{i=1}^{2} \mathbf{v}_i^T\mathbf{Mv}_i = \sum_{i=1}^{2} (\mathbf{v}_i^*)^T\mathbf{Mv}_i^*.$$

Since $\mathbf{v}_2^*$ is inside the span of $\mathbf{u}_2, \dots, \mathbf{u}_n$, it can be written as $\mathbf{v}^* = a_2\mathbf{u}_2 + \dots + a_n\mathbf{u}_n$, where $\mathbf{a} = [a_2 \dots a_n]^T$. It can be seen that:

$$(\mathbf{v}_2^*)^T\mathbf{Mv}_2^* = \sum_{i=2}^{n} \lambda_i(\mathbf{M}) a_i^2 \leq \lambda_2(\mathbf{M}).$$

Moreover from the previous step $(\mathbf{v}_1^*)^T\mathbf{Mv}_1^* \leq \lambda_1(\mathbf{M})$. Hence:

$$\sum_{i=1}^{2} \mathbf{v}_i^T\mathbf{Mv}_i \leq \lambda_1(\mathbf{M}) + \lambda_2(\mathbf{M}).$$

The upper bound is achievable by choosing $\mathbf{v}_1 = \mathbf{u}_1$ and $\mathbf{v}_2 = \mathbf{u}_2$. The procedure goes on iteratively. For a given k, the space spanned by $\mathbf{v}_1, \dots, \mathbf{v}_k$ has a $(k-1)$-dimensional subspace in intersection with the span of $\mathbf{u}_2, \dots, \mathbf{u}_n$. Find an orthonormal basis for this subspace $\mathbf{v}_2^*, \dots, \mathbf{v}_k^*$ and extend it with another $\mathbf{v}_1^*$ to an orthonormal basis for the space spanned by $\mathbf{v}_1, \dots, \mathbf{v}_k$. Then the sum $\sum_{i=2}^{k} (\mathbf{v}_i^*)^T\mathbf{Mv}_i^*$ is at most $\lambda_2(\mathbf{M}) + \dots + \lambda_k(\mathbf{M})$ and $\mathbf{v}_1^{*T}\mathbf{Mv}_1^* \leq \lambda_1(\mathbf{M})$. Therefore:

$$\sum_{i=1}^{k} \mathbf{v}_i^T\mathbf{Mv}_i \leq \lambda_1(\mathbf{M}) + \dots + \lambda_k(\mathbf{M}),$$

where the upper bound is achievable using $\mathbf{v}_i = \mathbf{u}_i$ for $i = 1, \ldots, k$.

□

The special case of above statements for $k = 1$ writes as:

$$\max_{\|\mathbf{v}\|=1, \mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{M}\mathbf{v} = \lambda_{\max}(\mathbf{M})$$

$$\min_{\|\mathbf{v}\|=1, \mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{M}\mathbf{v} = \lambda_{\min}(\mathbf{M}).$$

Note that:

$$\max_{\|\mathbf{v}\|=1, \mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{M}\mathbf{v} = \max_{\mathbf{v}\neq 0\in\mathbb{R}^n} \frac{\mathbf{v}^T\mathbf{M}\mathbf{v}}{\mathbf{v}^T\mathbf{v}}.$$

Theorem 2.5. *Given $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$, symmetric with eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n$ and $\mu_1 \geq \cdots \geq \mu_n$, respectively. Then*

$$\sum_{i=1}^{n} \lambda_i \mu_{n-i+1} \leq \mathrm{tr}(\mathbf{AB}) \leq \sum_{i=1}^{n} \lambda_i \mu_i.$$

Let $\lambda^+ = \max\{\lambda, 0\}$ denote the positive part of $\lambda \in \mathbb{R}$.

Theorem 2.6. *Given $\mathbf{M} \in \mathbb{R}^{n\times n}$ symmetric with spectral decomposition $\mathbf{M} = \mathbf{V}\mathrm{diag}(\lambda_1, \ldots, \lambda_n)\mathbf{V}^T$, $\lambda_1 \geq \cdots \geq \lambda_n$. Then*

$$\min_{\mathbf{A}\succeq 0, \mathrm{rk}(\mathbf{A})\leq k} \|\mathbf{M} - \mathbf{A}\|_F^2$$

is attained at $\mathbf{A}^* = \mathbf{V}\mathrm{diag}(\lambda_1^+, \ldots, \lambda_k^+, 0, \ldots, 0)\mathbf{V}^T$ with optimum value $\sum_{i=1}^{k}(\lambda_i - \lambda_i^+)^2 + \sum_{i=k+1}^{n} \lambda_i^2$.

Proof.

$$\begin{aligned}
\|\mathbf{M} - \mathbf{A}\|^2 &= \|\mathbf{M}\|^2 - 2\mathrm{tr}(\mathbf{MA}) + \|\mathbf{A}\|^2 \\
&\geq \sum_{i=1}^{n} \lambda_i^2 - 2\sum_{i=1}^{n} \lambda_i\mu_i + \sum_{i=1}^{n} \mu_i^2 \\
&= \sum_{i=1}^{n} (\lambda_i - \mu_i)^2 \\
&= \sum_{i=1}^{k} (\lambda_i - \mu_i)^2 + \sum_{i=k+1}^{n} (\lambda_i - 0)^2 \\
&\geq \sum_{i=1}^{k} (\lambda_i - \lambda_i^+)^2 + \sum_{i=k+1}^{n} \lambda_i^2.
\end{aligned}$$

Lower bound is attained if $\mathbf{A} = \mathbf{V}\mathrm{diag}(\lambda_1^+, \ldots, \lambda_k^+, 0, \ldots, 0)\mathbf{V}^T$. □

Definition 2.7. (Löwner semi-ordering) Given $\mathbf{V}, \mathbf{W} \succeq 0$. Define $\mathbf{V} \preceq \mathbf{W}$ if and only if $\mathbf{W} - \mathbf{V} \succeq 0$. It can be shown that the relation "$\preceq$" imposes a semi-ordering on the set of non-negative definite matrices, i.e., it satisfies the following properties

- (reflexive) $\mathbf{V} \preceq \mathbf{V}$
- (anti-symmetric) $\mathbf{V} \preceq \mathbf{W}$ and $\mathbf{W} \preceq \mathbf{V} \implies \mathbf{V} = \mathbf{W}$
- (transitive) $\mathbf{U} \preceq \mathbf{V}$ and $\mathbf{V} \preceq \mathbf{W} \implies \mathbf{U} \preceq \mathbf{W}$.

Theorem 2.8. *Let* $\mathbf{V}$ *and* $\mathbf{W}$ *be two* $n \times n$ *non-negative definite matrices, such that* $\mathbf{V} = (v_{ij}) \preceq \mathbf{W} = (w_{ij})$*, with the eignevalues as:*

- $\lambda_1(\mathbf{V}) \geq \cdots \geq \lambda_n(\mathbf{V})$,
- $\lambda_1(\mathbf{W}) \geq \cdots \geq \lambda_n(\mathbf{W})$

(a) $\lambda_i(\mathbf{V}) \leq \lambda_i(\mathbf{W})$*, for* $i = 1, \ldots, n$

(b) $v_{ii} \leq w_{ii}$*, for* $i = 1, \ldots, n$

(c) $v_{ii} + v_{jj} - 2v_{ij} \leq w_{ii} + w_{jj} - 2w_{ij}$

(d) $\mathrm{tr}(\mathbf{V}) \leq \mathrm{tr}(\mathbf{W})$

(e) $\det(\mathbf{V}) \leq \det(\mathbf{W})$

Proof. Exercise. □

2.1 Projection and Isometry

Definition 2.9. The matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$ is called a projection matrix, or idempotent, if $\mathbf{Q}^2 = \mathbf{Q}$. It is additionally called an orthogonal projection if additionally $\mathbf{Q}^T = \mathbf{Q}$.

The linear transformation $\mathbf{Q}$ maps onto $\mathrm{Im}(\mathbf{Q})$, a $k-$dimensional subspace of $\mathbb{R}^n$. Let $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{y} = \mathbf{Q}\mathbf{x} \in \mathrm{Im}(\mathbf{Q})$. Since $\mathbf{Q}$ is the projection matrix, $\mathbf{Q}\mathbf{y} = \mathbf{y}$. For an orthogonal projection, $\mathbf{x} - \mathbf{Q}\mathbf{x}$ is orthogonal to all vectors $\mathbf{y}$ in $\mathrm{Im}(\mathbf{Q})$ for every $\mathbf{x} \in \mathbb{R}^n$. To see this, note that there is a vector $\mathbf{z} \in \mathbb{R}^n$ such that $\mathbf{y} = \mathbf{Q}\mathbf{z}$. Then we have:

$$\mathbf{y}^T(\mathbf{x} - \mathbf{Q}\mathbf{x}) = \mathbf{z}^T\mathbf{Q}^T(\mathbf{x} - \mathbf{Q}\mathbf{x}).$$

Since for an orthogonal projection $\mathbf{Q}^T = \mathbf{Q}$ then:

$$\mathbf{z}^T\mathbf{Q}^T(\mathbf{x} - \mathbf{Q}\mathbf{x}) = \mathbf{z}^T\mathbf{Q}(\mathbf{x} - \mathbf{Q}\mathbf{x}) = \mathbf{z}^T(\mathbf{Q}\mathbf{x} - \mathbf{Q}^2\mathbf{x}) = \mathbf{z}^T(\mathbf{Q}\mathbf{x} - \mathbf{Q}\mathbf{x}) = 0.$$

Therefore $\mathbf{y}^T(\mathbf{x} - \mathbf{Q}\mathbf{x}) = 0$ and $\mathbf{x} - \mathbf{Q}\mathbf{x}$ is orthogonal to $\mathbf{y}$.

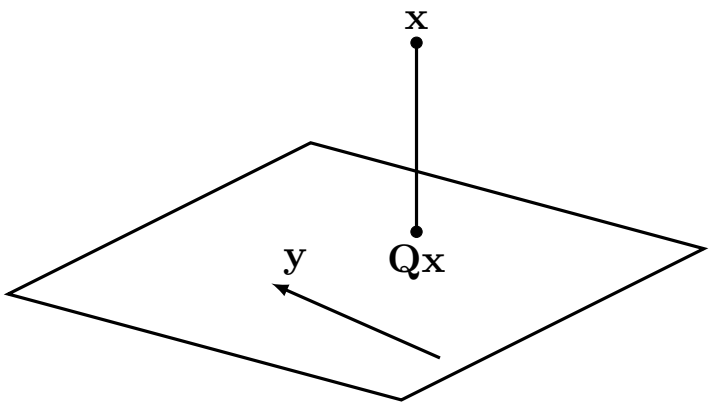


Figure 2.1: Orthogonal Projection

Lemma 2.10. *Let* $\mathbf{M} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ *be the spectral decomposition of* $\mathbf{M} \in \mathbb{R}^{n\times n}$ *and symmetric. For* $k \leq n$*, the matrix* $\mathbf{Q} = \sum_{i=1}^{k} \mathbf{v}_i\mathbf{v}_i^T$ *is an orthogonal projection onto* $\mathrm{Im}(\mathbf{Q}) = \langle \mathbf{v}_1, \ldots, \mathbf{v}_k \rangle$.

Proof. For $\mathbf{x} \in \mathbb{R}^n$, we have:

$$\mathbf{Q}\mathbf{x} = \sum_{i=1}^{k} \mathbf{v}_i\mathbf{v}_i^T x = \sum_{i=1}^{k} (\mathbf{v}_i^T\mathbf{x})\mathbf{v}_i = \sum_{i=1}^{k} \gamma_i\mathbf{v}_i \in \mathrm{Im}(\mathbf{Q}).$$

Moreover:

$$\mathbf{Q}^2 = (\sum_{i=1}^{k} \mathbf{v}_i\mathbf{v}_i^T)(\sum_{i=1}^{k} \mathbf{v}_i\mathbf{v}_i^T) = \sum_{i=1}^{k} \mathbf{v}_i\mathbf{v}_i^T = \mathbf{Q}.$$

Finally $\mathbf{Q}$ is symmetric and therefore it is an orthogonal projection. □

- *Let* $\mathbf{Q}$ *be an orthogonal projection on* $\mathrm{Im}(\mathbf{Q})$*, Then* $\mathbf{I}-\mathbf{Q}$*is an orthonormal projection onto* $\ker(\mathbf{Q})$.
  $\ker(\mathbf{Q})$ denotes the kernel of $\mathbf{Q}$, and $\mathrm{Im}(\mathbf{Q})$ denotes the image of $\mathbf{Q}$. First we verify the condition for a matrix to be a projection matrix:

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

  Therefore $\mathbf{I}-\mathbf{Q}$ is a projection matrix. Since $\mathbf{Q}$ is symmetric, so is $\mathbf{I}-\mathbf{Q}$ and hence an orthogonal projection. For the next par, let $\mathbf{y} \in \ker(\mathbf{Q})$, i.e., $\mathbf{Q}\mathbf{y} = \mathbf{0}$. Then:

  $$(\mathbf{I}-\mathbf{Q})\mathbf{y} = \mathbf{y} - \mathbf{Q}\mathbf{y} = \mathbf{y} \in \mathrm{Im}(\mathbf{I}-\mathbf{Q}).$$

  Therefore $\ker(\mathbf{Q}) \subseteq \mathrm{Im}(\mathbf{I}-\mathbf{Q})$. On the other hand, suppose that $y \in \mathrm{Im}(\mathbf{I}-\mathbf{Q})$. There is $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{y} = (\mathbf{I}-\mathbf{Q})\mathbf{x}$. We have:

  $$\mathbf{Q}\mathbf{y} = \mathbf{Q}(\mathbf{I}-\mathbf{Q})\mathbf{x} = \mathbf{Q}\mathbf{x} - \mathbf{Q}^2\mathbf{x} = \mathbf{Q}\mathbf{x} - \mathbf{Q}\mathbf{x} = \mathbf{0}.$$

  So $\mathbf{y} \in \ker(\mathbf{Q})$ and therefore $\mathrm{Im}(\mathbf{I}-\mathbf{Q}) \subseteq \ker(\mathbf{Q})$. So $\mathrm{Im}(\mathbf{I}-\mathbf{Q}) = \ker(\mathbf{Q})$.

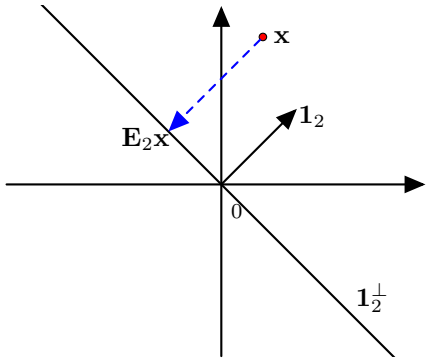


Figure 2.2: Orthogonal Projection of $\mathbf{E}_2$

- *Define* $\mathbf{E}_n$ *as follows:*

$$\mathbf{E}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_{n\times n} = \begin{bmatrix} 1-\frac{1}{n} & -\frac{1}{n} & \dots & -\frac{1}{n} \\ -\frac{1}{n} & 1-\frac{1}{n} & \dots & -\frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{n} & -\frac{1}{n} & \dots & 1-\frac{1}{n} \end{bmatrix}$$

Then $\mathbf{E}_n$ *is an orthogonal projection onto* $\mathbf{1}_n^\perp = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{1}_n^T\mathbf{x} = 0\}$ *where* $\mathbf{1}_n$ *is all-one-vector in* $\mathbb{R}^n$.
See that for all $\mathbf{x} \in \mathbb{R}^n$:

$$\mathbf{1}_n^T\mathbf{E}_n\mathbf{x} = \mathbf{1}_n^T(\mathbf{I}_n - \frac{1}{n}\mathbf{1}_{n\times n})\mathbf{x} = (\mathbf{1}_n^T - \mathbf{1}_n^T)\mathbf{x} = \mathbf{0}.$$

Therefore each vector in $\mathrm{Im}(\mathbf{E}_n)$ is orthogonal to $\mathbf{1}_n$.

Note that $\frac{1}{n}\mathbf{1}_{n\times n} \times \frac{1}{n}\mathbf{1}_{n\times n} = \frac{1}{n}\mathbf{1}_{n\times n}$ and $\frac{1}{n}\mathbf{1}_{n\times n}$ is symmetric. Therefore it is an orthogonal projection. Moreover its image is a one dimensional subspace spanned by $\mathbf{1}_n$. From the previous item, $\mathbf{I}_n - \frac{1}{n}\mathbf{1}_{n\times n}$ is also an orthogonal projection onto the kernel of $\frac{1}{n}\mathbf{1}_{n\times n}$ which is $\mathbf{1}_n^\perp$.

Theorem 2.11 (Inverse and determinant of partitioned matrix)**.** *Let* $\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix}$ *be a symmetric, invertible (regular) and* $\mathbf{A}$ *is also invertible (regular). Then:*

(a) The inverse matrix of $\mathbf{M}$ *is given by:*

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{F}\mathbf{E}^{-1}\mathbf{F}^T & -\mathbf{F}\mathbf{E}^{-1} \\ -\mathbf{E}^{-1}\mathbf{F}^T & \mathbf{E}^{-1} \end{bmatrix}$$

where $\mathbf{E}$ *is the Schur complement given by* $\mathbf{E} = \mathbf{C} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}$ *and* $\mathbf{F} = \mathbf{A}^{-1}\mathbf{B}$.

(b) The determinant of $\mathbf{M}$ *is given by:*

$$\det(\mathbf{M}) = \det(\mathbf{A})\det(\mathbf{C} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}).$$

There is also an extension of this theorem for general case where $\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}$ (see [Mur12, p.118]).

Definition 2.12 (Isometry)**.** A linear transformation $\mathbf{M} : \mathbb{R}^n \to \mathbb{R}^n$ is called an isometry if $\mathbf{x}^T\mathbf{x} = (\mathbf{M}\mathbf{x})^T(\mathbf{M}\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$.

Some properties of isometries are as follows:

- If $\mathbf{U}$ and $\mathbf{V}$ are isometries, then the product $\mathbf{UV}$ is also an isometry.
- If $\mathbf{U}$ is an isometry, $|\det(\mathbf{U})| = 1$.
- If $\mathbf{U}$ is an isometry, then $|\lambda(\mathbf{U})| = 1$ for all eigenvalues of $\mathbf{U}$.

3 Multivariate Distributions and Moments

The term *Probability* is used in our everyday life. In an experiment of tossing a fair coin for example, the probability it lands on head is 0.5. What does that mean? One explanation known as the Bayesian interpretation, it represents the probability as a measure of uncertainty about something [Mur12]. In other words, it is related to our information regarding the considered experiment. Different concepts and mathematical explanations regarding probabilities are presented in this chapter.

3.1 Random Vectors

Let $X_1, \ldots, X_n, n \in \mathbb{N}$ be random variables on the same probability space $(\Omega, \mathcal{F}, \mathbb{P})$:

$$X_i : (\Omega, \mathcal{F}, \mathbb{P}) \to (\mathbb{R}, \mathcal{R}), \quad i = 1, \ldots, p$$

where $\mathcal{R}$ is the Borel σ-algebra generated by the open sets of $\mathbb{R}$.

- The vector $\mathbf{X} = (X_1, \ldots, X_p)^T$ is called a random vector.
- Analogously, the matrix $\mathbf{X} = (X_{ij})_{\substack{1 \le i \le p, \\ 1 \le j \le n}}$, composed of the random variables X_{ij} as its elements, is called a random matrix.
- The joint distribution of a random vector is uniquely described by its multivariate distribution function:

  $$F(x_1, \ldots, x_p) = \mathbb{P}(X_1 \le x_1, \ldots, X_p \le x_p), \quad \text{where } (x_1, \ldots, x_p)^T \in \mathbb{R}^p,$$

  and x_i is a realization of X_i, with $i = 1, \ldots, p$.
- A random vector $\mathbf{X} = (X_1, \ldots, X_p)^T$ is called absolutely continuous if there exists an integrable function $f(x_1, \ldots, x_n) \ge 0$ such that:

  $$F(x_1, \ldots, x_p) = \int_{-\infty}^{x_p} \cdots \int_{-\infty}^{x_1} f(x_1, \ldots, x_p) \mathrm{d}x_1 \ldots \mathrm{d}x_p.$$

  where f is the probability density function (pdf) and F is the cumulative distribution function (cdf).

Example. (Multivariate normal distribution) The multivariate normal (or Gaussian) distribution of the random vector $\mathbf{X} \in \mathbb{R}^p$ has the following pdf:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\},$$

where $\mathbf{x} = (x_1, \ldots, x_p)^T \in \mathbb{R}^p$, and the parameters: $\boldsymbol{\mu} \in \mathbb{R}^p$, $\boldsymbol{\Sigma} \in \mathbb{R}^{p \times p}$, where $\boldsymbol{\Sigma} \succ 0$. This pdf can be denoted by $\mathbf{X} = (X_1, \ldots, X_p)^T \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Note that $\boldsymbol{\Sigma}$ must have full rank. There exists an $n-$dimensional Gaussian random variable, if $\mathrm{rk}(\boldsymbol{\Sigma}) < p$, however it has no density function with respect to $p-$ dimensional Lebesgue measure λ^p.

3.2 Expectation and Covariance

Definition 3.1. Given a random variable $\mathbf{X} = (X_1, \ldots, X_p)^T$.

