# Numerical Analysis
## *Theory*

Christian Rossi

Academic Year 2023-2024

**Abstract**

The topics of the course are:

- Floating-point arithmetic: different sources of the computational error; absolute vs relative errors; the floating point representation of real numbers; the round-off unit; the machine epsilon; floating-point operations; over- and under-flow; numerical cancellation.
- Numerical approximation of nonlinear equations: the bisection and the Newton methods; the fixed-point iteration; convergence analysis (global and local results); order of convergence; stopping criteria and corresponding reliability; generalization to the system of nonlinear equations (hints).
- Numerical approximation of systems of linear equations: direct methods (Gaussian elimination method; LU and Cholesky factorizations; pivoting; sparse systems: Thomas algorithm for tridiagonal systems); iterative methods (the stationary and the dynamic Richardson scheme; Jacobi, Gauss-Seidel, gradient, conjugate gradient methods (hints); choice of the preconditioner; stopping criteria and corresponding reliability); accuracy and stability of the approximation; the condition number of a matrix; over- and underdetermined systems: the singular value decomposition (hints).
- Numerical approximation of functions and data: Polynomial interpolation (Lagrange form); piecewise interpolation; cubic interpolating splines; least-squares approximation of clouds of data.
- Numerical approximation of derivatives: finite difference schemes of the first and second order; the undetermined coefficient method.
- Numerical approximation of definite integrals: simple and composite formulas; midpoint, trapezoidal, Cavalieri-Simpson quadrature rules; Gaussian formulas; degree of exactness and order of accuracy of a quadrature rule.
- Numerical approximation of ODEs: the Cauchy problem; one-step methods (forward and backward Euler and Crank-Nicolson schemes); consistency, stability, and convergence (hints).

# Contents

# CHAPTER 1

# Introduction

## 1.1 Numerical analysis and errors

Numerical analysis is the branch of mathematics concerned with employing electronic calculators to discover solutions for specific mathematical problems. It represents the fusion of mathematical principles and computer science. On the other hand, scientific computing also involves the formalization of models, requiring a foundation in engineering knowledge.

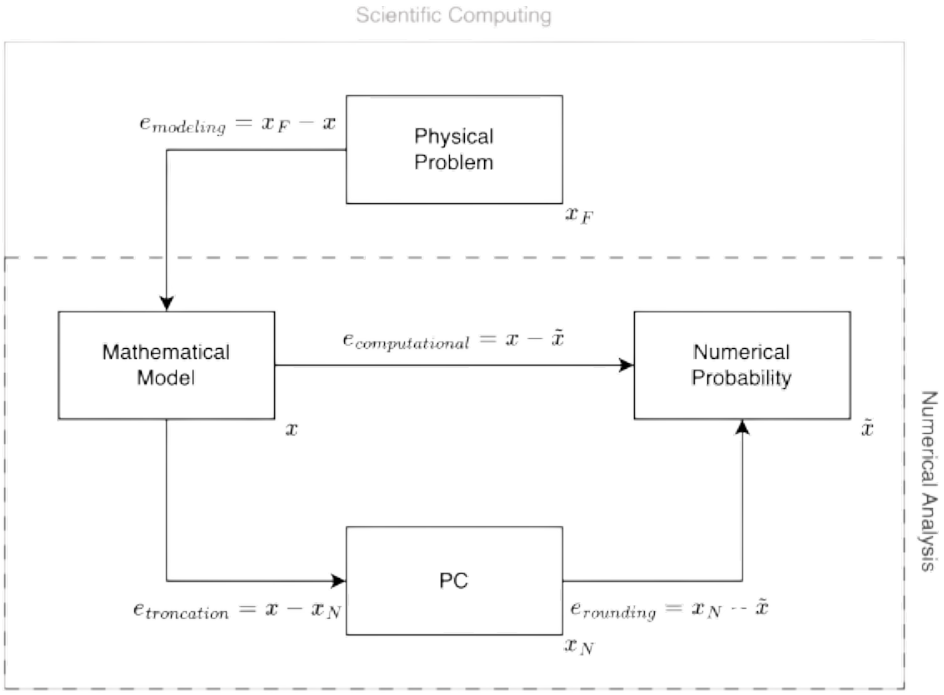



Figure 1.1: Difference between numerical analysis and scientific computing

As evident in the diagram, each computational step encounters errors. These errors can be categorized into two types:

- Absolute: $|x - \tilde{x}|$

- Relative: $\dfrac{|x-\tilde{x}|}{|x|}$, where $x \neq 0$

The relative error is considered more precise as it relates the error to the measured quantity.

**Example :** Let's take $x = 100$ and $\tilde{x} = 100.1$. In this case, the errors are as follows:

$$e_{abs} = |x - \tilde{x}| = |100 - 100.1| = 0.1$$

$$e_{rel} = \frac{|x-\tilde{x}|}{|x|} = \frac{|100 - 100.1|}{|100|} = 0.001$$

Now, consider $x = 0.2$ and $\tilde{x} = 0.1$. In this case, the errors are as follows:

$$e_{abs} = |x - \tilde{x}| = |0.2 - 0.1| = 0.1$$

$$e_{rel} = \frac{|x-\tilde{x}|}{|x|} = \frac{|0.2 - 0.1|}{|0.2|} = 0.5$$

The results show that in both examples, the absolute error is the same (0.1), which represents a 10% error. However, the relative error differs significantly. In the second example, the relative error is 50%, while in the first example, it is only 0.1%.

