Statistical and Mathematical Methods for Artificial Intelligence

Last update: 18 October 2023

Academic Year 2023 – 2024

Alma Mater Studiorum · University of Bologna

Contents

1 Finite numbers

1.1 Sources of error

Measure error Precision of the measurement instrument. Measure error

Arithmetic error Propagation of rounding errors in each step of an algorithm. Arithmetic error

Truncation error Approximating an infinite procedure to a finite number of iterations. Truncation error

Inherent error Caused by the finite representation of the data (floating-point). Inherent error

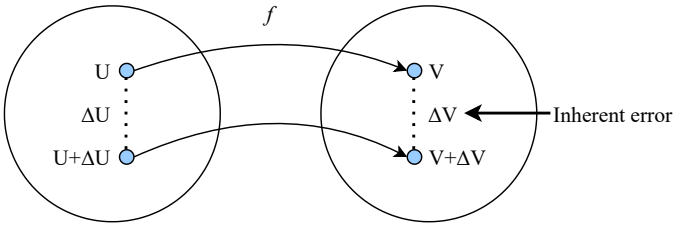


Figure 1.1: Inherent error visualization

1.2 Error measurement

Let x be a value and $\hat{x}$ its approximation. Then:

Absolute error

$$E_a = \hat{x} - x$$ Absolute error

Note that, out of context, the absolute error is meaningless.

Relative error

$$E_a = \frac{\hat{x} - x}{x}$$ Relative error

1.3 Representation in base β

Let $\beta \in \mathbb{N}_{>1}$ be the base. Each $x \in \mathbb{R} \setminus \{0\}$ can be uniquely represented as:

$$x = \texttt{sign}(x) \cdot (d_1\beta^{-1} + d_2\beta^{-2} + \cdots + d_n\beta^{-n})\beta^p \tag{1.1}$$

where:

- $0 \leq d_i \leq \beta - 1$
- $d_1 \neq 0$
- starting from an index i, not all d_j $(j \geq i)$ are equal to $\beta - 1$

Equation (1.1) can be represented using the normalized scientific notation as:

Normalized scientific notation

$$x = \pm(0.d_1 d_2 \dots)\beta^p$$

where $0.d_1 d_2 \dots$ is the **mantissa** and β^p the **exponent**.

Mantissa
Exponent

1.4 Floating-point

A floating-point system $\mathcal{F}(\beta, t, L, U)$ is defined by the parameters:

Floating-point

- β: base
- t: precision (number of digits in the mantissa)
- $[L, U]$: range of the exponent

Each $x \in \mathcal{F}(\beta, t, L, U)$ can be represented in its normalized form:

$$x = \pm(0.d_1 d_2 \dots d_t)\beta^p \quad L \leq p \leq U \tag{1.2}$$

We denote with $\texttt{fl}(x)$ the representation of $x \in \mathbb{R}$ in a given floating-point system.

Example. In $\mathcal{F}(10, 5, -3, 3)$, $x = 12.\bar{3}$ is represented as:

$$\texttt{fl}(x) = +0.12333 \cdot 10^2$$

1.4.1 Numbers distribution

Given a floating-point system $\mathcal{F}(\beta, t, L, U)$, the total amount of representable numbers is:

$$2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$$

Representable numbers are more sparse towards the exponent upper bound and more dense towards the lower bound. It must be noted that there is an underflow area around 0.

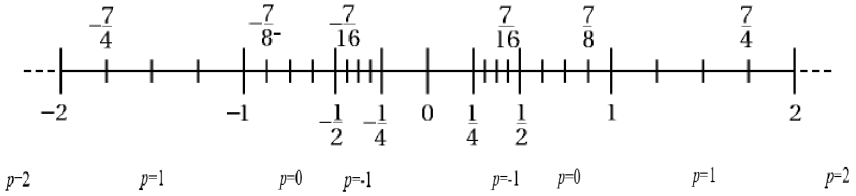


Figure 1.2: Floating-point numbers in $\mathcal{F}(2, 3, -1, 2)$

1.4.2 Number representation

Given a floating-point system $\mathcal{F}(\beta, t, L, U)$, the representation of $x \in \mathbb{R}$ can result in:

Exact representation if $p \in [L, U]$ and $d_i = 0$ for $i > t$.

Approximation if $p \in [L, U]$ but d_i may not be 0 for $i > t$. In this case, the representation is obtained by truncating or rounding the value.

Truncation
Rounding

Underflow if $p < L$. In this case, the value is approximated to 0.

Underflow

Overflow if $p > U$. In this case, an exception is usually raised.

Overflow

1.4.3 Machine precision

Machine precision $\varepsilon_{\text{mach}}$ determines the accuracy of a floating-point system. Depending on the approximation approach, machine precision can be computes as: Machine precision

Truncation $\varepsilon_{\text{mach}} = \beta^{1-t}$

Rounding $\varepsilon_{\text{mach}} = \frac{1}{2}\beta^{1-t}$

Therefore, rounding results in more accurate representations.
$\varepsilon_{\text{mach}}$ is the smallest distance among the representable numbers (Figure 1.3).

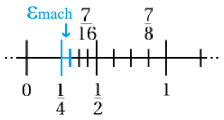


Figure 1.3: Visualization of $\varepsilon_{\text{mach}}$ in $\mathcal{F}(2, 3, -1, 2)$

In alternative, $\varepsilon_{\text{mach}}$ can be defined as the smallest representable number such that:

$$\texttt{fl}(1 + \varepsilon_{\text{mach}}) > 1.$$

1.4.4 IEEE standard

IEEE 754 defines two floating-point formats:

Single precision Stored in 32 bits. Represents the system $\mathcal{F}(2, 24, -128, 127)$. float32

1 (sign)	8 (exponent)	23 (mantissa)

Double precision Stored in 64 bits. Represents the system $\mathcal{F}(2, 53, -1024, 1023)$. float64

1 (sign)	11 (exponent)	52 (mantissa)

As the first digit of the mantissa is always 1, it does not need to be stored. Moreover, special configurations are reserved to represent `Inf` and `NaN`.

1.4.5 Floating-point arithmetic

Let:

- $+ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be a real numbers operation.
- $\oplus : \mathcal{F} \times \mathcal{F} \to \mathcal{F}$ be the corresponding operation in a floating-point system.

To compute $x \oplus y$, a machine:

1. Calculates $x + y$ in a high precision register (still approximated, but more precise than the floating-point system used to store the result)
2. Stores the result as $\texttt{fl}(x + y)$

A floating-point operation causes a small rounding error:

$$\left| \frac{(x \oplus y) - (x + y)}{x + y} \right| < \varepsilon_{\text{mach}}$$

However, some operations may be subject to the **cancellation** problem which causes information loss. Cancellation

Example. Given $x = 1$ and $y = 1 \cdot 10^{-16}$, we want to compute $x + y$ in $\mathcal{F}(10, 16, U, L)$.

$$\begin{aligned} z &= \texttt{fl}(x) + \texttt{fl}(y) \\ &= 0.1 \cdot 10^1 + 0.1 \cdot 10^{-15} \\ &= (0.1 + 0.\overbrace{0 \ldots 0}^{\text{16 zeros}}1) \cdot 10^1 \\ &= 0.1\overbrace{0 \ldots 0}^{\text{15 zeros}}1 \cdot 10^1 \end{aligned}$$

Then, we have that $\texttt{fl}(z) = 0.1\overbrace{0 \ldots 0}^{\text{15 zeros}} \cdot 10^1 = 1 = x$.

2 Linear algebra

2.1 Vector space

A **vector space** over $\mathbb{R}$ is a nonempty set V, whose elements are called vectors, with two operations: Vector space

$$\begin{array}{ll} \text{Addition} & +: V \times V \to V \\ \text{Scalar multiplication} & \cdot : \mathbb{R} \times V \to V \end{array}$$

A vector space has the following properties:

1. Addition is commutative and associative
2. A null vector exists: $\exists \bar{\mathbf{0}} \in V$ s.t. $\forall \mathbf{u} \in V : \bar{\mathbf{0}} + \mathbf{u} = \mathbf{u} + \bar{\mathbf{0}} = \mathbf{u}$
3. An identity element for scalar multiplication exists: $\forall \mathbf{u} \in V : 1\mathbf{u} = \mathbf{u}$
4. Each vector has its opposite: $\forall \mathbf{u} \in V, \exists \mathbf{a} \in V : \mathbf{a} + \mathbf{u} = \mathbf{u} + \mathbf{a} = \bar{\mathbf{0}}$.
   $\mathbf{a}$ is denoted as $-\mathbf{u}$.
5. Distributive properties:

$$\forall \alpha \in \mathbb{R}, \forall \mathbf{u}, \mathbf{w} \in V : \alpha(\mathbf{u} + \mathbf{w}) = \alpha\mathbf{u} + \alpha\mathbf{w}$$

$$\forall \alpha, \beta \in \mathbb{R}, \forall \mathbf{u} \in V : (\alpha + \beta)\mathbf{u} = \alpha\mathbf{u} + \beta\mathbf{u}$$

6. Associative property:

$$\forall \alpha, \beta \in \mathbb{R}, \forall \mathbf{u} \in V : (\alpha\beta)\mathbf{u} = \alpha(\beta\mathbf{u})$$

A subset $U \subseteq V$ of a vector space V is a **subspace** iff U is a vector space. Subspace

2.1.1 Basis

Let V be a vector space of dimension n. A basis $\beta = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ of V is a set of n linearly independent vectors of V. Basis
Each element of V can be represented as a linear combination of the vectors in the basis β:

$$\forall \mathbf{w} \in V : \mathbf{w} = \lambda_1\mathbf{v}_1 + \cdots + \lambda_n\mathbf{v}_n \text{ where } \lambda_i \in \mathbb{R}$$

The canonical basis of a vector space is a basis where each vector represents a dimension i (i.e. 1 in position i and 0 in all other positions). Canonical basis

Example. The canonical basis β of $\mathbb{R}^3$ is $\beta = \{(1,0,0), (0,1,0), (0,0,1)\}$

2.1.2 Dot product

The dot product of two vectors in $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is defined as: Dot product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T\mathbf{y} = \sum_{i=1}^{n} x_i \cdot y_i$$

2.2 Matrix

This is a (very formal definition of) matrix: Matrix

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

2.2.1 Invertible matrix

A matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is invertible (non-singular) if: Non-singular matrix

$$\exists \boldsymbol{B} \in \mathbb{R}^{n \times n} : \boldsymbol{AB} = \boldsymbol{BA} = \boldsymbol{I}$$

where $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{B}$ is denoted as $\boldsymbol{A}^{-1}$.

2.2.2 Kernel

The null space (kernel) of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a subspace such that: Kernel

$$\text{Ker}(\boldsymbol{A}) = \{\mathbf{x} \in \mathbb{R}^n : \boldsymbol{A}\mathbf{x} = \bar{\mathbf{0}}\}$$

Theorem 2.2.1. A square matrix $\boldsymbol{A}$ with $\text{Ker}(\boldsymbol{A}) = \{\bar{\mathbf{0}}\}$ is non singular.

2.2.3 Similar matrices

Two matrices $\boldsymbol{A}$ and $\boldsymbol{D}$ are **similar** if there exists an invertible matrix $\boldsymbol{P}$ such that: Similar matrices

$$\boldsymbol{D} = \boldsymbol{P}^{-1}\boldsymbol{AP}$$

2.3 Norms

2.3.1 Vector norms

The norm of a vector is a function: Vector norm

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

such that for each $\lambda \in \mathbb{R}$ and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$:

- $\|\mathbf{x}\| \geq 0$
- $\|\mathbf{x}\| = 0 \iff \mathbf{x} = \bar{\mathbf{0}}$
- $\|\lambda\mathbf{x}\| = |\lambda| \cdot \|\mathbf{x}\|$
- $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$

Common norms are:

2-norm $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$

1-norm $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$

∞-norm $\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|$

In general, different norms tend to maintain the same proportion. In some cases, unbalanced results may be obtained when comparing different norms.

Example. Let $\mathbf{x} = (1, 1000)$ and $\mathbf{y} = (999, 1000)$. Their norms are:

$$\begin{array}{ll} \|\mathbf{x}\|_2 = \sqrt{1000001} & \|\mathbf{y}\|_2 = \sqrt{1998001} \\ \|\mathbf{x}\|_\infty = 1000 & \|\mathbf{y}\|_\infty = 1000 \end{array}$$

2.3.2 Matrix norms

The norm of a matrix is a function: Matrix norm

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

such that for each $\lambda \in \mathbb{R}$ and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$:

- $\|\boldsymbol{A}\| \geq 0$
- $\|\boldsymbol{A}\| = 0 \iff \boldsymbol{A} = \boldsymbol{0}$
- $\|\lambda \boldsymbol{A}\| = |\lambda| \cdot \|\boldsymbol{A}\|$
- $\|\boldsymbol{A} + \boldsymbol{B}\| \leq \|\boldsymbol{A}\| + \|\boldsymbol{B}\|$

Common norms are:

2-norm $\|\boldsymbol{A}\|_2 = \sqrt{\rho(\boldsymbol{A}^T \boldsymbol{A})}$,
where $\rho(\boldsymbol{X})$ is the largest absolute value of the eigenvalues of $\boldsymbol{X}$ (spectral radius).

1-norm $\|\boldsymbol{A}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |a_{i,j}|$ (i.e. max sum of the columns in absolute value)

Frobenius norm $\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} a_{i,j}^2}$

2.4 Symmetric, positive definite matrices

Symmetric matrix A square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is symmetric $\iff \boldsymbol{A} = \boldsymbol{A}^T$ Symmetric matrix

Positive semidefinite matrix A symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is positive semidefinite iff Positive semidefinite matrix

$$\forall \mathbf{x} \in \mathbb{R}^n \setminus \{0\} : \mathbf{x}^T \boldsymbol{A} \mathbf{x} \geq 0$$

Positive definite matrix A symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is positive definite iff Positive definite matrix

$$\forall \mathbf{x} \in \mathbb{R}^n \setminus \{0\} : \mathbf{x}^T \boldsymbol{A} \mathbf{x} > 0$$

It has the following properties:

1. The null space of $\boldsymbol{A}$ has the null vector only: $\text{Ker}(\boldsymbol{A}) = \{\bar{\mathbf{0}}\}$.
   Which implies that $\boldsymbol{A}$ is non-singular (Theorem 2.2.1).
2. The diagonal elements of $\boldsymbol{A}$ are all positive.