(a) The expectation (vector) of $\mathbf{X}$, $\mathbb{E}(\mathbf{X})$, is defined by:

$$\mathbb{E}(\mathbf{X}) = (\mathbb{E}(X_1), \ldots, \mathbb{E}(X_p))^T .$$

(b) The covariance matrix of $\mathbf{X}$, $\mathrm{Cov}(\mathbf{X})$, is defined by:

$$\mathrm{Cov}(\mathbf{X}) = \mathbb{E}\left((\mathbf{X} - \mathbb{E}(\mathbf{X}))(\mathbf{X} - \mathbb{E}(\mathbf{X}))^T\right).$$

The expectation vector $\mathbb{E}(\mathbf{X})$ is constructed component-wise of $\mathbb{E}(X_i)$, where $i = 1, \ldots, p$. Furthermore, the covariance matrix has the covariance value $\mathrm{Cov}(X_i, X_j)$ as its (i,j)-th element given by

$$(\mathrm{Cov}(\mathbf{X}))_{i,j} = \mathrm{Cov}(X_i, X_j) = \mathbb{E}\left((X_i - \mathbb{E}(X_i))(X_j - \mathbb{E}(X_j))\right).$$

Theorem 3.2. *Given the random vectors $\mathbf{X} = (X_1, \ldots, X_p)^T$, and $\mathbf{Y} = (Y_1, \ldots, Y_p)^T$, the following statements hold:*

(a) $\mathbb{E}(\mathbf{AX} + \mathbf{b}) = \mathbf{A}\mathbb{E}(\mathbf{X}) + \mathbf{b}$

(b) $\mathbb{E}(\mathbf{X} + \mathbf{Y}) = \mathbb{E}(\mathbf{X}) + \mathbb{E}(\mathbf{Y})$

(c) $\mathrm{Cov}(\mathbf{AX} + \mathbf{b}) = \mathbf{A}\mathrm{Cov}(\mathbf{X})\mathbf{A}^T$

(d) $\mathrm{Cov}(\mathbf{X} + \mathbf{Y}) = \mathrm{Cov}(\mathbf{X}) + \mathrm{Cov}(\mathbf{Y})$*, if* $\mathbf{X}$ *and* $\mathbf{Y}$ *are stochastically independent.*

(e) $\mathrm{Cov}(\mathbf{X}) \succeq 0$*, i.e., the covariance matrix is non-negative definite.*

Proof. Prove (a)-(d) as exercise. Regarding e) assume that $\mathbf{a} \in \mathbb{R}^p$ be a vector, then

$$\mathbf{a}^T \mathrm{Cov}(\mathbf{X})\mathbf{a} \overset{(c)}{=} \mathrm{Cov}(\mathbf{a}^T\mathbf{X}) = \mathrm{Var}(\mathbf{a}^T\mathbf{X}) \geq 0.$$

□

Show as an exercise that if $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then

$$\mathbb{E}(\mathbf{X}) = \boldsymbol{\mu}, \quad \text{and} \quad \mathrm{Cov}(\mathbf{X}) = \boldsymbol{\Sigma}.$$

Theorem 3.3 (Steiner's rule)**.** *Given a random vector* $\mathbf{X} = (X_1, \ldots, X_p)^T$*, it holds that*

$$\mathbb{E}\left((\mathbf{X} - \mathbf{b})(\mathbf{X} - \mathbf{b})^T\right) = \mathrm{Cov}(\mathbf{X}) + (\mathbf{b} - \mathbb{E}(\mathbf{X}))(\mathbf{b} - \mathbb{E}(\mathbf{X}))^T, \qquad \forall\, \mathbf{b} \in \mathbb{R}^p.$$

Proof. Let $\boldsymbol{\mu} = \mathbb{E}(\mathbf{X})$. Note that

$$\mathbb{E}\left((\mathbf{X} - \boldsymbol{\mu})(\mathbf{b} - \boldsymbol{\mu})^T\right) = \mathbb{E}\left(\mathbf{X} - \boldsymbol{\mu}\right)\left(\mathbf{b} - \boldsymbol{\mu}\right)^T = 0,$$

and $\mathbb{E}(a) = a, \forall a \in \mathbb{R}^p$, then

$$\begin{aligned}
\mathbb{E}\left((\mathbf{X} - \boldsymbol{\mu} + \boldsymbol{\mu} - \mathbf{b})(\mathbf{X} - \boldsymbol{\mu} + \boldsymbol{\mu} - \mathbf{b})^T\right) &= \mathbb{E}\left((\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^T\right) + \mathbb{E}\left((\boldsymbol{\mu} - \mathbf{b})(\boldsymbol{\mu} - \mathbf{b})^T\right) \\
&\quad + \mathbb{E}\left((\mathbf{X} - \boldsymbol{\mu})(\boldsymbol{\mu} - \mathbf{b})^T\right) + \mathbb{E}\left((\boldsymbol{\mu} - \mathbf{b})(\mathbf{X} - \boldsymbol{\mu})^T\right) \\
&= \mathbb{E}\left((\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^T\right) + \mathbb{E}\left((\boldsymbol{\mu} - \mathbf{b})(\boldsymbol{\mu} - \mathbf{b})^T\right) \\
&= \mathrm{Cov}(\mathbf{X}) + (\mathbf{b} - \mathbb{E}(\mathbf{X}))(\mathbf{b} - \mathbb{E}(\mathbf{X}))^T,
\end{aligned}$$

where we used the linearity of expectation to show that

$$\mathbb{E}\left((\mathbf{X} - \boldsymbol{\mu})(\boldsymbol{\mu} - \mathbf{b})^T\right) = \left((\mathbb{E}(\mathbf{X}) - \boldsymbol{\mu})(\boldsymbol{\mu} - \mathbf{b})^T\right) = 0.$$

□

Theorem 3.4. *Let* $\mathbf{X}$ *be a random vector with* $\mathbb{E}(\mathbf{X}) = \boldsymbol{\mu}$ *and* $\mathrm{Cov}(\mathbf{X}) = \mathbf{V}$*. Then*

$$\mathbb{P}(\mathbf{X} \in \mathrm{Im}(\mathbf{V}) + \boldsymbol{\mu}) = 1.$$

Proof. Let $\ker(\mathbf{V}) = \{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{V}\mathbf{x} = 0\}$ be the kernel (or null space) of $\mathbf{V}$. Assume a basis as $\ker(\mathbf{V}) = \langle \mathbf{a}_1, \ldots, \mathbf{a}_r \rangle$. For $i = 1, \ldots, r$, it holds that

$$a_i^T \mathbf{V} a_i = \mathrm{Var}(a_i^T \mathbf{V}) = 0.$$

Hence, $\mathbf{a}_i^T \mathbf{X}$ should be almost surely equal to its expectation, namely, $\mathbb{E}(\mathbf{a}_i^T \mathbf{X}) = \mathbf{a}_i^T \boldsymbol{\mu}$. In other words,

$$\mathbb{P}(\mathbf{a}_i^T \mathbf{X} = \mathbf{a}_i^T \boldsymbol{\mu}) = 1, \text{ i.e., } \mathbb{P}(\mathbf{a}_i^T (\mathbf{X} - \boldsymbol{\mu}) = 0) = 1,$$

and

$$\mathbb{P}(\mathbf{X} - \boldsymbol{\mu} \in \mathbf{a}_i^{\perp}) = 1.$$

Given the fact that for an arbitrary random variable Z and two closed sets A and B, the following expression is valid

$$\mathbb{P}(Z \in A) = \mathbb{P}(Z \in B) = 1 \implies \mathbb{P}(Z \in A \cap B) = 1, \tag{3.1}$$

it holds that

$$\mathbb{P}((\mathbf{X} - \boldsymbol{\mu}) \in \mathbf{a}_1^{\perp} \cap \cdots \cap \mathbf{a}_r^{\perp}) = 1.$$

However, given that $\mathrm{Im}(\mathbf{V}) = \ker(\mathbf{V})^{\perp} =< \mathbf{a}_1, \ldots, \mathbf{a}_r >^{\perp} = \mathbf{a}_1^{\perp} \cap \cdots \cap \mathbf{a}_r^{\perp}$. Therefore,

$$\mathbb{P}((\mathbf{X} - \boldsymbol{\mu}) \in \mathrm{Im}(\mathbf{V})) = 1.$$

Prove (3.1) as an exercise. □

3.3 Conditional Distribution

Let $\mathbf{X} = (X_1, \ldots, X_p)^T$ be a random vector such that $\mathbf{X} = (\mathbf{Y}_1, \mathbf{Y}_2)^T$ where $\mathbf{Y}_1 = (X_1, \ldots, X_k)$ and $\mathbf{Y}_2 = (X_{k+1}, \ldots, X_p)$. Suppose that $\mathbf{X}$ is absolutely continuous with density $f_{\mathbf{X}}$. Then the conditional density of $\mathbf{Y}_1$ given $\mathbf{Y}_2 = \mathbf{y}_2$ is denoted by

$$f_{\mathbf{Y}_1|\mathbf{Y}_2}(\mathbf{y}_1 \mid \mathbf{y}_2) = \frac{f_{\mathbf{Y}_1,\mathbf{Y}_2}(\mathbf{y}_1, \mathbf{y}_2)}{f_{\mathbf{Y}_2}(\mathbf{y}_2)},$$

where $\mathbf{y}_1 \in \mathbb{R}^k$ is a realization of $\mathbf{Y_1}$. Furthermore, it also holds that

$$\mathbb{P}(\mathbf{Y}_1 \in B \mid \mathbf{Y}_2 = \mathbf{y}_2) = \int_B f_{\mathbf{Y}_1|\mathbf{Y}_2}(\mathbf{y}_1 \mid \mathbf{y}_2)\mathrm{d}\mathbf{y}_1, \quad \forall B \in \mathcal{R}^k.$$

Theorem 3.5 ([Mur12, Theorem 4.3.1])**.** *Suppose that* $(\mathbf{Y}_1, \mathbf{Y}_2) \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *where*

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}, \quad \textit{and} \quad \boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \boldsymbol{\Lambda}_{11} & \boldsymbol{\Lambda}_{12} \\ \boldsymbol{\Lambda}_{21} & \boldsymbol{\Lambda}_{22} \end{bmatrix}.$$

Then

(a) The distribution of $\mathbf{Y}_1$ *and* $\mathbf{Y}_2$ *are given by* $\mathbf{Y}_1 \sim N_k(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11})$ *and* $\mathbf{Y}_2 \sim N_{p-k}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22})$*, respectively.*

(b) The conditional density $f_{\mathbf{Y}_1|\mathbf{Y}_2}(\mathbf{y}_1 \mid \mathbf{y}_2)$ *is given by the multivariate normal distribution* $f_{\mathbf{Y}_1|\mathbf{Y}_2}(\mathbf{y}_1 \mid \mathbf{y}_2) \sim N_k(\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2})$*. The parameters* $\boldsymbol{\mu}_{1|2}$ *and* $\boldsymbol{\Sigma}_{1|2}$ *are defined as*

$$\begin{aligned} \boldsymbol{\mu}_{1|2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{y}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}(\mathbf{y}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\Sigma}_{1|2}(\boldsymbol{\Lambda}_{11}\boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{12}(\mathbf{y}_2 - \boldsymbol{\mu}_2)), \end{aligned}$$

and

$$\boldsymbol{\Sigma}_{1|2} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} = \boldsymbol{\Lambda}_{11}^{-1}.$$

Note that $\boldsymbol{\Sigma}_{1|2}$ is the Schur complement, introduced in the previous chapter.

3.4 Maximum Likelihood Estimation

Suppose $\mathbf{x} = (x_1, \ldots, x_n)$ is a random sample from a pdf $f(x; \boldsymbol{\vartheta})$, where $\boldsymbol{\vartheta}$ is a parameter vector. The function $L(\mathbf{x}; \boldsymbol{\vartheta})$ is referred to as the likelihood function, and defined as

$$L(\mathbf{x}; \boldsymbol{\vartheta}) = \prod_{i=1}^{n} f(x_i; \boldsymbol{\vartheta}). \tag{3.2}$$

Furthermore, the function $\ell(\mathbf{x};\boldsymbol{\vartheta})$ represents the log-likelihood function, and is defined as

$$\ell(\mathbf{x};\boldsymbol{\vartheta}) = \log L(\mathbf{x};\boldsymbol{\vartheta}) = \sum_{i=1}^{n} \log f(x_i;\boldsymbol{\vartheta}). \tag{3.3}$$

For a given sample $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, one can notice that both functions in (3.2) and (3.3) depend on the parameters in $\boldsymbol{\vartheta}$. Therefore, $\boldsymbol{\vartheta}$ is needed to be determined such that it fits the data in $\mathbf{x}$ through $L(\mathbf{x};\boldsymbol{\vartheta})$, or equivalently $\ell(\mathbf{x};\boldsymbol{\vartheta})$. The estimate of $\boldsymbol{\vartheta}$, denoted by $\hat{\boldsymbol{\vartheta}}$, is obtained as

$$\hat{\boldsymbol{\vartheta}} = \arg\max_{\boldsymbol{\vartheta}} \ell(\mathbf{x};\boldsymbol{\vartheta})$$

and called the maximum likelihood estimate (MLE) of $\boldsymbol{\vartheta}$.

Theorem 3.6. *Let $\mathbf{x}_1, \ldots, \mathbf{x}_n, n \in \mathbb{N}$, be i.i.d samples obtained from the distribution $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The MLEs of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are given by*

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i = \overline{\mathbf{x}}, \quad \textit{and} \quad \hat{\boldsymbol{\Sigma}} = \mathbf{S}_n = \frac{1}{n}\sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T.$$

Proof. In order to prove this theorem, Steiner's Lemma is required at this point along with the following fundamentals of matrix differentiation. For arbitrary matrices $\mathbf{V}, \mathbf{A}$ and vector $\mathbf{y}$, the following statements, which to be proved as exercise, are considered afterwards for simplification purposes.

- $\frac{\partial}{\partial \mathbf{V}} \log\det \mathbf{V} = (\mathbf{V}^{-1})^T$, if $\mathbf{V}$ is invertible.
- $\frac{\partial}{\partial \mathbf{V}} \mathrm{tr}(\mathbf{V}\mathbf{A}) = \mathbf{A}^T$.
- $\frac{\partial(\mathbf{y}^T\mathbf{A}\mathbf{y})}{\partial \mathbf{y}} = (\mathbf{A} + \mathbf{A}^T)\mathbf{y}$.

Starting with the log-likelihood function

$$\ell(\mathbf{x}_1, \ldots, \mathbf{x}_n; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{n} \left(\log \frac{1}{(2\pi)^{p/2}} + \log \frac{1}{|\ \boldsymbol{\Sigma}|^{1/2}} - \frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \right),$$

in order to be maximized, the additive constants can be dropped, hence the log-likelihood function is defined by

$$\ell^*(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{p}{2} \log |\ \boldsymbol{\Sigma}^{-1}\ | - \frac{1}{2}\sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \tag{3.4}$$

By utilizing the previously presented fundamentals, and choosing $\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Lambda}$, (3.4) can be

reformulated as

$$\begin{aligned}\ell^*(\boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \frac{n}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu})^T\boldsymbol{\Lambda}(\mathbf{x}_i - \boldsymbol{\mu}) \\ &= \frac{n}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\sum_{i=1}^{n}\mathrm{tr}\left(\boldsymbol{\Lambda}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T\right) \\ &= \frac{n}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Lambda}\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T\right). \end{aligned} \tag{3.5}$$

Based on Steiner's rule, the summation term in (3.5) can be rewritten as

$$\begin{aligned}\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T &= \sum_{i=1}^{n}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T + n(\overline{\mathbf{x}} - \boldsymbol{\mu})(\overline{\mathbf{x}} - \boldsymbol{\mu})^T \\ &= n\mathbf{S}_n + n(\overline{\mathbf{x}} - \boldsymbol{\mu})(\overline{\mathbf{x}} - \boldsymbol{\mu})^T, \end{aligned} \tag{3.6}$$

hence $\ell^*(\boldsymbol{\mu}, \boldsymbol{\Lambda})$ is given by

$$\begin{aligned}\ell^*(\boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \frac{n}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Lambda}(n\mathbf{S}_n + (\overline{\mathbf{x}} - \boldsymbol{\mu})(\overline{\mathbf{x}} - \boldsymbol{\mu})^T)\right) \\ &= \frac{n}{2}\log|\boldsymbol{\Lambda}| - \frac{n}{2}\mathrm{tr}\left(\boldsymbol{\Lambda}\mathbf{S}_n\right) - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Lambda}(\overline{\mathbf{x}} - \boldsymbol{\mu})(\overline{\mathbf{x}} - \boldsymbol{\mu})^T\right). \end{aligned} \tag{3.7}$$

Based on (3.6), note that $n\mathbf{S}_n \preceq \sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$, with equality when $\overline{\mathbf{x}} = \boldsymbol{\mu}$. Thus, $\hat{\boldsymbol{\mu}} = \overline{\mathbf{x}}$, and (3.7) is formulated as

$$\ell^*(\boldsymbol{\Lambda}) = \frac{n}{2}\log|\,\boldsymbol{\Lambda}\,| - \frac{n}{2}\mathrm{tr}\left(\boldsymbol{\Lambda}\mathbf{S}_n\right). \tag{3.8}$$

Moreover, in order to find $\hat{\boldsymbol{\Lambda}}$, the derivative of log-likelihood function in (3.8) with respect to $\boldsymbol{\Lambda}$ is calculated to find its zeros, as

$$\frac{\partial\ell^*(\boldsymbol{\Lambda})}{\partial\boldsymbol{\Lambda}} = 0 \implies \frac{n}{2}\boldsymbol{\Lambda}^{-1} - \frac{n}{2}\mathbf{S}_n = 0,$$

hence $\hat{\boldsymbol{\Lambda}}^{-1} = \hat{\boldsymbol{\Sigma}} = \mathbf{S}_n$.
Another way to obtain similar results is by taking the partial derivative of $\ell^*(\boldsymbol{\mu}, \boldsymbol{\Lambda})$ with respect to $\boldsymbol{\mu}$ and subsequently with respect to $\boldsymbol{\Lambda}$ to find the function's zeros, as follows

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\mu}}\ell^*(\boldsymbol{\mu}, \boldsymbol{\Lambda}) = -\frac{1}{2}(\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T)(\overline{\mathbf{x}} - \boldsymbol{\mu}) = -\boldsymbol{\Lambda}(\overline{\mathbf{x}} - \boldsymbol{\mu}) = 0 \implies \hat{\boldsymbol{\mu}} = \overline{\mathbf{x}}. \\ \frac{\partial}{\partial\boldsymbol{\Lambda}}\ell^*(\boldsymbol{\mu} = \hat{\boldsymbol{\mu}}, \boldsymbol{\Lambda}) = \frac{n}{2}\boldsymbol{\Lambda}^{-1} - \frac{n}{2}\mathbf{S}_n - \frac{1}{2}(\overline{\mathbf{x}} - \boldsymbol{\mu})(\overline{\mathbf{x}} - \boldsymbol{\mu})^T = 0 \implies \hat{\boldsymbol{\Lambda}}^{-1} = \hat{\boldsymbol{\Sigma}} = \mathbf{S}_n.\end{aligned}$$

□

4 Dimensionality Reduction

Given high dimensional data, the goal of dimensionality reduction algorithms is to find the "optimal" way of representing this data in a low dimensional space. Assigning low dimensional vectors of dimensions 1,2 or 3 to the available high dimensional data allows its graphical representation (in a 1D, 2D, or 3D plot for instance). Furthermore, the notion of "optimal" low dimensional representation must be specified. In the following sections, we will learn different concepts on dimensionality reduction.

4.1 Principal Component Analysis (PCA)

In general, PCA is a tool that searches for a few linear combinations to represent the given data, losing as little information as possible. More specifically, given data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^p$, the goal is to

- find a $k-$dimensional subspace such that the projections of $\mathbf{x}_1, \ldots, \mathbf{x}_n$ thereon represent the original points on its best.
- find the $k-$dimensional projections that preserve as much variance as possible.

Both of above accounts are equivalent as we will see.
Given $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ independently sampled from some distribution, the sample mean $\overline{\mathbf{x}}$ and sample covariance matrix $\mathbf{S}_n$ are defined as follows:

$$\overline{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i, \quad \mathbf{S}_n = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T.$$

Note that the sample mean $\overline{\mathbf{x}}$ is an unbiased estimator of $\mathbb{E}(\mathbf{X})$, and sample covariance matrix $\mathbf{S}_n$ is an unbiased estimator of $\mathbf{\Sigma} = \mathrm{Cov}(\mathbf{X})$.

4.1.1 Optimal Projection

Consider the following optimization problem:

$$\min_{\substack{\mathbf{a}\in\mathbb{R}^p \\ \mathbf{Q}\in\mathcal{O}_n \\ \mathrm{rk}(\mathbf{Q})=k}} \sum_{i=1}^{n}\|\mathbf{x}_i - \mathbf{a} - \mathbf{Q}(\mathbf{x}_i - \mathbf{a})\|_F^2$$

where $\mathcal{O}_n$ is the space of $n \times n$ orthogonal projections. The idea is to find a shift vector $\mathbf{a}$ and an orthogonal projection $\mathbf{Q}$ on a $k-$dimensional subspace, such that the projection

points are closest to the original ones. We have:

$$\begin{aligned}
\min_{\mathbf{a},\mathbf{Q}} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{a} - \mathbf{Q}(\mathbf{x}_i - \mathbf{a})\|_F^2 &= \min_{\mathbf{a},\mathbf{Q}} \sum_{i=1}^{n} \|(\mathbf{I} - \mathbf{Q})(\mathbf{x}_i - \mathbf{a})\|_F^2 \\
&= \min_{\mathbf{a},\mathbf{R}} \sum_{i=1}^{n} \|\mathbf{R}(\mathbf{x}_i - \mathbf{a})\|_F^2 \\
&= \min_{\mathbf{a},\mathbf{R}} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{a})^T \mathbf{R}^T \mathbf{R} (\mathbf{x}_i - \mathbf{a}) \\
&= \min_{\mathbf{a},\mathbf{R}} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{a})^T \mathbf{R} (\mathbf{x}_i - \mathbf{a}) \\
&= \min_{\mathbf{a},\mathbf{R}} \sum_{i=1}^{n} \mathrm{tr}(\mathbf{R}(\mathbf{x}_i - \mathbf{a})(\mathbf{x}_i - \mathbf{a})^T)
\end{aligned}$$

Note that $\mathbf{R} = (\mathbf{I} - \mathbf{Q})$ is also an orthogonal projection, and hence $\mathbf{R}^T\mathbf{R} = \mathbf{R}^2 = \mathbf{R}$. Moreover using Steiner's lemma, we have:

$$\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \mathbf{a})(\mathbf{x}_i - \mathbf{a})^T = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T + (\overline{\mathbf{x}} - \mathbf{a})(\overline{\mathbf{x}} - \mathbf{a})^T$$

Hence:

$$\begin{aligned}
\min_{\mathbf{a},\mathbf{Q}} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{a} - \mathbf{Q}(\mathbf{x}_i - \mathbf{a})\|_F^2 &= \min_{\mathbf{a},\mathbf{R}} \sum_{i=1}^{n} \mathrm{tr}(\mathbf{R}(\mathbf{x}_i - \mathbf{a})(\mathbf{x}_i - \mathbf{a})^T) \\
&= \min_{\mathbf{a},\mathbf{R}} \mathrm{tr}(\mathbf{R} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{a})(\mathbf{x}_i - \mathbf{a})^T) \\
&\geq \min_{\mathbf{R}} \mathrm{tr}(\mathbf{R} \sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T) \\
&= \min_{\mathbf{R}} \mathrm{tr}(\mathbf{R}(n-1)\mathbf{S}_n) \\
&= \min_{\mathbf{Q}} (n-1)\mathrm{tr}(\mathbf{S}_n(\mathbf{I} - \mathbf{Q})).
\end{aligned}$$

It remains to solve:

$$\max_{\mathbf{Q}} \mathrm{tr}(\mathbf{S}_n \mathbf{Q}).$$

Since $\mathbf{Q}$ is an orthogonal projection matrix of rank k, it is non-negative definite, it can be written as $\mathbf{Q} = \sum_{i=1}^{k} \mathbf{q}_i \mathbf{q}_i^T$, where $\mathbf{q}_i$'s are orthonormal. Therefore if $\tilde{\mathbf{Q}} = (\mathbf{q}_1, \ldots, \mathbf{q}_k)$, the matrix $\mathbf{Q}$ can be written as $\tilde{\mathbf{Q}}\tilde{\mathbf{Q}}^T$. Therefore Ky Fan's theorem implies that

$$\max_{\mathbf{Q}} \mathrm{tr}(\mathbf{S}_n \mathbf{Q}) = \max_{\substack{\tilde{\mathbf{Q}} \\ \tilde{\mathbf{Q}}^T\tilde{\mathbf{Q}} = \mathbf{I}_k}} \mathrm{tr}(\tilde{\mathbf{Q}}^T \mathbf{S}_n \tilde{\mathbf{Q}}) = \sum_{i=1}^{k} \lambda_i(\mathbf{S}_n)\,,$$

where $\lambda_1(\mathbf{S}_n) \geq \cdots \geq \lambda_n(\mathbf{S}_n)$ are the eigenvalues of $\mathbf{S}_n$ in decreasing order. The maximum is attained if $\mathbf{q}_1, \ldots, \mathbf{q}_k$ are the orthonormal eigenvectors corresponding to $\lambda_1(\mathbf{S}_n) \geq \cdots \geq \lambda_k(\mathbf{S}_n)$.

4.1.2 Variance-Preserving Projection

The goal is to find the $k-$dimensional projection that preserves the most variance. This idea can be formulated as follows:

$$\begin{aligned}
\max_{\mathbf{Q}} \sum_{i=1}^{n} \|\mathbf{Q}\mathbf{x}_i - \frac{1}{n}\sum_{j=1}^{n}\mathbf{Q}\mathbf{x}_j\|^2 &= \max_{\mathbf{Q}} \sum_{i=1}^{n} \|\mathbf{Q}\mathbf{x}_i - \mathbf{Q}\overline{\mathbf{x}}\|^2 \\
&= \max_{\mathbf{Q}} \sum_{i=1}^{n} \|\mathbf{Q}(\mathbf{x}_i - \overline{\mathbf{x}})\|^2 \\
&= \max_{\mathbf{Q}} \sum_{i=1}^{n} (\mathbf{Q}(\mathbf{x}_i - \overline{\mathbf{x}}))^T \mathbf{Q}(\mathbf{x}_i - \overline{\mathbf{x}}) \\
&= \max_{\mathbf{Q}} \sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}})^T \mathbf{Q}^T\mathbf{Q}(\mathbf{x}_i - \overline{\mathbf{x}}) \\
&= \max_{\mathbf{Q}} \sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}})^T \mathbf{Q}(\mathbf{x}_i - \overline{\mathbf{x}}) \\
&= \max_{\mathbf{Q}} \sum_{i=1}^{n} \mathrm{tr}(\mathbf{Q}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T) \\
&= \max_{\mathbf{Q}} \mathrm{tr}(\mathbf{Q}\sum_{i=1}^{n}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T) \\
&= \max_{\mathbf{Q}} \mathrm{tr}((n-1)\mathbf{Q}\mathbf{S}_n)\,.
\end{aligned}$$

However the last optimization problem appeared also above and therefore following a similar solution, the optimal projection $\mathbf{Q}$ is equal to $\sum_{i=1}^{k} \mathbf{q}_i\mathbf{q}_i^T$ where $\mathbf{q}_1, \ldots, \mathbf{q}_k$ are the orthonormal eigenvectors corresponding to $\lambda_1(\mathbf{S}_n) \geq \cdots \geq \lambda_k(\mathbf{S}_n)$.