## 1.2 Floating point

A finite set of machine numbers, denoted as $\mathbb{F} = \{-\tilde{a}_{min}, \ldots, \tilde{a}_{max}\}$ and referred to as floating-point numbers, is employed by calculators due to their limited capacity to store numbers and perform operations. The process of mapping a real number from the set $\mathbb{R}$ to a value in $\mathbb{F}$ is achieved using the function $fl(x)$, which involves both truncation and rounding operations.

The set $\mathbb{F} = \mathbb{F}(\beta, t, L, U)$ is defined by four parameters: $\beta, t, L$ and $U$. These parameters collectively characterize every real number $fl(x) \in \mathbb{F}$, which can be expressed as:

$$fl(x) = (-1)^s (0.a_1 a_2 \ldots a_t)\beta^e$$

Here's what each component represents:

- $\beta \geq 2$ is the base, an integer determining the numeric system.
- $m = (0.a_1 a_2 \ldots a_t)$ represents the mantissa.
- $e \in \mathbb{Z}$ is the exponent, subject to the constraints $L < e < U$, with $L < 0$ and $U > 0$.
- $s = \{0, 1\}$ denotes the sign.

In defining the numbers in the mantissa set, a crucial condition is that $a_1 \neq 0$ to ensure the uniqueness of the representation. In such cases, the number is referred to as normalized.

The set of floating-point numbers exhibits several characteristic values and properties:

- Machine epsilon: it represents the gap between 1 and the smallest floating-point number greater than 1, and it is given by the formula:

$$\varepsilon_M = \beta^{1-t}$$

- Round-off error: this error reflects the relative error introduced when replacing a real number $x \in \mathbb{R} - \{0\}$ with its corresponding $fl(x) \in \mathbb{F}$. It is bounded by the condition:

$$\frac{|x - fl(x)|}{|x|} \leq \frac{1}{2}\varepsilon_M$$

  where $x \neq 0$.

- The biggest and the smallest numbers in the set: these can be calculated using the following formulas:

$$x_{min} = \beta^{L-1}$$

$$x_{max} = \beta^{U}(1 - \beta^{-t})$$

**Example :** In MATLAB, the floating-point set is defined with the following parameters:

$$(\beta = 2, t = 53, L = -1021, U = 1024)$$

With the command "eps" we can find the machine epsilon, that in MATLAB case is:

$$\epsilon_M = 2.22 \cdot 10^{-16}$$

With the command "realmin" and "realmax" we can find the smallest and the largest numbers representable that are equal to:

$$x_{min} = 2.225073858507201 \cdot 10^{-308}$$

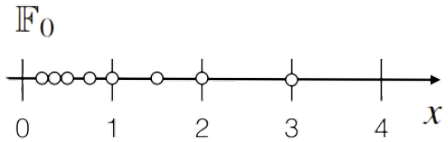
$$x_{max} = 1.797693134862316 \cdot 10^{308}$$

An important observation is that not all real numbers are representable in the floating-point set, resulting in a lack of continuity in the latter. Increasing the magnitude of numbers in $\mathbb{R}$ also leads to an increase in the gap between consecutive numbers in $\mathbb{F}$.

**Example :** Let us consider the floating number set $\mathbb{F}(2, 2, -1, 2)$. The characteristic values of this set are:

- $\epsilon_M = \beta^{1-t} = 0.5$.
- $x_{min} = \beta^{L-1} = 0.25$.
- $x_{max} = \beta^{U}(1 - \beta^{t}) = 3$.
- $\#\mathbb{F} = 2\beta^{t-1}(\beta - 1)(U - L + 1) + 1 = 16$.

The exponent can take values of $-1, 0, 1$ and 2, and the mantissa is represented as $(a_1 a_2)_\beta$ due to $t = 2$. A figure illustrates the possible positive values in this set.




| $e$ | $-1$ | $0$ | $1$ | $2$ |
|---|---|---|---|---|
| $m = (10)_2 = 2$ | $\frac{1}{4}$ | $\frac{1}{2}$ | $1$ | $2$ |
| $m = (11)_2 = 3$ | $\frac{3}{8}$ | $\frac{3}{4}$ | $\frac{3}{2}$ | $3$ |

One notable consequence of transitioning between the two sets ($\mathbb{R}$ and $\mathbb{F}$) is the loss of two essential properties: associativity and the neutral element for addition.

# CHAPTER 2

# Nonlinear equations

## 2.1 Introduction

To find a solution for a nonlinear equation $f(x)$, we aim to determine $\alpha \in \mathbb{R}$ that serves as a root of $f$ such that $f(\alpha) = 0$.

**Definition**

A point $\alpha$ is termed a *zero with multiplicity* $m$ if the following conditions are met:

- $f(\alpha) = f'(\alpha) = \cdots = f^{(m-1)}(\alpha) = 0$.
- $f^{(m)}(\alpha) \neq 0$.

When $m$ is an odd number, the function exhibits a change in sign as it crosses the zero. In contrast, when $m$ is even, the function maintains the same sign and does not undergo a sign change.

## 2.2 Iterative methods

The set comprising all polynomials of degree $n$ is represented by the symbol $\mathbb{P}_n$ encompassing all polynomials with degrees less than or equal to $n$.

**Theorem**

There is no solution in radicals to general polynomial equations of degree five or higher with arbitrary coefficients.