2.5 Orthogonality

Angle between vectors The angle ω between two vectors $\mathbf{x}$ and $\mathbf{y}$ can be obtained from: Angle between vectors

$$\cos\omega = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\|_2 \cdot \|\mathbf{y}\|_2}$$

Orthogonal vectors Two vectors $\mathbf{x}$ and $\mathbf{y}$ are orthogonal ($\mathbf{x} \perp \mathbf{y}$) when: Orthogonal vectors

$$\langle \mathbf{x}, \mathbf{y} \rangle = 0$$

Orthonormal vectors Two vectors $\mathbf{x}$ and $\mathbf{y}$ are orthonormal when: Orthonormal vectors

$$\mathbf{x} \perp \mathbf{y} \text{ and } \|\mathbf{x}\| = \|\mathbf{y}\| = 1$$

Theorem 2.5.1. The canonical basis of a vector space is orthonormal.

Orthogonal matrix A matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is orthogonal if its columns are orthonormal vectors. It has the following properties: Orthogonal matrix

1. $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{I} = \boldsymbol{A}^T\boldsymbol{A}$, which implies $\boldsymbol{A}^{-1} = \boldsymbol{A}^T$.
2. The length of a vector is unchanged when mapped through an orthogonal matrix:
$$\|\boldsymbol{A}\mathbf{x}\|^2 = \|\mathbf{x}\|^2$$
3. The angle between two vectors is unchanged when both are mapped through an orthogonal matrix:
$$\cos\omega = \frac{(\boldsymbol{A}\mathbf{x})^T(\boldsymbol{A}\mathbf{y})}{\|\boldsymbol{A}\mathbf{x}\| \cdot \|\boldsymbol{A}\mathbf{y}\|} = \frac{\mathbf{x}^T\mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|}$$

Note: an orthogonal matrix represents a rotation.

Orthogonal basis Given a n-dimensional vector space V and a basis $\beta = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of V. β is an orthogonal basis if: Orthogonal basis

$$\mathbf{b}_i \perp \mathbf{b}_j \text{ for } i \neq j \text{ (i.e. } \langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0)$$

Orthonormal basis Given a n-dimensional vector space V and an orthogonal basis $\beta = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of V. β is an orthonormal basis if: Orthonormal basis

$$\|\mathbf{b}_i\|_2 = 1 \text{ (or } \langle \mathbf{b}_i, \mathbf{b}_i \rangle = 1)$$

Orthogonal complement Given a n-dimensional vector space V and a m-dimensional subspace $U \subseteq V$. The orthogonal complement $U^\perp$ of U is a $(n-m)$-dimensional subspace of V such that it contains all the vectors orthogonal to every vector in U: Orthogonal complement

$$\forall \mathbf{w} \in V : \mathbf{w} \in U^\perp \iff (\forall \mathbf{u} \in U : \mathbf{w} \perp \mathbf{u})$$

Note that $U \cap U^\perp = \{\bar{\mathbf{0}}\}$ and it is possible to represent all vectors in V as a linear combination of both the basis of U and $U^\perp$.

The vector $\mathbf{w} \in U^\perp$ s.t. $\|\mathbf{w}\| = 1$ is the **normal vector** of U. Normal vector

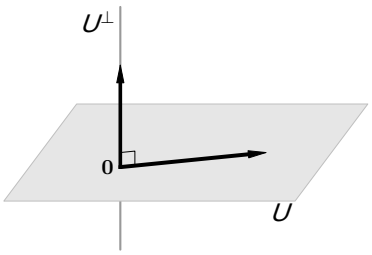


Figure 2.1: Orthogonal complement of a subspace $U \subseteq \mathbb{R}^3$

2.6 Projections

Projections are methods to map high-dimensional data into a lower-dimensional space while minimizing the compression loss.
Let V be a vector space and $U \subseteq V$ a subspace of V. A linear mapping $\pi : V \to U$ is a (orthogonal) projection if:

Orthogonal projection

$$\pi^2 = \pi \circ \pi = \pi$$

In other words, applying π multiple times gives the same result (i.e. idempotency).
π can be expressed as a transformation matrix $\boldsymbol{P}_\pi$ such that:

$$\boldsymbol{P}_\pi^2 = \boldsymbol{P}_\pi$$

2.6.1 Projection onto general subspaces

To project a vector $\mathbf{x} \in \mathbb{R}^n$ into a lower-dimensional subspace $U \subseteq \mathbb{R}^n$, it is possible to use the basis of U.
Let $m = \dim(U)$ be the dimension of U and $\boldsymbol{B} = (\mathbf{b}_1, \dots, \mathbf{b}_m) \in \mathbb{R}^{n \times m}$ an ordered basis of U. A projection $\pi_U(\mathbf{x})$ represents $\mathbf{x}$ as a linear combination of the basis:

Projection onto subspace basis

$$\pi_U(\mathbf{x}) = \sum_{i=1}^{m} \lambda_i \mathbf{b}_i = \boldsymbol{B}\boldsymbol{\lambda}$$

where $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_m)^T \in \mathbb{R}^m$ are the new coordinates of $\mathbf{x}$ and is found by minimizing the distance between $\pi_U(\mathbf{x})$ and $\mathbf{x}$.

2.7 Eigenvectors and eigenvalues

Given a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $\lambda \in \mathbb{C}$ is an eigenvalue of $\boldsymbol{A}$ with corresponding eigenvector $\mathbf{x} \in \mathbb{R}^n \setminus \{\bar{\mathbf{0}}\}$ if:

Eigenvalue
Eigenvector

$$\boldsymbol{A}\mathbf{x} = \lambda\mathbf{x}$$

It is equivalent to say that:

- λ is an eigenvalue of $\boldsymbol{A} \in \mathbb{R}^{n \times n}$
- $\exists \mathbf{x} \in \mathbb{R}^n \setminus \{\bar{\mathbf{0}}\}$ s.t. $\boldsymbol{A}\mathbf{x} = \lambda\mathbf{x}$
  Equivalently the system $(\boldsymbol{A} - \lambda\boldsymbol{I}_n)\mathbf{x} = \bar{\mathbf{0}}$ is non-trivial ($\mathbf{x} \neq \bar{\mathbf{0}}$).
- $\text{rank}(\boldsymbol{A} - \lambda\boldsymbol{I}_n) < n$

- $\det(\boldsymbol{A} - \lambda \boldsymbol{I}_n) = 0$ (i.e. $(\boldsymbol{A} - \lambda \boldsymbol{I}_n)$ is singular (i.e. not invertible))

Note that eigenvectors are not unique. Given an eigenvector $\mathbf{x}$ of $\boldsymbol{A}$ with eigenvalue λ, we can prove that $\forall c \in \mathbb{R} \setminus \{0\} : c\mathbf{x}$ is an eigenvector of $\boldsymbol{A}$:

$$\boldsymbol{A}(c\mathbf{x}) = c(\boldsymbol{A}\mathbf{x}) = c\lambda\mathbf{x} = \lambda(c\mathbf{x})$$

Theorem 2.7.1. $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is symmetric positive definite $\iff$ its eigenvalues are all positive.

Eigenvalues and positive definiteness

Eigenspace Set of all the eigenvectors of $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ associated to an eigenvalue λ. This set is a subspace of $\mathbb{R}^n$.

Eigenspace

Eigenspectrum Set of all eigenvalues of $\boldsymbol{A} \in \mathbb{R}^{n\times n}$.

Eigenspectrum

Geometric multiplicity Given an eigenvalue λ of a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. The geometric multiplicity of λ is the number of linearly independent eigenvectors associated to λ.

Geometric multiplicity

Theorem 2.7.2. Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. If its n eigenvectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ are associated to distinct eigenvalues, then $\mathbf{x}_1, \dots, \mathbf{x}_n$ are linearly independent (i.e. they form a basis of $\mathbb{R}^n$).

Linearly independent eigenvectors

Defective matrix A matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is defective if it has less than n linearly independent eigenvectors.

Defective matrix

Theorem 2.7.3 (Spectral theorem)**.** Given a symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. Its eigenvectors form a orthonormal basis and its eigenvalues are all in $\mathbb{R}$.

Spectral theorem

2.7.1 Diagonalizability

A matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is diagonalizable if it is similar to a diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$:

Diagonalizable matrix

$$\exists \boldsymbol{P} \in \mathbb{R}^{n\times n} \text{ s.t. } \boldsymbol{P} \text{ invertible and } \boldsymbol{D} = \boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P}$$

Theorem 2.7.4. Similar matrices have the same eigenvalues.

Theorem 2.7.5. A symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is always diagonalizable.

Symmetric matrix diagonalizability

3 Linear systems

A linear system:

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n = b_2 \\ \vdots \\ a_{m,1}x_1 + a_{m,2}x_2 + \cdots + a_{m,n}x_n = b_m \end{cases}$$

can be represented as:

$$\boldsymbol{A}\mathbf{x} = \mathbf{b}$$

where:

$$\boldsymbol{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{pmatrix} \in \mathbb{R}^{m \times n} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} \in \mathbb{R}^m$$

3.1 Square linear systems

Square linear system

A square linear system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$ with $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ has an unique solution iff one of the following conditions is satisfied:

1. $\boldsymbol{A}$ is non-singular (invertible)
2. $\text{rank}(\boldsymbol{A}) = n$ (full rank)
3. $\boldsymbol{A}\mathbf{x}$ admits only the solution $\mathbf{x} = \bar{\mathbf{0}}$

Algebraic solution to linear systems

The solution can be algebraically determined as

$$\boldsymbol{A}\mathbf{x} = \mathbf{b} \iff \mathbf{x} = \boldsymbol{A}^{-1}\mathbf{b}$$

However, this approach requires to compute the inverse of a matrix, which has a time complexity of $O(n^3)$.

3.2 Direct methods

Direct methods

Direct methods compute the solution of a linear system in a finite number of steps. Compared to iterative methods, they are more precise but more expensive.
The most common approach consists in factorizing the matrix $\boldsymbol{A}$.

3.2.1 Gaussian factorization

Gaussian factorization (LU decomposition)

Given a square linear system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$, the matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is factorized into $\boldsymbol{A} = \boldsymbol{LU}$ such that:

- $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ is a lower triangular matrix
- $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ is an upper triangular matrix

The system can be decomposed to:

$$\begin{aligned} \boldsymbol{A}\mathbf{x} = \mathbf{b} &\iff \boldsymbol{LU}\mathbf{x} = \mathbf{b} \\ &\iff \mathbf{y} = \boldsymbol{U}\mathbf{x} \;\&\; \boldsymbol{L}\mathbf{y} = \mathbf{b} \end{aligned}$$

To find the solution, it is sufficient to solve in order:

1. $\boldsymbol{L}\mathbf{y} = \mathbf{b}$ (solved w.r.t. $\mathbf{y}$)
2. $\mathbf{y} = \boldsymbol{U}\mathbf{x}$ (solved w.r.t. $\mathbf{x}$)

The overall complexity is $O(\frac{n^3}{3}) + 2 \cdot O(n^2) = O(\frac{n^3}{3})$.
$O(\frac{n^3}{3})$ is the time complexity of the LU factorization. $O(n^2)$ is the complexity to directly solving a system with a triangular matrix (forward or backward substitutions).

3.2.2 Gaussian factorization with pivoting

Gaussian factorization with pivoting

During the computation of $\boldsymbol{A} = \boldsymbol{LU}$ (using Gaussian elimination[1]), a division by 0 may occur. A method to prevent this problem (and to lower the algorithmic error) is to change the order of the rows of $\boldsymbol{A}$ before decomposing it. This is achieved by using a permutation matrix $\boldsymbol{P}$, which is obtained as a permutation of the identity matrix.
The permuted system becomes $\boldsymbol{PA}\mathbf{x} = \boldsymbol{P}\mathbf{b}$ and the factorization is obtained as $\boldsymbol{PA} = \boldsymbol{LU}$. The system can be decomposed to:

$$\begin{aligned} \boldsymbol{PA}\mathbf{x} = \boldsymbol{P}\mathbf{b} &\iff \boldsymbol{LU}\mathbf{x} = \boldsymbol{P}\mathbf{b} \\ &\iff \mathbf{y} = \boldsymbol{U}\mathbf{x} \;\&\; \boldsymbol{L}\mathbf{y} = \boldsymbol{P}\mathbf{b} \end{aligned}$$

An alternative formulation (which is what `SciPy` uses) is defined as:

$$\boldsymbol{A} = \boldsymbol{PLU} \iff \boldsymbol{P}^T\boldsymbol{A} = \boldsymbol{LU}$$

It must be noted that $\boldsymbol{P}$ is orthogonal, so $\boldsymbol{P}^T = \boldsymbol{P}^{-1}$. The solution to the system ($\boldsymbol{P}^T\boldsymbol{A}\mathbf{x} = \boldsymbol{P}^T\mathbf{b}$) can be found as above.

3.2.3 Cholesky factorization

Given a symmetric definite positive matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. It is possible to decompose $\boldsymbol{A}$ as:

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$$

where $\boldsymbol{L}$ is lower triangular.
A square system where $\boldsymbol{A}$ is symmetric definite positive can be solved as above using the Cholesky factorization. This method has time complexity $O(\frac{n^3}{6})$.

[1] `https://en.wikipedia.org/wiki/LU_decomposition#Using_Gaussian_elimination`

3.3 Iterative methods

Iterative methods

Iterative methods solve a linear system by computing a sequence that converges to the exact solution. Compared to direct methods, they are less precise but computationally faster and more adapt for large systems.
The overall idea is to build a sequence of vectors $\mathbf{x}_k$ that converges to the exact solution $\mathbf{x}^*$:

$$\lim_{k\to\infty} \mathbf{x}_k = \mathbf{x}^*$$

Generally, the first vector $\mathbf{x}_0$ is given (or guessed). Subsequent vectors are computed w.r.t. the previous iteration as $\mathbf{x}_k = g(\mathbf{x}_{k-1})$.
The two most common families of iterative methods are:

Stationary methods

Stationary methods compute the sequence as:

$$\mathbf{x}_k = \boldsymbol{B}\mathbf{x}_{k-1} + \mathbf{d}$$

where $\boldsymbol{B}$ is called iteration matrix and $\mathbf{d}$ is computed from the $\mathbf{b}$ vector of the system. The time complexity per iteration is $O(n^2)$.