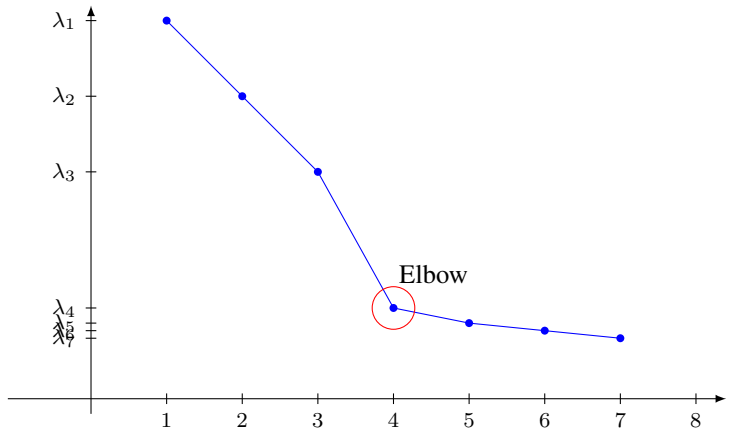


Figure 4.1: Scree Plot

4.1.3 How to carry out PCA

In order carry out PCA on the given $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ data points, we first fix $k \ll p$. Then, we proceed to the following steps

- Compute $\mathbf{S}_n = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T$. Find its spectral decomposition as $\mathbf{S}_n = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ where $\mathbf{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_p)$ with $\lambda_1 \geq \cdots \geq \lambda_p$ and $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_p) \in \mathcal{O}(p)$.
- $\mathbf{v}_1, \ldots, \mathbf{v}_k$ are called the k Principal eigenvectors to the principal eigenvalues $\lambda_1 \geq \cdots \geq \lambda_k$.
- Projected points are found by

$$\hat{\mathbf{x}}_i = \begin{bmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_k^T \end{bmatrix} \mathbf{x}_i, \qquad i = 1, \ldots, n\,.$$

Let us discuss computational complexity of PCA. Using the conventional method, discussed above, the complexity of constructing $\mathbf{S}_n$ is $O(np^2)$ [1] and the complexity of spectral decomposition is $O(p^3)$ [Ban08]. Therefore the computational complexity of both steps together are $O(\max\{np^2, p^3\})$.
However this can be improved. Assume $p < n$, then we write

$$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \quad \text{and} \quad \mathbf{S}_n = \frac{1}{n-1}(\mathbf{X} - \overline{\mathbf{x}}\mathbf{1}_n^T)(\mathbf{X} - \overline{\mathbf{x}}\mathbf{1}_n^T)^T\,.$$

[1] This is called Big-O notation or Bachmann-Landau notation. A function $f(n)$ is $O(g(n))$ if for some $n_0 > 0$ and a constant $c > 0$, $|f(n)| \leq c|g(n)|$ for $n \geq n_0$. For example, if an algorithm over n objects takes at most $n^2 + n$ time to run, then its complexity is $O(n^2)$.

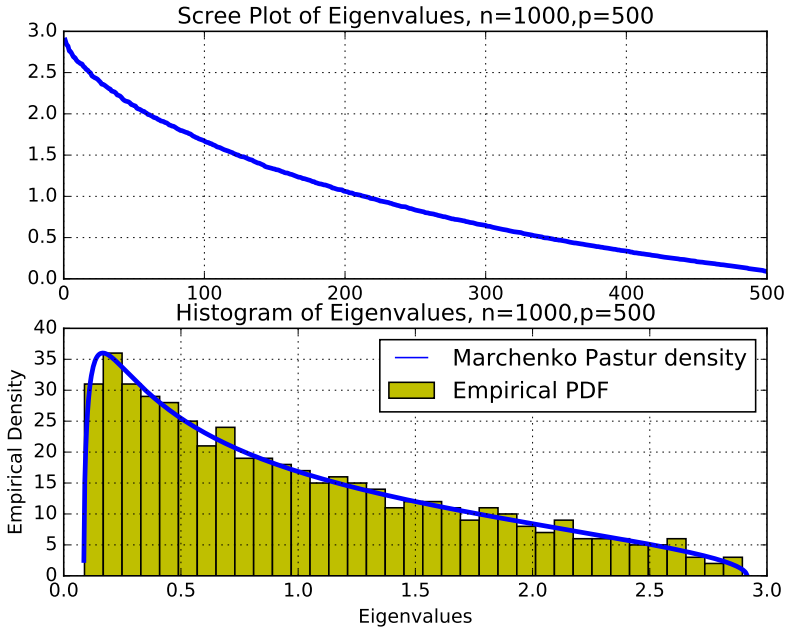


Figure 4.2: Eigenvalues of $\mathbf{S}_n$ and its scree plot

Consider singular value decomposition (SVD) of $\mathbf{X}-\overline{\mathbf{x}}\mathbf{1}_n^T = \mathbf{U}_{p\times p}\mathbf{D}\mathbf{V}_{p\times n}^T$ where $\mathbf{U} \in \mathcal{O}(p)$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}_p$, $\mathbf{D} = \text{diag}(\sigma_1, \ldots, \sigma_p)$. Using this decomposition, we have

$$\mathbf{S}_n = \frac{1}{n-1}\mathbf{U}\mathbf{D}\mathbf{V}^T\mathbf{V}\mathbf{D}\mathbf{U}^T = \frac{1}{n-1}\mathbf{U}\mathbf{D}^2\mathbf{U}^T.$$

Hence $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_p]$ contains the eigenvectors of $\mathbf{S}_n$. Computational complexity of finding SVD for $\mathbf{X} - \overline{\mathbf{x}}\mathbf{1}_n^T$ is given by $O(\min\{n^2p, p^2n\})$. However if one is only interested in top k eigenvectors, the cost reduces to $O(knp)$.
Another issue is about PCA is the choice of k. If the goal of PCA is data visualization, then $k = 2$ or $k = 3$ are reasonable choices. But PCA is also used for dimensionality reduction. In practice, it can happen that the data lies in a low dimensional subspace but it is corrupted by a high dimensional noise. Also, it is possible that some algorithms are computationally expensive to run on high dimensions and it makes sense to bring the data to lower dimensions and run the algorithm more efficiently on the lower dimensional space.
To choose proper k, one heuristic is to look at the scree plot or scree graph. The scree plot is the plot of ordered eigenvalues of $\mathbf{S}_n$. The scree graph was introduced by Raymond B. Cattell [Cat66]. It is a very subjective way of determining k. The idea is to find k from the plot such that the line through the points to the left of k is steep and the line through the points to the right of k is not steep. This looks like an elbow in the scree plot. In Figure 4.1, a scree plot is shown. The value of k can be chosen by recognizing an elbow in the graph of ordered eigenvalues.

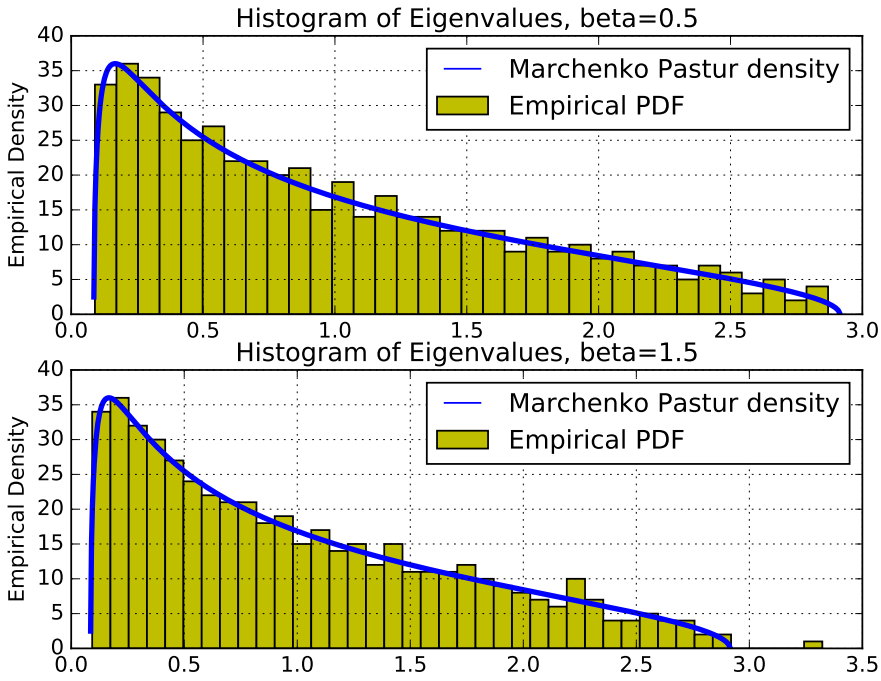


Figure 4.3: Eigenvalues of $\mathbf{S}_n$ for Spike model with $\beta = 1.5, 0.5$

4.1.4 Eigenvalue structure of S_n in high dimensions

Suppose that $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ are independent samples of a Gaussian random variable $\mathbf{X} \sim N_p(\mathbf{0}, \mathbf{\Sigma})$. Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$. Estimate $\mathbf{\Sigma}$ by $\mathbf{S}_n = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{n}\mathbf{X}\mathbf{X}^T$.
If p is fixed, from law of large numbers, $\mathbf{S}_n$ will tend to $\mathbf{\Sigma}$ as $n \to \infty$ almost everywhere. However if both n and p are large, then it is not clear anymore what the relation between $\mathbf{S}_n$ and $\mathbf{\Sigma}$ is. To see this, consider the case where $\mathbf{\Sigma} = \mathbf{I}$ [Ban08]. Figure 4.2 shows the scree plot and histogram of the eigenvalues for $n = 1000$ and $p = 500$. The plot shows that there are many eigenvalues bigger than 1 unlike $\mathbf{\Sigma} = \mathbf{I}$ which has all eigenvalues equal to one. Scree plot also implies that data lies on a low dimensional space which is also not true.
Following theorem is about distribution of eigenvalues of $\mathbf{S}_n$ when p and n are comparable.

Theorem 4.1 (Marchenko-Pastur, 1967)**.** *Let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be i.i.d. random vectors on $\mathbb{R}^p$ with $\mathbb{E}(\mathbf{X}_i) = 0$ and $\mathrm{Cov}(\mathbf{X}_i) = \sigma^2 \mathbf{I}_p$. Let $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n) \in \mathbb{R}^{p \times n}$ and $\mathbf{S}_n = \frac{1}{n}\mathbf{X}\mathbf{X}^T \in \mathbb{R}^{p \times p}$. Let $\lambda_1, \ldots, \lambda_p$ be the eigenvalues of $\mathbf{S}_n$. Suppose that $p, n \to \infty$ such that $\frac{p}{n} \to \gamma \in (0, 1]$ as $n \to \infty$. Then the sample distribution of $\lambda_1, \ldots, \lambda_p$ converges almost surely to the following density*

$$f_\gamma(u) = \frac{1}{2\pi\sigma^2 u\gamma}\sqrt{(b-u)(u-a)}, \quad a \le u \le b$$

with $a(\gamma) = \sigma^2(1 - \sqrt{\gamma})^2$ *and* $b(\gamma) = \sigma^2(1 + \sqrt{\gamma})^2$.

Proof. Refer to [Bai99] for various proofs. □

Marchenko-Pastur distribution is presented in Figure 4.2 by the blue curve.

Remark 2. If $\gamma > 1$, there will be a mass point at zero with probability $1 - \frac{1}{\gamma}$. Since $\gamma > 1$, then $n < p$. Moreover the rank of $\mathbf{S}_n = \frac{1}{n}\mathbf{X}\mathbf{X}^T$ will be at most $\min(p, n)$ which is $n < p$ in this case. This means that $\mathbf{S}_n$ is not full rank and zero is definitely one of the eigenvalues.

The theorem shows that there is a wide spread of spectrum of eigenvalues even in the case i.i.d. distributed random variables. The main question is to what degree PCA can recover low dimensional structure from the data. Is PCA useful at all?

4.1.5 Spike Models

Suppose that there is a low dimensional structure in data. Let us say that each sample results from a point on a one dimensional space with an additional high dimensional noise perturbation. The one dimensional part is modeled by $\sqrt{\beta}G\mathbf{v}$ where $\mathbf{v}$ is a unit norm vector in $\mathbb{R}^p$, β is a non-negative constant and G is the standard normal random variable. The high dimensional noise is modeled by $\mathbf{U} \sim N_p(0, \mathbf{I}_p)$. Therefore the samples are $\mathbf{X}_i = \mathbf{U}_i + \sqrt{\beta}G_i\mathbf{v}$ with $\mathbb{E}(\mathbf{X}_i) = 0$. Since G_i and $\mathbf{U}_i$ are independent, using Theorem 3.2, we have that

$$\mathrm{Cov}(\mathbf{X}_i) = \mathrm{Cov}(\mathbf{U}_i) + \mathrm{Cov}(\sqrt{\beta}G_i\mathbf{v}) = \mathbf{I}_p + \mathbf{v}\mathrm{Cov}(\sqrt{\beta}G_i)\mathbf{v}^T = \mathbf{I}_p + \beta\mathbf{v}\mathbf{v}^T.$$

Suppose that $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are i.i.d. distributed with $\mathrm{Cov}(\mathbf{X}_i) = \mathbf{I}_p + \beta\mathbf{v}\mathbf{v}^T$. Let us look at distribution of eigenvalues for some numerical examples. Figure 4.3 shows the distribution of eigenvalues for $\beta = 1.5$ and $\beta = 0.5$ and $p = 500$ and $n = 1000$, and $\mathbf{v} = \mathbf{e}_1$. It can be seen that all eigenvalues appear inside the interval proposed by Marchenko-Pastur distribution when $\beta = 0.5$. However, the situation is different when $\beta = 1.5$. One eigenvalue pops out of the interval in this case. Note that in general the maximum eigenvalue of $\mathbf{I}_p + \beta\mathbf{e}_1\mathbf{e}_1^T$ is 1+1.5 which is 2.5, and all other eigenvalues are 1.
The question is whether there is a threshold for β above which we will see one eigenvalue popping out. The following theorem provides the transition point known as BPP (Baik, Ben Arous and Péché) transition.

Theorem 4.2 ([BAP05])**.** *Let* $\mathbf{X}_1, \ldots, \mathbf{X}_n$ *be i.i.d. random vectors on* $\mathbb{R}^p$ *with* $\mathbb{E}(\mathbf{X}_i) = 0$ *and* $\mathrm{Cov}(\mathbf{X}_i) = \mathbf{I}_p + \beta\mathbf{v}\mathbf{v}^T$, $\beta \geq 0$, $\mathbf{v} \in \mathbb{R}^p$, $\|\mathbf{v}\| = 1$. *Let* $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n) \in \mathbb{R}^{p\times n}$ *and* $\mathbf{S}_n = \frac{1}{n}\mathbf{X}\mathbf{X}^T \in \mathbb{R}^{p\times p}$. *Suppose that* $p, n \to \infty$ *such that* $\frac{p}{n} \to \gamma \in (0, 1]$ *as* $n \to \infty$.

- *If* $\beta \leq \sqrt{\gamma}$ *then* $\lambda_{\max}(\mathbf{S}_n) \to (1 + \sqrt{\gamma})^2$ *and* $|\langle \mathbf{v}_{\max}, \mathbf{v}\rangle|^2 \to 0$.
- *If* $\beta > \sqrt{\gamma}$ *then* $\lambda_{\max}(\mathbf{S}_n) \to (1 + \beta)(1 + \frac{\gamma}{\beta}) > (1 + \sqrt{\gamma})^2$ *and* $|\langle \mathbf{v}_{\max}, \mathbf{v}\rangle|^2 \to \frac{1-\gamma/\beta^2}{1-\gamma/\beta}$.

The interpretation of this theorem is that, only if $\beta > \sqrt{\gamma}$, the largest eigenvalue exceeds the upper asymptotic bound of the asymptotic support and the corresponding eigenvector has a non-trivial correlation with the eigenvector $\mathbf{v}$.

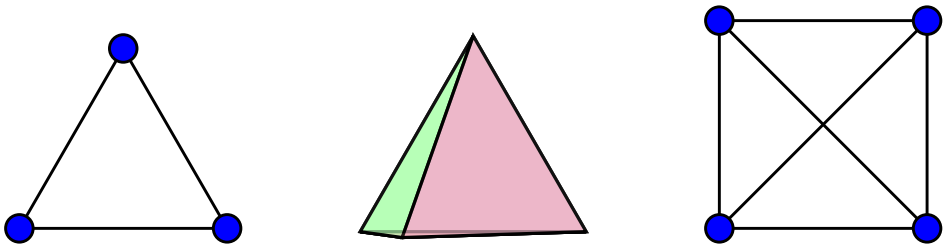


Figure 4.4: Embedding of $\mathbf{\Delta}_1$,$\mathbf{\Delta}_2$ and $\mathbf{\Delta}_3$

4.2 Multidimensional Scaling

Suppose that the pairwise distance of three points are given by the distance matrix

$$\mathbf{\Delta}_1 = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

The question is whether these points can be presented in a low dimensional space such that their pairwise distances are preserved. It is easy to see that this matrix has an embedding in a 2−dimensional space, given by a equilateral triangle. Now consider the following distance matrix for four points

$$\mathbf{\Delta}_2 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}.$$

An embedding of this matrix should have four points with equal distances. This is not possible in a 2-dimensional space but it is possible in a 3−dimensional space. This will have tetrahedron shape. Consider another distance matrix for four points

$$\mathbf{\Delta}_3 = \begin{bmatrix} 0 & 1 & \sqrt{2} & 1 \\ 1 & 0 & 1 & \sqrt{2} \\ \sqrt{2} & 1 & 0 & 1 \\ 1 & \sqrt{2} & 1 & 0 \end{bmatrix}.$$

This can be embedded in 2−dimensional space using a square with side length of 1. These examples are all about finding points in a low dimensional Euclidean space given only their pairwise distances. Evidently, the result is rotation and translation invariant.
Given n objects and $O_1, \ldots, O_n$ and pairwise dissimilarities δ_{ij} between objects i and j. Assume that $\delta_{ij} = \delta_{ji} \geq 0$ and $\delta_{ii} = 0$ for all $i, j = 1, \ldots, n$. Define $\mathbf{\Delta} = (\delta_{ij})_{1 \leq i,j \leq n}$ as the dissimilarity matrix. Define $\mathcal{U}_n$, the set of dissimilarity matrices as follows:

$$\mathcal{U}_n = \{\mathbf{\Delta} = (\delta_{ij})_{1 \leq i,j \leq n} | \delta_{ij} = \delta_{ji} \geq 0, \delta_{ii} = 0, \text{ for all } i, j = 1, \ldots, n\}.$$

Our objective now is to find n points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in a Euclidean space, typically $\mathbb{R}^k$, such that the distances $\|\mathbf{x}_i - \mathbf{x}_j\|$ fit the dissimilarities δ_{ij} best.

Example ([Mat97])**.** Consider towns in a country, say Germany and δ_{ij} is the driving distance from the town A to the town B. Find an embedding in $\mathbb{R}^2$, i.e., the map.

Consider $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^T \in \mathbb{R}^{n \times k}$ and distances $d_{ij}(\mathbf{X}) = \|\mathbf{x}_i - \mathbf{x}_j\|$, thus $\mathbf{D}(\mathbf{X}) = (d_{ij}(\mathbf{X}))_{1 \leq i,j \leq n} \in \mathbb{R}^{n \times n}$. Often, it is mathematically more convenient to consider a transformation of original distances and dissimilarities. One way is to consider the power $q \geq 1$ of these values, i.e., $\delta_{ij}^q, d_{ij}^q(\mathbf{X})$. For this purpose, the element-wise powered matrices are denoted by

$$\mathbf{\Delta}^{(q)} = (\delta_{ij}^q)_{1 \leq i,j \leq n}, \mathbf{D}^{(q)}(\mathbf{X}) = (d_{ij}^q(\mathbf{X}))_{1 \leq i,j \leq n}\,.$$

In its completely general formulation, the approximation problem of matrix multidimensional scaling (MDS) is as follows. Given a dissimilarity matrix $\mathbf{\Delta}$, a power transformation $q \geq 1$, a metric d on $\mathbb{R}^k$ and a matrix norm $\|.\|$, solve

$$\min_{\mathbf{X} \in \mathbb{R}^{n \times k}} \|\mathbf{\Delta}^{(q)} - \mathbf{D}^{(q)}(\mathbf{X})\|\,. \tag{4.1}$$

4.2.1 Characterization of Euclidean Distance Matrices

The dissimilarity matrix $\mathbf{\Delta} = (\delta_{ij})_{1 \leq i,j \leq n} \in \mathcal{U}_n$ is called Euclidean distance matrix, or it has a Euclidean embedding in $\mathbb{R}^k$, if there exist vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^k$ such that $\delta_{ij}^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2$ for all i, j where $\|.\|$ is the Euclidean norm (i.e., $\|\mathbf{y}\| = \sqrt{\sum_{i=1}^k y_i^2}$). This would be the case if the approximation problem from eq. (4.1) can be solved with error 0 for $q = 2$. In the reminder of this section, an explicit solution is constructed for a general class of matrix norms. For this task, the projection matrix $\mathbf{E}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$ plays an important role as we will see. Note that $\mathbf{1}_n\mathbf{1}_n^T = \mathbf{1}_{n \times n}$.

Theorem 4.3. *The dissimilarities matrix $\mathbf{\Delta} \in \mathcal{U}_n$ have an Euclidean embedding in $\mathbb{R}^k$ if and only if $-\frac{1}{2}\mathbf{E}_n\mathbf{\Delta}^{(2)}\mathbf{E}_n$ is non-negative definite and $\mathrm{rk}(\mathbf{E}_n\mathbf{\Delta}^{(2)}\mathbf{E}_n) \leq k$. The least k which allows for an embedding is called dimensionality of $\mathbf{\Delta}$.*

Proof. Given $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^T \in \mathbb{R}^{n \times k}$, it holds (proof as exercise) that

$$-\frac{1}{2}\mathbf{D}^{(2)}(\mathbf{X}) = \mathbf{X}\mathbf{X}^T - \mathbf{1}_n\hat{\mathbf{x}}^T - \hat{\mathbf{x}}\mathbf{1}_n^T\,,$$

where $\hat{\mathbf{x}} = \frac{1}{2}[\mathbf{x}_1^T\mathbf{x}_1, \ldots, \mathbf{x}_n^T\mathbf{x}_n]^T$. Using this relation and the fact that $\mathbf{E}_n\mathbf{1} = 0$, we get

$$-\frac{1}{2}\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X})\mathbf{E}_n = \mathbf{E}_n\mathbf{X}\mathbf{X}^T\mathbf{E}_n \succeq 0\,.$$

Note that the right hand side of the equality is non-negative definite, since $\mathbf{X} \in \mathbb{R}^{n \times k}$, $\mathrm{rk}(\mathbf{X}\mathbf{X}^T) \leq k$ thus $\mathrm{rk}(\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X})\mathbf{E}_n) \leq k$. If there is an Euclidean embedding of $\mathbf{\Delta}$ then $\mathbf{\Delta}^{(2)} = \mathbf{D}^{(2)}(\mathbf{X})$ for some $\mathbf{D}^{(2)}$, then

$$\begin{aligned} -\frac{1}{2}\mathbf{E}_n\mathbf{\Delta}^{(2)}\mathbf{E}_n = -\frac{1}{2}\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X})\mathbf{E}_n \succeq 0\,, \\ \mathrm{rk}(-\frac{1}{2}\mathbf{E}_n\mathbf{\Delta}^{(2)}\mathbf{E}_n) = \mathrm{rk}(-\frac{1}{2}\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X})\mathbf{E}_n) \leq k\,. \end{aligned}$$

For the opposite direction suppose that $-\frac{1}{2}\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n \succeq 0$ and $\mathrm{rk}(\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n) \leq k$. Then there exists $n \times k$ matrix $\mathbf{X}$ such that (proof as exercise)

$$-\frac{1}{2}\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n = \mathbf{X}\mathbf{X}^T, \text{ and } \mathbf{X}^T\mathbf{E}_n = \mathbf{X}^T.$$

Then $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^T$ is an appropriate configuration, since

$$-\frac{1}{2}\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X})\mathbf{E}_n = \mathbf{E}_n\mathbf{X}\mathbf{X}^T\mathbf{E}_n = \mathbf{X}\mathbf{X}^T = -\frac{1}{2}\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n.$$

It follows that $\mathbf{D}^{(2)}(\mathbf{X}) = \boldsymbol{\Delta}^{(2)}$. (proof as exercise) □

4.2.2 The Best Euclidean Fit to a Given Dissimilarity Matrix

Let $\|.\|$ denote the Frobenius norm, $\|\mathbf{A}\| = \left(\sum_{i,j} a_{ij}^2\right)^{\frac{1}{2}}$. Let λ^+ denote the positive part of λ as $\lambda^+ = \max\{\lambda, 0\}$.