Therefore, for polynomials with degrees exceeding four, iterative methods are required for solutions. The fundamental concept of these methods can be outlined as follows:

1. Begin with an initial guess, denoted as $x^{(0)}$, which serves as a speculative value for $\alpha$.
2. Utilize this selected value as an input for a black-box function.

3. Take the output of the black-box function as the new $x^{(0)}$ and return to step one.

After several iterations, a sequence of values $\{x^{(n)}\}$ converges such that:

$$\lim_{n\to\infty} = \alpha$$

Additionally, the error associated with the approximated value for $\alpha$ approaches zero:

$$\lim_{n\to\infty} e^n = 0$$

This implies that the error can also be expressed as:

$$e^n = \alpha - x^{(n)}$$

All the methods we are going to discuss generate a sequence $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$ of numbers that ideally converges to $\alpha$:

$$\lim_{k\to+\infty} \left|x^{(k)} - \alpha\right| = 0$$

**Definition (*order of convergence*)**

An iterative method for approximating the zero $\alpha$ of the function $f(x)$ is considered to have a convergence order $q$ if and only if for $k > k_0$:

$$\left|x^{(k)} - \alpha\right| \leq c\left|x^{(k+1)} - \alpha\right|^q$$

There are two possible cases:

- If $q = 1$, it is termed linear convergence, with the constraint $0 < c < 1$.
- If $q > 1$, $c$ can be any positive number greater than zero.

## 2.3 Stopping conditions

Iterative methods necessitate a stopping criterion to determine when to halt the iteration process. This criterion can be based on one of four possible conditions:

- Error criterion: terminate if the absolute error satisfies $\left|x^{(k)} - \alpha\right| \leq \epsilon_e$.
- Residual criterion: Halt the process if the absolute value of the function's residual meets the condition $\left|f\left(x^{(k)}\right)\right| \leq \epsilon_r$.
- Step length criterion: stop the iteration when the absolute difference between consecutive steps adheres to $\left|x^{(k)} - x^{(k-1)}\right| \leq \epsilon_s$.
- Maximum iterations criterion: terminate the iterations if the number of iterations reaches or exceeds a specified maximum value $k_{max}$.

If none of the first three stopping criteria are satisfied, it indicates a lack of convergence in the iterative process.

## 2.4 Bisection method

**Theorem**

Let $f(x)$ be a continuous function on the interval $I = (a, b)$, that is $f \in C^0([a, b])$. If $f(a)f(b) < 0$, then there exists at least one zero $\alpha \in I$ of $f(x)$.

Suppose there exists a unique zero, denoted as $\alpha$. The bisection method employs a strategy involving the given interval and selects sub-intervals within which the function $f$ exhibits a change in sign. By following this procedure, it is assured that each interval chosen in this manner will encompass $\alpha$.

The sequence $\{x^{(k)}\}$ consisting of the midpoints of these sub-intervals will inevitably converge to $\alpha$. This convergence occurs because the lengths of these sub-intervals decrease to zero as $k$ approaches infinity. We can establish the following relationship:

$$\left|x^{(k)} - \alpha\right| \leq \frac{1}{2}\left|b^{(k)} - a^{(k)}\right|$$

Given the similarity between the left-hand side of this equation and the condition for the error, we can determine the stopping criterion as follows:

$$\left|b^{(k)} - a^{(k)}\right| \leq 2\epsilon_e$$

Now, with the tolerance $\epsilon$ at our disposal, we can calculate the minimum number of iterations required:

$$k_{min} = \left\lceil \log_2\left(\frac{|b-a|}{\epsilon}\right) - 1 \right\rceil$$

## Algorithm

The algorithm takes as input a continuous function $f$ belonging to $C(\mathbb{R})$ and an interval $[a, b]$ with the property that $f(a)f(b) \leq 0$. The output of the algorithm is an approximate value for the zero of the function.

**Algorithm 1** Algorithm for the bisection method

**for** $k = 0, 1, \ldots, n$ **do**
    $x^{(k)} = \dfrac{a+b}{2}$
    **if** $\left|b^{(k)} - a^{(k)}\right| \leq 2\epsilon_e$ **then**
        **return** $x^{(k)}$
    **else if** $f(x^{(k)})f(a) < 0$ **then**
        $b \leftarrow x^{(k)}$
    **else**
        $a \leftarrow x^{(k)}$
    **end if**
**end for**

## Summary

The advantages of this method are as follows:

- I can control the maximum allowable error.

- Convergence is assured.
- It relies solely on the evaluation of $f$

However, there are some drawbacks:

- It is effective only when $f$ changes sign within the interval.
- Convergence tends to be slow.

## 2.5 Newton method

The bisection method relies solely on the sign of the function $f$ at the endpoints of sub-intervals. However, a more efficient approach can be developed by leveraging both the function values and its derivative. For a differentiable function $f$, we can use the tangent line at a point $x^{(k)}$:

$$y(x) = f(x^{(k)}) + f^{'}(x^{(k)})(x - x^{(k)})$$

This equation represents the tangent to the curve $(x, f(x))$ at the point $x^{(k)}$. Assuming that $x^{(k+1)}$ is a point where $f(x^{(k+1)}) = 0$, we can derive:

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f^{'}(x^{(k)})} \qquad k \geq 0$$

provided $f^{'}(x^{(k)}) \neq 0$.

This method is commonly known as Newton's method and involves locally approximating the zero of $f$ by replacing $f$ with its tangent line. Notably, this method converges in a single step when $f$ is linear.