Gradient-like methods

Gradient-like methods have the form:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_{k-1}\mathbf{p}_{k-1}$$

where $\alpha_{k-1} \in \mathbb{R}$ and the vector $\mathbf{p}_{k-1}$ is called direction.

3.3.1 Stopping criteria

Stopping criteria

One or more stopping criteria are needed to determine when to truncate the sequence (as it is theoretically infinite). The most common approaches are:

Residual based The algorithm is terminated when the current solution is close enough to the exact solution. The residual at iteration k is computed as $\mathbf{r}_k = \mathbf{b} - \boldsymbol{A}\mathbf{x}_k$. Given a tolerance ε, the algorithm may stop when:

- $\|\mathbf{r}_k\| \leq \varepsilon$ (absolute)
- $\frac{\|\mathbf{r}_k\|}{\|\mathbf{b}\|} \leq \varepsilon$ (relative)

Update based The algorithm is terminated when the difference between iterations is very small. Given a tolerance τ, the algorithm stops when:

$$\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leq \tau$$

Obviously, as the sequence is truncated, a truncation error is introduced when using iterative methods.

3.4 Condition number

Inherent error causes inaccuracies during the resolution of a system. This problem is independent from the algorithm and is estimated using exact arithmetic.
Given a system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$, we perturbate $\boldsymbol{A}$ and/or $\mathbf{b}$ and study the inherited error. For instance, if we perturbate $\mathbf{b}$, we obtain the following system:

$$\boldsymbol{A}\tilde{\mathbf{x}} = (\mathbf{b} + \Delta\mathbf{b})$$

After finding $\tilde{\mathbf{x}}$, we can compute the inherited error as $\Delta\mathbf{x} = \tilde{\mathbf{x}} - \mathbf{x}$.
By comparing $\left\|\frac{\Delta\mathbf{x}}{\mathbf{x}}\right\|$ and $\left\|\frac{\Delta\mathbf{b}}{\mathbf{b}}\right\|$, we can compute the error introduced by the perturbation.
It can be shown that the distance is:

$$\left\|\frac{\Delta\mathbf{x}}{\mathbf{x}}\right\| \leq \|\boldsymbol{A}\| \cdot \|\boldsymbol{A}^{-1}\| \cdot \left\|\frac{\Delta\mathbf{b}}{\mathbf{b}}\right\|$$

Finally, we can define the **condition number** of a matrix $\boldsymbol{A}$ as: Condition number

$$K(\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \|\boldsymbol{A}^{-1}\|$$

A system is **ill-conditioned** if $K(\boldsymbol{A})$ is large (i.e. a small perturbation of the input causes a large change of the output). Otherwise it is **well-conditioned**. Ill-conditioned Well-conditioned

3.5 Linear least squares problem

See Section 4.2.4.

4 Matrix decomposition

4.1 Eigendecomposition

Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. If the eigenvectors of $\boldsymbol{A}$ form a basis of $\mathbb{R}^n$, then $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ can be decomposed into:

Eigendecomposition

$$\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1}$$

where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ contains the eigenvectors of $\boldsymbol{A}$ as its columns and $\boldsymbol{D}$ is a diagonal matrix whose diagonal contains the eigenvalues of $\boldsymbol{A}$.

4.2 Singular value decomposition

Given a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ of rank $r \in [0, \min\{m, n\}]$. The singular value decomposition (SVD) of $\boldsymbol{A}$ is always possible and has form:

Singular value decomposition

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$$

$$= \begin{pmatrix} \begin{pmatrix} \\ \mathbf{u}_1 \\ \\ \end{pmatrix} & \dots & \begin{pmatrix} \\ \mathbf{u}_m \\ \\ \end{pmatrix} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_{\min\{m,n\}} \end{pmatrix} \begin{pmatrix} (& \mathbf{v}_1 &) \\ & \vdots & \\ (& \mathbf{v}_n &) \end{pmatrix}$$

where:

- $\boldsymbol{U} \in \mathbb{R}^{m \times m}$ is an orthogonal matrix with columns $\mathbf{u}_i$ called left-singular vectors.
- $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ is an orthogonal matrix with columns $\mathbf{v}_i$ called right-singular vectors.
- $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ is a matrix with $\boldsymbol{\Sigma}_{i,j} = 0$ (i.e. diagonal if it was a square matrix) and the singular values $\sigma_i, i = 1 \dots \min\{m, n\}$ on the diagonal. By convention $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r \geq 0$. Note that singular values $\sigma_j = 0$ for $(r{+}1) \leq j \leq \min\{m, n\}$ (i.e. singular values at indexes after $\text{rank}(\boldsymbol{A})$ are always 0).

We can also represent SVD as a **singular value equation**, which resembles the eigenvalue equation:

Singular value equation

$$\boldsymbol{A}\mathbf{v}_i = \sigma_i \mathbf{u}_i \text{ for } i = 1, \dots, r$$

This is derived from:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \iff \boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{V} \iff \boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{\Sigma}$$

4.2.1 Singular values and eigenvalues

Given $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, we can obtain the eigenvalues and eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^T$ through SVD.

Eigendecomposition of $\boldsymbol{A}^T\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^T$

For $\boldsymbol{A}^T\boldsymbol{A}$, we can compute:

$$\begin{aligned}\boldsymbol{A}^T\boldsymbol{A} &= (\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T)^T(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T) \text{ using } (\boldsymbol{A}\boldsymbol{B})^T = \boldsymbol{B}^T\boldsymbol{A}^T \\ &= (\boldsymbol{V}\boldsymbol{\Sigma}^T\boldsymbol{U}^T)(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T) \\ &= \boldsymbol{V}\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}\boldsymbol{V}^T \\ &= \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^T\end{aligned}$$

As $\boldsymbol{V}$ is orthogonal ($\boldsymbol{V}^T = \boldsymbol{V}^{-1}$), we can apply the eigendecomposition theorem:

- The diagonal of $\boldsymbol{\Sigma}^2$ (i.e. the square of the singular values of A) are the eigenvalues of $\boldsymbol{A}^T\boldsymbol{A}$
- The columns of $\boldsymbol{V}$ (right-singular vectors) are the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$

The same process holds for $\boldsymbol{A}\boldsymbol{A}^T$. In this case, the columns of $\boldsymbol{U}$ (left-singular vectors) are the eigenvectors.

4.2.2 Singular values and 2-norm

Given a symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, we have that $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{A}^2 = \boldsymbol{A}\boldsymbol{A}^T$ (as $\boldsymbol{A}^T = \boldsymbol{A}$). The eigenvalues of $\boldsymbol{A}^2$ are $\lambda_1^2, \dots, \lambda_n^2$, where λ_i are eigenvalues of $\boldsymbol{A}$. Alternatively, the eigenvalues of $\boldsymbol{A}^2$ are the squared singular values of $\boldsymbol{A}$: $\lambda_i^2 = \sigma_i^2$. Moreover, the eigenvalues of $\boldsymbol{A}^{-1}$ are $\frac{1}{\lambda_1}, \dots, \frac{1}{\lambda_n}$.
We can compute the 2-norm as: 2-norm using SVD

$$\|\boldsymbol{A}\|_2 = \sqrt{\rho(\boldsymbol{A}^T\boldsymbol{A})} = \sqrt{\rho(\boldsymbol{A}^2)} = \sqrt{\max\{\sigma_1^2, \dots, \sigma_r^2\}} = \sigma_1$$

$$\|\boldsymbol{A}^{-1}\|_2 = \sqrt{\rho((\boldsymbol{A}^{-1})^T(\boldsymbol{A}^{-1}))} = \sqrt{\rho((\boldsymbol{A}\boldsymbol{A}^T)^{-1})} = \sqrt{\rho((\boldsymbol{A}^2)^{-1})} = \sqrt{\max\{\frac{1}{\sigma_1^2}, \dots, \frac{1}{\sigma_r^2}\}} = \frac{1}{\sigma_r}$$

Furthermore, we can compute the condition number of $\boldsymbol{A}$ as:

$$K(\boldsymbol{A}) = \|\boldsymbol{A}\|_2 \cdot \|\boldsymbol{A}^{-1}\|_2 = \sigma_1 \cdot \frac{1}{\sigma_r}$$

4.2.3 Application: Matrix approximation

Given a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and its SVD decomposition $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, we can construct a rank-1 matrix (dyad) $\boldsymbol{A}_i \in \mathbb{R}^{m\times n}$ as: Dyad

$$\boldsymbol{A}_i = \mathbf{u}_i\mathbf{v}_i^T$$

where $\mathbf{u}_i \in \mathbb{R}^m$ is the i-th column of $\boldsymbol{U}$ and $\mathbf{v}_i \in \mathbb{R}^n$ is the i-th column of $\boldsymbol{V}$. Then, we can compose $\boldsymbol{A}$ as a sum of dyads:

$$\boldsymbol{A}_i = \sum_{i=1}^{r} \sigma_i\mathbf{u}_i\mathbf{v}_i^T = \sum_{i=1}^{r} \sigma_i\boldsymbol{A}_i$$

By considering only the first $k < r$ singular values, we can obtain a rank-k approximation of $\boldsymbol{A}$: Rank-k approximation

$$\hat{\boldsymbol{A}}(k) = \sum_{i=1}^{k} \sigma_i\mathbf{u}_i\mathbf{v}_i^T = \sum_{i=1}^{k} \sigma_i\boldsymbol{A}_i$$

Theorem 4.2.1 (Eckart-Young)**.** Given $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ of rank r. For any $k \leq r$ (this theorem is interesting for $k < r$), the rank-k approximation is:

$$\hat{\boldsymbol{A}}(k) = \arg \min_{\boldsymbol{B} \in \mathbb{R}^{m \times n}, \text{rank}(\boldsymbol{B}) = k} \|\boldsymbol{A} - \boldsymbol{B}\|_2$$

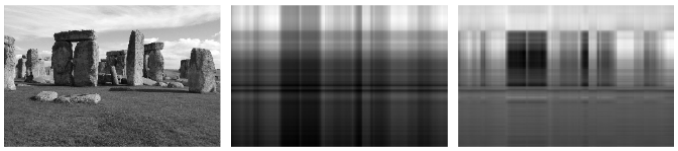
In other words, among all the possible projections, $\hat{\boldsymbol{A}}(k)$ is the closer one to $\boldsymbol{A}$. Moreover, the error of the rank-k approximation is:

$$\|\boldsymbol{A} - \hat{\boldsymbol{A}}(k)\|_2 = \left\| \sum_{i=1}^{r} \sigma_i \boldsymbol{A}_i - \sum_{j=1}^{k} \sigma_j \boldsymbol{A}_j \right\|_2 = \left\| \sum_{i=k+1}^{r} \sigma_i \boldsymbol{A}_i \right\|_2 = \sigma_{k+1}$$

Image compression

Each dyad requires $1 + m + n$ (respectively for σ_i, $\mathbf{u}_i$ and $\mathbf{v}_i$) numbers to be stored. A rank-k approximation requires to store $k(1+m+n)$ numbers. Therefore, the compression factor is given by: Compression factor

$$c_k = 1 - \frac{k(1+m+n)}{mn}$$



(a) Original image $\boldsymbol{A}$. (b) Rank-1 approximation $\widehat{\boldsymbol{A}}(1)$.(c) Rank-2 approximation $\widehat{\boldsymbol{A}}(2)$.

(d) Rank-3 approximation $\widehat{\boldsymbol{A}}(3)$.(e) Rank-4 approximation $\widehat{\boldsymbol{A}}(4)$.(f) Rank-5 approximation $\widehat{\boldsymbol{A}}(5)$.

Figure 4.1: Approximation of an image

4.2.4 Application: Linear least squares problem

A system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$ with $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $m > n$ does not generally have a solution. Therefore, instead of finding the exact solution, it is possible to search for a $\tilde{\mathbf{x}}$ such that: Linear least squares

$$\boldsymbol{A}\tilde{\mathbf{x}} - \mathbf{b} \approx \bar{\mathbf{0}}$$

In other words, we aim to find a $\tilde{\mathbf{x}}$ that is close enough to solve the system. This problem is usually formulated as:

$$\tilde{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathbb{R}^n} \|\boldsymbol{A}\mathbf{x} - \mathbf{b}\|_2^2$$

It always admits a solution and, depending on $\text{rank}(\boldsymbol{A})$, there two possible cases:

rank$(\boldsymbol{A}) = n$ The solution is unique for each $b \in \mathbb{R}^m$. It is found by solving the normal equation: Normal equation

$$\boldsymbol{A}^T\boldsymbol{A}\mathbf{x} = \boldsymbol{A}^T\mathbf{b}$$

$\boldsymbol{A}^T\boldsymbol{A}$ is symmetric definite positive and the system can be solved using the Cholesky factorization.

rank$(\boldsymbol{A}) < n$ The system admits infinite solutions. Of all the solutions S, we are interested in the one with minimum norm:

Least squares using SVD

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in S} \|\mathbf{x}\|_2$$

This problem can be solved using SVD:

$$\mathbf{x}^* = \sum_{i=1}^{\text{rank}(\boldsymbol{A})} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i$$

4.2.5 Application: Polynomial interpolation

Given a set of m data $(x_i, y_i), i = 1, \ldots, m$, we want to find a polynomial of degree n $(m > n)$ that approximates it. In other words, we want to find a function:

Polynomial interpolation

$$f(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_n x^n$$

that minimizes the residual vector $\mathbf{r} = (r_1, \ldots, r_m)$, where $r_i = |y_i - f(x_i)|$. We can formulate this as a linear system:

$$\mathbf{r} = \mathbf{y} - \boldsymbol{A}\mathbf{c} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} - \begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^n \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix}$$

that can be solved as a linear least squares problem:

$$\min_{\mathbf{c} \in \mathbb{R}^n} \|\mathbf{y} - \boldsymbol{A}\mathbf{c}\|_2^2$$