Theorem 4.4 ([Mat97, p. 31])**.** *Let $\boldsymbol{\Delta} \in \mathcal{U}_n$ be a dissimilarity matrix, and $-\frac{1}{2}\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n$ has the spectral decomposition $-\frac{1}{2}\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n = \mathbf{V}\mathrm{diag}(\lambda_1, \ldots, \lambda_n)\mathbf{V}^T$ with $\lambda_1 \geq \cdots \geq \lambda_n$ and orthogonal matrix $\mathbf{V}$. Then the optimization problem*

$$\min_{\mathbf{X}\in\mathbb{R}^{n\times k}} \|\mathbf{E}_n\big(\boldsymbol{\Delta}^{(2)} - \mathbf{D}^{(2)}(\mathbf{X})\big)\mathbf{E}_n\|$$

has a solution given by

$$\mathbf{X}^* = \left[\sqrt{\lambda_1^+}\mathbf{v}_1, \ldots, \sqrt{\lambda_k^+}\mathbf{v}_k\right] \in \mathbb{R}^{n\times k}.$$

Proof. Note that a solution to

$$\min_{\mathbf{A}\succeq 0, \mathrm{rk}(\mathbf{A})\leq k} \| -\frac{1}{2}\mathbf{E}_n\boldsymbol{\Delta}^{(2)}\mathbf{E}_n - \mathbf{A}\|$$

is given by $\mathbf{A}^* = \mathbf{V}\mathrm{diag}(\lambda_1^+, \ldots, \lambda_k^+, 0, \ldots, 0)\mathbf{V}^T$, according to Theorem 2.6. Then, it holds

$$\begin{aligned}
-\frac{1}{2}\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X}^*)\mathbf{E}_n &= \mathbf{E}_n\mathbf{X}^*\mathbf{X}^{*T}\mathbf{E}_n \\
&= \mathbf{E}_n[\mathbf{v}_1, \ldots, \mathbf{v}_k]\mathrm{diag}(\lambda_1^+, \ldots, \lambda_k^+)[\mathbf{v}_1, \ldots, \mathbf{v}_k]^T\mathbf{E}_n \\
&= \mathbf{V}\mathrm{diag}(\lambda_1^+, \ldots, \lambda_k^+, 0, \ldots, 0)\mathbf{V}^T = \mathbf{A}^*.
\end{aligned}$$

So that the minimum is attained in the set $\left\{-\frac{1}{2}\mathbf{E}_n\mathbf{D}^{(2)}(\mathbf{X})\mathbf{E}_n | \mathbf{X} \in \mathbb{R}^{n\times k}\right\}$.

□

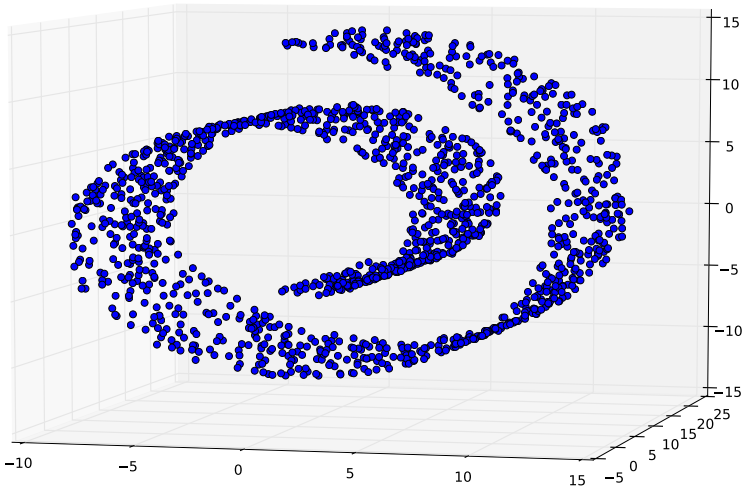


Figure 4.5: Swiss roll data with 1500 samples

4.2.3 Non-linear Dimensionality Reduction

Suppose that the data points do not lie near a linear subspace but have a low dimensional structure anyway. Consider the data point in Figure 4.5. The model is call two-dimensional swiss roll. The points lie on a two dimensional manifold which is a non-linear one. Previous dimensionality reduction methods for finding a low-dimensional embedding of this data cannot detect the proper structure of the data. The main reason is that the geodesic distance of points should be considered instead of Euclidean distances. The points that are far apart on the manifold, measured through their shortest path on the manifold, may look very close in high-dimensional space.
The complete isometric feature mapping, ISOMAP is an example of non-linear dimensionality reduction [TDSL00]. Given data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$, lying on a manifold, e.g., the swiss roll, the idea is to approximate the geodesic distance of the data points by constructing a weighted graph and finding the shortest path between vertices. For the sake of clarity, an example of a weighted graph is depicted in Figure 4.6. The algorithm consists of three steps:

1. Construct neighborhood graph: find a weighted graph $G(V, E, \mathbf{W})$ with vertices $v_i = \mathbf{x}_i$ such that two vertices v_i and v_j are connected only if $\|\mathbf{x}_i - \mathbf{x}_j\| \leq \epsilon$. Another way is to connect each point to its K nearest neighbors.

2. Compute the shortest paths: for each pair (v_i, v_j) compute the shortest path (Di-

jkstra's algorithm). The geodesic distance $\delta(v_i, v_j)$ can be taken as number of hops/links from v_i to v_j or sum of $\|\mathbf{x}_l - \mathbf{x}_k\|$ on a shortest path. [2]

3. Construct $d-$dimensional embedding: apply MDS on the basis of geodesic distances $\mathbf{\Delta} = (\delta(v_i, v_j))_{1 \leq i,j \leq n}$.

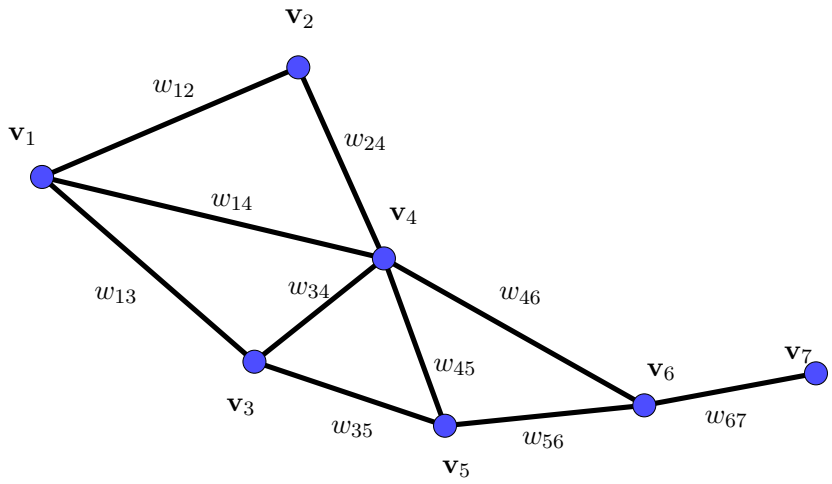


Figure 4.6: Example weighted graph $G(V, E, \mathbf{W})$ with vertices $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_7\}$, edges $E = \{e_{12}, \ldots, e_{67}\}$ where every edge e_{ij} has a corresponding associated weight $w_{ij} = w_{ji}$, thus leading to the graph's weight matrix $\mathbf{W} = (w_{ij})_{i,j=1,\ldots,7} \in \mathbb{R}^{7\times 7}$.

Note that, this algorithm's performance is sensitive to the choice of ϵ. A very small choice of ϵ leads to disconnected graph and a large ϵ misses the low dimensional structure of data. Shortcomings of this approach are:

- Very large distances may distort local neighborhoods.
- Computational complexity: Dijkstra's algorithm, MDS.
- Not robust to noise perturbation: as mentioned above, ϵ should be adapted to noise perturbation. There are some ways of choosing ϵ based on the given data [BS02] by looking at the trade-off between two parameters. One is the fraction of the variance in geodesic distance estimates not accounted for in the Euclidean embedding and the other is the fraction of points not included in the largest connected component of the neighborhood graph [BS02, Fig. 1].

[2] Locally the geodesic distance can be well appropriated by the Euclidean one.

4.3 Diffusion Maps

Diffusion Maps is a non-linear dimensionality reduction technique or feature extraction, introduced by Coifman and Lafon [CL06]. With ISOMAP, it is another example of manifold learning algorithms that capture the geometry of the data set. In these algorithms, the data is represented by parameters of its underlying geometry in a low dimensional Euclidean space. The main intention is to discover the underlying manifold that the data has been sampled from. The main idea is to construct a weight function (or kernel) based on the connection between data. The eigenvectors obtained using this kernel represent the data in a lower dimension. The diffusion map framework consists of the following steps [TCGC13]:

1. Construct a weighted graph $G(V, E, \mathbf{W})$ on the data. The pairwise weights measure the closeness between data points.

2. Define a random walk on the graph determined by a transition matrix constructed from the weights $\mathbf{W}$.

3. Perform a non-linear embedding of the points in a lower dimensional space based on the parameters of the graph and its respective transition matrix.

To this end, let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ be n samples. We start from constructing a weighted graph $G(V, E, \mathbf{W})$. In a diffusion map, the nodes which are connected by an edge with large weight are considered to be close. Each sample $\mathbf{x}_i$ is associated with a vertex v_i. The weight of an edge between $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by the weight function or kernel $w_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. The selected kernel should satisfy three properties:

- Symmetry: $K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)$
- Non-negativity: $K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$
- Locality: there is a scale parameter ϵ such that if $\|\mathbf{x}_i - \mathbf{x}_j\| \ll \epsilon$ then $K(\mathbf{x}_i, \mathbf{x}_j) \to 1$, and if $\|\mathbf{x}_i - \mathbf{x}_j\| \gg \epsilon$ then $K(\mathbf{x}_i, \mathbf{x}_j) \to 0$.

Note that such kernel functions encapsulate the notion of closeness between the data points. Setting the scale parameter ϵ, similar to the choice of ϵ in ISOMAP, is important. A small ϵ may lead to a disconnected graph, while a large ϵ may miss the underlying geometry. The Gaussian kernel is one of the well known weight functions and its defined as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\epsilon^2}\right).$$

Using these kernel functions, the weight matrix is constructed.
Next, we construct a random walk X_t, $t = 0, 1, 2, \ldots$ on the vertices of the graph $V = \{v_1, \ldots, v_n\}$ with transition matrix

$$\mathbf{M} = (M_{ij})_{i,j=1,\ldots,n} \text{ with } M_{ij} = \frac{w_{ij}}{\deg(i)}, 1 \leq i, j \leq n\,,$$

where $\mathbf{W} = (w_{ij})_{1\leq i,j\leq n}$ and $\deg(i) = \sum_j w_{ij}$. This transition matrix represents the probability of moving from the node v_i at time t to v_j at time $t+1$, namely

$$\mathbb{P}(X_{t+1} = j | X_t = i) = M_{ij}\,.$$

The transition matrix $\mathbf{M}$ can be written as $\mathbf{D}^{-1}\mathbf{W}$ where $\mathbf{D} = \mathrm{diag}(\deg(1), \ldots, \deg(n))$. Moreover, the conditional distribution of being at the vertex v_j having started at the vertex v_i is given by

$$\mathbb{P}(X_t = j | X_0 = i) = (\mathbf{M}^t)_{i,j}, j = 1, \ldots, n\,.$$

Then, the probability of being at each vertex after step time t starting from v_i is given by i^{th} row of $\mathbf{M}^t = (M_{ij}^{(t)})_{1\leq i,j\leq n}$. This distribution is

$$v_i \to \mathbf{e}_i^T\mathbf{M}^t = (M_{i1}^{(t)}, \ldots, M_{in}^{(t)})\,.$$

Therefore, a vector of probabilities is assigned to each vertex v_i. This vector contains information about underlying geometry. If v_i and v_j are close - strongly connected in the graph - then $\mathbf{e}_i^T\mathbf{M}^t$ and $\mathbf{e}_j^T\mathbf{M}^t$ will be similar.
However it is still not clear how this representation can be embedded in a low-dimensional space. To do this, we focus on the spectrum of $\mathbf{M}^t$. The transition $\mathbf{M} = \mathbf{D}^{-1}\mathbf{W}$ is not symmetric, however the normalized matrix $\mathbf{S} = \mathbf{D}^{1/2}\mathbf{M}\mathbf{D}^{-1/2}$ is symmetric, since $\mathbf{S} = \mathbf{D}^{1/2}\mathbf{D}^{-1}\mathbf{W}\mathbf{D}^{-1/2} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ and $\mathbf{W}$ is symmetric. The spectral decomposition of $\mathbf{S}$ is then given by $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$, with $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ eigenvalue matrix such that $\lambda_1 \geq \cdots \geq \lambda_n$. Therefore, $\mathbf{M}$ can be written as

$$\mathbf{M} = \mathbf{D}^{-1/2}\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\mathbf{D}^{1/2} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^{\mathrm{T}}$$

where $\mathbf{\Phi} = \mathbf{D}^{-1/2}\mathbf{V} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n)$ and $\mathbf{\Psi} = \mathbf{D}^{1/2}\mathbf{V} = (\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_n)$.
Note that $\mathbf{\Phi}$ and $\mathbf{\Psi}$ are bi-orthogonal, i.e., $\mathbf{\Phi}^T\mathbf{\Psi} = \mathbf{I}_n$, or equivalently $\boldsymbol{\phi}_i^T\boldsymbol{\psi}_j = \delta_{ij}$. λ_k's are the eigenvalues of $\mathbf{M}$ with right and left eigenvectors $\boldsymbol{\phi}_k$ and $\boldsymbol{\psi}_k$, thus

$$\mathbf{M}\boldsymbol{\phi}_k = \lambda_k\boldsymbol{\phi}_k, \boldsymbol{\psi}_k^T\mathbf{M} = \lambda_k\boldsymbol{\psi}_k^T\,.$$

In summary, we have that

$$\mathbf{M} = \sum_{k=1}^{n} \lambda_k\boldsymbol{\phi}_k\boldsymbol{\psi}_k^T$$

and hence

$$\mathbf{M}^t = \sum_{k=1}^{n} \lambda_k^t\boldsymbol{\phi}_k\boldsymbol{\psi}_k^T.$$

$$\mathbf{e}_i^T\mathbf{M}^t = \sum_{k=1}^{n} \lambda_k^t\mathbf{e}_i^T\boldsymbol{\phi}_k\boldsymbol{\psi}_k^T = \sum_{k=1}^{n} \lambda_k^t\phi_{k,i}\boldsymbol{\psi}_k^T,$$

Therefore, the distribution $\mathbf{e}_i^T\mathbf{M}^t$ can be represented in terms of basis vectors $\boldsymbol{\psi}_k$ with coefficients $\lambda_k^t\phi_{k,i}$ for $k = 1, \ldots, n$ with $\boldsymbol{\phi}_k = (\phi_{k,1}, \ldots, \phi_{k,n})^T$. These coefficients are used to define the diffusion map.

Definition 4.5. The diffusion map at step time t is defined as:

$$\boldsymbol{\phi}_t(v_i) = \begin{bmatrix} \lambda_1^t \phi_{1,i} \\ \vdots \\ \lambda_n^t \phi_{n,i} \end{bmatrix}, i = 1, \ldots, n$$

In the diffusion map, $\phi_{k,i}$ does not vary with t but each element is dependent on t via λ_i^t. The eigenvalues of transition matrix therefore capture the main components of the data. The following theorem provides some information about the eigenvalues of $\mathbf{M}$.

Theorem 4.6. *The eigenvalues $\lambda_1, \ldots, \lambda_n$ of $\mathbf{M}$ satisfy $|\lambda_k| \leq 1$. It also holds that $\mathbf{M}\mathbf{1}_n = \mathbf{1}_n$ and 1 is an eigenvalue of $\mathbf{M}$.*

Proof. Since $\mathbf{M}$ is an stochastic matrix, then the sum of each row elements is one, which implies $\mathbf{M}\mathbf{1}_n = \mathbf{1}_n$. Let $\mathbf{m}_k = (m_{k,1}, \ldots, m_{k,n})^T$ be the eigenvector corresponding to λ_k and suppose that $|m_{k,l}| = \max_{1 \leq j \leq n} |m_{k,j}|$, which means that $|m_{k,j}| \leq |m_{k,l}|$. It can be seen that

$$\sum_{j=1}^{n} M_{lj} m_{k,j} = \lambda_k m_{k,l} \implies |\lambda_k| \leq \sum_{j=1}^{n} M_{lj} \frac{|m_{k,j}|}{|m_{k,l}|} \leq \sum_{j=1}^{n} M_{lj} = 1.$$

□

An interesting point is that $\lambda_1 = 1$ and $\boldsymbol{\phi}_1 = \mathbf{1}_n$. Therefore the first element of the diffusion map in above definition is always one for all points. Therefore we simply drop this from the diffusion map and rewrite it as

$$\boldsymbol{\phi}_t(v_i) = \begin{bmatrix} \lambda_2^t \phi_{2,i} \\ \vdots \\ \lambda_n^t \phi_{n,i} \end{bmatrix}, i = 1, \ldots, n.$$

It is possible to have more than one eigenvalues with absolute value equal to one. In this case, the underlying graph is either disconnected or bipartite. If λ_k is small, λ_k^t is rather small for moderate t. This motivates truncating the diffusion maps to d dimensions.

Definition 4.7. The diffusion map truncated to d dimensions is defined as

$$\boldsymbol{\phi}_t^{(d)}(v_i) = \begin{bmatrix} \lambda_2^t \phi_{1,i} \\ \vdots \\ \lambda_{d+1}^t \phi_{n,i} \end{bmatrix}, i = 1, \ldots, n$$

$\boldsymbol{\phi}_t^{(d)}(v_i)$ is an approximate embedding of $v_1, \ldots, v_n$ in a $d-$dimensional Euclidean space. If the graph structure $G(V, E, \mathbf{W})$ is appropriately chosen, non-linear geometries can also be recovered using diffusion maps. The connection between the Euclidean distance in the diffusion map coordinates (diffusion distance) and the distance between the probability distributions is described in the following theorem [Ban08, Theorem 2.11].

Theorem 4.8. *For any pair of nodes v_i and v_j it holds that:*

$$\|\boldsymbol{\phi}_t(v_i) - \boldsymbol{\phi}_t(v_j)\|^2 = \sum_{l=1}^{n} \frac{1}{\deg(l)} \big(\mathbb{P}(X_t = l | X_0 = i) - \mathbb{P}(X_t = l | X_0 = j)\big)^2.$$

Proof. Exercise. □

5 Classification and Clustering

Classification and clustering are one of the central tasks in machine learning. Given a set of data points, the purpose is to classify the points into subgroups, which express closeness or similarity of the points and which are represented by a cluster head.

5.1 Discriminant Analysis

Suppose that g populations/groups/classes $C_1, \dots, C_g$ are given, each represented by a p.d.f. $f_i(\mathbf{x})$ on $\mathbb{R}^p$, $i = 1, \dots, g$.
A discriminant rule divides $\mathbb{R}^p$ into disjoint regions $R_1, \dots, R_g$, $\cup_{i=1}^{p} R_i = \mathbb{R}^p$. The discriminant rule is defined by

$$\text{allocate some observation } \mathbf{x} \text{ to } C_i \text{ if } \mathbf{x} \in R_i \,.$$

Often the densities $f_i(\mathbf{x})$ are completely unknown or its parameters, such as its mean and variance, must be estimated from a training set $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^p$ with known class allocation. This setup, where a training set with known class allocation is given, is called a "supervised" learning setup.

5.1.1 Fisher's Linear Discriminant Function

Fisher's linear discriminant function is a tool for supervised learning, where a training set $\mathbf{x}_1, \dots, \mathbf{x}_n$ with known classification is given. When a new observation $\mathbf{x}$ with unknown classification is obtained, a linear discriminant rule $\mathbf{a}^T\mathbf{x}$ is calculated such that $\mathbf{x}$ is allocated to some class in an optimal way.
Hence, an appropriate linear transformation $\mathbf{a} \in \mathbb{R}^p$ must be computed using the training set. Particularly, in this analysis, $\mathbf{a}$ is chosen such that the ratio of the *between-groups sum of squares* and the *within group sum of squares* is maximized. To formally define this ratio, let $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$ be samples from g groups $C_1, \dots, C_g$. We as well define $\mathbf{X}_l = [\mathbf{x}_j]_{j \in C_l}$ and $n_l = |\{j : 1 \le j \le n; j \in C_l\}|$. Then, the average of the training set is given by

$$\overline{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \in \mathbb{R}^p \,,$$

and the average over the group C_l is given by

$$\overline{\mathbf{x}}_l = \frac{1}{n_l} \sum_{j \in C_l} \mathbf{x}_j \in \mathbb{R}^p \,.$$

Moreover, since $\mathbf{a} \in \mathbb{R}^p$ is defined to be the linear discriminant of data, the vector $\mathbf{y} \in \mathbb{R}^n$ that stores the discriminant values of the vectors within the training set is given by

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \mathbf{X}^T \mathbf{a}\,.$$

The discriminant values corresponding to the vectors of the training set, that belong to the group C_l, are stored in $\mathbf{y}_l = (y_j)_{j \in C_l}$. Similarly define the general average and between-groups average as

$$\overline{y} = \frac{1}{n} \sum_{i=1}^{n} y_i \quad \text{and} \quad \overline{y}_l = \frac{1}{n_l} \sum_{j \in C_l} y_j\,.$$

Note that

$$\begin{aligned} \sum_{i=1}^{n} (y_i - \overline{y})^2 &= \sum_{l=1}^{g} \sum_{j \in C_l} (y_j - \overline{y}_l + \overline{y}_l - \overline{y})^2 \\ &\overset{(a)}{=} \sum_{l=1}^{g} \left[\sum_{j \in C_l} (y_j - \overline{y}_l)^2 + \sum_{j \in C_l} (\overline{y}_l - \overline{y})^2 \right] \\ &= \sum_{l=1}^{g} \sum_{j \in C_l} (y_j - \overline{y}_l)^2 + \sum_{l=1}^{g} n_l (\overline{y}_l - \overline{y})^2 \end{aligned}$$

where (a) follows from a similar argument behind Steiner's rule -Theorem 3.3. Finally, $\sum_{l=1}^{g} \sum_{j \in C_l} (y_j - \overline{y}_l)^2$ is the sum of squares within groups and $\sum_{l=1}^{g} n_l (\overline{y}_l - \overline{y})^2$ is the sum of squares between groups, so the problem of selecting the optimal $\mathbf{a}$ is formally defined.

