Statistical and Mathematical Methods for Artificial Intelligence

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 into a finite number of iterations. Truncation error

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

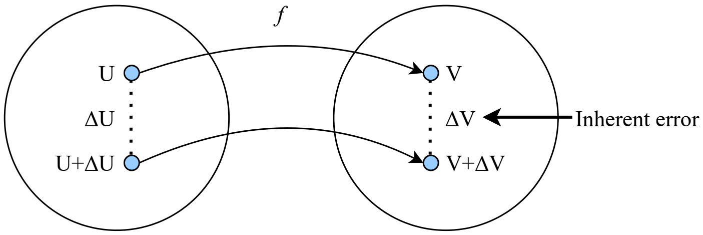

Figure 1: Inherent error visualization

1.2 Error measurement

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

Absolute error

$$E_a = \hat{x} - x \tag{1}$$

Absolute error

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

Relative error

$$E_a = \frac{\hat{x} - x}{x} \tag{2}$$

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:

$$x = \texttt{sign}(x) \cdot (d_1\beta^{-1} + d_2\beta^{-2} + \dots d_n\beta^{-n})\beta^p \tag{3}$$

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 (3) can be represented using the normalized scientific notation as: Normalized scientific notation

$$x = \pm(0.d_1d_2\dots)\beta^p \tag{4}$$

where $0.d_1d_2\dots$ 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:

$$x = \pm(0.d_1 d_2 \dots d_t)\beta^p \quad L \leq p \leq U \tag{5}$$

Example 1.1. 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.

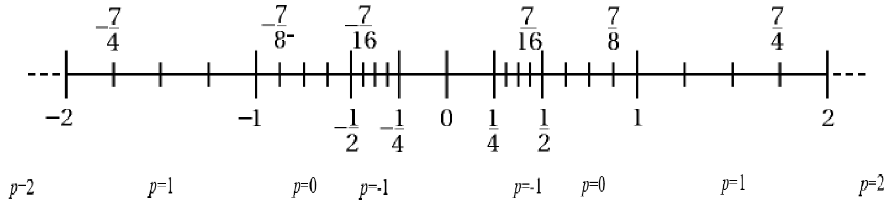

Figure 2: Floating-point numbers in $\mathcal{F}(2, 3, -1, 2)$

1.4.2 Numbers 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 values is approximated as 0.

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

1.4.3 Machine precision

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:

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 3).

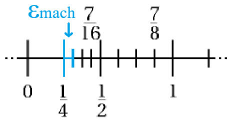

Figure 3: Visualization of $\varepsilon_{\text{mach}}$ in $\mathcal{F}(2, 3, -1, 2)$

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 storing system)
2. Stores the result as $\texttt{fl}(x + y)$

A floating-point operation causes a small rounding error:

$$\left|\frac{(x \oplus y) - (x + y)}{x + y}\right| < \varepsilon_{\text{mach}} \tag{6}$$

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

Cancellation

Example 1.2. Given $x = 1$ and $y = 1 \cdot 10^{-16}$, we want to compute $x + y$ in $\mathcal{F}(10, 16, U, L)$.

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

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

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. $\forall \boldsymbol{u} \in V : \bar{\mathbf{0}} + \boldsymbol{u} = \boldsymbol{u} + \bar{\mathbf{0}} = \boldsymbol{u}$
3. An identity element for scalar multiplication exists: $\forall \boldsymbol{u} \in V : 1\boldsymbol{u} = \boldsymbol{u}$
4. Each vector has its opposite: $\forall \boldsymbol{u} \in V, \exists \boldsymbol{a} \in V : \boldsymbol{a} + \boldsymbol{u} = \boldsymbol{u} + \boldsymbol{a} = \bar{\mathbf{0}}$
5. Distributive properties:

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

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

6. Associative property:

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

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

2.1.1 Basis

Basis

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

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

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 2.1. 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 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ is defined as: Dot product

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

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

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

Theorem 2.1. A square matrix $\boldsymbol{A}$ with $\text{Ker}(\boldsymbol{A}) = \{\bar{\boldsymbol{0}}\}$ is non singular.

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 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$:

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

Common norms are:

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

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

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

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

Example 2.2. Let $\boldsymbol{x} = (1, 1000)$ and $\boldsymbol{y} = (999, 1000)$. Their norms are:

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

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 $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$:

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

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 $\|\boldsymbol{A}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |a_{i,j}|$

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

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

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

Positive definite matrix

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

It has the following properties:

1. The null space of $\boldsymbol{A}$ has the null vector only: $\text{Ker}(\boldsymbol{A}) = \{\bar{\boldsymbol{0}}\}$.
   Which implies that $\boldsymbol{A}$ is non-singular (Theorem 2.1).
2. The diagonal elements of $\boldsymbol{A}$ are all positive.