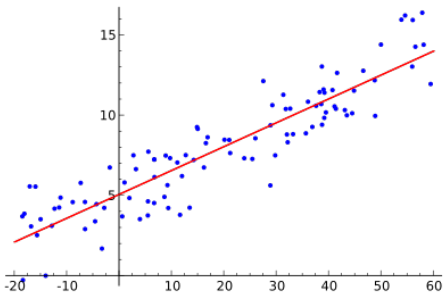


Figure 4.2: Interpolation using a polynomial of degree 1

4.3 Eigendecomposition vs SVD

Eigendecomposition $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1}$	**SVD** $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$
Only defined for square matrices $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ with eigenvectors that form a basis of $\mathbb{R}^n$	Always exists
$\boldsymbol{P}$ is not necessarily orthogonal	$\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal
The elements on the diagonal of $\boldsymbol{D}$ may be in $\mathbb{C}$	The elements on the diagonal of $\boldsymbol{\Sigma}$ are all non-negative reals
For symmetric matrices, eigendecomposition and SVD are the same	

5 Vector calculus

5.1 Gradient of real-valued multivariate functions

Gradient Given a function $f : \mathbb{R}^n \to \mathbb{R}$, the gradient is a row vector containing the partial derivatives of f: Gradient

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} & \frac{\partial f(\mathbf{x})}{\partial x_2} & \dots & \frac{\partial f(\mathbf{x})}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{1 \times n}$$

Hessian Given a function $f : \mathbb{R}^n \to \mathbb{R}$, the Hessian matrix $\boldsymbol{H} \in \mathbb{R}^{n \times n}$ contains the second derivatives of f: Hessian matrix

$$\boldsymbol{H} = \begin{pmatrix} \frac{\partial f}{\partial x_1^2} & \frac{\partial f}{\partial x_1 \partial x_2} & \dots & \frac{\partial f}{\partial x_1 \partial x_n} \\ \frac{\partial f}{\partial x_2 \partial x_1} & \frac{\partial f}{\partial x_2^2} & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial x_n \partial x_1} & \dots & \dots & \frac{\partial f}{\partial x_n^2} \end{pmatrix}$$

In other words, $H_{i,j} = \frac{\partial f}{\partial x_i \partial x_j}$. Moreover, $\boldsymbol{H}$ is symmetric.

5.1.1 Partial differentiation rules

Product rule Let $f, g : \mathbb{R}^n \to \mathbb{R}$: Product rule

$$\frac{\partial}{\partial \mathbf{x}}(f(\mathbf{x})g(\mathbf{x})) = \frac{\partial f}{\partial \mathbf{x}} g(\mathbf{x}) + f(\mathbf{x}) \frac{\partial g}{\partial \mathbf{x}}$$

Sum rule Let $f, g : \mathbb{R}^n \to \mathbb{R}$: Sum rule

$$\frac{\partial}{\partial \mathbf{x}}(f(\mathbf{x}) + g(\mathbf{x})) = \frac{\partial f}{\partial \mathbf{x}} + \frac{\partial g}{\partial \mathbf{x}}$$

Chain rule Let $f : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{g}$ a vector of n functions $g_i : \mathbb{R}^m \to \mathbb{R}$: Chain rule

$$\frac{\partial}{\partial \mathbf{x}}(f \circ \mathbf{g})(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}}(f(\mathbf{g}(\mathbf{x}))) = \frac{\partial f}{\partial \mathbf{g}} \frac{\partial \mathbf{g}}{\partial \mathbf{x}}$$

More precisely, considering a $f : \mathbb{R}^2 \to \mathbb{R}$ of two variables $g_1(t), g_2(t) : \mathbb{R} \to \mathbb{R}$ that are functions of t. The gradient of f with respect to t is:

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \begin{pmatrix} \frac{\partial f}{\partial g_1} & \frac{\partial f}{\partial g_2} \end{pmatrix} \begin{pmatrix} \frac{\partial g_1}{\partial t} \\ \frac{\partial g_2}{\partial t} \end{pmatrix} = \frac{\partial f}{\partial g_1} \frac{\partial g_1}{\partial t} + \frac{\partial f}{\partial g_2} \frac{\partial g_2}{\partial t}$$

In other words, the first matrix represents the gradient of f w.r.t. its variables and the second matrix contains in the i-th row the gradient of g_i.

Therefore, if g_i are in turn multivariate functions $g_1(s,t), g_2(s,t) : \mathbb{R}^2 \to \mathbb{R}$, the chain rule can be applies as:

$$\frac{\mathrm{d}f}{\mathrm{d}(s,t)} = \begin{pmatrix} \frac{\partial f}{\partial g_1} & \frac{\partial f}{\partial g_2} \end{pmatrix} \begin{pmatrix} \frac{\partial g_1}{\partial s} & \frac{\partial g_1}{\partial t} \\ \frac{\partial g_2}{\partial s} & \frac{\partial g_2}{\partial t} \end{pmatrix}$$

Example. Let $f(x_1, x_2) = x_1^2 + 2x_2$, where $x_1 = \sin(t)$ and $x_2 = \cos(t)$.

$$\begin{aligned}\frac{\mathrm{d}f}{\mathrm{d}t} &= \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial t} \\ &= (2x_1)(\cos(t)) + (2)(-\sin(t)) \\ &= 2\sin(t)\cos(t) - 2\sin(t)\end{aligned}$$

Example. Let $h : \mathbb{R} \to \mathbb{R}$ be defined as $h(t) = (f \circ g)(t)$ where:

$$f : \mathbb{R}^2 \to \mathbb{R} \text{ is defined as } f(\mathbf{x}) = \exp(x_1 x_2^2)$$

$$g : \mathbb{R} \to \mathbb{R}^2 \text{ is defined as } \mathbf{g}(t) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} t\cos(t) \\ t\sin(t) \end{pmatrix}$$

The gradient of h with respect to t can be computed as:

$$\frac{\mathrm{d}h}{\mathrm{d}t} = \frac{\partial f}{\partial \mathbf{g}}\frac{\partial \mathbf{g}}{\partial t} = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{pmatrix} \begin{pmatrix} \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial t} \end{pmatrix}$$

$$= \begin{pmatrix} \exp(x_1 x_2^2)x_2^2 & 2\exp(x_1 x_2^2)x_1 x_2 \end{pmatrix} \begin{pmatrix} \cos(t) + (-t\sin(t)) \\ \sin(t) + t\cos(t) \end{pmatrix}$$

Least squares loss gradient

Example (Gradient of a least squares loss)**.** Given a linear model defined on $\boldsymbol{\theta}$:

$$\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\theta}$$

with $\boldsymbol{\theta} \in \mathbb{R}^D$, $\boldsymbol{\Phi} \in \mathbb{R}^{N \times D}$ and $\mathbf{y} \in \mathbb{R}^N$. We can define the least squares loss function as:

$$L(\mathbf{e}) = \|\mathbf{e}\|_2^2$$
$$\mathbf{e}(\boldsymbol{\theta}) = \mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}$$

It must be noted that:

$$L(\mathbf{e}) = \|\mathbf{e}\|_2^2 = \mathbf{e}^T\mathbf{e} = \sum_{i=1}^{N} \mathbf{e}_i^2$$

To compute the gradient of L with respect to $\boldsymbol{\theta}$, we can use the chain rule:

$$\begin{aligned}\nabla L(\boldsymbol{\theta}) = \frac{\partial L}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \boldsymbol{\theta}} &= (2\mathbf{e}^T)(-\boldsymbol{\Phi}) \\ &= -2(\mathbf{y}^T - \boldsymbol{\theta}^T\boldsymbol{\Phi}^T)\boldsymbol{\Phi} \\ &= -2(\mathbf{y}^T\boldsymbol{\Phi} - \boldsymbol{\theta}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi})\end{aligned}$$

Note that if we enforce $\nabla L(\boldsymbol{\theta}) = \bar{\mathbf{0}}$, we obtain the normal equation of Section 4.2.4:

$$\begin{aligned}\nabla L = 0 &\iff -2(\mathbf{y}^T\boldsymbol{\Phi} - \boldsymbol{\theta}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}) = \bar{\mathbf{0}} \\ &\iff \mathbf{y}^T\boldsymbol{\Phi} - \boldsymbol{\theta}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \bar{\mathbf{0}} \\ &\iff \boldsymbol{\Phi}^T\mathbf{y} - \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\theta} = \bar{\mathbf{0}}\end{aligned}$$

5.2 Gradient of vector-valued multivariate functions

Vector-valued function Function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ with $n \geq 1$ and $m > 1$. Given $\mathbf{x} \in \mathbb{R}^n$, the output can be represented as:

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{pmatrix} \in \mathbb{R}^m$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$.

Jacobian Given $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, the Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ contains the first-order derivatives of $\mathbf{f}$: Jacobian matrix

$$\boldsymbol{J} = \nabla \mathbf{f}(\mathbf{x}) = \begin{pmatrix} \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_n} \end{pmatrix} = \begin{pmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_m(\mathbf{x})}{\partial x_n} \end{pmatrix}$$

In other words, $J_{i,j} = \frac{\partial f_i}{\partial x_j}$. Note that the Jacobian matrix is a generalization of the gradient in the real-valued case.

5.3 Backpropagation

Backpropagation is used to tune the parameters of a neural network. A neural network can be seen as a composition of many functions: Backpropagation

$$\mathbf{y} = (\mathbf{f}_K \circ \mathbf{f}_{K-1} \circ \cdots \circ \mathbf{f}_1)(\mathbf{x}) = \mathbf{f}_K(\mathbf{f}_{K-1}(\cdots \mathbf{f}_1(\mathbf{x}) \cdots))$$

Each $\mathbf{f}_i$ takes as input the output of the previous layer $\mathbf{x}_{i-1}$ and has the form:

$$\mathbf{f}_i(\mathbf{x}_{i-1}) = \sigma_i(\boldsymbol{A}_{i-1}\mathbf{x}_{i-1} + \mathbf{b}_{i-1})$$

where σ_i is an activation function[1] (a function to add nonlinearity), while $\boldsymbol{A}_{i-1}$ (linear mapping) and $\mathbf{b}_{i-1}$ (biases) are the parameters of $\mathbf{f}_i$.

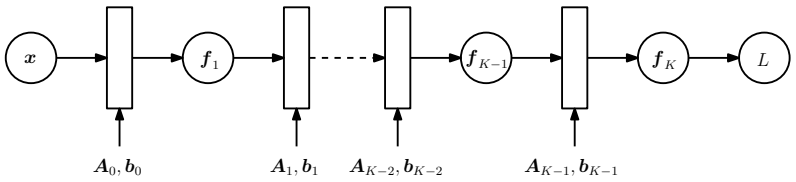


Figure 5.1: Forward pass

We can more compactly denote a neural network with input $\mathbf{x}$ and K layers as:

$$\begin{aligned} \mathbf{f}_0 &= \mathbf{x} \\ \mathbf{f}_i &= \sigma_i(\boldsymbol{A}_{i-1}\mathbf{f}_{i-1} + \mathbf{b}_{i-1}) \; i = 1, \ldots, K \end{aligned}$$

Given the ground truth $\mathbf{y}$, we want to find the parameters $\boldsymbol{A}_j$ and $\mathbf{b}_j$ that minimizes the squared loss:

$$L(\boldsymbol{\theta}) = \|\mathbf{y} - \mathbf{f}_K(\boldsymbol{\theta}, \mathbf{x})\|^2$$

[1] `https://en.wikipedia.org/wiki/Activation_function`

where $\boldsymbol{\theta} = \{\boldsymbol{A}_0, \mathbf{b}_0, \ldots, \boldsymbol{A}_{K-1}, \mathbf{b}_{K-1}\}$ are the parameters of each layer. This can be done by using the chain rule to compute the partial derivatives of L with respect to the parameters $\boldsymbol{\theta}_j = \{\boldsymbol{A}_j, \mathbf{b}_j\}$:

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-1}} = \overbrace{\frac{\partial L}{\partial \mathbf{f}_K}\frac{\partial \mathbf{f}_K}{\partial \boldsymbol{\theta}_{K-1}}}^{\text{New}}$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-2}} = \overbrace{\frac{\partial L}{\partial \mathbf{f}_K}}^{\text{Known}}\overbrace{\frac{\partial \mathbf{f}_K}{\partial \mathbf{f}_{K-1}}\frac{\partial \mathbf{f}_{K-1}}{\partial \boldsymbol{\theta}_{K-2}}}^{\text{New}}$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-3}} = \overbrace{\frac{\partial L}{\partial \mathbf{f}_K}\frac{\partial \mathbf{f}_K}{\partial \mathbf{f}_{K-1}}}^{\text{Known}}\overbrace{\frac{\partial \mathbf{f}_{K-1}}{\partial \mathbf{f}_{K-2}}\frac{\partial \mathbf{f}_{K-2}}{\partial \boldsymbol{\theta}_{K-3}}}^{\text{New}}$$

$$\vdots$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_i} = \overbrace{\frac{\partial L}{\partial \mathbf{f}_K}\frac{\partial \mathbf{f}_K}{\partial \mathbf{f}_{K-1}}\cdots}^{\text{Known}}\overbrace{\frac{\partial \mathbf{f}_{i+2}}{\partial \mathbf{f}_{i+1}}\frac{\partial \mathbf{f}_{i+1}}{\partial \boldsymbol{\theta}_i}}^{\text{New}}$$

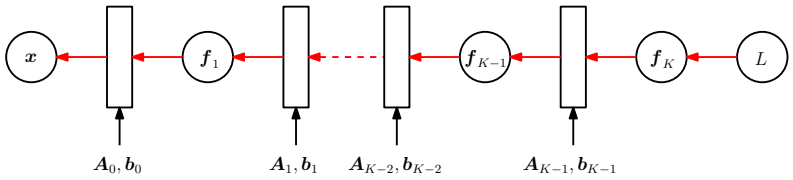


Figure 5.2: Backward pass

5.4 Automatic differentiation

Starting from the example below first is recommended.