To address this problem with compact notation, let $\mathbf{E}_n$ and $\mathbf{E}_{n_l} = \mathbf{E}_l$, $l = 1, \ldots, g$ be centering operators. Using matrix notation, we have

$$\begin{aligned} \sum_{l=1}^{g} \sum_{j \in C_l} (y_j - \overline{y}_l)^2 &= \sum_{l=1}^{g} \mathbf{y}_l^T \mathbf{E}_l \mathbf{y}_l \\ &= \sum_{l=1}^{g} \mathbf{a}^T \mathbf{X}_l^T \mathbf{E}_l \mathbf{X}_l \mathbf{a} \\ &= \mathbf{a}^T (\sum_{l=1}^{g} \mathbf{X}_l^T \mathbf{E}_l \mathbf{X}_l) \mathbf{a} = \mathbf{a}^T \mathbf{W} \mathbf{a}. \end{aligned}$$

where $\mathbf{W} = \sum_{l=1}^{g} \mathbf{X}_l^T \mathbf{E}_l \mathbf{X}_l$. Similarly,

$$\begin{aligned}\sum_{l=1}^{g} n_l(\overline{y}_l - \overline{y})^2 &= \sum_{l=1}^{g} n_l(\mathbf{a}^T(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}}))^2 \\ &= \sum_{l=1}^{g} n_l \mathbf{a}^T(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}})(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}})^T\mathbf{a} \\ &= \mathbf{a}^T\left(\sum_{l=1}^{g} n_l(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}})(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}})^T\right)\mathbf{a} = \mathbf{a}^T\mathbf{B}\mathbf{a},\end{aligned}$$

where $\mathbf{B} = \sum_{l=1}^{g} n_l(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}})(\overline{\mathbf{x}}_\mathbf{l} - \overline{\mathbf{x}})^T$. Then, the linear discriminant analysis requires obtaining the $\mathbf{a}$ that solves

$$\max_{\mathbf{a}\in\mathbb{R}^p} \frac{\mathbf{a}^T\mathbf{B}\mathbf{a}}{\mathbf{a}^T\mathbf{W}\mathbf{a}} \quad (\star)$$

Theorem 5.1. *The maximum value of* $(\star)$ *is attained at the eigenvector of* $\mathbf{W}^{-1}\mathbf{B}$ *corresponding to the largest eigenvalue.*

Proof. Assuming $\mathbf{a} = \mathbf{W}^{-1/2}\mathbf{b}$, we have

$$\max_{\mathbf{a}\in\mathbb{R}^p} \frac{\mathbf{a}^T\mathbf{B}\mathbf{a}}{\mathbf{a}^T\mathbf{W}\mathbf{a}} = \max_{\mathbf{b}\in\mathbb{R}^p} \frac{\mathbf{b}^T\mathbf{W}^{-1/2}\mathbf{B}\mathbf{W}^{-1/2}\mathbf{b}}{\mathbf{b}^T\mathbf{b}} = \lambda_{\max}(\mathbf{W}^{-1/2}\mathbf{B}\mathbf{W}^{-1/2}),$$

where the last part results from Theorem 2.4. Furthermore $\mathbf{W}^{-1/2}\mathbf{B}\mathbf{W}^{-1/2}$ and $\mathbf{W}^{-1}\mathbf{B}$ have the same eigenvalues, since:

$$\mathbf{W}^{-1}\mathbf{B}\mathbf{v} = \lambda\mathbf{v} \iff \mathbf{W}^{-1/2}\mathbf{B}\mathbf{v} = \lambda\mathbf{W}^{1/2}\mathbf{v} \iff \mathbf{W}^{-1/2}\mathbf{B}\mathbf{W}^{-1/2}\mathbf{W}^{1/2}\mathbf{v} = \lambda\mathbf{W}^{1/2}\mathbf{v}.$$

Therefore the two matrices have the same eigenvalues. Moreover suppose that $\mathbf{v}$ is the eigenvector of $\mathbf{W}^{-1}\mathbf{B}$ corresponding to $\lambda_{\max}$. Then we have

$$\frac{\mathbf{v}^T\mathbf{B}\mathbf{v}}{\mathbf{v}^T\mathbf{W}\mathbf{v}} = \frac{\mathbf{v}^T\mathbf{B}\mathbf{v}}{\mathbf{v}^T\mathbf{W}(\frac{1}{\lambda_{\max}}\mathbf{W}^{-1}\mathbf{B}\mathbf{v})} = \lambda_{\max}.$$

□

The linear function $\mathbf{a}^T\mathbf{x}$ is called Fisher's linear discriminant function or the first canonical variate. The ratio is invariant with the respect to scaling of $\mathbf{a}$.
The application of the linear discriminant analysis is as follows.

- Given the training set $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ with known groups, compute the optimum $\mathbf{a}$ from Theorem 5.1.
- For a new observation $\mathbf{x}$, compute $\mathbf{a}^T\mathbf{x}$.

- Allocate $\mathbf{x}$ to the group with closest value of $\mathbf{a}^T\overline{\mathbf{x}}_\mathbf{l} = \overline{y}_l$. Discriminant rule can be formulated as

  Discriminant Rule: Allocate $\mathbf{x}$ to the group l if $|\mathbf{a}^T\mathbf{x} - \mathbf{a}^T\overline{\mathbf{x}}_\mathbf{l}| < |\mathbf{a}^T\mathbf{x} - \mathbf{a}^T\overline{\mathbf{x}}_\mathbf{j}|$ for all $j = 1, \ldots, l-1, l+1, \ldots, g$.

Fisher's discriminant function is immediately relevant for the special case of $g = 2$, where there are two groups of size n_1 and n_2 with $n = n_1 + n_2$. In this case we have

$$\begin{aligned}
\mathbf{B} &= n_1(\overline{\mathbf{x}}_\mathbf{1} - \overline{\mathbf{x}})(\overline{\mathbf{x}}_\mathbf{1} - \overline{\mathbf{x}})^T + n_2(\overline{\mathbf{x}}_\mathbf{2} - \overline{\mathbf{x}})(\overline{\mathbf{x}}_\mathbf{2} - \overline{\mathbf{x}})^T \\
&= n_1(\overline{\mathbf{x}}_\mathbf{1} - \frac{n_1}{n}\overline{\mathbf{x}_\mathbf{1}} - \frac{n_2}{n}\overline{\mathbf{x}}_\mathbf{2})(\overline{\mathbf{x}}_\mathbf{1} - \frac{n_1}{n}\overline{\mathbf{x}_\mathbf{1}} - \frac{n_2}{n}\overline{\mathbf{x}}_\mathbf{2})^T + n_2(\overline{\mathbf{x}}_\mathbf{2} - \frac{n_2}{n}\overline{\mathbf{x}_\mathbf{2}} - \frac{n_1}{n}\overline{\mathbf{x}}_\mathbf{1})(\overline{\mathbf{x}}_\mathbf{2} - \frac{n_2}{n}\overline{\mathbf{x}_\mathbf{2}} - \frac{n_1}{n}\overline{\mathbf{x}}_\mathbf{1})^T \\
&= n_1(\frac{n_2}{n}\overline{\mathbf{x}_\mathbf{1}} - \frac{n_2}{n}\overline{\mathbf{x}}_\mathbf{2})(\frac{n_2}{n}\overline{\mathbf{x}_\mathbf{1}} - \frac{n_2}{n}\overline{\mathbf{x}}_\mathbf{2})^T + n_2(\frac{n_1}{n}\overline{\mathbf{x}_\mathbf{2}} - \frac{n_1}{n}\overline{\mathbf{x}}_\mathbf{1})(\frac{n_1}{n}\overline{\mathbf{x}_\mathbf{2}} - \frac{n_1}{n}\overline{\mathbf{x}}_\mathbf{1})^T \\
&= \frac{n_1 n_2^2}{n^2}(\overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2})(\overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2})^T + \frac{n_2 n_1^2}{n^2}(\overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2})(\overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2})^T \\
&= \frac{n_1 n_2}{n}(\overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2})(\overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2})^T = \frac{n_1 n_2}{n}\mathbf{d}\mathbf{d}^T,
\end{aligned}$$

where $\mathbf{d} = \overline{\mathbf{x}_\mathbf{1}} - \overline{\mathbf{x}}_\mathbf{2}$. Therefore $\mathbf{B}$ has rank one and only one eigenvalue that is not equal to 0. Therefore $\mathbf{W}^{-1}\mathbf{B}$ has only one non-zero eigenvalue, which is given by

$$\mathrm{tr}(\mathbf{W}^{-1}\mathbf{B}) = \frac{n_1 n_2}{n}\mathbf{d}^T\mathbf{W}^{-1}\mathbf{d}.$$

Since $\mathbf{W}$ is nonnegative definite, the above value is nonnegative and therefore is the maximum eigenvalue. Note that $\mathbf{d}$ is an eigenvector of $\mathbf{B}$. We have

$$\begin{aligned}
(\mathbf{W}^{-1}\mathbf{B})\mathbf{W}^{-1}\mathbf{d} &= \mathbf{W}^{-1}(\frac{n_1 n_2}{n}\mathbf{d}\mathbf{d}^T)\mathbf{W}^{-1}\mathbf{d} \\
&= \frac{n_1 n_2}{n}\mathbf{W}^{-1}\mathbf{d}\left(\mathbf{d}^T\mathbf{W}^{-1}\mathbf{d}\right) \\
&= \left(\frac{n_1 n_2}{n}\mathbf{d}^T\mathbf{W}^{-1}\mathbf{d}\right)\mathbf{W}^{-1}\mathbf{d}.
\end{aligned}$$

Therefore, $\mathbf{W}^{-1}\mathbf{d}$ is an eigenvector of $\mathbf{W}^{-1}\mathbf{B}$ corresponding to the eigenvalue $\frac{n_1 n_2}{n}\mathbf{d}^T\mathbf{W}^{-1}\mathbf{d}$. Then, the discriminant rule becomes

- Allocate $\mathbf{x}$ to C_1 if $\mathbf{d}^T\mathbf{W}^{-1}(\mathbf{x} - \frac{1}{2}(\overline{\mathbf{x}}_\mathbf{1} + \overline{\mathbf{x}}_\mathbf{2})) > 0$.

$\mathbf{a} = \mathbf{W}^{-1}\mathbf{d}$ is normal to the discriminating hyperplane between the classes.

One advantage of Fischer's approach is that it is distribution free. It is based on the general principle that the between-groups sum of squares is large relative to the within-groups sum of squares, which measured by the quotient of these two quantities.

5.1.2 Gaussian Maximum Likelihood (ML) Discriminant Rule

In general, the maximum likelihood rule allocates observation $\mathbf{x}$ to the class C_l which maximizes the likelihood $L_l(\mathbf{x}) = \max_j L_j(\mathbf{x})$. A Gaussian ML rule is the particular case when the likelihood functions $L_l(\mathbf{x})$ are Gaussian. Assume that the class distributions are Gaussian and known as $\mathcal{N}_p(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ with $\boldsymbol{\mu}_l$ and $\boldsymbol{\Sigma}_l$ fixed and with densities

$$f_l(\mathbf{u}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}_l|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{u}-\boldsymbol{\mu}_l)^T\boldsymbol{\Sigma}_l^{-1}(\mathbf{u}-\boldsymbol{\mu}_l)\right\}, \mathbf{u} \in \mathbb{R}^p.$$

Then, the Gaussian ML discriminant rule would assign a given $\mathbf{x}$ to the class C_l that maximizes $f_l(x)$ over l.

Theorem 5.2. *The ML discriminant allocates* $\mathbf{x}$ *to class* C_l *which maximizes* $f_l(\mathbf{x})$ *over* $l = 1, \ldots, g$.

(a) If $\boldsymbol{\Sigma}_l = \boldsymbol{\Sigma}$ *for all* l*, then the ML rule allocates* $\mathbf{x}$ *to* C_l *which minimizes the Mahalanobis distance:*

$$(\mathbf{x}-\boldsymbol{\mu}_l)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_l).$$

(b) If $g = 2$*, and* $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$*, then the ML rule allocates* $\mathbf{x}$ *to the class* C_1 *if*

$$\boldsymbol{\alpha}^T(\mathbf{x}-\boldsymbol{\mu}) > 0,$$

where $\boldsymbol{\alpha} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$ *and* $\boldsymbol{\mu} = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)$.

Proof. Part (a) follows directly from the definition of ML discriminant rule. The ML discriminant finds the class l such that:

$$l = \arg\max_{1\le j\le g} f_j(\mathbf{x}).$$

Since $\boldsymbol{\Sigma}$ is fixed for all classes, the maximization of $f_l(\mathbf{x})$ amounts to maximization of exponent which is minimization of the Mahalanobis distance. Part (b) is an exercise. □

Note that the rule (b) is analogue to Fisher's discriminant rule with parameters $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$ and $\boldsymbol{\Sigma}$ substituting estimates $\overline{\mathbf{x}}_\mathbf{1}$, $\overline{\mathbf{x}}_\mathbf{2}$ and $\mathbf{W}$.
Application in practice: $\boldsymbol{\Sigma}_l$ and $\boldsymbol{\mu}_l$ are mostly not known. One can estimate these parameters from a training set with known allocations as $\hat{\boldsymbol{\Sigma}}_\mathbf{l}$ and $\hat{\boldsymbol{\mu}}_l$ for $l = 1, \ldots, g$. Substitute $\boldsymbol{\Sigma}_l$ and $\boldsymbol{\mu}_l$ by their ML estimates $\hat{\boldsymbol{\Sigma}}_\mathbf{l}$ and $\hat{\boldsymbol{\mu}}_l$ and compute the ML discriminant rule.

5.2 Cluster Analysis

The aim of cluster analysis is to group n objects into g homogeneous classes. g is normally unknown, but usually assumed to be much smaller than n. Homogeneous means that objects are close to each other. Members of different groups are significantly discriminable. A certain metric is needed to define success. Note that this problem is a case of unsupervised learning, since none of the n objects are known to belong to a certain group.

5.2.1 $k-$means Clustering

Given the training set $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$, the purpose of $k-$means Clustering is to partition the data set into clusters $C_1, \ldots, C_g$ with centers in each cluster $\mathbf{u}_1, \ldots, \mathbf{u}_g$ as solution to:

$$\min_{C_1,\ldots,C_g,\mathbf{u}_1,\ldots,\mathbf{u}_g} \sum_{l=1}^{k} \sum_{i \in C_l} \|\mathbf{x}_i - \mathbf{u}_l\|^2$$

Since the optimization problem above is quite difficult to solve, the problem can be modified as the optimal centers are $\overline{\mathbf{x}}_l = \frac{1}{n_l} \sum_{j \in C_l} \mathbf{x}_j \in \mathbf{R}^p$, when given the partition.

Algorithm 1 $k-$means algorithm - Lloyd's algorithm

1: **procedure** $k-$MEANS ALGORITHM
2: **repeat**
3: Given centers $\mathbf{u}_1, \ldots, \mathbf{u}_g$, each point $\mathbf{x}_i$ is assigned to cluster $l = \arg\min_j \|\mathbf{x}_i - \mathbf{u}_j\|^2$
4: update the centers $\mathbf{u}_l = \frac{1}{n_l} \sum_{i \in C_l} \mathbf{x}_i$
5: **until** no more new data

The disadvantages of $k-$means clustering is that the number of clusters of g needs knowing priori, and Euclidean space is needed as well. The iteration may end in sub-optimal solutions which has always convex clusters. This is particularly difficult for the data pattern in Figure 5.1. We discuss this clustering problem in the next part.

5.2.2 Spectral Clustering

To overcome these difficulties, when given the data set $\mathbf{x}_1, \ldots, \mathbf{x}_n$, a weighted graph could be constructed $G = (V, E, \mathbf{W})$, each point $\mathbf{x}_i$ is a vertex $\mathbf{v}_i, i = 1, \ldots, n$, and the edges weights $\mathbf{w}_{i,j}$ are $\mathbf{w}_{i,j} = \mathbf{K}_\varepsilon(\|\mathbf{x}_i - \mathbf{x}_j\|)$ with kernel $\mathbf{K}_\varepsilon$, e.g., $\mathbf{K}_\varepsilon(u) = \exp(-\frac{1}{2\varepsilon}u^2)$. Here, it should be noted that $\|\mathbf{x}_i - \mathbf{x}_j\|$ can be substituted by any dissimilarity measure. Now, lets consider a random walk with transition matrix

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

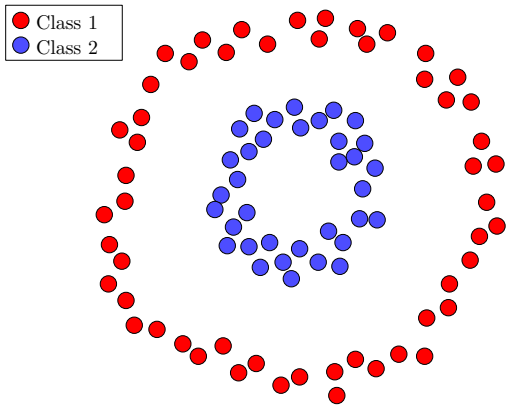


Figure 5.1: Example where k-means clustering is sub-optimal.

We assume that this matrix is constructed using a diffusion map as in Section 4.3, thus

$$\mathbf{P}(\mathbf{X}_{t+1} = j|\mathbf{X}_t = i) = \frac{w_{ij}}{\deg(i)} = M_{ij}, \tag{5.1a}$$

$$\mathbf{D} = \mathrm{diag}(\deg(i)), \deg(i) = \sum_{l=1}^{n} w_{il}\,. \tag{5.1b}$$

Therefore, $\mathbf{M}$ can be decomposed into

$$\mathbf{M} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^T = \sum_{k=1}^{n} \lambda_k \varphi_k \psi_k^T\,, \tag{5.2a}$$

$$\mathbf{\Phi} = (\varphi_1, \ldots, \varphi_n), \quad \mathbf{\Psi} = (\psi_1, \ldots, \psi_n)\,, \tag{5.2b}$$

since $\mathbf{M}$ is a biorthonormal system with $\mathbf{\Phi}(\mathbf{\Psi})$ as its right(left) eigenvectors. Then, $\mathbf{M}^t = \sum_{k=1}^{n} \lambda_k^t \varphi_k \psi_k^T$, with $\mathbf{m}_{i,j}^{(t)}$ denoting distribution of being in vertex j having started from i. So the whole distribution of being having started from i can be denoted as

$$\mathbf{v}_i \to \mathbf{e}_i^T \mathbf{M}^t = \sum_{k=1}^{n} \lambda_k^t e_i \varphi_k \psi_k^T = \sum_{k=1}^{n} \lambda_k^t \varphi_{k,i} \psi_k^T$$

where $\lambda_k^t \varphi_{k,i}$ are the coefficients, and ψ_k^T the orthonormal basis.
If $\mathbf{v}_i, \mathbf{v}_j$ are close or strongly connected, then $e_i^T \mathbf{M}^t$ and $e_j^T \mathbf{M}^t$ are similar. Moreover, this diffusion map can be truncated to d dimensions,

$$\mathbf{\Phi}_t^{(d)}(i) = \begin{pmatrix} \lambda_2^t \varphi_{2,i} \\ \vdots \\ \lambda_{d+1}^t \varphi_{d+1,i} \end{pmatrix}. \tag{5.3}$$

Algorithm 2 Spectral Clustering Algorithm

1: **procedure** SPECTRAL CLUSTERING
2: Given a graph $G = (V, E, \mathbf{W})$, k denotes number of clusters, and time t
3: compute the $(k-1)$dimension diffusion map

$$\mathbf{\Phi}_t^{(k-1)}(i) = \begin{pmatrix} \lambda_2^t \varphi_{2,i} \\ \vdots \\ \lambda_k^t \varphi_{k,i} \end{pmatrix}$$

4: cluster $\mathbf{\Phi}_t^{(k-1)}(1), \ldots, \mathbf{\Phi}_t^{(k-1)}(n) \in \mathbb{R}^{k-1}$ using, eg, k-means clustering

The aim of spectral clustering is to cluster vertices of the graph into k clusters, as summarized in algorithm 2.
Particularly for two the case of clusters C and C^c we have that

$$\mathbf{\Phi}_t^{(1)} \in \mathbb{R}^1, i = 1, \ldots, n$$

will be on a line ($\mathbf{\Phi}_t^{(1)}$ is 1-dimensional, i.e., a scalar). Then, a natural way of clustering on a line is to define a threshold q such that $\mathbf{v}_i \in C$ if $\mathbf{\Phi}_t^{(1)}(i) \leqslant q$. For example, for 5 points we would get 1D clustering problem as the one shown in Figure 5.2.

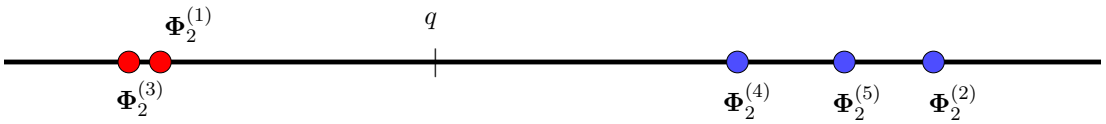


Figure 5.2: Example of spectral clustering with 2 classes and 5 points.

5.2.3 Hierarchical Clustering

Another type of unsupervised clustering algorithms are the hierarchical clustering methods. Hierarchical clustering algorithms are mainly divided into agglomerative (bottom up) and divisive (top down) clustering. Agglomerative methods assign a cluster to each observation and then reduce the number of clusters by iteratively merging smaller clusters into larger ones. On the other hand, divisive methods start with one large cluster containing all the observations and iteratively divide it into smaller clusters.
Since the principles behind agglomerative and divisive clustering are quite analogous, in this lecture we only explain agglomerative algorithms. To this end, given n objects $\mathbf{v}_1, \ldots, \mathbf{v}_n$ and pairwise dissimilarities $\delta_{i,j} = \delta_{j,i}, \mathbf{\Delta} = (\delta_{i,j})$ $i, j = 1, \ldots, n$, a linkage

function between two clusters C_1, C_2 is defined as

$$d(C_1, C_2) = \begin{cases} \min_{i \in C_1, j \in C_2} \delta_{i,j} & \textbf{single linkage} \\ \max_{i \in C_1, j \in C_2} \delta_{i,j} & \textbf{complete linkage} \\ \frac{1}{|C_1||C_2|} \sum_{i \in C_1, j \in C_2} \delta_{i,j} & \textbf{average linkage} \end{cases}. \tag{5.4}$$

Then, as summarized in algorithm 3, an agglomerative clustering algorithm builds larger clusters by merging similar clusters as we move up the hierarchy (see Figure 5.3). Note that algorithm 3 merges clusters until we are left with a single cluster containing all observations. In practice, we can stop iterating once the desired amount of clusters is reached.