However, the Newton method doesn't converge for all possible initial values $x^{(0)}$, but only for those values sufficiently close to $\alpha$. To determine an appropriate initial value when the value of $\alpha$ is unknown, one can employ a few iterations of the bisection method or visually inspect the graph of $f$.

### Modified Newton method

If $f$ is twice continuously differentiable $\big(f \in C^2(\mathbb{R}), f^{'}(\alpha) \neq 0\big)$, and $x^{(0)}$ is chosen sufficiently close to $\alpha$, the Newton method exhibits quadratic convergence. In cases where $\alpha$ has a multiplicity $m$ greater than one, the Newton method converges linearly. To mitigate this linear convergence, a modified Newton method can be used:

$$x^{(k+1)} = x^{(k)} - m\frac{f(x^{(k)})}{f^{'}(x^{(k)})} \qquad k \geq 0$$

assuming $f^{'}(x^{(k)}) \neq 0$.

### Quasi-Newton method

Another variant of this method is the quasi-Newton method, which approximates the derivative using finite differences:

$$f^{'}(x^{(k)}) \simeq \frac{f(x^{(k)} + h) - f(x^{(k)})}{h}$$

### Algorithm

The inputs for this algorithm consist of a function $f \in C^1(\mathbb{R})$ and an initial guess $x^{(0)} \in \mathbb{R}$. The output of the algorithm is an approximate value for the zero of the function.

**Algorithm 2** Algorithm for the basic Newton method

1: **for** $k = 0, 1, \ldots, n$ **do**
2: $\quad x^{(k+1)} = x^{(k)} - \dfrac{f(x^{(k)})}{f'(x^{(k)})}$
3: $\quad$ **if** $k > k_{max} \vee \left|x^{(k)} - x^{(k-1)}\right| \leq \epsilon_s \vee \left|f\left(x^{(k+1)}\right)\right| \leq \epsilon_r$ **then**
4: $\quad\quad$ **return** $x^{(k+1)}$
5: $\quad$ **end if**
6: **end for**

### Summary

The advantages of the Newton method are as follows:

- Rapid convergence.
- Applicable to zeros with even multiplicity.

However, there are certain disadvantages:

- Demands computation of the derivative.
- Requires careful selection of the initial guess $x^{(0)}$.

## 2.6 Secant method

In situations where the derivative of a function $f$ is not readily available in analytical form, the Newton method cannot be employed for finding its zeros. However, we still have the capability to compute the function $f$ at arbitrary points, and in such cases, we can replace the exact value of $f'(x^{(k)})$ with an incremental ratio based on previously computed values of $f$. The secant method capitalizes on this strategy, and it converges super-linearly with a rate of $q = 1.6$.

### Algorithm

The algorithm takes two initial guesses, $x^{(0)} \in \mathbb{R}$ and $x^{(1)} \in \mathbb{R}$, as inputs. The output of the algorithm is an approximate value for the zero of the function.

**Algorithm 3** Algorithm for the secant method

1: **for** $k = 0, 1, \ldots, n$ **do**
2: $\quad x^{(k+1)} = x^{(k)} - f(x^{(k)}) \dfrac{x^{(k)} - x^{(k-1)}}{f(x^{(k)}) - f(x^{(k-1)})}$
3: $\quad$ **if** $k > k_{max} \vee \left|x^{(k)} - x^{(k-1)}\right| \leq \epsilon_s \vee \left|f\left(x^{(k+1)}\right)\right| \leq \epsilon_r$ **then**
4: $\quad\quad$ **return** $x^{(k+1)}$
5: $\quad$ **end if**
6: **end for**

## 2.7 Fixed point method

Given a function $\phi : [a, b] \to \mathbb{R}$, the objective is to find $\alpha \in [a, b]$ such that $\alpha = \phi(\alpha)$. If such $\alpha$ exists, it is referred to as a fixed point of $\phi$, and it can be computed using the following formula:

$$x^{(k+1)} = \phi(x^{(k)}) \qquad k \geq 0$$

This algorithm is known as fixed point iteration, and $\phi$ is termed iteration function.

**Proposition**

Let us suppose that $\phi(x)$ is continuous in $[a, b]$ and such that $\phi(x) \in [a, b]$ for every $x \in [a, b]$; then, there exists at least a fixed point $\alpha \in [a, b]$.

Moreover, if

$$\exists L < 1 \text{ such that } |\phi(x_1) - \phi(x_2)| \leq L\,|x_1 - x_2| \qquad \forall x_1, x_2 \in [a, b]$$

then there exists a unique fixed point $\alpha \in [a, b]$ of $\phi$ and the sequence converges to $\alpha$, for any choice of initial guess $x^{(0)} \in [a, b]$.

**Proof of the first proposition:** The function $g(x) = \phi(x) - x$ is continuous in $[a, b]$ and, thanks to assumption made on the range of $\phi$, it holds $g(a) = \phi(a) - a \geq 0$ and $g(b) = \phi(b) - b \geq 0$. By applying the theorem of zeros of continuous functions, we can conclude that $g$ has at least one zero in $[a, b]$. ■

**Proof of the second proposition:** Indeed, should two different fixed points $\alpha_1$ and $\alpha_2$ exist, then:

$$|\alpha_1 - \alpha_2| = |\phi(\alpha_1) - \phi(\alpha_2)| \leq L\,|\alpha_1 - \alpha_2| < |\alpha_1 - \alpha_2|$$

which cannot be. We prove now that the sequence $x^{(k)}$ converges to the unique fixed point $\alpha$ when $k \to \infty$, for any choice of initial guess $x^{(0)} \in [a, b]$. It holds:

$$\frac{\left|x^{(k)} - \alpha\right|}{\left|x^{(0)} - \alpha\right|} \leq L^k$$

Passing to the limit as $k \to \infty$, we obtain $\lim_{k\to\infty} \left|x^{(k)} - \alpha\right| = 0$, which is the desired result. ■

In practical applications, selecting an interval $[a, b]$ a priori that satisfies the assumptions of the previous proposition can be challenging. In such scenarios, the Ostrowski theorem provides a valuable local convergence result.