Automatic differentiation

Automatic differentiation allows to numerically compute the gradient of complex functions using elementary functions, intermediate variables and the chain rule through a computation graph. When the gradient has many components, it also allows to compute it more efficiently.
Let f be a function, $x_1, \ldots, x_d$ the input variables of f, $x_{d+1}, \ldots, x_{D-1}$ the intermediate variables and x_D the output variable. The computation graph can be expressed as:

$$\forall i \in \{d+1, \ldots, D\} : x_i = g_i(x_{\mathrm{Pa}(x_i)})$$

where g_i are elementary functions and $x_{\mathrm{Pa}(x_i)}$ are the parent nodes of x_i in the graph. In other words, each intermediate variable is expressed as an elementary function of its preceding nodes. The derivatives of f can then be computed step-by-step going backwards as:

$$\frac{\partial f}{\partial x_D} = 1, \text{ as by definition } f = x_D$$

$$\frac{\partial f}{\partial x_i} = \sum_{\forall x_j : x_i \in \mathrm{Pa}(x_j)} \frac{\partial f}{\partial x_j}\frac{\partial x_j}{\partial x_i} = \sum_{\forall x_j : x_i \in \mathrm{Pa}(x_j)} \frac{\partial f}{\partial x_j}\frac{\partial g_j}{\partial x_i}$$

where $\mathrm{Pa}(x_j)$ is the set of parent nodes of x_j in the graph. In other words, to compute the partial derivative of f w.r.t. x_i, we apply the chain rule by first computing the partial derivative of f w.r.t. the variables following x_i in the graph (as the computation goes backwards).
Automatic differentiation is applicable to all functions that can be expressed as a computational graph and when the elementary functions are differentiable. Note that backpropagation is a special case of automatic differentiation.

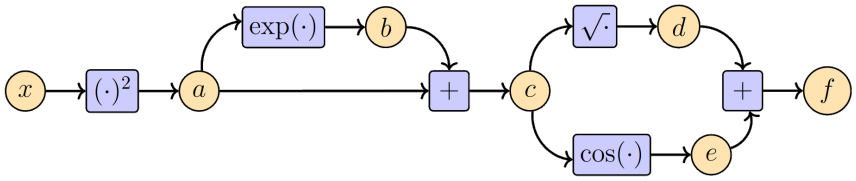
Example. Given the function:

$$f(x) = \sqrt{x^2 + \exp(x^2)} + \cos(x^2 + \exp(x^2))$$

and the elementary functions $\{(\cdot)^2, \exp(\cdot), +, \sqrt{\cdot}, \cos(\cdot)\}$, f can be decomposed in the following intermediate variables:

$$\begin{aligned} a &= x^2 \\ b &= \exp(a) \\ c &= a + b \\ d &= \sqrt{c} \end{aligned} \qquad \begin{aligned} e &= \cos(c) \\ f &= d + e \end{aligned}$$

Which corresponds to the following computation graph:



We can then compute the derivatives of the intermediate variables w.r.t. their inputs (i.e. inbound edges):

$$\begin{aligned} \frac{\partial a}{\partial x} &= 2x \\ \frac{\partial b}{\partial a} &= \exp(a) \\ \frac{\partial c}{\partial a} &= 1 \\ \frac{\partial c}{\partial b} &= 1 \end{aligned} \qquad \begin{aligned} \frac{\partial d}{\partial c} &= \frac{1}{2\sqrt{c}} \\ \frac{\partial e}{\partial c} &= -\sin(c) \\ \frac{\partial f}{\partial d} &= 1 \\ \frac{\partial f}{\partial e} &= 1 \end{aligned}$$

Finally, we can compute $\frac{\partial f}{\partial x}$ by going backward from the output (f) to the input (x):

$$\begin{aligned} \frac{\partial f}{\partial d} &= \text{ already known (previous step)} \\ \frac{\partial f}{\partial e} &= \text{ already known (previous step)} \\ \frac{\partial f}{\partial c} &= \frac{\partial f}{\partial d}\frac{\partial d}{\partial c} + \frac{\partial f}{\partial e}\frac{\partial e}{\partial c} \end{aligned} \qquad \begin{aligned} \frac{\partial f}{\partial b} &= \frac{\partial f}{\partial c}\frac{\partial c}{\partial b} \\ \frac{\partial f}{\partial a} &= \frac{\partial f}{\partial b}\frac{\partial b}{\partial a} + \frac{\partial f}{\partial c}\frac{\partial c}{\partial a} \\ \frac{\partial f}{\partial x} &= \frac{\partial f}{\partial a}\frac{\partial a}{\partial x} \end{aligned}$$

In other words, to compute the partial derivative of f w.r.t. a variable x_i, all variables w_j that follows x_i in the graph are considered.

Now, by substituting we obtain:

$$\frac{\partial f}{\partial c} = 1 \cdot \frac{1}{2\sqrt{c}} + 1 \cdot (-\sin(c))$$
$$\frac{\partial f}{\partial b} = \frac{\partial f}{\partial c} \cdot 1$$
$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial b} \cdot \exp(a) + \frac{\partial f}{\partial c} \cdot 1$$
$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial a} \cdot 2x$$

6 Gradient methods

6.1 Minimum of a function

Let $f : \mathbb{R}^N \to \mathbb{R}$ be continuous and differentiable in $\mathbb{R}^N$.

Stationary point $\mathbf{x}^*$ is a stationary point of f iff: Stationary point

$$\nabla f(\mathbf{x}^*) = \bar{\mathbf{0}}$$

Local minimum $\mathbf{x}^* \in \mathbb{R}^N$ is a local minimum of f iff: Local minimum

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \; \forall \mathbf{x} \in \mathbb{R}^N : \|\mathbf{x} - \mathbf{x}^*\| < \varepsilon$$

Strict local minimum $\mathbf{x}^* \in \mathbb{R}^N$ is a strict local minimum of f iff: Strict local minimum

$$f(\mathbf{x}^*) < f(\mathbf{x}) \; \forall \mathbf{x} \in \mathbb{R}^N : \|\mathbf{x} - \mathbf{x}^*\| < \varepsilon$$

Global minimum $\mathbf{x}^* \in \mathbb{R}^N$ is a global minimum of f iff: Global minimum

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \; \forall \mathbf{x} \in \mathbb{R}^N$$

Strict global minimum $\mathbf{x}^* \in \mathbb{R}^N$ is a strict global minimum of f iff: Strict global minimum

$$f(\mathbf{x}^*) < f(\mathbf{x}) \; \forall \mathbf{x} \in \mathbb{R}^N$$

Note that $\max f(x) = \min -f(x)$.

6.1.1 Optimality conditions

First order condition Let $f : \mathbb{R}^N \to \mathbb{R}$ be continuous and differentiable in $\mathbb{R}^N$. First order condition

$$\text{If } \mathbf{x}^* \text{ local minimum of } f \Rightarrow \nabla f(\mathbf{x}^*) = \bar{\mathbf{0}}$$

Second order condition Let $f : \mathbb{R}^N \to \mathbb{R}$ be continuous and twice differentiable. Second order condition

$$\text{If } \nabla f(\mathbf{x}^*) = \bar{\mathbf{0}} \text{ and } \nabla^2 f(\mathbf{x}^*) \text{ positive definite} \Rightarrow \mathbf{x}^* \text{ strict local minimum of } f$$

As the second order condition requires to compute the Hessian matrix, which is expensive, in practice only the first order condition is checked.

6.2 Descent methods

Descent methods are iterative methods that have the property: Descent methods

$$f(\mathbf{x}_k) < f(\mathbf{x}_{k+1})$$

The iteration is defined as:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_{k-1}\mathbf{p}_{k-1}$$

where $\mathbf{p}_{k-1} \in \mathbb{R}^N$ is the search direction and $\alpha_{k-1} \in \mathbb{R}$ is the step length. Search direction Step length
Note: descent methods usually converge to a local minimum.

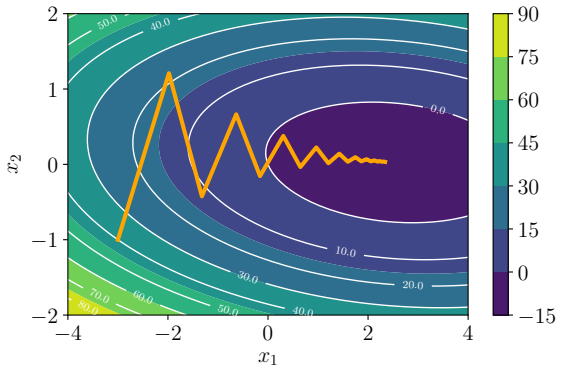


Figure 6.1: Descent method steps in $\mathbb{R}^2$ (i.e. moving across contour lines)

6.2.1 Choice of the search direction

Descent direction $\mathbf{p} \in \mathbb{R}^N$ is a descent direction of f in $\mathbf{x}$ if: Descent direction

$$\exists \bar{\alpha} > 0, \forall \alpha \in [0, \bar{\alpha}] : f(\mathbf{x} + \alpha \mathbf{p}) < f(\mathbf{x})$$

Theorem 6.2.1. Let $\mathbf{p} \in \mathbb{R}^N$, $\mathbf{p} \neq \bar{\mathbf{0}}$.

$$\text{If } \mathbf{p}^T \nabla f(\mathbf{x}) < 0 \Rightarrow \mathbf{p} \text{ descent direction of } f \text{ in } x$$

Theorem 6.2.2. For all $\mathbf{x}$, $\mathbf{p} = -\nabla f(\mathbf{x})$ is a descent direction of f in x.

Proof.

$$\begin{aligned} \mathbf{p}^T \nabla f(\mathbf{x}) < 0 &\iff -(\nabla f(\mathbf{x}))^T \nabla f(\mathbf{x}) < 0 \\ &\iff -\|\nabla f(\mathbf{x})\|_2^2 < 0 \end{aligned}$$

This holds as the norm is always positive. □

Gradient-like methods Gradient-like methods are descent methods that use $-\nabla f$ as step. Gradient-like methods

6.2.2 Choice of the step length

Constant In machine learning, it is common to set a constant value for the step (learning rate), but it can be proved that this does not guarantee convergence.

Backtracking procedure α_k is chose such that it respects the Wolfe condition[1]: Backtracking procedure

```
def backtracking(τ, c1):
    αk = 1 # Initial guess
    while f(xk + αk∇f(xk)) > f(xk) + c1αk∇f(xk)^T∇f(xk):
        αk = αk / τ
    return αk
```

It can be proved that, by using the backtracking procedure, gradient methods converge to a local minimum.

[1] https://en.wikipedia.org/wiki/Wolfe_conditions

6.2.3 Stopping condition

Stopping condition

We can stop iterating when $\mathbf{x}_k \approx \mathbf{x}^*$, that is, $\nabla f(\mathbf{x}_k) \approx \bar{\mathbf{0}}$. We can verify this by checking the norm of the gradient against a tolerance τ:

Absolute condition $\|\nabla f(x_k)\|_2 < \tau$

Relative condition $\frac{\|\nabla f(x_k)\|_2}{\|\nabla f(x_0)\|_2} < \tau$

A generic gradient-like method can then be defined as:

```
def gradientMethod(f, x_0):
    k = 0
    while stoppingCondition(f, x_k, x_0):
        p_k = −∇f(x_k)
        α_k = backtracking(...)
        x_{k+1} = x_k + α_k p_k
        k = k + 1
    return x_k
```

6.2.4 Problems

Initialization point

Choice of the initialization point The starting point of an iterative method is a user defined parameter. For simple problems, it is usually chosen randomly in $[-1, +1]$.

For complex problems, the choice of the initialization point is critical as it may cause numerical instabilities or bad results. Heuristics can be used to select an adequate starting point.

Flag regions and local optima

Flag regions and local optima Flat regions slow down the learning speed, while a local optima causes the method to converge at a poor solution.

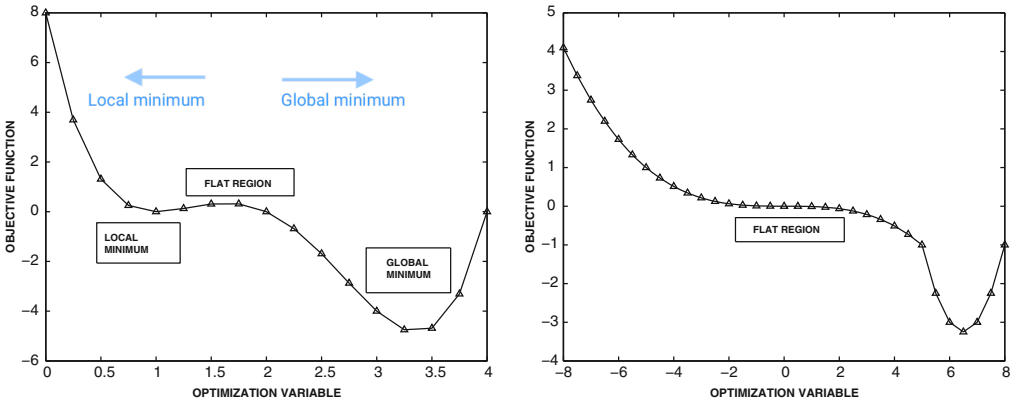


Figure 6.2: Flat regions and local minima

Vanishing gradient
Exploding gradient

Differential curvature Different magnitudes of the partial derivatives may cause the problem of vanishing and exploding gradient. This causes the learning process to require more iterations to correct the direction.

In practice, as the gradient of complex functions is only an instantaneous direction of best decrease and does not represent the direction to the minimum in the long term, many updates are required for a gradient method to converge.

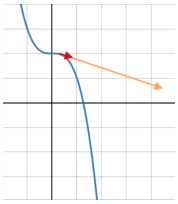
A method to mitigate this issue is to use feature normalization techniques.

Non-differentiable objective function If the objective function has a small number of non-differentiable points, the gradient descent method can be applied with minor modifications.

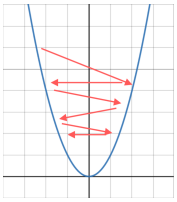
If lots of points are non-differentiable, the gradients will not be informative enough to determine a decrease direction.