Algorithm 3 Agglomerative Clustering

procedure AGGLOMERATIVE CLUSTERING
  Initialize clusters as singletons: **for** $i = 1, \ldots, n$ **do** $C_i \leftarrow i$
  Initialize the set of clusters available for merging as $S \leftarrow \{1, \ldots, n\}$
  repeat
    Pick the two most similar clusters to merge: $(j, k) \leftarrow \arg\min_{j,k \in S} d(C_j, C_k)$
    Merge C_k into C_j as $C_j \leftarrow C_j \bigcup C_k$
    Mark k as unavailable, $S \leftarrow S - \{k\}$
    If $C_j = \{1, \ldots, n\}$, then mark j as unavailable, $S \leftarrow S - \{j\}$.
    for each $i \in S$ **do** update dissimilarities $d(C_i, C_j)$
  until no more clusters are available for merging

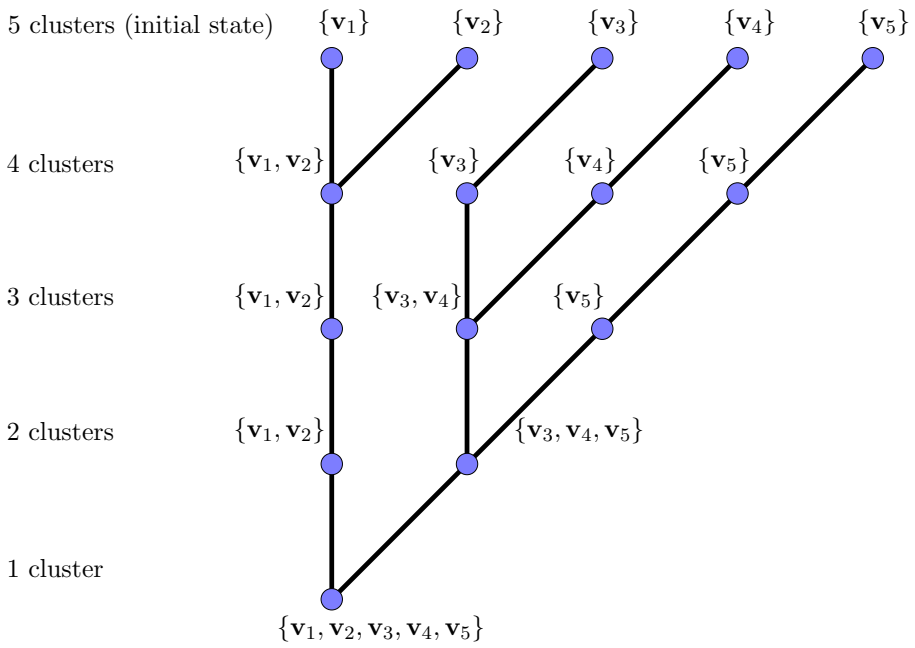


Figure 5.3: Graphical example of agglomerative clustering with 5 points ($n = 5$).

6 Support-Vector Machines

Given a training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, $\mathbf{x}_i \in \mathbb{R}^p$ are data points belonging to two different classes or groups. The class membership is indicated by $y_i \in \{-1, 1\}$.

The key idea is to select a particular hyperplane that separates the points in the two classes and maximizes the *margin*, i.e., the distance between the hyperplane and the closest points of the training set from each class.

The basic concept is drafted in Fig. 6.1.

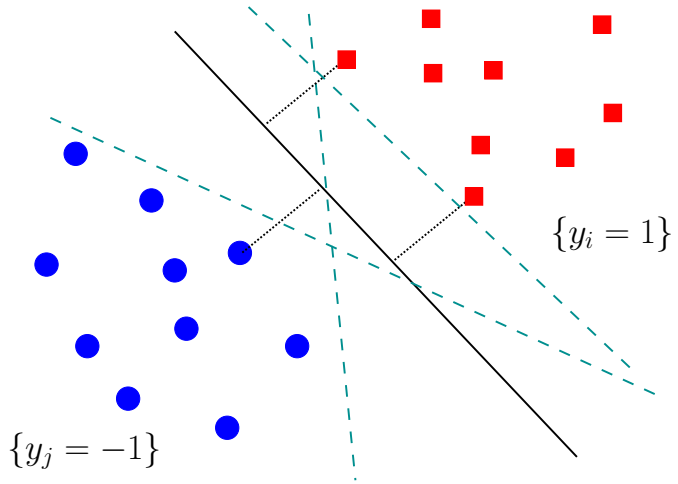


Figure 6.1: Potential separating hyperplanes (dashed, cyan) and the margin optimizing one (solid, black).

Support-Vector Machines are a set of learning methods used for classification, regression and outlier detection. They are among the best "off-the-shelf" supervised learning algorithms, if not even the best. Since only support-vectors are used for decisions SVMs are also memory efficient. They perform extremely effective if the number of dimensions is high, even in cases where the number of samples is smaller than the number of dimensions. The application of so called *kernel functions* is a way to generalize SVMs to nonlinear decision patterns, which makes SVMs very flexible and versatile.

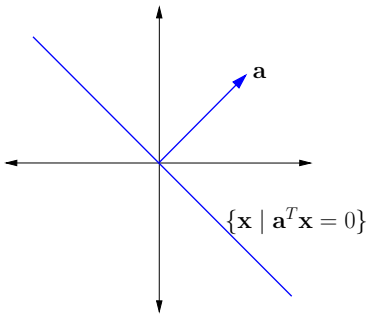
6.1 Hyperplanes and Margins

We commence by briefly considering the representation of hyperplanes and half-spaces in $\mathbb{R}^p$.
Suppose that $\mathbf{a} \in \mathbb{R}^p$ and $b \in \mathbb{R}$ are given.

a) Given $\mathbf{a} \in \mathbb{R}^p$

$$\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} = 0\}$$

is the $(p-1)$-dimensional linear subspace orthogonal to $\mathbf{a}$.

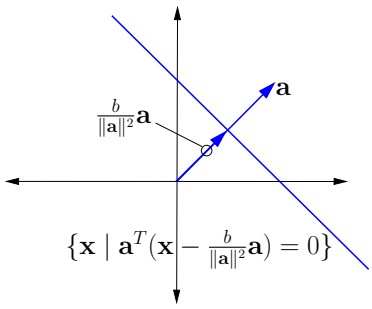


b) Given $\mathbf{a} \in \mathbb{R}^p$ and $b \in \mathbb{R}$

$$\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} - b = 0\}$$

is the linear space shifted by the vector $\frac{\mathbf{b}}{\|\mathbf{a}\|^2}\mathbf{a}$. This holds since $\mathbf{a}^T\mathbf{x} - b = 0$ if and only if $\mathbf{a}^T\mathbf{x} - \frac{\mathbf{a}^T\mathbf{a}}{\|\mathbf{a}\|^2}b = 0$ if and only if $\mathbf{a}^T\left(\mathbf{x} - \frac{b}{\|\mathbf{a}\|^2}\mathbf{a}\right) = 0$.

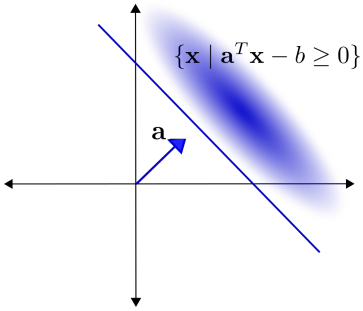
Hence, a linear space shifted by $\frac{b}{\|\mathbf{a}\|^2}\mathbf{a}$, is a hyperplane of distance $\frac{b}{\|\mathbf{a}\|}$ from $\{\mathbf{a}^T\mathbf{x} = 0\}$.



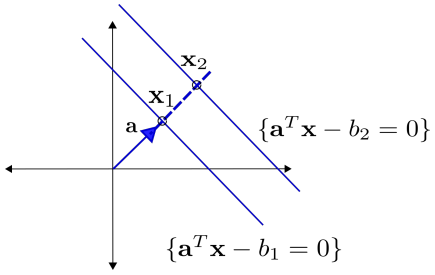
c) Given $\mathbf{a} \in \mathbb{R}^p$ and $b \in \mathbb{R}$

$$\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} \geq b\}$$

represents a half-space of points lying on one side of the corresponding hyperplane.



d) Given $\mathbf{a} \in \mathbb{R}^p$, $b_1, b_2 \in \mathbb{R}$, the distance between two hyperplanes is required $H_1 = \{\mathbf{a}^T\mathbf{x} - b_1 = 0\}$ and $H_2 = \{\mathbf{a}^T\mathbf{x} - b_2 = 0\}$.



Both hyperplanes are parallel and orthogonal to $\mathbf{a}$. Pick $\mathbf{x}_1$ and $\mathbf{x}_2$ such that:

$$\begin{aligned} \mathbf{x}_1 = \lambda_1\mathbf{a}, \quad & \mathbf{x}_2 = \lambda_2\mathbf{a} \\ \mathbf{a}^T\mathbf{x}_1 - b_1 = 0, \quad & \mathbf{a}^T\mathbf{x}_2 - b_2 = 0. \end{aligned}$$

Then:

$$\begin{aligned} \lambda_1\mathbf{a}^T\mathbf{a} - b_1 = 0, \quad & \lambda_2\mathbf{a}^T\mathbf{a} - b_2 = 0 \\ \lambda_1\|\mathbf{a}\|^2 - b_1 = 0, \quad & \lambda_2\|\mathbf{a}\|^2 - b_2 = 0 \\ \lambda_1 = \frac{b_1}{\|\mathbf{a}\|^2}, \quad & \lambda_2 = \frac{b_2}{\|\mathbf{a}\|^2}. \end{aligned}$$

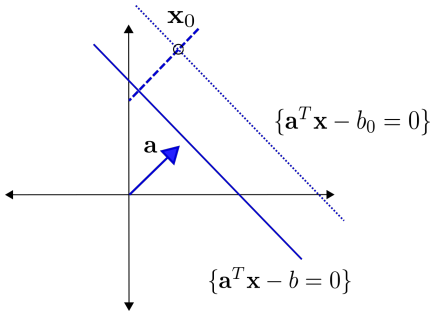
See that $\mathbf{x}_1 - \mathbf{x}_2$ is orthogonal to both hyperplanes and therefore the norm of this vector gives the distance between the two hyperplanes:

$$\begin{aligned} \|\mathbf{x}_1 - \mathbf{x}_2\| &= \|\lambda_2\mathbf{a} - \lambda_1\mathbf{a}\| = |\lambda_1 - \lambda_2|\|\mathbf{a}\| \\ &= \left(\frac{b_2}{\|\mathbf{a}\|^2} - \frac{b_1}{\|\mathbf{a}\|^2}\right)\|\mathbf{a}\| = \frac{|b_2 - b_1|}{\|\mathbf{a}\|}. \end{aligned}$$

Hence the distance between parallel H_1 and H_2 is:

$$\frac{1}{\|\mathbf{a}\|}|b_1 - b_2|.$$

e) Given $\mathbf{a} \in \mathbb{R}^p$, $b \in \mathbb{R}$ and $\mathbf{x}_0 \in \mathbb{R}^p$, the distance between the hyperplane $H_1 = \{\mathbf{x} \mid \mathbf{a}^T\mathbf{x} - b = 0\}$ and the point $\mathbf{x}_0$ is required.



Consider the auxiliary hyperplane containing $\mathbf{x}_0$:

$$H_0 = \{\mathbf{x} \mid \mathbf{a}^T\mathbf{x} - b_0 = 0\} = \{\mathbf{x} \mid \mathbf{a}^T\mathbf{x} - \mathbf{a}^T\mathbf{x}_0 = 0\}.$$

Note that $b_0 = \mathbf{a}^T\mathbf{x}_0$ since $\mathbf{a}^T\mathbf{x}_0 - b_0 = 0$. By the previous step, the distance between H and H_0 is:

$$\frac{1}{\|\mathbf{a}\|}|b - \mathbf{a}^T\mathbf{x}_0|.$$

The distance is called the margin of $\mathbf{x}_0$.

A hyperplane $\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} = b\}$ is separating points of two classes, if one group is contained in the half-space $\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} \geq b\}$ and the other group in $\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} \leq b\}$, see Fig. 6.1. The minimum distance from points to the separating hyperplane is called *margin*. Our aim is to find a hyperplane which maximizes the margin.

6.2 The Optimal Margin Classifier

We now set out to formulate an optimization problem that will give us the separating hyperplane with maximum margins. Given a training set

$$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}, \quad \mathbf{x}_i \in \mathbb{R}^p,\ y_i \in \{-1, 1\},$$

assume there exists a separating hyperplane

$$\{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{a}^T\mathbf{x} + b = 0\}.$$

Then for some $\gamma \geq 0$ and for all $i = 1, \ldots, n$, we have:

$$\begin{aligned} y_i = +1 \quad &\Longrightarrow \quad \mathbf{a}^T\mathbf{x}_k + b \geq \gamma \\ y_i = -1 \quad &\Longrightarrow \quad \mathbf{a}^T\mathbf{x}_k + b \leq -\gamma. \end{aligned}$$

Hence

$$y_i\big(\mathbf{a}^T\mathbf{x}_i + b\big) \geq \gamma \text{ for some } \gamma \geq 0, \text{ for all } i = 1, \ldots, n.$$

The minimum margin of the members of each class from the separating hyperplane is given by

$$\frac{1}{\|\mathbf{a}\|}\gamma.$$

The objective is to find a separating hyperplane such that this minimum margin is maximum.

$$\max_{\gamma,\mathbf{a},b} \frac{\gamma}{\|\mathbf{a}\|} \quad \text{s.t.} \quad y_i\big(\mathbf{a}^T\mathbf{x}_i + b\big) \geq \gamma \text{ for all } i = 1,\ldots,n.$$

This problem is scale invariant. In other words if $(\gamma, \mathbf{a}, b)$ is a solution so is $(2\gamma, 2\mathbf{a}, 2b)$. Therefore a normalization by γ leads to the following formulation

$$\begin{aligned}
&\min_{\gamma,\mathbf{a},b} \|\frac{\mathbf{a}}{\gamma}\| \quad \text{s.t.} \quad y_i\left((\frac{\mathbf{a}}{\gamma})^T\mathbf{x}_i + \frac{b}{\gamma}\right) \geq 1 \text{ for all } i = 1,\ldots,n \\
\iff &\min_{\mathbf{a},b} \|\mathbf{a}\| \quad \text{s.t.} \quad y_i\big(\mathbf{a}^T\mathbf{x}_i + b\big) \geq 1 \text{ for all } i = 1,\ldots,n \\
\iff &\min_{\mathbf{a},b} \frac{1}{2}\|\mathbf{a}\|^2 \quad \text{s.t.} \quad y_i\big(\mathbf{a}^T\mathbf{x}_i + b\big) \geq 1 \text{ for all } i = 1,\ldots,n.
\end{aligned}$$

This leads to the optimization problem for finding *Optimal Margin Classifier* (OMC):

$$\begin{aligned}
&\min_{\mathbf{a}\in\mathbb{R}^p,\, b\in\mathbb{R}} \frac{1}{2}\|\mathbf{a}\|^2 \\
&\text{such that } y_i\big(\mathbf{a}^T\mathbf{x}_i + b\big) \geq 1,\ i = 1,\ldots,n.
\end{aligned} \tag{OMC}$$

This is a quadratic optimization problem with linear inequality constraints, a special case of a convex general optimization problem.

- Assume $\mathbf{a}^*$ is an optimal solution of (OMC) and a nearest support point $\mathbf{x}_k$ and $\mathbf{x}_\ell$ from each class is known, as will become clear later from duality theory, then

  $$\begin{aligned}
  \mathbf{a}^{*T}\mathbf{x}_k + b &= 1, \\
  \mathbf{a}^{*T}\mathbf{x}_\ell + b &= -1.
  \end{aligned}$$

  Hence

  $$\mathbf{a}^{*T}\mathbf{x}_k + \mathbf{a}^{*T}\mathbf{x}_\ell + 2b = 0$$

  yielding

  $$b^* = -\frac{1}{2}\mathbf{a}^{*T}\big(\mathbf{x}_k + \mathbf{x}_\ell\big) \tag{6.1}$$

  as the optimum b-value.

  Even using a single point $\mathbf{x}_k$ satisfying $y_k(\mathbf{a}^{*T}\mathbf{x}_k + b^*) = 1$ is enough to obtain b^* as

  $$\begin{aligned}
  &y_k(\mathbf{a}^{*T}\mathbf{x}_k + b^*) = 1 \\
  \iff &\mathbf{a}^{*T}\mathbf{x}_k + b^* = y_k \quad (\text{since } y_k^2 = 1) \\
  \iff &b^* = y_k - \mathbf{a}^{*T}\mathbf{x}_k.
  \end{aligned}$$

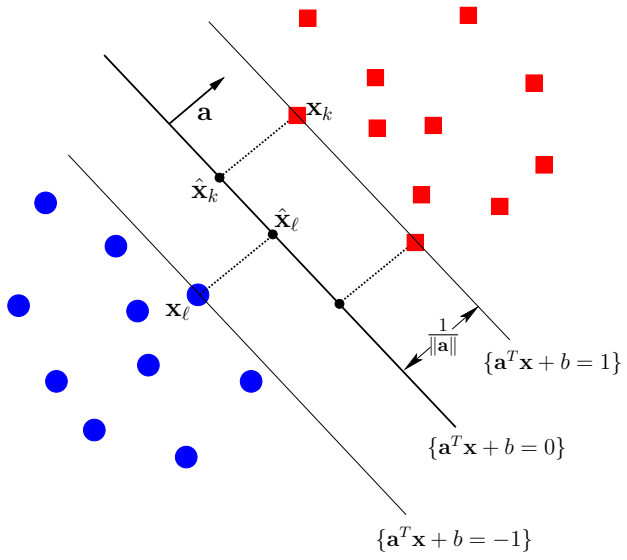


Figure 6.2: The margin maximizing hyperplane (solid, black) and its parallel sisters.

- The solution $(\mathbf{a}^*, b^*)$ is called the *optimal margin classifier*.
- It can be solved by commercial or public domain generic quadratic programming (QP) software.

Is the problem completely solved by now? Yes and no. We can do better by applying Lagrange duality theory. By this we not only get the optimum solution efficiently but also identify all support points. Moreover, a simple decision rule can be derived, which only depends on the support points. Moreover the case of non-separable data should also be considered.

6.3 SVM and Lagrange Duality

Let's start with the brief excursion on convex optimization.

- Normally a convex optimization problem can be formulated as,

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq b_i,\ i = 1, \ldots, m. \\ & h_j(x) = 0,\ j = 1, \ldots, p \end{aligned}$$

  where $f_0(x)$ and $f_i(x)$ are convex and $h_j(x)$ are linear.

- Lagrangian function is derived from primal optimization problem:

$$L(x, \boldsymbol{\lambda}, \boldsymbol{\upsilon}) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{p} \upsilon_j h_j(x).$$

- Lagrangian dual function is defined as the infimum of Lagrangian function:

$$g(\boldsymbol{\lambda}, \boldsymbol{\upsilon}) = \inf_{x \in \mathcal{D}} L(x, \boldsymbol{\lambda}, \boldsymbol{\upsilon}) \tag{6.2}$$

$$\mathcal{D} = \bigcap_{i=0}^{m} \mathrm{dom}(f_i) \cap \bigcap_{j=0}^{p} \mathrm{dom}(h_j) \tag{6.3}$$

  The Lagrangian dual function is a concave function.

- The Lagrange dual problem is derived as follows,

$$\begin{aligned} &\text{maximize} \quad g(\boldsymbol{\lambda}, \boldsymbol{\upsilon}) \\ &\text{subject to} \quad \boldsymbol{\lambda} \geq 0. \end{aligned} \tag{6.4}$$

- Suppose that $\boldsymbol{\lambda}^*, \boldsymbol{\upsilon}^*$ are the optimal solutions of Lagrangian duality problem, and x^* is the optimal solution of primal optimization problem. The weak duality theorem states that

$$g(\boldsymbol{\lambda}^*, \boldsymbol{\upsilon}^*) \leq f_0(x^*).$$

  The strong duality holds if

$$g(\boldsymbol{\lambda}^*, \boldsymbol{\upsilon}^*) = f_0(x^*).$$

- If the constraints are linear then "the Slater's condition" holds, which implies that $g(\boldsymbol{\lambda}^*, \boldsymbol{\upsilon}^*) = f_0(x^*)$, i.e., strong duality holds, i.e., and therefore the duality gap is zero.

- If strong duality holds, then

$$\begin{aligned} f_0(x^*) &= g(\boldsymbol{\lambda}^*, \boldsymbol{\upsilon}^*) \\ &= \inf_x \left(f_0(x) + \sum_{i=1}^{m} \lambda_i^* f_i(x) + \sum_{i=1}^{p} \upsilon_i^* h_i(x) \right) \\ &\leq f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* f_i(x^*) + \sum_{i=1}^{p} \upsilon_i^* h_i(x^*) \\ &\leq f_0(x^*), \end{aligned}$$

  since $\lambda_i^* \geq 0$, $f_i(x^*) \leq 0$ and $h_i(x^*) = 0$. Hence, equality holds everywhere such that

  (i) x^* minimizes $L(x, \boldsymbol{\lambda}^*, \boldsymbol{\upsilon}^*)$,

(ii) $\lambda_i^* f_i(x^*) = 0, i = 1, \ldots, m$ (Complementary slackness), i.e.,

$$\begin{aligned} \lambda_i^* > 0 &\implies f_i(x^*) = 0 \\ f_i(x^*) < 0 &\implies \lambda_i^* = 0. \end{aligned}$$

- Karush-Kuhn-Tucher conditions (KKT) is defined by
  1. $f_i(x) \leq 0, i = 1, \ldots, m$, $h_j(x) = 0, j = 1, \ldots, p$ (primal constraints)
  2. $\lambda_i \geq 0, i = 1, \ldots, m$ (dual constraints)
  3. $\lambda_i f_i(x) = 0, i = 1, \ldots, m$ (complementary slackness)
  4. $\nabla_x L(x, \boldsymbol{\lambda}, \boldsymbol{\upsilon}) = 0$

Theorem 6.1. *If slater's condition is satisfied then the strong duality holds. If in addition f_i, h_j are differentiable, then for x^*, (λ^*, υ^*) to be primal and dual optimal, it is necessary and sufficient that the KKT condition holds.*

We can see the application to SVM. For example, given training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$. The primal optimization problem is given by

$$\begin{aligned} \min_{\mathbf{a} \in \mathbb{R}^p, b \in \mathbb{R}} \quad & \frac{1}{2}\|\mathbf{a}\|^2 \\ \text{s.t} \quad & y_i(\mathbf{a}^T \mathbf{x}_i + b) \geq 1, i = 1, \ldots, n \end{aligned} \tag{6.5}$$

The Lagrangian function writes as

$$L(\mathbf{a}, b, \boldsymbol{\lambda}) = \frac{1}{2}\|\mathbf{a}\|^2 - \sum_{i=1}^{n} \lambda_i (y_i(\mathbf{a}^T \mathbf{x}_i + b) - 1)$$

$$\frac{\partial(\mathbf{a}, b, \boldsymbol{\lambda})}{\partial \mathbf{a}} = \mathbf{a} - \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i = 0 \Longrightarrow \mathbf{a}^* = \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i,$$

$$\frac{\partial(\mathbf{a}, b, \boldsymbol{\lambda})}{\partial b} = \sum_{i=1}^{n} \lambda_i y_i = 0 \Longrightarrow \sum_{i=1}^{n} \lambda_i y_i = 0$$

Dual function writes as

$$\begin{aligned} g(\boldsymbol{\lambda}) = L(\mathbf{a}^*, b^*, \boldsymbol{\lambda}) &= \frac{1}{2}\|\mathbf{a}^*\|^2 - \sum_{i=1}^{n} \lambda_i (y_i(\mathbf{a}^{*T} \mathbf{x}_i + b^*) - 1) \\ &= \sum_{i=1}^{n} \lambda_i + \frac{1}{2}(\sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i)^T (\sum_{j=1}^{n} \lambda_j y_j \mathbf{x}_j) - \sum_{i=1}^{n} \lambda_i y_i (\sum_{j=1}^{n} \lambda_j y_j \mathbf{x}_j)^T x_i - \sum_{i=1}^{n} \lambda_i y_i b^* \\ &= \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} y_i y_j \lambda_i \lambda_j \mathbf{x}_i^T \mathbf{x}_j \end{aligned}$$

Finally the dual problem can be obtained as

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \quad & g(\boldsymbol{\lambda}) = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} y_i y_j \lambda_i \lambda_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{s.t} \quad & \lambda_i \geq 0 \\ & \sum_{i=1}^{n} \lambda_i y_i = 0 \end{aligned} \tag{(DP)}$$

If λ_i^*'s are the solution of the dual problem, then $\mathbf{a}^* = \sum_{i=1}^{n} \lambda_i^* y_i \mathbf{x}_i$ and $b^* = y_k - \mathbf{a}^{*T} \mathbf{x}_k$ with $\mathbf{x}_k$ are some support vector. The Slater's condition is also satisfied, so strong duality holds. From KKT for optimal $\boldsymbol{\lambda}^*$ complementary slackness follows

$$\lambda_i^* (y_i (\mathbf{a}^{*T} \mathbf{x}_i + b^*) - 1) = 0, i = 1, \ldots, n$$

Hence

$$\lambda_i^* > 0 \Longrightarrow y_i (\mathbf{a}^{*T} \mathbf{x}_i + b^*) = 1 \tag{6.6}$$

$$\lambda_i^* = 0 \Longrightarrow y_i (\mathbf{a}^{*T} \mathbf{x}_i + b^*) \geq 1 \tag{6.7}$$

$\lambda_i^* > 0$ indicates supporting vectors, those which have smallest distance to the separating hyperplane. After solving ((DP)) the support vectors are determined by $\lambda_i^* > 0$.
Let $S = \{i | \lambda_i^* > 0\}$, $S_+ = \{i \in S | y_i = +1\}$ and $S_- = \{i \in S | y_i = -1\}$. Then

$$\begin{aligned} \mathbf{a}^* &= \sum_{i \in S} \lambda_i^* y_i \mathbf{x}_i \\ b^* &= -\frac{1}{2} \mathbf{a}^{*T} (\mathbf{x}_k + \mathbf{x}_l), \quad k \in S_+, l \in S_- \end{aligned}$$

SVM can be then simply stated as follows

- Training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$.
- Determine λ^* and $\mathbf{a}^*, b^*$
- For each new point $\mathbf{x}$, to find class label $y \in \{-1, 1\}$, first compute

  $$d(x) = \mathbf{a}^{*T} \mathbf{x} + b^* = (\sum_{i \in S} \lambda_i^* y_i \mathbf{x}_i)^T \mathbf{x} + b^* = \sum_{i \in S} \lambda_i^* y_i \mathbf{x}_i^T \mathbf{x} + b^*$$

  and then predict $y = 1$, if $d(x) \geq 0$, otherwise $y = -1$.