Theorem 2.2. If the eigenvalues of a symmetric matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ are all positive. Then $\boldsymbol{B}$ is positive definite.

2.5 Orthogonality

Angle between vectors

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

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

Orthogonal vectors

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

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

Orthonormal vectors

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

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

Theorem 2.3. The canonical basis of a vector space is orthonormal.

Orthogonal matrix

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

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:

$$\|\boldsymbol{A}\boldsymbol{x}\|^2 = \|\boldsymbol{x}\|^2$$

3. The angle between two vectors is unchanged when both are mapped through an orthogonal matrix:

$$\cos\omega = \frac{(\boldsymbol{A}\boldsymbol{x})^T(\boldsymbol{A}\boldsymbol{y})}{\|\boldsymbol{A}\boldsymbol{x}\| \cdot \|\boldsymbol{A}\boldsymbol{y}\|} = \frac{\boldsymbol{x}^T\boldsymbol{y}}{\|\boldsymbol{x}\| \cdot \|\boldsymbol{y}\|}$$

Orthogonal basis

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

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

Orthonormal basis

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

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

Orthogonal complement

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:

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

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

Normal vector

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

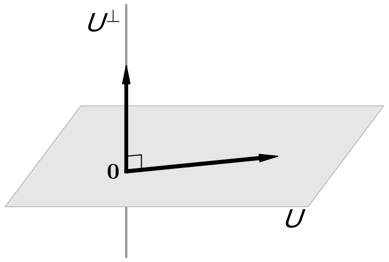

Figure 4: Orthogonal complement of a subspace $U \subseteq \mathbb{R}^3$

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 $\boldsymbol{x} \in \mathbb{R}^n$ 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 $\boldsymbol{B} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_m) \in \mathbb{R}^{n \times m}$ an ordered basis of U.

A projection $\pi_U(\boldsymbol{x})$ represents $\boldsymbol{x}$ as a linear combination of the basis:

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

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

3 Linear systems

A linear system:

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

can be represented as:

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

where:

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

3.1 Square linear systems

Square linear system

A square linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ with $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{R}^n$ 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. $\boldsymbol{A}\boldsymbol{x}$ admits only the solution $\boldsymbol{x} = \bar{\boldsymbol{0}}$

Algebraic solution to linear systems

The solution can be algebraically determined as

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

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 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, the matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is factorized into $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$ 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

As directly solving a system with a triangular matrix has complexity $O(n^2)$ (forward or backward substitutions), the system can be decomposed to:

$$\begin{aligned} \boldsymbol{Ax} = \boldsymbol{b} &\iff \boldsymbol{LUx} = \boldsymbol{b} \\ &\iff \boldsymbol{y} = \boldsymbol{Ux} \;\&\; \boldsymbol{Ly} = \boldsymbol{b} \end{aligned} \tag{7}$$

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

1. $\boldsymbol{Ly} = \boldsymbol{b}$ (solved w.r.t. $\boldsymbol{y}$)

2. $\boldsymbol{y} = \boldsymbol{Ux}$ (solved w.r.t. $\boldsymbol{x}$)

The overall complexity is $O(\frac{n^3}{3}) + 2 \cdot O(n^2) = O(\frac{n^3}{3})$

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 $\boldsymbol{PAx} = \boldsymbol{Pb}$ and the factorization is obtained as $\boldsymbol{PA} = \boldsymbol{LU}$. The system can be decomposed to:

$$\begin{aligned} \boldsymbol{PAx} = \boldsymbol{Pb} &\iff \boldsymbol{LUx} = \boldsymbol{Pb} \\ &\iff \boldsymbol{y} = \boldsymbol{Ux} \;\&\; \boldsymbol{Ly} = \boldsymbol{Pb} \end{aligned} \tag{8}$$

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 ($\boldsymbol{P}^T\boldsymbol{Ax} = \boldsymbol{P}^T\boldsymbol{b}$) can be found as above.

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 $\boldsymbol{x}_k$ that converges to the exact solution $\boldsymbol{x}^*$:

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

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

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

Stationary methods

Stationary methods compute the sequence as:

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

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

Gradient-like methods

Gradient-like methods have the form:

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

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

3.3.1 Stopping criteria

Stopping criteria

One ore 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 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$. Given a tolerance ε, the algorithm stops when:

- $\|\boldsymbol{r}_k\| \leq \varepsilon$
- $\frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{b}\|} \leq \varepsilon$

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

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

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 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, we perturbate $\boldsymbol{A}$ and/or $\boldsymbol{b}$ and study the inherited error. For instance, if we perturbate $\boldsymbol{b}$, we obtain the following system:

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

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

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

Condition number

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

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

Ill-conditioned
Well-conditioned

A system is **ill-conditioned** if $K(\boldsymbol{A})$ is large (i.e. small perturbation on the input causes large changes in the output). Otherwise it is **well-conditioned**.