Difficult topologies A cliff in the objective function causes problems when evaluating the gradient at the edge. With a small step size, there is a slow down in convergence. With a large step size, there is an overshoot that may cause the algorithm to diverge. Cliff

A valley in the objective function causes a gradient method to bounce between the sides to a point where no significant progress can be made. Valley



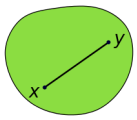
(a) Cliff region



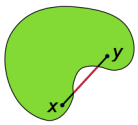
(b) Ping pong tournament in a valley

6.3 Convex functions

Convex set Informally, a set is convex if, for any two points of the set, the points laying on the segment connecting them are also part of the set. Convex set



(a) Convex set



(b) Non-convex set

Convex function Let $\Omega \subseteq \mathbb{R}^n$ be a convex set and $f : \Omega \to \mathbb{R}$. f is convex if: Convex function

$$\forall \mathbf{x}_1, \mathbf{x}_2 \in \Omega, \forall t \in [0, 1] : f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) \leq tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2)$$

In other words, the segment connecting two points of the function lays above the graph.

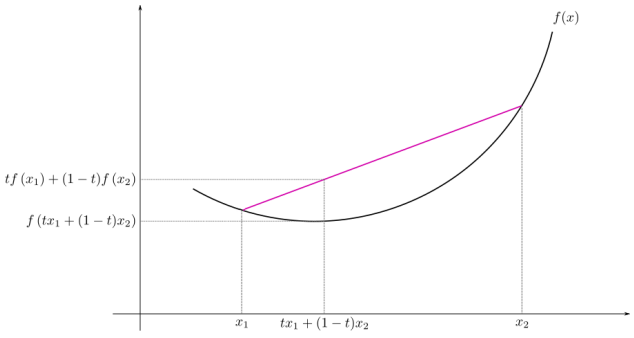


Figure 6.5: Convex function

Strictly convex function Let $\Omega \subseteq \mathbb{R}^n$ be a convex set and $f : \Omega \to \mathbb{R}$. f is strictly convex if:

Strictly convex function

$$\forall \mathbf{x}_1, \mathbf{x}_2 \in \Omega, \forall t \in [0,1] : f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) < tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2)$$

6.3.1 Properties

Convex properties

- if f convex $\Rightarrow$ any local minimum of f is also global
- if f strictly convex $\Rightarrow$ the global minimum of f is unique
- if f convex and differentiable $\Rightarrow$ any stationary point of f is a global minimum

6.3.2 Quadratic functions

A quadratic function has form:

Quadratic function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T \boldsymbol{A}\mathbf{x} - \mathbf{x}^T\mathbf{b} + c$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $\mathbf{b} \in \mathbb{R}^n$ and $c \in \mathbb{R}$.

Theorem 6.3.1. If f is a quadratic form with $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ symmetric positive semidefinite, then f is convex.

Theorem 6.3.2. If f is a quadratic form with $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ symmetric positive definite, then f is strictly convex.

Theorem 6.3.3. The least squares problem $\|\boldsymbol{A}\mathbf{x} - \mathbf{b}\|_2^2$ is a quadratic function.

Least squares quadratic function

Proof.

$$\begin{aligned}(\boldsymbol{A}\mathbf{x} - \mathbf{b})^T(\boldsymbol{A}\mathbf{x} - \mathbf{b}) &= (\mathbf{x}^T\boldsymbol{A}^T - \mathbf{b}^T)(\boldsymbol{A}\mathbf{x} - \mathbf{b}) \\ &= \mathbf{x}^T\boldsymbol{A}^T\boldsymbol{A}\mathbf{x} - \mathbf{b}^T\boldsymbol{A}\mathbf{x} - \mathbf{x}^T\boldsymbol{A}^T\mathbf{b} + \mathbf{b}^T\mathbf{b}\end{aligned}$$

As $\mathbf{b}^T\boldsymbol{A}\mathbf{x} = \mathbf{x}^T\boldsymbol{A}^T\mathbf{b}$, we have:

$$\mathbf{x}^T\boldsymbol{A}^T\boldsymbol{A}\mathbf{x} - 2\mathbf{x}^T\boldsymbol{A}^T\mathbf{b} + \mathbf{b}^T\mathbf{b}$$

Let $\boldsymbol{B} = \boldsymbol{A}^T\boldsymbol{A}$, $\mathbf{q} = \boldsymbol{A}^T\mathbf{b}$ and $c = \mathbf{b}^T\mathbf{b}$, we have the quadratic form:

$$\mathbf{x}^T\boldsymbol{B}\mathbf{x} - 2\mathbf{x}^T\mathbf{q} + c$$

$\boldsymbol{B}$ is symmetric positive semidefinite (i.e. $\|\boldsymbol{A}\mathbf{x} - \mathbf{b}\|_2^2$ is convex). Moreover, when $\boldsymbol{A}$ is full-rank, $\boldsymbol{B}$ is symmetric positive definite (i.e. strictly convex). □

6.4 Gradient descent with momentum

The momentum is an additional term to keep track of previous iterations:

Momentum

$$\Delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_{k-1} = \gamma\Delta\mathbf{x}_{k-1} - \alpha_{k-1}\nabla f(\mathbf{x}_{k-1})$$

where $\gamma \in [0,1]$. An iteration is therefore defined as:

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \alpha_{k-1}\nabla f(\mathbf{x}_{k-1}) + \gamma\Delta\mathbf{x}_{k-1}$$

6.5 Stochastic gradient descent (SGD)

Stochastic gradient descent

SGD is a stochastic approximation of gradient descent that uses an approximation of the gradient. Given N data points, the loss can be defined as the sum of the individual losses:

$$L(\mathbf{x}) = \sum_{n=1}^{N} L_n(\mathbf{x})$$

where $\mathbf{x}$ is the vector of parameters. The corresponding gradient can be computed as:

$$\nabla L(\mathbf{x}) = \sum_{n=1}^{N} \nabla L_n(\mathbf{x})$$

Mini-batch

SGD reduces the amount of computation by approximating the gradient with a subset (mini-batch) B of ∇L_n:

$$\nabla L(\mathbf{x}) = \sum_{i \in B} \nabla L_i(\mathbf{x})$$

Theorem 6.5.1. Under some assumptions and with an appropriate decrease in learning rate, SGD is guaranteed to converge to a local minimum.

Different sizes of the mini-batch result in different behavior:

Large mini-batches accurate estimates of the gradient.

Small mini-batches faster computation.

7 Probability and statistics

Probability model of a process where the underlying uncertainty is captured by random variables.

Statistics determine the underlying process that explains an observation.

7.1 Probability

State space Set Ω of all the possible results of an experiment. State space

Example. A coin is tossed two times. $\Omega = \{(\mathrm{T},\mathrm{T}),(\mathrm{T},\mathrm{H}),(\mathrm{H},\mathrm{T}),(\mathrm{H},\mathrm{H})\}$

Event Set of possible results (i.e. A is an event if $A \subseteq \Omega$) Event

Probability Let $\mathcal{E}$ be the set of all the possible events (i.e. power set of Ω). The probability is a function: Probability

$$\mathcal{P}(A) : \mathcal{E} \to [0,1]$$

Example. Let Ω be as above. Given an event $A = \{(\mathrm{T},\mathrm{H}),(\mathrm{H},\mathrm{T})\}$, its probability is: $\mathcal{P}(A) = \frac{2}{4} = \frac{1}{2}$

Conditional probability Probability of an event B, knowing that another event A happened: Conditional probability

$$\mathcal{P}(B|A) = \frac{\mathcal{P}(A \cap B)}{\mathcal{P}(A)}, \text{ with } \mathcal{P}(A) \neq 0$$

Example. A coin is tossed three times. Given the events $A = \{\text{tails two times}\}$ and $B = \{\text{one heads and one tails}\}$ We have that:

$$\Omega = \{(\mathrm{T},\mathrm{T},\mathrm{T}),(\mathrm{T},\mathrm{T},\mathrm{H}),(\mathrm{T},\mathrm{H},\mathrm{T})(\mathrm{T},\mathrm{H},\mathrm{H}),(\mathrm{H},\mathrm{T},\mathrm{T}),(\mathrm{H},\mathrm{T},\mathrm{H})(\mathrm{H},\mathrm{H},\mathrm{T}),(\mathrm{H},\mathrm{H},\mathrm{H})\}$$

$$\mathcal{P}(A) = \tfrac{4}{8} = \tfrac{1}{2} \qquad \mathcal{P}(B) = \tfrac{6}{8} = \tfrac{3}{4} \qquad \mathcal{P}(A \cap B) = \tfrac{3}{8}$$

$$\mathcal{P}(A|B) = \tfrac{3/8}{3/4} = \tfrac{1}{2} \qquad \mathcal{P}(B|A) = \tfrac{3/8}{1/2} = \tfrac{3}{4}$$

Independent events Two events A and B are independent if: Independent events

$$\mathcal{P}(A \cap B) = \mathcal{P}(A)\mathcal{P}(B)$$

It follows that:

$$\mathcal{P}(A|B) = \mathcal{P}(A) \qquad \mathcal{P}(B|A) = \mathcal{P}(B)$$

In general, given n events $A_1, \ldots, A_n$, they are independent if:

$$\mathcal{P}(A_1 \cap \cdots \cap A_n) = \prod_{i=1}^{n} \mathcal{P}(A_i)$$

7.2 Random variables

Random variable (RV) A random variable X is a function: Random variable

$$X : \Omega \to \mathbb{R}$$

Target space/Support Given a random variable X, the target space (or support) $\mathcal{T}_X$ of X is the set of all its possible values: Target space

$$\mathcal{T}_X = \{x \mid x = X(\omega), \forall \omega \in \Omega\}$$

7.2.1 Discrete random variables

Discrete random variable A random variable X is discrete if its target space $\mathcal{T}_X$ is finite or countably infinite. Discrete random variable

Example. A coin is tossed twice.

The random variable is $X(\omega) = \{\text{number of heads}\}$. We have that $\mathcal{T}_X = \{0, 1, 2\}$, therefore X is discrete.

Example. Roll a die until 6 comes out.

The random variable is $Y(\omega) = \{\text{number of rolls before 6}\}$. We have that $\mathcal{T}_Y = \{1, 2, \dots\} = \mathbb{N} \setminus \{0\}$, therefore Y is discrete as $\mathcal{T}_Y$ is a countable set.

Probability mass function (PMF) Given a discrete random variable X, its probability mass function is a function $p_X : \mathcal{T}_X \to [0, 1]$ such that: Probability mass function (PMF)

$$p_X(x) = \mathcal{P}(X = x), \forall x \in \mathcal{T}_X$$

A PMF has the following properties:

1. $p_X(x) \geq 0, \forall x \in \mathcal{T}_X$
2. $\sum_{x \in \mathcal{T}_X} p_X(x) = 1$
3. Let $A \subseteq \Omega$, $\mathcal{P}(X = x \in A) = \sum_{x \in A} p_X(x)$

We denote with $X \sim p_X$ a random variable X with PMF p_X.

Example. Let $\Omega = \{(\mathrm{T}, \mathrm{T}), (\mathrm{T}, \mathrm{H}), (\mathrm{H}, \mathrm{T}), (\mathrm{H}, \mathrm{H})\}$. Given a random variable $X = \{\text{number of heads}\}$ with $\mathcal{T}_X = \{0, 1, 2\}$. The PMF is:

$$p_X = \mathcal{P}(X = 0) = \frac{1}{4}$$
$$p_X = \mathcal{P}(X = 1) = \frac{2}{4}$$
$$p_X = \mathcal{P}(X = 2) = \frac{1}{4}$$

7.2.2 Continuous random variables

Continuous random variable A random variable X is continuous if its target space $\mathcal{T}_X$ is uncountably infinite (i.e. a subset of $\mathbb{R}$). Usually, $\mathcal{T}_X$ is an interval or union of intervals. Continuous random variable

Example. Given a random variable $Z = \{\text{Time before the arrival of a client}\}$. Z is continuous as $\mathcal{T}_Z = [a, b] \subseteq [0, +\infty[$ is an uncountable set.

Probability density function (PDF) Given a continuous random variable X, its probability density function is a function $p_X : \mathcal{T}_X \to \mathbb{R}$ such that:

Probability density function (PDF)

$$\mathcal{P}(X \in A) = \int_A p_X(x)\,dx$$

$$\mathcal{P}(a \leq X \leq b) = \int_a^b p_X(x)\,dx$$

Note that $\mathcal{P}(X = a) = \mathcal{P}(a \leq X \leq a) = \int_a^a p_X(x)\,dx = 0$

A PDF has the following properties:

1. $p_X(x) \geq 0, \forall x \in \mathcal{T}_X$
2. $\int_{x \in \mathcal{T}_X} p_X(x)\,dx = 1$
3. $\mathcal{P}(X \in A) = \int_A p_X(x)\,dx$

We denote with $X \sim p_X$ a random variable X with PDF p_X.

7.3 Discrete joint distribution

Univariate distribution Distribution with one random variable.

Univariate distribution

Multivariate distribution Distribution with multiple random variables.

Multivariate distribution

Joint probability Let X and Y be random variables respectively with target space $\mathcal{T}_X$ and $\mathcal{T}_Y$. The joint probability of X and Y has target space $\mathcal{T}_{XY} = \mathcal{T}_X \times \mathcal{T}_Y$ and its PMF is:

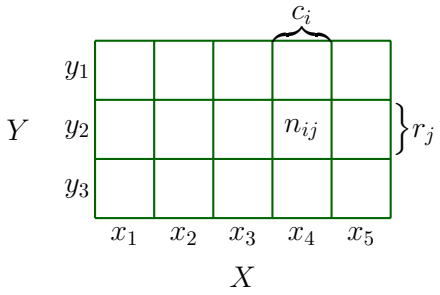
Joint probability

$$p_{XY}(x_i, y_j) = \mathcal{P}(X = x_i \cap Y = y_j)$$

$p_X(x)$ and $p_Y(y)$ are the **marginal probabilities**.

Marginal probability

Example. Let X and Y be random variables respectively with five and three possible states.