Remarks:

- $|S|$ is normally much less than n.
- The decision only depends on the inner products $\mathbf{x}_i^T \mathbf{x}$ for support-vectors $\mathbf{x}_i, i \in S$.

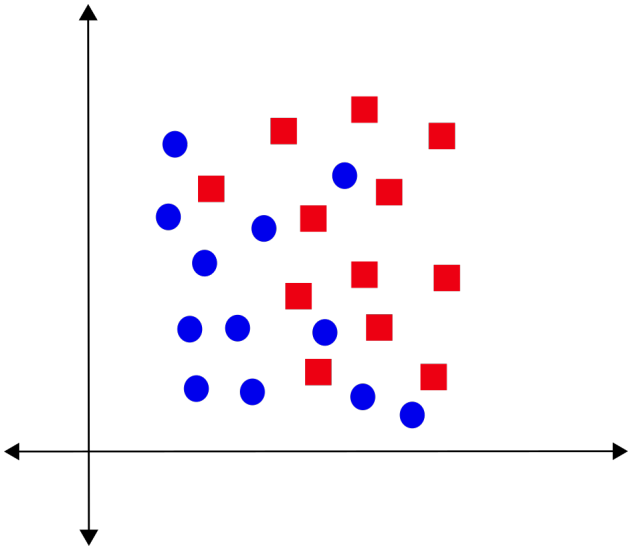
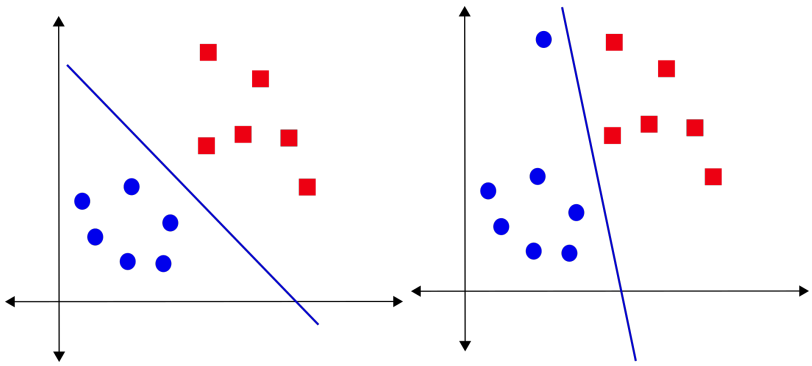


Figure 6.3: A linearly non-separable dataset

6.4 Robustness and Non-separability

So far, the assumption that there exists a separating hyperplane between two classes. What happens if not? For example, the points in Fig. 6.3 are not linearly separable. Moreover the optimum margin classifier is sensitive to outliers. Outliers cause drastic swing of the optimal margin classifier.



Both problems are addressed by the following approach: ℓ_1-regularization.

$$\begin{aligned} \min_{\mathbf{a}\in\mathbb{R}^p, b, \xi} \quad & \frac{1}{2}\|\mathbf{a}\|^2 - c\sum_{i=1}^{n}\xi_i \\ \text{s.t} \quad & y_i(\mathbf{a}^T\mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, \ldots, n \\ & \xi_i \geq 0, i = 1, \ldots, n \end{aligned} \tag{6.8}$$

For the optimal solution $\mathbf{a}^*, b^*$, it is allowed that margins are less than $\frac{1}{\|\mathbf{a}^*\|}$, i.e.,

$$y_i(\mathbf{a}^{*T}\mathbf{x}_i + b^*) \leq 1.$$

If $y_i(\mathbf{a}^{*T}\mathbf{x}_i + b^*) = 1 - \xi_i, \xi_i > 0$, then a cost of $c\xi_i$ is paid. Parameter c controls the balance between the two goals in (6.8).
Lagrangian for (6.8) is given by:

$$L(\mathbf{a}, b, \xi, \boldsymbol{\lambda}, \boldsymbol{\gamma}) = \frac{1}{2}\|\mathbf{a}\|^2 + c\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\lambda_i(y_i(\mathbf{a}^T\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{n}\gamma_i\xi_i$$

where γ_i, λ_i's are Lagrangian multipliers. Analogously to the above, obtain the dual problem as

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \quad & \sum_{i=1}^{n}\lambda_i - \frac{1}{2}\sum_{i,j} y_i y_j \lambda_i \lambda_j \mathbf{x}_i^T\mathbf{x}_j \\ \text{s.t} \quad & 0 \leq \lambda_i \leq c \\ & \sum_{i=1}^{n}\lambda_i y_i = 0 \end{aligned} \tag{6.9}$$

Let λ_i^* be the optimum solution of (6.9). As before, let $S = \{i|\lambda_i^* > 0\}$. Corresponding $\mathbf{x}_i$'s are called support vectors. Then $\mathbf{a}^* = \sum_{i\in S}\lambda_i^* y_i \mathbf{x}_i$ is the optimum $\mathbf{a}$.
Complementary slackness conditions are

$$\lambda_i = 0 \Longrightarrow y_i(\mathbf{a}^{*T}\mathbf{x}_i + b_i) \geq 1 \tag{6.10}$$

$$\lambda_i = c \Longrightarrow y_i(\mathbf{a}^{*T}\mathbf{x}_i + b_i) \leq 1 \tag{6.11}$$

$$0 < \lambda_i < c \Longrightarrow y_i(\mathbf{a}^{*T}\mathbf{x}_i + b_i) = 1 \tag{6.12}$$

$$\tag{6.13}$$

If $0 < \lambda_k < c$ for some k ($\mathbf{x}_k$ support vector), then $b^* = y_k - \mathbf{a}^{*T}\mathbf{x}_k$ providing the optimum b.
Also it is possible to pick two support vectors $\mathbf{x}_k$ and $\mathbf{x}_\ell$ with $y_k = +1$ and $y_\ell = -1$, then

$$\left.\begin{aligned} b^* &= y_k - \mathbf{a}^{*T}\mathbf{x}_k \\ b^* &= y_\ell - \mathbf{a}^{*T}\mathbf{x}_\ell \end{aligned}\right\} \implies 2b^* = -\mathbf{a}^{*T}(\mathbf{x}_k + \mathbf{x}_\ell),$$

hence $b^* = -\frac{1}{2}\mathbf{a}^{*T}(\mathbf{x}_k + \mathbf{x}_\ell)$ is the optimum b.
To classify a new point $\mathbf{x} \in R^p$, there are two classifiers.

- Hard classifier: first compute:

$$\mathbf{a}^{*T}\mathbf{x} + b^* = (\sum_{i\in S}\lambda_i^* y_i \mathbf{x}_i)^T\mathbf{x} + b^* = \sum_{i\in S}\lambda_i^* y_i \mathbf{x}_i^T\mathbf{x} + b^* = d(\mathbf{x}).$$

  Decide $y = 1$ if $d(\mathbf{x}) \geq 0$, otherwise $y = -1$.

- Soft classifier: Compute $d(\mathbf{x}) = h(\mathbf{a}^{*T}\mathbf{x} + b^*)$ where

$$h(t) = \begin{cases} -1, & t < -1 \\ t, & -1 \leq t \leq +1 \\ +1, & t > 1 \end{cases} \tag{6.14}$$

  $d(\mathbf{x})$ is a real number in $[-1, +1]$ if $\mathbf{a}^{*T}\mathbf{x} + b^* \in [-1, +1]$, i.e., if x is residing in the overlapping area.

Both classifiers only depend on the inner products $\mathbf{x}_i^T\mathbf{x} = \langle \mathbf{x}_i, \mathbf{x} \rangle$ with support vectors $\mathbf{x}_i, i \in S$.

6.5 The SMO Algorithm

The Sequential Minimal Optimization (SMO) algorithm is an algorithm to solve the dual problem, which is given as

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \quad & W(\boldsymbol{\lambda}) = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} y_i y_j \lambda_i \lambda_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{s.t.} \quad & 0 \leq \lambda_i \leq c \\ & \sum_{i=1}^{n} \lambda_i y_i = 0. \end{aligned} \tag{6.15}$$

Assume $\boldsymbol{\lambda}$ is a feasible point, i.e., $\boldsymbol{\lambda}$ satisfies the constraints. Note that *cyclic coordinate optimization* does not work, since, e.g.,

$$\lambda_1 y_1 = -\sum_{i=2}^{n} \lambda_i y_i \quad \text{or} \quad \lambda_1 = -y_1 \sum_{i=2}^{n} \lambda_i y_i.$$

Hence each λ_j is determined by fixing λ_i, $i \neq j$. The idea is to update at least two λ_j simultaneously, which is SMO algorithm.

Algorithm 4 SMO algorithm

1: **procedure** SMO
2: **repeat**
3: 1. Select a pair (i, j) to be updated next, the one which promises the most progress
4: 2. Optimize $W(\boldsymbol{\lambda})$ w.r.t. λ_i and λ_j while keeping λ_k, $k \neq i, j$, fixed.
5: **until** Convergence

One can check KKT within a tolerance limit $\epsilon = 0.01$ or 0.001, to verify Convergence. Optimize $W(\boldsymbol{\lambda})$ w.r.t. λ_1, λ_2 with $\lambda_3, \ldots, \lambda_n$ fixed and $\boldsymbol{\lambda}$ feasible. It holds

$$\lambda_1 y_1 + \lambda_2 y_2 = -\sum_{i=3}^{n} \lambda_i y_i = \zeta, \quad \zeta \text{ fixed.}$$

Derive the following:

$$\begin{aligned} 0 \leq \lambda_1, \lambda_2 \leq c \\ L \leq \lambda_2 \leq H. \end{aligned} \tag{6.16}$$

Moreover:

$$\lambda_1 = y_1(\zeta - \lambda_2 y_2). \tag{6.17}$$

Hence $W(\lambda_1, \ldots, \lambda_n) = W(y_1(\zeta - \lambda_2 y_2), \lambda_2, \underbrace{\lambda_3, \ldots, \lambda_n}_{\text{fixed}})$ and therefore the objective function turns out to be a quadratic function of λ_2 and it can be written as :

$$\gamma_2 \lambda_2^2 + \gamma_1 \lambda_2 + \gamma_0.$$

Determine the maximum by differentiation:

$$2\gamma_2 \lambda_2 + \gamma_1 = 0 \implies \lambda_2 = -\frac{\gamma_1}{2\gamma_2}$$

with optimum solution $\lambda_2^{(r)}$. The final solution, following (6.16), is

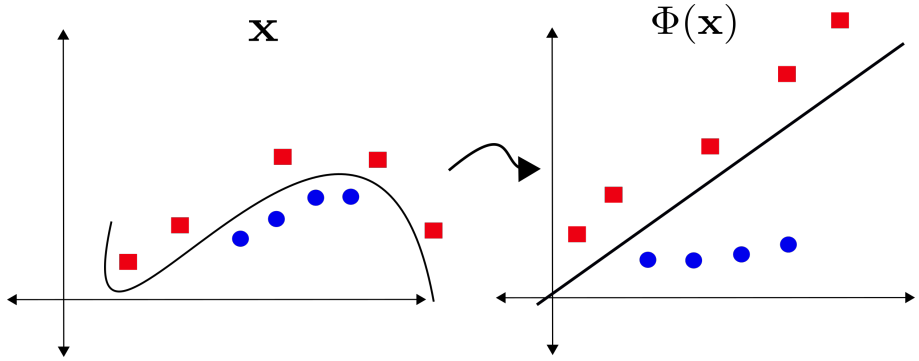
$$\lambda_2^{(c)} = \begin{cases} H & \text{if } \lambda_2^{(r)} > H \\ \lambda_2^{(r)} & \text{if } L \leq \lambda_2^{(r)} \leq H \\ L & \text{if } \lambda_2^{(r)} < L \end{cases} \tag{6.18}$$

λ_1 is computed using (6.17).
It still remains to clarify:

- What is the best choice of the next pair (i, j) to update?
- How to update the coefficients $\gamma_0, \gamma_1, \gamma_2$ in the run of SMO.
- The algorithm converges, however, the right choice of (i, j) in each step accelerates the rate of Convergence.
- Generalization of SMO algorithm [OFG97]

6.6 Kernels

Instead of applying SVM to the raw data ("attributes") $\mathbf{x}_i$, one can apply it to transformed data ("features") $\Phi(\mathbf{x}_i)$. The function Φ is called feature mapping. The aim of kernels is to achieve better separability.



Consider the dual SVM problem.

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \quad & g(\boldsymbol{\lambda}) = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} y_i y_j \lambda_i \lambda_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{s.t.} \quad & 0 \leq \lambda_i \leq c \\ & \sum_{i=1}^{n} \lambda_i y_i = 0. \end{aligned} \tag{6.19}$$

$g(\boldsymbol{\lambda})$ only depends on the inner products $\mathbf{x}_i^T \mathbf{x}_j$. Substitute $\mathbf{x}_i$ by $\Phi(\mathbf{x}_i)$ and use the same inner product $\langle , \rangle$. Replace $\mathbf{x}_i^T \mathbf{x}_j$ by,

$$\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = K(\mathbf{x}_i, \mathbf{x}_j).$$

In other words, the inner product of the features are given by the function $K(\mathbf{x}_i, \mathbf{x}_j)$. Note that $K(\mathbf{x}_i, \mathbf{x}_j)$ is often easier to compute than $\Phi(\mathbf{x})$ itself.
The intuition why we need kernels is that if $\Phi(\mathbf{x}), \Phi(\mathbf{y})$ are close, $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$ is large. If $\Phi(\mathbf{x}) \perp \Phi(\mathbf{y})$ then $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = 0$. Hence, $K(\mathbf{x}_i, \mathbf{x}_j)$ measures how similar $\mathbf{x}$ and $\mathbf{y}$ are. What is needed for Kernel based methods is an inner product in some feature space $\{\Phi(\mathbf{x}) | \mathbf{x} \in \mathbb{R}^p\}$.

Example. Given $\mathbf{x}, \mathbf{z} \in \mathbb{R}^p$ define the Kernel functions as

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle^2 = \left(\sum_{i=1}^{p} x_i z_i \right)^2$$

The question is whether there is some function Φ such that $\langle \mathbf{x}, \mathbf{z} \rangle^2$ is an inner product in the feature space. Let $p = 2$ and $\mathbf{x} = (x_1, x_2)^T$ and $\mathbf{z} = (z_1, z_2)^T$. Use $\Phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2) : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$. Then

$$\begin{aligned} \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \\ &= (x_1 z_1 + x_2 z_2)^2 = \langle \mathbf{x}, \mathbf{z} \rangle^2. \end{aligned}$$

Example. (Gaussian Kernel) Given $\mathbf{x}, \mathbf{z} \in \mathbb{R}^p$, the kernel is defined as

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2})$$

Question: is there a feature mapping Φ and a feature space with inner product specified as above?

Definition 6.2. Kernel $K(\mathbf{x}, \mathbf{z})$ is called valid, if there exists a feature Φ such that $K(\mathbf{x}, \mathbf{z}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle$ for all $\mathbf{x}, \mathbf{z} \in \mathbb{R}^p$

Theorem 6.3 (Mercer's theorem)**.** *Given $K : \mathbb{R}^p \times \mathbb{R}^p \longrightarrow \mathbb{R}$, K is a valid kernel if and only if for any $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ the kernel matrix $(K(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1,\ldots,n}$ is nonnegative definite.*

Proof. (Only $\Longrightarrow$) If K is valid then there exists a function Φ such that:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)) \rangle = \langle \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_i) \rangle = K(\mathbf{x}_j, \mathbf{x}_i)$$

Moreover,

$$\mathbf{z}^T (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j} \mathbf{z} = \sum_{k,l} z_k z_l \langle \Phi(\mathbf{x}_k), \Phi(\mathbf{x}_l) \rangle = \langle \sum_k z_k \Phi(\mathbf{x}_k), \sum_l z_l \Phi(\mathbf{x}_l) \rangle \geq 0.$$

□

Example. (Polynomial Kernel) Consider the following kernel:

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z} + c)^d, \mathbf{x}, \mathbf{z} \in \mathbb{R}^p, c \in \mathbb{R}, d \in \mathbf{N}, d \geq 2.$$

Feature space of this kernel is of dimension $\binom{p+d}{d}$, containing all monomials of degree less that or equal to d.
(Exercise: Determine $\Phi(x)$.)

Kernels can also be constructed over infinite dimensional spaces,e.g, function space or probability distributions, providing a lot of modeling power. The solution of the optimal problem is still a convex problem with linear constraints.

7 Machine Learning

Recently, machine learning has captured the attention in many fields and been utilized in different applications, such as voice, image recognition and stock market prediction. Machine learning can be best defined as a set of methods which are able to extract information or pattern from a given data in order to learn and understand the system under consideration, and if necessary predict its future behavior [Mur12]. However, to ensure reliable functionalities, proper learning algorithms have to be considered based on the problem in hand.

7.1 Supervised Learning

Given the pair (x_i, y_i), with $i = 1, \ldots, n$, the set

$$\{(x_i, y_i) \mid i = 1, \ldots, n\}$$

is called the training set of $n \in \mathbb{N}$ number of training examples, where $x_i \in \mathcal{X}$ is known as the input or feature variable, and $y_i \in \mathcal{Y}$ as the output or target variable. Via supervised learning approach, machine learning algorithms utilize such a training set to determine a function or a hypothesis

$$h : \mathcal{X} \longrightarrow \mathcal{Y},$$

such that $h(x_i)$ is a "good" predictor of y_i. Once the learning is over, i.e, the parameters of the function h is determined, it can be utilized to test or identify new samples. Based on the nature of y_i, the problem is

- a regression one, if $\mathcal{Y}$ is continuous, or
- a classification one, if $\mathcal{Y}$ is discrete.

7.1.1 Linear Regression

Given the training set $\{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, n\}$, where $\mathbf{x}_i \in \mathbb{R}^p$, and $y_i \in \mathbb{R}$. The linear function

$$\begin{aligned} y_i &= \nu_0 + x_1\nu_1 + \cdots + x_p\nu_p + \epsilon_i \\ &= (1, \mathbf{x}_i^T)\boldsymbol{\nu} + \epsilon_i, \end{aligned}$$

is considered a linear regression model, where $\epsilon_i \in \mathbb{R}$ is a random error. Hence, the learning function is represented by $h_{\boldsymbol{\nu}}(\mathbf{x}) = (1, \mathbf{x}^T)\boldsymbol{\nu}$, where $\boldsymbol{\nu} = (\nu_0, \ldots, \nu_p)^T$ are the regression parameters. Thus, the regression problem can be redefined as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\nu} + \boldsymbol{\epsilon}, \tag{7.1}$$

where

$$\mathbf{X}_{n\times(p+1)} = \begin{pmatrix} 1 & \mathbf{x}_1^T \\ \vdots & \vdots \\ 1 & \mathbf{x}_n^T \end{pmatrix}, \qquad \mathbf{y} = (y_1, \ldots, y_n)^T, \qquad \text{and} \qquad \boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)^T.$$

In (7.1), the best $\boldsymbol{\nu}$, denoted by $\boldsymbol{\nu}^\star$, is required to be obtained by solving the minimization problem

$$\min_{\boldsymbol{\nu}\in\mathbb{R}^{p+1}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\nu}\|, \tag{7.2}$$

The solution can be performed in the following steps.

1. Find $\hat{\mathbf{y}}$ by projecting $\mathbf{y}$ onto $\mathrm{Im}(\mathbf{X})$
   The term $\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is an orthogonal projection onto $\mathrm{Im}(\mathbf{X})$, provided the inverse $(\mathbf{X}^T\mathbf{X})^{-1}$ exists. Note that the inverse $(\mathbf{X}^T\mathbf{X})^{-1}$ must exist, however if not, replace $(\mathbf{X}^T\mathbf{X})^{-1}$ by the so-called Moore-Penrose-inverse $(\mathbf{X}^T\mathbf{X})^{+}$.
   Thus,

$$\begin{aligned} \hat{\mathbf{y}} &= \arg\min_{\mathbf{z}\in\mathrm{Im}(\mathbf{X})} \|\mathbf{y} - \mathbf{z}\| \\ &= \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \end{aligned}$$

2. Find $\boldsymbol{\nu}$ such that $\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\nu}$

$$\begin{aligned} \hat{\mathbf{y}} &= \mathbf{X}\boldsymbol{\nu} \\ \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} &= \mathbf{X}\boldsymbol{\nu} \end{aligned}$$

   multiplying both sides by $\mathbf{X}^T$ results

$$\mathbf{X}^T\mathbf{y} = \mathbf{X}^T\mathbf{X}\boldsymbol{\nu}.$$

   Thus,

$$\boldsymbol{\nu}^\star = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

   and,

$$\mathbf{X}\boldsymbol{\nu}^\star = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{y} = \hat{\mathbf{y}}.$$

As an exercise, prove that $\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is the orthogonal projection onto $\mathrm{Im}\mathbf{X}$, provided that $(\mathbf{X}^T\mathbf{X})^{-1}$ exists.

Example. A 1-dimensional regression problem: Figure 7.1 shows a 1-dimensional regression problem, which can be formulated as follows.

$$y_i = \nu_0 + \nu_1 x_i + \epsilon_i, \quad \text{for } i = 1, \ldots, n.$$

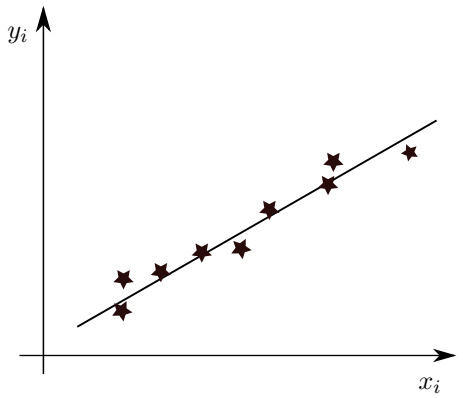


Figure 7.1: A 1-D regression problem.