**Theorem**

Let $\alpha$ be a fixed point of a function $\phi$ which is continuous and continuously differentiable in a suitable neighborhood $\mathcal{J}$ of $\alpha$. If $\left|\phi'(\alpha)\right| < 1$, then there exists $\delta > 0$ for which $\{x^{(k)}\}$ converges to $\alpha$, for every $x^{(0)}$ such that $\left|x^{(0)} - \alpha\right| < \delta$. Moreover, it holds:

$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{x^{(k)} - \alpha} = \phi'(\alpha)$$

To satisfy the conditions of the Ostrowski theorem, the following hypothesis is considered:

$$\exists\delta|\left|\phi^{'}(\alpha)\right|<1 \quad \forall x\,|x-\alpha|<\delta$$

We can refer to this interval as $I_\delta$. When a point $x$ is chosen within this interval, it guarantees that $\phi(x)\in I_\delta$.

**Proof:** Let us take $|\phi(x)-\alpha|$, where it is satisfied the relation $|x-\alpha|<\delta$. We have:

$$|\phi(x)-\alpha|=|\phi(x)-\phi(\alpha)|\leq\left|\phi^{'}(\xi)(x-\alpha)\right|$$

So, we have that $\xi$ is between $x$ and *alpha*, and so it holds $\xi\in I_\delta$. At the meantime we have by hypothesis that:

$$\left|\phi^{'}(\xi)\right|<1$$

Now we can write:

$$\left|\phi^{'}(\xi)(x-\alpha)\right|=\left|\phi^{'}(\xi)\right||x-\alpha|\leq|x-\alpha|<\delta$$

In the end we found that:

$$|\phi(x)-\alpha|<\delta$$

So we have proved that $\phi(x)\in I_\delta$. ■

**Proof of Ostrowski:** Thanks to the Lagrange theorem, for any $k\geq 0$, there exists a point $\xi_k$ between $x^{(k)}$ and $\alpha$ such that:

$$\left|x^{(x)}-\alpha\right|=\left|\phi(x^{(x)})-\phi(\alpha)\right|=\left|\phi^{'}(\xi_k)(x^{(x)}-\alpha)\right|$$

We've determined that $\xi_k$ lies between $x^{(k)}$ and $x^{(k+1)}$. Thus, if $x^{(k)}\in I_\delta$, as shown in the previous proof, we also have $x^{(k+1)}\in I_\delta$ and $\xi_k\in I_\delta$. Consequently, we can derive:

$$\frac{\left|x^{(k+1)}-\alpha\right|}{\left|x^{(k)}-\alpha\right|}=\left|\phi^{'}(\xi_k)\right|$$

Since $\xi_k\to\alpha$, we can establish the convergence formula. ■

The fixed-point iteration converges at least linearly. When $\left|\phi^{'}(\alpha)\right|>1$, if $x^{(k)}$ is sufficiently close to $\alpha$ such that $\left|\phi^{'}(x^{(k)})\right|>1$, then $\left|\alpha-x^{(k+1)}\right|>\left|\alpha-x^{(k)}\right|$. In such cases, the sequence cannot converge to the fixed point. Conversely, when $\left|\phi^{'}(\alpha)\right|=1$, no definite conclusion can be drawn because either convergence or divergence could occur, depending on the properties of the iteration function $\phi^{'}(x)$.

**Proposition**

Assume that all hypothesis of Ostrowski's theorem are satisfied. In addition, assume that $\phi$ is twice continuously differentiable and that $\phi^{'}(\alpha)=0$ and $\phi^{''}(\alpha)\neq 0$. Then, the fixed point iteration converge with order two and:

$$\lim_{k\to\infty}\frac{x^{(k+1)}-\alpha}{\left(x^{(k)}-\alpha\right)^2}=\frac{1}{2}\phi^{''}(\alpha)$$

Given a simple zero such that $f(\alpha) = 0$ we can use the fixed point method to derive the Newton method:

$$\phi_N(x) = x - \frac{f(x)}{f'(x)}$$

If we derive the previous function we obtain:

$$\phi_N^{'}(x) = \frac{f(x)f^{''}(x)}{[f'(x)]^2}$$

And we can say that:

$$\phi_N^{'}(\alpha) = \frac{f(\alpha)f^{''}(\alpha)}{[f'(\alpha)]^2} = 0$$

**Proof:** We can note that the denominator is not null (the zero is simple, so the first derivative is not null). We assumed that $f(\alpha) = 0$, and we can say nothing about $f^{''}(\alpha)$. But since we have

$$\phi_N^{'}(\alpha) = \frac{f(\alpha)f^{''}(\alpha)}{[f'(\alpha)]^2} = \frac{0 \cdot n}{n} = \frac{0}{n} = 0$$