We denote with:

- N the number of events
- n_{ij} the number of events with state $X = x_i$ and $Y = y_j$ (i.e. $p(x, y) = n_{ij}$)
- $c_i = \sum_{j=1}^{3} n_{ij}$ the sum of the i-th column
- $r_j = \sum_{i=1}^{5} n_{ij}$ the sum of the j-th row

The marginal probabilities are:

$$p(x_i) = \mathcal{P}(X = x_i) = \frac{c_i}{N} \qquad p(y_j) = \mathcal{P}(Y = y_j) = \frac{r_j}{N}$$

The conditional probabilities can be computed as:

$$\mathcal{P}(Y = y_j | X = x_i) = \frac{p(x_i, y_i)}{p(x_i)} = \frac{n_{ij}/N}{c_i/N} = \frac{n_{ij}}{c_i}$$

$$\mathcal{P}(X = x_i | Y = y_j) = \frac{p(x_i, y_i)}{p(y_j)} = \frac{n_{ij}/N}{r_j/N} = \frac{n_{ij}}{r_j}$$

7.4 Rules of probability

7.4.1 Sum rule

Sum rule
Marginalization property

Given X and Y random variables. The sum rule states that:

$$p(\boldsymbol{x}) = \begin{cases} \sum_{\boldsymbol{y} \in \mathcal{T}_Y} p(\boldsymbol{x}, \boldsymbol{y}) & \text{if } \boldsymbol{y} \text{ discrete} \\ \int_{\mathcal{T}_Y} p(\boldsymbol{x}, \boldsymbol{y})\, d\boldsymbol{y} & \text{if } \boldsymbol{y} \text{ continuous} \end{cases}$$

The sum rule relates the joint distribution and a marginal distribution. In fact, the sum rule can be applied to any subset of the random variables of a joint distribution. Given $\boldsymbol{x} = \big(x_1, \ldots, x_D\big)^T$, the marginal w.r.t. x_i can be obtained by integrating/summing out all random variables except x_i:

$$p(x_i) = \int p(x_1, \ldots, x_D)\, d\boldsymbol{x}_{\setminus i}$$

7.4.2 Product rule

Product rule

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x}) = p(\boldsymbol{x}|\boldsymbol{y})p(\boldsymbol{y})$$

7.5 Bayes' theorem

Bayes' theorem

Theorem 7.5.1. Given two random variables X and Y:

$$\overbrace{p(\boldsymbol{x}|\boldsymbol{y})}^{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{x})}^{\text{likelihood}} \overbrace{p(\boldsymbol{x})}^{\text{prior}}}{\underbrace{p(\boldsymbol{y})}_{\text{evidence}}}$$

where:

Prior is the prior knowledge of the unobserved data $\boldsymbol{x}$. — Prior

Likelihood describes the relation between $\boldsymbol{x}$ and $\boldsymbol{y}$. — Likelihood

Posterior represents the quantity of interest (i.e. knowledge on $\boldsymbol{x}$ after observing $\boldsymbol{y}$). — Posterior

Evidence/Marginal likelihood normalizes the posterior. It is defined independently from $\boldsymbol{x}$ (i.e. is constant) as: — Evidence/Marginal likelihood

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})\, d\boldsymbol{x}$$

Proof. This is a direct consequence of the product rule:

$$p(\boldsymbol{x}|\boldsymbol{y})p(\boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x}) \iff p(\boldsymbol{x}|\boldsymbol{y})p(\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{y})}$$

□

Note: sometimes, instead of the full posterior, the maximum is considered (with loss of information):

$$\max_x p(x|y) = \max_x \frac{p(y|x)p(x)}{\underbrace{p(y)}_{\text{constant}}} = \max_x p(y|x)p(x)$$

7.6 Statistics

Statistic A statistic of a random variable is a deterministic function of it. Statistic

7.6.1 Mean

Expected value (univariate) Given a function g of a random variable $X \sim p(x)$, its expected value is: Expected value (univariate)

$$\mathbb{E}_X[g(x)] = \begin{cases} \sum_{x \in \mathcal{T}_X} g(x)p(x) & \text{if X is discrete} \\ \int_{\mathcal{T}_X} g(x)p(x)\,dx & \text{if X is continuous} \end{cases}$$

Expected value (multivariate) A multivariate random variable X can be seen as a vector of univariate random variables $\left(X_1, \ldots, X_D\right)^T$. Its expected value can be computed element wise as: Expected value (multivariate)

$$\mathbb{E}_X[g(\boldsymbol{x})] = \begin{pmatrix} \mathbb{E}_{X_1}[g(x_1)] \\ \vdots \\ \mathbb{E}_{X_D}[g(x_D)] \end{pmatrix} \in \mathbb{R}^D$$

Mean Given a random variable $X \sim p(x)$, the mean of X is its expected value with g defined as the identity: Mean

$$\mathbb{E}_X[x] = \begin{cases} \sum_{x \in \mathcal{T}_X} x \cdot p(x) & \text{if X is discrete} \\ \int_{\mathcal{T}_X} x \cdot p(x)\,dx & \text{if X is continuous} \end{cases}$$

7.6.2 Variance

Covariance (univariate) Given two univariate random variables X and Y, their covariance is: Covariance (univariate)

$$\mathrm{Cov}_{XY}[x, y] = \mathbb{E}_{XY}[(x - \mathbb{E}_X[x])(y - \mathbb{E}_Y[y])]$$

Lemma 7.6.1. $\mathrm{Cov}_{XY}[x, y] = \mathbb{E}_{XY}[x, y] - \mathbb{E}_X[x]\mathbb{E}_Y[y]$

Variance (univariate) The variance of a univariate random variable is given by: Variance (univariate)

$$\mathbb{V}_X[x] = \mathrm{Cov}_X[x, x]$$

Its square root is the standard deviation $\sigma(x)$.

Covariance (multivariate) Given two multivariate random variables X and Y with states $\boldsymbol{x} \in \mathbb{R}^D$ and $\boldsymbol{y} \in \mathbb{R}^E$, their covariance is: Covariance (multivariate)

$$\text{Cov}_{XY}[\boldsymbol{x}, \boldsymbol{y}] = \text{Cov}_{XY}[\boldsymbol{y}, \boldsymbol{x}]^T = \mathbb{E}_{XY}[\boldsymbol{x}\boldsymbol{y}^T] - \mathbb{E}_X[\boldsymbol{x}]\mathbb{E}_Y[\boldsymbol{y}]^T \in \mathbb{R}^{D \times E}$$

Variance (multivariate) Given a multivariate random variable X with states $\boldsymbol{x} \in \mathbb{R}^D$ and mean vector $\boldsymbol{\mu} \in \mathbb{R}^D$. Its variance is given by: Variance (multivariate)

$$\begin{aligned}
\mathbb{V}_X[\boldsymbol{x}] &= \text{Cov}_X[\boldsymbol{x}, \boldsymbol{x}] \\
&= \mathbb{E}_X[\boldsymbol{x}\boldsymbol{x}^T] - \mathbb{E}_X[\boldsymbol{x}]\mathbb{E}_X[\boldsymbol{x}]^T \\
&= \begin{pmatrix} \text{Cov}[x_1, x_1] & \text{Cov}[x_1, x_2] & \cdots & \text{Cov}[x_1, x_D] \\ \text{Cov}[x_2, x_1] & \text{Cov}[x_2, x_2] & \cdots & \text{Cov}[x_2, x_D] \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}[x_D, x_1] & \text{Cov}[x_D, x_2] & \cdots & \text{Cov}[x_D, x_D] \end{pmatrix} \in \mathbb{R}^{D \times D}
\end{aligned}$$

This matrix is called covariance matrix and is symmetric positive semidefinite.

Correlation Given two random variables X and Y, their correlation is: Correlation

$$\text{corr}[x, y] = \frac{\text{Cov}[x, y]}{\sqrt{\mathbb{V}[x]\mathbb{V}[y]}} \in [-1, 1]$$

- When $\text{corr}[x, y] \to +1$, x and y are expected to grow together.
- When $\text{corr}[x, y] \to -1$, x grows when y decreases and vice versa.
- When $\text{corr}[x, y] \to 0$, x and y are not correlated.

7.6.3 Empirical mean and variance

In practice, it is not always possible to compute statistics on the real population. Empirical observations can be made on a (finite) subset of the real population sampled as a finite number of identical random variables $X_1, \ldots, X_N$.

Empirical mean Empirical mean

$$\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n$$

Empirical variance Empirical variance

$$\sigma^2 = \frac{1}{N} \sum_{n=1}^{N} (x_n - \bar{x})^2$$

7.7 Random variables properties

7.7.1 Manipulations

- $\mathbb{E}[\boldsymbol{x} + \boldsymbol{y}] = \mathbb{E}[\boldsymbol{x}] + \mathbb{E}[\boldsymbol{y}]$ Manipulations of random variables
- $\mathbb{E}[\boldsymbol{x} - \boldsymbol{y}] = \mathbb{E}[\boldsymbol{x}] - \mathbb{E}[\boldsymbol{y}]$
- $\mathbb{V}[\boldsymbol{x} + \boldsymbol{y}] = \mathbb{V}[\boldsymbol{x}] + \mathbb{V}[\boldsymbol{y}] + \text{Cov}[\boldsymbol{x}, \boldsymbol{y}] + \text{Cov}[\boldsymbol{y}, \boldsymbol{x}]$
- $\mathbb{V}[\boldsymbol{x} - \boldsymbol{y}] = \mathbb{V}[\boldsymbol{x}] + \mathbb{V}[\boldsymbol{y}] - \text{Cov}[\boldsymbol{x}, \boldsymbol{y}] - \text{Cov}[\boldsymbol{y}, \boldsymbol{x}]$

7.7.2 Statistical independence

Two random variables X and Y are statistically independent iff: Statistical independence

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x})p(\boldsymbol{y})$$

Theorem 7.7.1. If X and Y are statistically independent, then:

- $p(\boldsymbol{x}|\boldsymbol{y}) = p(\boldsymbol{x})$ and $p(\boldsymbol{y}|\boldsymbol{x}) = p(\boldsymbol{y})$
- $\mathbb{V}_{XY}[\boldsymbol{x} + \boldsymbol{y}] = \mathbb{V}_X[\boldsymbol{x}] + \mathbb{V}_Y[\boldsymbol{y}]$
- $\text{Cov}_{XY}[\boldsymbol{x}, \boldsymbol{y}] = \bar{\boldsymbol{0}}$

7.7.3 Conditional independence

Two random variables X and Y are conditionally independent given Z iff: Conditional independence

$$p(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{y}|\boldsymbol{z}) \, \forall \boldsymbol{z} \in \mathcal{T}_Z$$

7.7.4 Inner product

Given two zero mean random variables X and Y, their inner product is defined as: Inner product of random variables

$$\langle X, Y \rangle = \text{Cov}[x, y]$$

The covariance matrix is symmetric, positive definite.
Moreover, we have that:

- $\|X\| = \sqrt{\langle X, X \rangle} = \sqrt{\text{Cov}[x, x]} = \sqrt{\mathbb{V}[x]} = \sigma[x]$
- $\cos\theta = \frac{\langle X,Y \rangle}{\|X\| \cdot \|Y\|} = \frac{\text{Cov}[x,y]}{\sqrt{\mathbb{V}[x]\mathbb{V}[y]}}$, where θ is the angle between X and Y.
- $X \perp Y \iff \langle X, Y \rangle = 0 \iff \text{Cov}[x, y] = 0 \iff X$ and Y uncorrelated

7.8 Common distributions

7.8.1 Discrete random variables

Uniform distribution Given a discrete random variable X with $\#(\mathcal{T}_X) = N$, X has an uniform distribution if: Uniform distribution

$$p_X(x) = \frac{1}{N}, \forall x \in \mathcal{T}_X$$

Poisson distribution Given a discrete random variable X with mean λ, X has a poisson distribution if: Poisson distribution

$$p_X(x) = e^{-\lambda}\frac{\lambda^x}{x!}, \forall x \in \mathcal{T}_X$$

A poisson distribution has $\mathbb{E}[x] = \lambda$ and $\mathbb{V}[x] = \lambda$.

7.8.2 Continuous random variables

Continuous uniform distribution

Continuous uniform distribution Given a continuous random variable X with $\mathcal{T}_X = [a, b]$, X has a continuous uniform distribution if:

$$p_X(x) = \frac{1}{b-a}, \forall x \in \mathcal{T}_X$$

Normal distribution

Normal distribution Given a continuous random variable X and the parameters μ (mean) and σ (variance). X has a normal distribution if:

$$p_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}, \forall x \in \mathcal{T}_X$$

In the multivariate case, it is defined as:

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{D}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} e^{(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))} \in \mathbb{R}$$

where $\boldsymbol{\mu}$ is the mean vector and $\boldsymbol{\Sigma}$ the covariance matrix.

Standard normal distribution

Standard normal distribution Normal distribution with $\mu = 0$ and $\sigma = 1$ (univariate) or $\boldsymbol{\mu} = \bar{\mathbf{0}}$ and $\boldsymbol{\Sigma} = \boldsymbol{I}$ (multivariate).

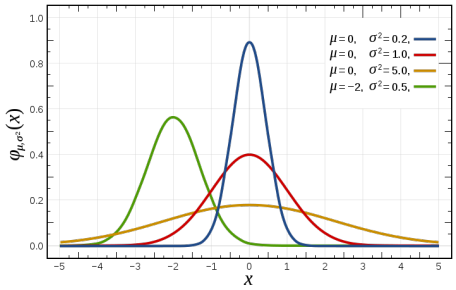


Figure 7.1: Normal distributions and standard normal distribution

Gaussian sum and linear transformations

Theorem 7.8.1 (Linearity)**.** Given X and Y independent Gaussian random variables with $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$ and $p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$. It holds that:

$$p(a\boldsymbol{x} + b\boldsymbol{y}) = \mathcal{N}(a\boldsymbol{\mu}_x + b\boldsymbol{\mu}_y, a^2\boldsymbol{\Sigma}_x + b^2\boldsymbol{\Sigma}_y)$$

8 Machine learning

8.1 Models

Function model The model (predictor) is a deterministic function: Function model

$$f : \mathbb{R}^D \to \mathbb{R}$$

In this course, only linear functions are considered:

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \cdots + \theta_D x_D = \boldsymbol{\theta}^T \mathbf{x}$$

where $\mathbf{x} = \big(1, x_1, \ldots, x_D\big)$ is the input vector and $\boldsymbol{\theta} = \big(\theta_0, \ldots, \theta_D\big)$ is the parameter vector.