Find the regression parameters ν_0 and ν_1.
Solution: Given that

$$\mathbf{X} = \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}, \qquad \text{and,} \qquad \boldsymbol{\nu} = \begin{pmatrix} \nu_0, \nu_1 \end{pmatrix}^T,$$

the term $\mathbf{X}^T\mathbf{X}$ is obtained as

$$\begin{aligned} \mathbf{X}^T\mathbf{X} &= \begin{pmatrix} 1 & \dots & 1 \\ x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \\ &= \begin{pmatrix} n & \sum_i x_i \\ \sum_i x_i & \sum_i x_i^2 \end{pmatrix}, \end{aligned}$$

hence

$$\begin{aligned} (\mathbf{X}^T\mathbf{X})^{-1} &= \frac{1}{n\sum_i x_i^2 - (\sum_i x_i)^2} \begin{pmatrix} \sum_i x_i^2 & -\sum_i x_i \\ -\sum_i x_i & n \end{pmatrix} \\ &= \frac{\frac{1}{n^2}}{\frac{1}{n}\sum_i x_i^2 - (\frac{1}{n}\sum_i x_i)^2} \begin{pmatrix} \sum_i x_i^2 & -\sum_i x_i \\ -\sum_i x_i & n \end{pmatrix} \\ &= \frac{1}{n}\frac{1}{\overline{x^2} - \overline{x}^2} \begin{pmatrix} \overline{x^2} & -\overline{x} \\ -\overline{x} & 1 \end{pmatrix}. \end{aligned}$$

Thus,

$$\begin{aligned}
\boldsymbol{\nu}^\star &= \begin{pmatrix} \nu_0^\star \\ \nu_1^\star \end{pmatrix} \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y \\
&= \frac{1}{n}\frac{1}{\overline{x^2} - \overline{x}^2}\begin{pmatrix} \overline{x^2} & -\overline{x} \\ -\overline{x} & 1 \end{pmatrix}\begin{pmatrix} n\overline{y} \\ \sum_i x_i y_i \end{pmatrix} \\
&= \frac{1}{\sigma_x^2}\begin{pmatrix} \overline{x^2} & -\overline{x} \\ -\overline{x} & 1 \end{pmatrix}\begin{pmatrix} \overline{y} \\ pxy \end{pmatrix} \\
&= \frac{1}{\sigma_x^2}\begin{pmatrix} \overline{x^2}\,\overline{y} - \overline{x}\,p\,x\,y \\ -\overline{x}\,\overline{y} + p\,x\,y \end{pmatrix} \\
&= \frac{1}{\sigma_x^2}\begin{pmatrix} \overline{x^2}\,\overline{y} - \overline{x}\,p\,x\,y \\ \sigma_{xy} \end{pmatrix}.
\end{aligned}$$

Hence, the regression parameters for a 1-dimensional problem can be directly calculated as

$$\nu_1^\star = \frac{\sigma_{xy}}{\sigma_x^2},$$

and

$$\nu_0^\star = \overline{y} - \nu_1^\star \overline{x},$$

where,

$$\begin{aligned}
\bar{x} &= \frac{1}{n}\sum_i x_i, \\
\bar{y} &= \frac{1}{n}\sum_i y_i, \\
\sigma_{xy} &= \frac{1}{n}\sum_i x_i y_i - \bar{x}\bar{y}, \\
\sigma_{x^2} &= \frac{1}{n}\sum_i x_i^2 - \bar{x}^2.
\end{aligned}$$

Prove as an exercise, that

$$\begin{aligned}
\nu_0^\star &= \overline{y} - \nu_1^\star \overline{x} \\
&= \frac{1}{\sigma_x^2}(\overline{x^2}\,\overline{y} - p\,x\,y\overline{x}).
\end{aligned}$$

7.1.2 Logistic Regression

The concept of regression can be extended for binary classification problems, i.e., problems where the output variable y can be either 0 or 1. Correspondingly, the hypothesis is give

by

$$h_{\boldsymbol{\nu}}(\mathbf{x}) = g(\boldsymbol{\nu}^T\mathbf{x}) = \frac{1}{1+\mathbb{E}(-\boldsymbol{\nu}^T\mathbf{x})} \in [0,1], \tag{7.3}$$

where $g(z) = \frac{1}{1+\mathbb{E}(-z)}$ is called the logistic or sigmoid function. Figure 7.2 represents the logistic function $g(z)$ for values $z \in [-5,5]$. As an exercise, prove that the statement $g'(z) = g(z)(1-g(z))$ holds.

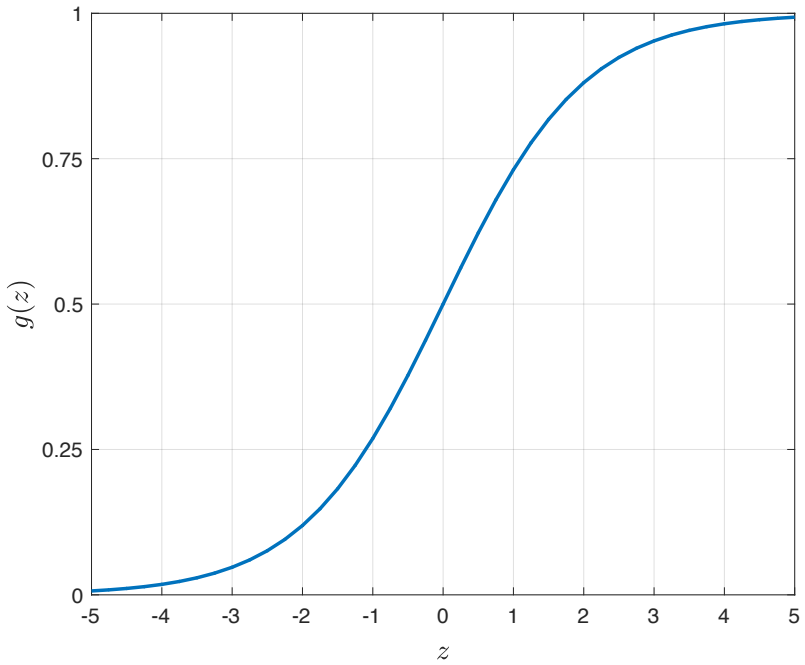


Figure 7.2: The logistic or sigmoid function $g(z)$ for $z \in [-5,5]$.

Given the training set $\{(\mathbf{x}_i, y_i) \mid i = 1,\dots,n\}$ training samples, where $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \{0,1\}$. Similar to the linear regression, we set $x_{i,0} = 1$, subject to

$$\boldsymbol{\nu}^T\mathbf{x} = \nu_0 + \sum_{j=1}^{p} \nu_j x_j, \quad \text{where } \mathbf{x} = (1, x_1, \dots, x_p).$$

Probabilistic Interpretation of Logistic Regression

The posterior probability of class $y = 1$ is interpreted via the logistic function, such that

$$\begin{aligned} \mathrm{P}(y=1|\,\mathbf{x},\boldsymbol{\nu}) &= h_{\boldsymbol{\nu}}(\mathbf{x}) \\ \mathrm{P}(y=0|\,\mathbf{x},\boldsymbol{\nu}) &= 1 - h_{\boldsymbol{\nu}}(\mathbf{x}). \end{aligned}$$

In other words, the posterior probability can be written as

$$\mathrm{P}(y=1|\,\mathbf{x},\boldsymbol{\nu}) = (h_{\boldsymbol{\nu}}(\mathbf{x}))^y\,(1-h_{\boldsymbol{\nu}}(\mathbf{x}))^{1-y}, \qquad y \in \{0,1\}.$$

Assuming $n \in \mathbb{N}$ independent training samples, such that

$$\mathbf{X} = \begin{pmatrix} 1 & \mathbf{x}_1^T \\ \vdots & \vdots \\ 1 & \mathbf{x}_n^T \end{pmatrix} \in \mathbb{R}^{n\times(p+1)}$$

or equivalently $\mathbf{X} = (x_{i,j})_{\substack{1\leq i\leq n \\ 0\leq j\leq p}}$. The corresponding likelihood function is given as

$$\begin{aligned} L(\boldsymbol{\nu}) = \mathrm{P}(\mathbf{y}|\,\mathbf{X}, \boldsymbol{\nu}) &= \prod_{i=1}^{n} \mathrm{P}(\,y_i|\,\mathbf{x}_i, \boldsymbol{\nu}) \\ &= \prod_{i=1}^{n} (h_{\boldsymbol{\nu}}(\mathbf{x}_i))^{y_i} (1 - h_{\boldsymbol{\nu}}(\mathbf{x}_i))^{1-y_i}, \end{aligned}$$

and the log-likelihood as

$$\begin{aligned} \ell(\boldsymbol{\nu}) &= \log L(\boldsymbol{\nu}) \\ &= \sum_{i=1}^{n} y_i \log h_{\boldsymbol{\nu}}(\mathbf{x}_i) + (1 - y_i) \log(1 - h_{\boldsymbol{\nu}}(\mathbf{x}_i)). \end{aligned} \tag{7.4}$$

In order to estimate of the logistic regression parameters $\boldsymbol{\nu}$, the following objective function is considered

$$\max_{\boldsymbol{\nu}\in\mathbb{R}^{p+1}} \ell(\boldsymbol{\nu}).$$

For this matter, the gradient ascent optimization has been considered. The $(k+1)^{\text{th}}$ update of the logistic parameters, denoted by $\boldsymbol{\nu}^{(k+1)}$, are given by

$$\boldsymbol{\nu}^{(k+1)} = \boldsymbol{\nu}^{(k)} + \alpha \, \nabla_{\boldsymbol{\nu}} \, \ell(\boldsymbol{\nu}^{(k)}),$$

where α is the learning parameters. By setting $(\mathbf{x}_i, y_i) = (\mathbf{x}, y)$, with

$$\mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_p \end{pmatrix} = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_p \end{pmatrix},$$

and for each addent in (7.4), $\nabla_{\boldsymbol{\nu}}$ is calculated as

$$\begin{aligned} \frac{\partial}{\partial \nu_j} \left[y \log h_{\boldsymbol{\nu}}(\mathbf{x}) + (1 - y) \log(1 - h_{\boldsymbol{\nu}}(\mathbf{x})) \right] &= A(\boldsymbol{\nu}, \mathbf{x}) \frac{\partial}{\partial \nu_j} g(\boldsymbol{\nu}^T \mathbf{x}) \\ &= A(\boldsymbol{\nu}, \mathbf{x}) g(\boldsymbol{\nu}^T \mathbf{x})(1 - g(\boldsymbol{\nu}^T \mathbf{x})) \frac{\partial}{\partial \nu_j} \boldsymbol{\nu}^T \mathbf{x} \\ &= \big(y(1 - g(\boldsymbol{\nu}^T \mathbf{x})) - (1 - y) g(\boldsymbol{\nu}^T \mathbf{x}) \big) \, \mathbf{x}_j \\ &= (y - h_{\boldsymbol{\nu}}(\mathbf{x})) \mathbf{x}_j, \end{aligned}$$

where

$$A(\boldsymbol{\nu}, \mathbf{x}) = y\frac{1}{g(\boldsymbol{\nu}^T\mathbf{x})} - (1-y)\frac{1}{1-g(\boldsymbol{\nu}^T\mathbf{x})},$$

and

$$\mathbf{x}_j = \begin{pmatrix} 1 \\ x_{1,j} \\ \vdots \\ x_{p,j} \end{pmatrix}.$$

Hence,

$$\frac{\partial}{\partial \nu_j}\ell(\boldsymbol{\nu}) = \sum_{i=1}^{n}(y_i - h_{\boldsymbol{\nu}}(\mathbf{x}_i))x_{i,j},$$

with the update rule

$$\nu_j^{(k+1)} := \nu_j^{(k)} + \alpha\sum_{i=1}^{n}(y_i - h_{\boldsymbol{\nu}}(\mathbf{x}_i))x_{i,j}.$$

Alternative approach: using Newton's method. The parameter updates are hence given by

$$\boldsymbol{\nu}^{(k+1)} := \boldsymbol{\nu}^{(k)} - H^{-1}\,\nabla_{\boldsymbol{\nu}}\,\ell(\boldsymbol{\nu}),$$

where H is the Hessian matrix.
Furthermore, as (7.3) is utilized to represent the posterior probability, such that

$$\mathrm{P}(y|\,\mathbf{x}, \boldsymbol{\nu}) = (h_{\boldsymbol{\nu}}(\mathbf{x}))^y\,(1-h_{\boldsymbol{\nu}}(\mathbf{x}))^{1-y}, \qquad y \in \{0,1\},$$

hence it can be interpreted via the Bernoulli distribution, as

$$f(k) = p^k(1-p)^{1-k},$$

where $p \in (0,1), k \in \{0,1\}$, and $f(k)$ is given by

$$\begin{aligned} f(k) &= \mathbb{E}(k\log p + (1-k)\log(1-p)) \\ &= \mathbb{E}(k\log\frac{p}{1-p} + \log(1-p)) \\ &= \mathbb{E}(k\log\frac{p}{1-p})\mathbb{E}(1-p), \end{aligned}$$

with $\log\frac{p}{1-p} = q \Leftrightarrow p = \frac{1}{1+\mathbb{E}(-p)}$.

7.1.3 The Perceptron Learning Algorithms

Another approach for binary classification method is via the Perceptron learning algorithm. The function $h_{\boldsymbol{\nu}}(\mathbf{x}) = g(\boldsymbol{\nu}^T\mathbf{x})$ forces the output variable y to be either 0 or 1, i.e, $y = \{0, 1\}$, as

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0, \end{cases}$$

unlike the logistic regression, where $g(z) \in [0, 1]$. In analogy to the logistic regression, the update rule is given by

$$\nu_j^{(k+1)} := \nu_j^{(k)} + \sum_{i=1}^{n} \alpha(y_i - h_{\boldsymbol{\nu}}(\mathbf{x}_i))x_{i,j},$$

where $i = 1, \ldots, n$ and $j = 1, \ldots, p + 1$. The perceptron algorithm is considered to be a rough model of how neurons in the human brain works. Thus, it is one of the initial artificial neural networks with a probabilistic interpretation, without the need of MLE.

7.2 Reinforcement Learning

So far, the presented machine learning algorithms employ supervised learning methods, i.e., they rely on having a training set of a certain number of examples in order to estimate their parameters. This section presents the so-called *reinforcement learning* approach. Unlike the supervised learning, it does not need a training set to earn the knowledge of the system under consideration. This approach has to find a suitable action based on the current situation or environment to minimize a certain reward function [Bis06]. In other words, the learning is performed via a trial and error process, where its reliability and accuracy are indicated via the reward function. Applications on the reinforcement learning can be: robot leg coordination, autonomous flying, cell phone routing, factory control, etc,. As no training set is utilized, a reward function is instead considered to give an indication about the learning method itself: good or bad. For an analytical description, Markov decision process (MDP) is to be presented.

7.2.1 Markov Decision Process (MDP)

MDP can be represented via the tuple $(S, A, \{P_{sa}\}, \gamma, \mathbf{R})$, where

- S is a set of states.
- A is a set of actions.
- $P_{s,a}$ transition probabilities indicating the probability the system is presented in state $s \in S$ and taking the action $a \in A$. $P_{s,a}$ is a probability distribution over the state space.

- $\gamma \in [0,1]$ is a discount factor.
- R is a reward function, denoted by $R : S \times A \longrightarrow \mathbb{R}$, or simply by $R : S \longrightarrow \mathbb{R}$.

The dynamics of the process can be represented as

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \xrightarrow{a_3} \dots,$$

where $s_0, s_1, s_2, s_3, \cdots \in S$, and $a_0, a_1, a_2, a_3 \cdots \in A$. The total payoff is given by

$$R(s_0, a_0) + \gamma R(s_1, a_1) + \gamma^2 R(s_2, a_2) + \gamma^3 R(s_3, a_3) + \dots,$$

or simply by

$$R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \dots,$$

if the dependency is on the states only. The goal of the reinforcement learning can be formulated as

$$\max\ \mathbb{E}(\mathbf{R}(s_0) + \gamma \mathbf{R}(s_1) + \gamma^2 \mathbf{R}(s_2) + \gamma^3 R(s_3) + \dots),$$

over all actions in A. For this matter, a policy function π is defined as

$$\pi : S \longrightarrow A : s \longrightarrow \pi(s),$$

where the state $s \in S$ takes an action $a = \pi(s)$. Furthermore, the value function for the policy π is defined by

$$V^\pi(s) = \mathbb{E}(R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \cdots \mid s_0 = s, \pi),$$

which represent the expected total payoff upon starting in state s, and applying the policy π. Note that for any policy π, the Bellman equations hold, i.e.,

$$V^\pi(s) = R(s) + \gamma \sum_{s' \in S} P_{s,\pi(s)}(s') V^\pi(s'). \tag{7.5}$$

The right-hand side in (7.5) represents an immediate reward $R(s)$ plus the expected sum of future discounted rewards. As an exercise, prove that (7.5) holds.
Furthermore, solving for V^π for fixed policy π means to solve $\mid S \mid$ number of linear equations with $\mid S \mid$ variables if $\mid S \mid < \infty$ holds where $S = \{1, \dots, m\}, m \in \mathbb{N}$.
Let $\mathbf{R}, \mathbf{\Psi}^\pi$ and $\mathbf{V}^\pi$ be variables defined as

$$\mathbf{R} = (R(1), \dots, R(m))^T \in \mathbb{R}^m,$$
$$\mathbf{\Psi}^\pi = \begin{pmatrix} P_{1,\pi(1)} \\ \vdots \\ P_{m,\pi(m)} \end{pmatrix} \in \mathbb{R}^{m \times m},$$

and

$$\mathbf{V}^\pi = (V^\pi(1), \dots, V^\pi(m))^T \in \mathbb{R}^m,$$

thus (7.5) can be represented in matrix form as

$$\mathbf{V}^{\pi} = \mathbf{R} + \gamma \mathbf{\Psi}^{\pi} \mathbf{V}^{\pi} \Longleftrightarrow \mathbf{V}^{\pi} = (\mathbf{I}_m - \gamma \mathbf{\Psi}^{\pi})^{-1} \mathbf{R}, \tag{7.6}$$

provided the inverse exists. In (7.6), $\mathbf{I}_m$ represents an $m \times m$ identity matrix. Thus, the optimal value function is defined as

$$\begin{aligned} V^{\star}(s) &= \max_{\pi} V^{\pi}(s) \\ &= \max_{\pi} \mathbb{E}(R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots \mid s_0 = s, \pi), \end{aligned}$$

which indicates that the maximum expected payoff over all policies π, when starting at state $s \in S$. In analogous to (7.5), it holds that

$$V^{\star}(s) = R(s) + \max_{a \in A} \sum_{s' \in S} P_{s,a}(s') V^{\star}(s').$$

Futhermore, we also define

$$\pi^{\star}(s) = \arg\max_{a \in A} \sum_{s' \in S} P_{s,a}(s') V^{\star}(s'). \tag{7.7}$$

A partial ordering over policies is defined as

$$\pi \geq \pi' \text{ if } V^{\pi}(s) \geq V^{\pi'}(s), \quad \text{for all } s \in S.$$

Theorem 7.1. *a) There exists an optimal policy $\pi^{\star}$ with*

$$\pi^{\star} \geq \pi, \quad \textit{for all policies } \pi, \textit{such that } \pi^{\star} \textit{ is given by (7.7)}$$

b) All optimal policies achieve the same value,

$$V^{\pi *}(s) = V^{*}(s)$$

In other words, both *a)* and *b)* state that an optimal policy π^{*} is independent of the initial state $s \in S$.
At this point, it is still needed to compute the optimal policy $\pi^{\star}$. For this matter, there are mainly two algorithms: the value iteration, and the policy iteration algorithms.

7.2.2 Computing the optimal policy

The policy iteration algorithm is utilized to find the optimal $\pi^{\star}$. It is indicated in Algorithm 5.
Based on Algorithm 5, it can be note that,
a) for fixed policy π, $\mathbf{V}^{\pi}$ can be computed from (7.5),
b) $V(s')$ is obtained with respect to the actual policy π. Update of π is greedy with respect to $\mathbf{V}$.
After a finite number of iterations, $\mathbf{V}$ converges to $\mathbf{V}^{\star}$, and π to $\pi^{\star}$.

Algorithm 5 Policy iteration algorithm

1: **procedure** INITIALIZE π RANDOMLY
2: **repeat**
3: a) $\mathbf{V} := \mathbf{V}^{\pi}$
4: b) For each $s \in S$, let $\pi(s) := \arg\max_{a \in A} \sum_{s' \in S} P_{s,a}(s')V(s')$
5: **until** Convergence

7.2.3 Model learning for MDPs

In practice, the transition probability $P_{s,a}$ and sometimes the reward function R are unknown. Therefore, it is required to estimate $P_{s,a}(s'), s' \in S$, and $R(s')$ from a given (observed) data, which are obtained by carrying out trials or experiments in the form of

$$s_0^{(\ell)} \xrightarrow{a_0^{(\ell)}} s_1^{(\ell)} \xrightarrow{a_1^{(\ell)}} s_2^{(\ell)} \xrightarrow{a_2^{(\ell)}} s_3^{(\ell)} \dots,$$

for $\ell = 1, 2, \dots, \in \mathbb{N}$ number of trails. Thus, the estimate of $P_{s,a}$, denoted by $P^{\star}s, a$ is calculated as

$$P^*_{s,a} = \frac{\text{times took action } a \text{ in state } s \text{ and got to } s'}{\text{times tool action } a \text{ in state } s}.$$

In case of $\frac{0}{0}$, $P^{\star}s, a(s')$ is chosen to be uniformly distributed, such that $P^{\star}_{s,a}(s') = \frac{1}{|S|}$. Similarity, $R(s)$ can be estimated from observed data. Possible algorithm for learning a MDP with unknown $P_{s,a}$ is shown in Algorithm 6.

Algorithm 6 Possible algorithm for learning a MDP with unknown $P_{s,a}$

1: **procedure** INITIALIZE π RANDOMLY
2: **repeat**
3: a) Execute π in the MDP for some number of trials ℓ
4: b) Update the estimates $P^*_{s,a}$(and potentially R)
5: c) Update π using Algorithm 5
6: **until** Convergence

Bibliography

[Agg15] Charu C. Aggarwal. *Data Mining*. Springer International Publishing, 2015.

[Bai99] Z. D. Bai. Methodologies in spectral analysis of large dimensional random matrices, A review. *Statistica Sinica*, 9(3):611–662, 1999.

[Ban08] Afonso S. Bandeira. Ten Lectures and Forty-Two Open Problems in the Mathematics of Data Science. `http://www.cims.nyu.edu/~bandeira/TenLecturesFortyTwoProblems.pdf`, 2008. [Online].

[BAP05] Jinho Baik, Gérard Ben Arous, and Sandrine Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5):1643–1697, September 2005.

[Bis06] C. M. Bishop. *Pattern recognition and machine learning*. Springer, New York, 2006.

[BS02] Mukund Balasubramanian and Eric L. Schwartz. The Isomap Algorithm and Topological Stability. *Science*, 295(5552):7–7, January 2002.

[Cat66] Raymond B. Cattell. The Scree Test For The Number Of Factors. *Multivariate Behavioral Research*, 1(2):245–276, April 1966.

[CL06] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, July 2006.

[Fan50] Ky Fan. On a Theorem of Weyl Concerning Eigenvalues of Linear Transformations. *Proceedings of the National Academy of Sciences of the United States of America*, 36(1):31–35, January 1950.

[HTF09] Trevor Hastie, Robert Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer series in statistics. Springer, New York, NY, 2nd edition, 2009.

[LRU14] Jurij Leskovec, Anand Rajaraman, and Jeffrey D. Ullman. *Mining of massive datasets*. Cambridge University Press, Cambridge, 2nd edition, 2014.

[Mat97] Rudolf Mathar. *Multidimensionale Skalierung: mathematische Grundlagen und algorithmische Aspekte*. Teubner-Skripten zur mathematischen Stochastik. Teubner, Stuttgart, 1997. OCLC: 605887892.

[Mey00] C. D. Meyer. *Matrix analysis and applied linear algebra*. SIAM, Philadelphia, 2000.

[MKB79] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate analysis*. Probability and mathematical statistics. Academic Press, London; New York, 1979.

[Mur12] Kevin P. Murphy. *Machine learning: a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, Cambridge, MA, 2012.

[OFG97] Edgar Osuna, Robert Freund, and Federico Girosit. Training support vector machines: an application to face detection. In *Computer vision and pattern recognition, 1997. Proceedings., 1997 IEEE computer society conference on*, pages 130–136. IEEE, 1997.

[TCGC13] R. Talmon, I. Cohen, S. Gannot, and R. R. Coifman. Diffusion Maps for Signal Processing: A Deeper Look at Manifold-Learning Techniques Based on Kernels and Graphs. *IEEE Signal Processing Magazine*, 30(4):75–86, July 2013.

[TDSL00] Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.