We have that the function evaluated in $\alpha$ is zero. ■

The stopping criterion for the fixed point iteration method is as follows:

$$\left|x^{(k+1)} - x^{(k)}\right| \leq \epsilon_s$$

**Proof:** For the case where the derivative is not null, we have:

$$\left|x^{(k+1)} - \alpha\right| = \left|\phi(x^{(k)}) - \phi(\alpha)\right| \leq \left|\phi^{'}(\xi_k)\right| \left|x^{(k)} - \alpha\right|$$

Then I add and subtract $x^{(k+1)}$:

$$\left|\phi^{'}(\xi_k)\right| \left|x^{(k)} - \alpha + x^{(k+1)} - x^{(k+1)}\right|$$

Applying the triangular inequality we obtain:

$$\left|\phi^{'}(\xi_k)\right| \left|x^{(k)} - \alpha + x^{(k+1)} - x^{(k+1)}\right| \leq \left|\phi^{'}(\xi_k)\right| \left|x^{(k+1)} - \alpha\right| \left|x^{(k)} - x^{(k+1)}\right|$$

From where we can derive:

$$\left(1 - \left|\phi^{'}(\xi_k)\right|\right) \left|x^{k+1} - \alpha\right| \leq \left|\phi^{'}(\xi_k)\right| \left|x^{(k+1)} - x^{(k)}\right|$$

That is:

$$\left|x^{(k+1)} - \alpha\right| \leq \frac{\phi^{'}(\xi_k)}{1 - |\phi'(\xi_k)|} \left|x^{(k+1)} - x^{(k)}\right|$$

■

**Proof:** For the case where the derivative is null, we have:

$$\left|x^{(k+1)} - \alpha\right| \leq \left|x^{(k+1)} - \alpha + x^{(k)} - x^{(k)}\right| \leq \left|x^{(k+1)} - x^{(k)}\right| \left|\phi x^{(k+1)} - \phi(\alpha)\right|$$

That is less or equal than:

$$\left|x^{(k+1)} - x^{(k)}\right| + \left|\phi^{'}(\xi_k)\right| \left|x^{(k+1)} - \alpha\right|$$

But for hypothesis we have that $\phi^{'}(\alpha) = 0$ we obtain:

$$\left|x^{(k+1)} - \alpha\right| \leq \left|x^{(k+1)} - x^{(k)}\right|$$

■

## 2.8 Aitken method

We have the following relation:

$$x^{(k+1)} - \alpha = \phi(x^{(k+1)}) - \phi(\alpha) = \lambda_k \left(x^{(k+1)} - \alpha\right)$$

We know that $\lambda_k$ is the first derivative of $\phi$ at a certain point $\xi$. If we know $\lambda_k$, we can express $\alpha$ as:

$$\alpha = \frac{\phi(x^{(k)}) - \lambda_k x^k}{1 - \lambda_k}$$

Additionally, the quantity:

$$A_k = \frac{\phi(\phi(x^{(k)})) - \phi(x^{(k)})}{\phi(x^{(k)}) - x^{(k)}}$$

provides a good approximation of $\lambda_k$, so we can replace $\lambda_k$ with $A_k$. By substituting, we obtain the formula used in the Aitken method.

**Theorem**

Let $\phi(x) = x - f(x)$ and $f(\alpha)$ be a simple zero. Let $\phi^{(k)}$ converge to the first order to $\alpha$, then $\phi_A$ converge to the second order. If $\phi(x)$ converges with order $p$ and $f(x)$ is still a simple zero, then $\phi_A$ converges with order $2p - 1$.

Occasionally, $\phi_A$ converges even when $\phi$ does not.

### Algorithm

**Algorithm 4** Algorithm for the Aitken method

1: **for** $k = 0, 1, \ldots, n$ **do**
2: $\quad x^{(k+1)} = x^{(k)} - \dfrac{\left[\phi(x^{(k)}) - x^{(k)}\right]^2}{\phi(\phi(x^{(k)})) - 2\phi(x^{(k)}) + x^{(k)}}$
3: $\quad$ **if** stopping criterion is satisfied **then**
4: $\quad\quad$ **return** $x^{(k+1)}$
5: $\quad$ **end if**
6: **end for**

## 2.9 Systems of nonlinear equations

### Newton method

Let's consider a system of nonlinear equations in the form:

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases}$$

where $f_1, \ldots, f_n$ are nonlinear functions. By defining $f = (f_1, \ldots, f_n)^T$ and $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^d$, we can represent the system compactly as:

$$f(x) = 0$$

To extend Newton's method to systems of equations, we replace the first derivative of the scalar function $f$ with the Jacobian matrix $J_f$ of the vector function $f$. The components of the Jacobian matrix are defined as:

$$(J_f)_{ij} = \frac{\partial f_1}{\partial x_1} \quad i, j = 1, \ldots, n$$

## Algorithm

The input of the algorithm is the initial guess $x^{(0)}$.