Probabilistic model The model is a multivariate probabilistic distribution that is able to quantify uncertainty in noisy data. Probabilistic model

8.2 Learning

8.2.1 Empirical risk minimization

Used for function models. The parameters of the predictor are directly obtained as an optimization problem that aims to minimize the distance between the prediction and the ground truth. Empirical risk minimization
Let $(\mathbf{x}_n, y_n)$ be a dataset of N elements where $\mathbf{x}_n \in \mathbb{R}^D$ are the examples and $y_n \in \mathbb{R}$ are the labels. We want to estimate a predictor $f_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x}$ with parameters $\boldsymbol{\theta}$ such that, with the ideal parameters $\boldsymbol{\theta}^*$, it fits the data well:

$$f_{\boldsymbol{\theta}^*}(\mathbf{x}_n) \approx y_n$$

We denote the output of the estimator as $\hat{y}_n = f_{\boldsymbol{\theta}}(\mathbf{x}_n)$.

Loss function A loss function $\ell(y_n, \hat{y}_n)$ indicates how a predictor fits the data. Loss function

An assumption commonly made in machine learning is that the dataset $(\mathbf{x}_n, y_n)$ is independent and identically distributed. Therefore, the empirical mean is a good estimate of the population mean.

Empirical risk Given the example matrix $\boldsymbol{X} = \big(\mathbf{x}_1, \ldots, \mathbf{x}_N\big) \in \mathbb{R}^{N \times D}$ and the label vector $\mathbf{y} = \big(y_1, \ldots, y_N\big) \in \mathbb{R}^N$. The empirical risk is given by the average loss: Empirical risk

$$\mathbf{R}_{\text{emp}}(f_{\boldsymbol{\theta}}, \boldsymbol{X}, \mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, \hat{y}_n)$$

Least-squares loss The least-squares loss is defined as: Least-squares loss

$$\ell(y_n, \hat{y}_n) = (y_n - \hat{y}_n)^2$$

Therefore, the minimization task is:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^D} \frac{1}{N} \sum_{n=1}^{N} (y_n - f_{\boldsymbol{\theta}}(\mathbf{x}_n))^2 = \min_{\boldsymbol{\theta} \in \mathbb{R}^D} \frac{1}{N} \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \mathbf{x}_n)^2 = \min_{\boldsymbol{\theta} \in \mathbb{R}^D} \frac{1}{N} \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\theta}\|^2$$

Expected risk The expected risk is defined as: Expected risk

$$\mathbf{R}_{\text{true}}(f_{\boldsymbol{\theta}}) = \mathbb{E}_{\mathbf{x},y}[\ell(y, f_{\boldsymbol{\theta}}(\mathbf{x}_{\text{test}}))]$$

where the parameters $\boldsymbol{\theta}$ are fixed and the samples are taken from a test set.

Overfitting A predictor $f_{\boldsymbol{\theta}}$ is overfitting when $\mathbf{R}_{\text{emp}}(f, \boldsymbol{X}_{\text{train}}, \mathbf{y}_{\text{train}})$ underestimates $\mathbf{R}_{\text{true}}(f_{\boldsymbol{\theta}})$ (i.e. the loss on the training set is low, but on the test set is high). Overfitting

Regularization Method that introduces a penalty term to the loss that helps to find a compromise between the accuracy and the complexity of the solution: Regularization

$$\bar{\ell}(y_n, \hat{y}_n) = \ell(y_n, \hat{y}_n) + \lambda \mathcal{R}(\boldsymbol{\theta})$$

where $\lambda \in \mathbb{R}^+$ is the regularization parameter and $\mathcal{R}$ is the regularizer (penalty term).

Regularized least squares A simple regularization term for the least squares problem is $\|\boldsymbol{\theta}\|^2$. The problem becomes: Regularized least squares

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^D} \{\frac{1}{N} \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\theta}\|^2 + \lambda \|\boldsymbol{\theta}\|^2\}$$

8.2.2 Maximum likelihood estimation (MLE)

Used for probabilistic models. The parameters are determined as the most likely to predict the correct label given an input.

Negative log-likelihood Given a random variable $\boldsymbol{x}$, a probability density $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ parametrized by $\boldsymbol{\theta}$ and a predictor, the negative log-likelihood of $\boldsymbol{x}$ is: Negative log-likelihood

$$\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{\theta}) = -\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$$

Note that:

- The minus is added as we are converting the problem of maximizing the likelihood to a minimization problem.
- The logarithm is useful for numerical stability.

$\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{\theta})$ indicates how likely it is to observe $\boldsymbol{x}$ with $\boldsymbol{\theta}$ as the parameters of the predictor.

Given a dataset $(\boldsymbol{x}_n, y_n)$ of N independent and identically distributed (i.i.d.) elements, optimizing the likelihood allows to find the most likely parameters to represent the dataset. As the dataset is independent, we have that:

$$p_{\boldsymbol{\theta}}(\mathbf{y}|\boldsymbol{X}) = \prod_{n=1}^{N} p_{\boldsymbol{\theta}}(y_n|\boldsymbol{x}_n)$$

where $\boldsymbol{X} = \big(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\big)$ and $\mathbf{y} = \big(y_1, \dots, y_N\big)$. Moreover, as the dataset is identically distributed, each $p_{\boldsymbol{\theta}}(y_n|\boldsymbol{x}_n)$ of the product has the same distribution.

By applying the logarithm, we have that the negative log-likelihood of a i.i.d. dataset is define as:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log p_{\boldsymbol{\theta}}(y_n|\boldsymbol{x}_n)$$

and to find good parameters $\boldsymbol{\theta}$, we solve the problem:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^D} \mathcal{L}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}\in\mathbb{R}^D} -\sum_{n=1}^{N} \log p_{\boldsymbol{\theta}}(y_n|\boldsymbol{x}_n)$$

Gaussian likelihood Using a linear model $\boldsymbol{x}^T\boldsymbol{\theta}$ as predictor and assuming that the likelihood has a Gaussian distribution as follows: Gaussian likelihood

$$p_{\boldsymbol{\theta}}(y_n \,|\, \boldsymbol{x}_n) = \mathcal{N}(y_n \,|\, \boldsymbol{x}_n^T\boldsymbol{\theta}, \sigma^2)$$

where the Gaussian distribution has mean $\boldsymbol{x}_n^T\boldsymbol{\theta}$ (i.e. $f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)$) and variance σ^2 for the n-th data point.

The negative log-likelihood is:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}) &= -\sum_{n=1}^{N} \log p_{\boldsymbol{\theta}}(y_n|\boldsymbol{x}_n) \\
&= -\sum_{n=1}^{N} \log \mathcal{N}(y_n|\boldsymbol{x}_n^T\boldsymbol{\theta}, \sigma^2) \\
&= -\sum_{n=1}^{N} \log \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_n - \boldsymbol{x}_n^T\boldsymbol{\theta})^2}{2\sigma^2} \right) \right) \\
&= -\sum_{n=1}^{N} \log \exp\left(-\frac{(y_n - \boldsymbol{x}_n^T\boldsymbol{\theta})^2}{2\sigma^2} \right) - \sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} \\
&= \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - \boldsymbol{x}_n^T\boldsymbol{\theta})^2 - \sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}}
\end{aligned}$$

The minimization problem becomes:

$$\begin{aligned}
\min_{\boldsymbol{\theta}\in\mathbb{R}^D} \mathcal{L}(\boldsymbol{\theta}) &= \min_{\boldsymbol{\theta}\in\mathbb{R}^D} \overbrace{\frac{1}{2\sigma^2}}^{\text{constant}} \sum_{n=1}^{N} (y_n - \boldsymbol{x}_n^T\boldsymbol{\theta})^2 - \overbrace{\sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}}}^{\text{constant}} \\
&= \min_{\boldsymbol{\theta}\in\mathbb{R}^D} \sum_{n=1}^{N} (y_n - \boldsymbol{x}_n^T\boldsymbol{\theta})^2 \\
&= \min_{\boldsymbol{\theta}\in\mathbb{R}^D} \|\mathbf{y} - \boldsymbol{\theta}\boldsymbol{X}\|^2
\end{aligned}$$

which corresponds to the least squares problem.

(a) When the parameters are good, the label will be near the mean (i.e. predictor)

(b) When the parameters are bad, the label will be far the mean

Figure 8.1: Geometric interpretation of the Gaussian likelihood. (not sure if this is correct)

8.2.3 Maximum a posteriori estimation (MAP)

Maximum a posteriori estimation uses the opposite distribution of MLE and maximizes: Maximum a posteriori (MAP)

$$\max_{\theta \in \mathbb{R}^D} p(\theta | \boldsymbol{X}, \mathbf{y}) = \min_{\theta \in \mathbb{R}^D} -p(\theta | \boldsymbol{X}, \mathbf{y})$$

In other words, it maximizes the probability of a set of parameters θ given the observation of the dataset $(\boldsymbol{X}, \mathbf{y})$. By applying the Bayes' theorem, the problem becomes:

$$\begin{aligned} \min_{\theta \in \mathbb{R}^D} -\frac{p(\mathbf{y}|\boldsymbol{X}, \theta)p(\theta)}{\underbrace{p(\mathbf{y}|\boldsymbol{X})}_{\text{constant}}} &= \min_{\theta \in \mathbb{R}^D} -p(Y|\boldsymbol{X}, \theta)p(\theta) \\ &= \min_{\theta \in \mathbb{R}^D} \{-\log p(\mathbf{y}|\boldsymbol{X}, \theta) - \log p(\theta)\} \end{aligned}$$

Gaussian posteriori By assuming that the conditional probability of the dataset follows a Gaussian distribution (as in MLE), the problem becomes: Gaussian posteriori

$$\min_{\theta \in \mathbb{R}^D} \{-\log p(\mathbf{y}|\boldsymbol{X}, \theta) - \log p(\theta)\} = \min_{\theta \in \mathbb{R}^D} \{\|\mathbf{y} - \theta \boldsymbol{X}\|^2 - \log p(\theta)\}$$

Moreover, assuming that $p(\theta) \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, we have that:

$$-\log p(\theta) = \frac{1}{2\sigma^2}\|\theta\|^2$$

Therefore, the problem becomes:

$$\min_{\theta \in \mathbb{R}^D} \{\|\mathbf{y} - \theta \boldsymbol{X}\|^2 + \lambda \|\theta\|^2\}$$

MAP can be seen as a regularization factor for MLE.

8.3 Linear regression

Given a dataset of inputs $\mathbf{x}_n \in \mathbb{R}^D$ with corresponding labels $y_n = f(\mathbf{x}_n) + \varepsilon$, where $f : \mathbb{R}^D \to \mathbb{R}$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian noise, we want to estimate the function f. Linear regression

Model We use as predictor:

$$f(\mathbf{x}) = \mathbf{x}^T \theta$$

Because of the noise, we use a probabilistic model with likelihood:

$$p_\theta(y \mid \mathbf{x}) = \mathcal{N}(y \mid f(\mathbf{x}), \sigma^2)$$

Parameter estimation To estimate $\boldsymbol{\theta}$, we can use MLE:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^D} -p_{\boldsymbol{\theta}}(\mathbf{y}|\boldsymbol{X})$$

8.3.1 Maximum likelihood estimation with features

MLE with features

Linear regression is linear only with respect to the parameters $\boldsymbol{\theta}$. Therefore, it is possible to apply any transformation to the inputs of the predictor f such that:

$$f(\mathbf{x}_n) = (\phi(\mathbf{x}_n))^T\boldsymbol{\theta}$$

where $\phi : \mathbb{R}^D \to \mathbb{R}^K$ is a transformation and $\boldsymbol{\theta} \in \mathbb{R}^K$ are the parameters.
Given a dataset of N entries $\mathbf{x}_n \in \mathbb{R}^D$ with labels $y_n \in \mathbb{R}$ and a transformation function $\phi : \mathbb{R}^D \to \mathbb{R}^K$, the transformed features can be expressed through a feature matrix $\boldsymbol{\Phi} \in \mathbb{R}^{N\times K}$:

$$\boldsymbol{\Phi} = \begin{pmatrix} (\phi(\mathbf{x}_1))^T \\ \vdots \\ (\phi(\mathbf{x}_N))^T \end{pmatrix} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \cdots & \phi_{K-1}(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \cdots & \phi_{K-1}(\mathbf{x}_N) \end{pmatrix}$$

The negative log-likelihood can be defined as:

$$-\log p_{\boldsymbol{\theta}}(\mathbf{y} \mid \boldsymbol{X}) = \frac{1}{2\sigma^2}(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta})^T(\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}) + \text{constant}$$

As $\boldsymbol{\Phi}$ is (usually) full-rank and convex, the problem can be solved directly using normal equations:

$$\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\theta} = \boldsymbol{\Phi}^T\mathbf{y} \iff \boldsymbol{\theta} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\mathbf{y}$$

Obviously, the negative log-likelihood can also be minimized by using a gradient method.

Root mean square error (RMSE) RMSE is computed as:

Root mean square error (RMSE)

$$\sqrt{\frac{1}{N}\|\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - (\phi(\mathbf{x}_n))^T\boldsymbol{\theta})^2}$$

Differently from MSE, RMSE allows to compare errors of datasets with different sizes and scales its result to the labels.

By comparing the RMSE of the train and test sets, it is possible to check if a model is overfitting.

Polynomial regression The transformation function $\phi : \mathbb{R} \to \mathbb{R}^K$ is defined as:

Polynomial regression

$$\phi(x) = \begin{pmatrix} \phi_0(x) \\ \phi_1(x) \\ \phi_2(x) \\ \vdots \\ \phi_{K-1}(x) \end{pmatrix} = \begin{pmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^{K-1} \end{pmatrix}$$

The predictor is then defined as:

$$\begin{aligned} f(x) &= (\phi(x))^T\boldsymbol{\theta} \\ &= \sum_{i=0}^{K-1}\phi_i(x)\vartheta_i = \sum_{i=0}^{K-1} x^i\vartheta_i \end{aligned}$$

<end of course>