**Algorithm 5** Algorithm for the Newton method for systems

1: **for** $k = 0, 1, \ldots, n$ **do**
2: solve $J_F(x^{(k)})\delta x^{(k)} = -F(x^{(k)})$
3: $x^{(k+1)} \leftarrow x^{(k)} + \delta x^{(k)}$
4: **if** $\left\|\delta x^{(k+1)} \leq \epsilon\right\| \vee \left\|F(x^{(k+1)}) \leq \epsilon_r\right\|$ **then**
5: **return** $x^{(k+1)}$
6: **end if**
7: **end for**

## Broyden method

Computing the Jacobian matrix $J_f$ can be computationally expensive, especially when dealing with a large number of equations $d$. To simplify the computation of this matrix, one can use the finite difference approximation, as follows:

$$(J_f)_{ij} \approx \frac{f_i(x_1^k, \ldots, x_{j-1}^k, x_{j+h}^k, x_{j+1}^k, \ldots, x_d^k) - f(x_k)}{h}$$

However, this method can also be computationally costly. There are alternative methods for approximating the Jacobian that are less expensive, known as quasi-Newton methods. One of the most important quasi-Newton methods used for this purpose is called the Broyden method.

The secant method can be adapted for solving systems of nonlinear equations while maintaining a super-linear rate of convergence.

This adaptation involves replacing the Jacobian matrix $J_f(x^{(k)})$ of Newton's method, where $k \geq 0$ with suitable matrices $B_k$. These matrices are recursively defined, starting from an initial matrix $B_0$ that approximates of $J_f(x^{(0)})$.

The starting point is to replace the identity matrix $I_k$ with a matrix $B_k$ that satisfies the equation:

$$B_k\,(x_k - x_{k-1}) = F(x_k) - F(x_{k-1})(S)$$

Since there are infinitely many matrices that satisfy the previous equation, Broyden's idea was to select $B_k$ in a way that fulfills condition $(S)$ and minimizes the expression:

$$\min \|B_k - B_{k-1}\|_F^2$$

## Algorithm

The algorithm takes as input the initial guess $x^{(0)} \in \mathbb{R}^n$ and a given $B_0 \in \mathbb{R}^{n \times n}$ (commonly set to $I$).

**Algorithm 6** Algorithm for the Broyden method for systems

1: **for** $k = 0, 1, \ldots, n$ **do**
2: solve $B_k \delta x^{(k)} = -F(x^{(k)})$
3: $x^{(k+1)} \leftarrow x^{(k)} + \delta x^{(k)}$
4: $\delta F^{(k)} \leftarrow F(x^{(k+1)}) + F(x^{(k)})$
5: $B_{k+1} = B_k + \dfrac{\left(\delta F^{(k)} - B_k \delta x^{(k)}\right)}{\left\|\delta x^{(k)}\right\|^2} \delta x^{(k)^T}$
6: **if** $\left\|\delta x^{(k+1)} \leq \epsilon\right\| \vee \left\|F(x^{(k+1)}) \leq \epsilon_r\right\|$ **then**
7: **return** $B_{k+1}$
8: **end if**
9: **end for**

We do not need the sequence $\{B_k\}$ to converge to the Jacobian matrix $J_f(\alpha)$. Instead, it can be proven that:

$$\lim_{k \to \infty} \frac{\| \left(B_k - J_f(\alpha)\right) \left(x^{(k)} - \alpha\right) \|}{\|x^{(k)} - \alpha\|} = 0$$

This property ensures that $B_k$ serves as a useful approximation of $J_f(\alpha)$ along the error direction $x_{(k)} - \alpha$. The Broyden method exhibits a convergence rate of approximately $q \simeq 1.6$, making it super-linear.

As the Newton method for systems of nonlinear equations requires the inverse of the Jacobian for computation, it's more efficient to directly compute the inverse rather than the normal matrix. The matrix obtained with the Broyden formula always has a rank of one, allowing us to find the inverse using the Sherman-Morrison formula:

$$\left(A + uw^T\right) = A^{-1} - \frac{Auw^T A^{-1}}{1 + w^T A u}$$

Here, $A \in \mathbb{R}^{n \times n}$ represents an invertible square matrix, and $u, v \in \mathbb{R}^n$ are column vectors. With this approach, we can compute the approximation of the Jacobian using the Broyden method:

$$B_{k+1}^{-1} = B_k^{-1} + \frac{\delta x^{(k)} - B_k^{-1} \delta f_k}{\delta x^{(k)} B_k^{-1} \delta f_k} \left(\delta x^{(k)}\right) B_k^{-1}$$

## Bad Broyden method

The flawed version of the Broyden method computes the difference between two steps using the secant condition: $x_k - x_{k+1} = B_k^{-1} \left[F(x_k) - F(x_{k-1})\right]$. It then constructs the approximation of the Jacobian directly by imposing the secant condition and minimizing:

$$\min \left\|B_k^{-1} - B_{k-1}^{-1}\right\|_F^2$$

With these formulas, the Broyden method becomes:

$$B_{k+1}^{-1} = B_k^{-1} + \frac{\delta x^{(k)} - B_k^{-1} \delta f^{(k)}}{\left\|\delta f^{(k)}\right\|^2} \left[\delta f^{(k)}\right]^T$$

# CHAPTER 3

# Linear systems

## 3.1 Introduction

Nonlinear equations systems can be represented as:

$$Ax = B$$

Here, $A$ is a non-singular square matrix of dimension $n \times n$, with elements $a_{ij}$ that can be real or complex, while $x$ and $B$ are column vectors of size $n$. The vector $x$ represents the unknown solution, while $B$ is a predefined vector. In component-wise form, it can be expressed as:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

**Definition**

The matrix $A$ is *non-singular* if and only if:

$$Av = 0 \leftrightarrow v = 0$$

In other words, the kernel of $A$ only contains the vector 0 or equivalently:

$$\dim\left(\ker\left(A\right)\right) = 0$$

Additionally, it is essential that $\text{rank}(A) = n$ or $\det(A) \neq 0$.

The solution to this system can be determined using Cramer's rule:

$$x_i = \frac{\det A_i}{\det A} \qquad i = 1, \ldots, n$$

Here, $A_i$ is the matrix derived from $A$ by replacing the $i$-th column with $B$. However, the time complexity of Cramer's rule operations is on the order of $3(n+1)!$, making it impractical for most cases. There are two alternative approaches to tackle this:

- Direct methods, which provide the system's solution in a finite number of steps
- Iterative methods, which, in principle, demand an infinite number of steps.

## 3.2 Direct methods

### LU factorization method

Let $A \in \mathbb{R}^{n \times n}$. Suppose there exist two appropriate matrices $L$ and $U$, which are lower triangular and upper triangular, respectively, such that $A = LU$. This is referred to as an LU factorization of $A$. In the event that $A$ is non-singular, both $L$ and $U$ are also non-singular, and consequently, their diagonal elements are non-zero. Under these circumstances, solving $Ax = B$ can be reduced to solving two triangular systems: $Ly = B$ and $Ux = y$.

#### Forward substitution algorithm

The initial equation can be efficiently solved using the forward substitution algorithm with a time complexity of $O(n^2)$. The algorithm can be outlined as follows:

**Algorithm 7** Forward substitution algorithm

$$y_1 \leftarrow \frac{b_1}{l_{11}}$$
$$y_n \leftarrow \frac{1}{l_{11}} \left( b_i - \sum_{j=1}^{i-1} l_{ij} y_j \right) \qquad i = 2, \ldots, n$$

#### Backward substitution algorithm

Similarly, the second equation can be effectively solved using the backward substitution algorithm, also with a time complexity of $O(n^2)$. The algorithm can be summarized as follows:

**Algorithm 8** Backward substitution algorithm

$$x_n \leftarrow \frac{y_n}{u_{nn}}$$
$$x_i \leftarrow \frac{1}{u_{ii}} \left( y_i - \sum_{j=i+1}^{n} u_{ij} x_j \right) \qquad i = n-1, \ldots, 1$$

The elements within matrices $L$ and $U$ satisfy the system of nonlinear equations, represented as:

$$\sum_{r=1}^{\min(i,j)} l_{ir} u_{rj} = a_{ij} \qquad i, j = 1, \ldots, n$$

This system is underdetermined because there are $n^2$ equations and $n^2 + n$ unknowns. As a result, the LU factorization is not unique. However, by enforcing that the $n$ diagonal elements of $L$ are set to 1, the previous system becomes a determined one. This determined system can be efficiently solved using the Gauss algorithm.

### Gauss algorithm

**Algorithm 9** Gauss algorithm

**for** $k = 1, \ldots, n-1$ **do**
  **for** $i = k+1, \ldots, n$ **do**
    $l_{ik} \leftarrow \dfrac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$
    **for** $j = k+1, \ldots, n$ **do**
      $a_{ij}^{(k+1)} \leftarrow a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)}$
    **end for**
  **end for**
**end for**

The elements denoted as $a_{kk}^{(k)}$ must all possess non-zero values and are referred to as pivot elements. After completing this procedure, the upper triangular matrix $U$ is composed of elements represented by $u_{ij}$, while the coefficients $l_{ij}$ generated by this algorithm make up the matrix $L$. The determination of the elements within $L$ and $U$ necessitates approximately $\frac{2}{3}n^3$ operations.

**Proposition**

For a given matrix $A \in \mathbb{R}^{n \times n}$, its LU factorization exists and is unique if and only if the principal submatrices $A_i$ of $A$ of order $i = 1, \ldots, n-1$ are non-singular.

The conditions outlined in this proposition are satisfied by specific classes of matrices:

- Strictly dominant matrices.
- Real symmetric and positive definite matrices.
- Complex definite positive matrices.

**Example :** Given the following system:

$$\begin{cases} x_1 + 3x_2 = b_1 \\ 2x_1 + 2x_2 + 2x_3 = b_2 \\ 3x_1 + 6x_2 + 4x_3 = b_3 \end{cases}$$

Find the lower triangular and the upper triangular matrices using the Gauss algorithm. The initial matrix is:

$$A^{(0)} = \begin{bmatrix} 1 & 0 & 3 \\ 2 & 2 & 2 \\ 3 & 6 & 4 \end{bmatrix}$$

For the first iteration we compute the values for the lower matrices using the first column:

$$l_{21} = \frac{a_{21}}{a_{11}} = \frac{2}{1} = 2$$

$$l_{31} = \frac{a_{31}}{a_{11}} = \frac{3}{1} = 3$$

So, it is now possible to compute the new second and third rows with these formulas:

$$r_2 \leftarrow r_2 - l_{21} r_1$$

$$r_3 \leftarrow r_3 - l_{31} r_1$$

The new matrix now becomes:

$$A^{(1)} = \begin{bmatrix} 1 & 0 & 3 \\ 2-2 & 2-0 & 2-6 \\ 3-3 & 6-0 & 4-9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 2 & -4 \\ 0 & 6 & -5 \end{bmatrix}$$

For the first iteration we compute the values for the lower matrices using the first column (always from $A^{(0)}$):

$$l_{32} = \frac{a_{32}}{a_{22}} = \frac{6}{2} = 3$$

So, it is now possible to compute the new third row with this formula:

$$r_3 \leftarrow r_3 - l_{32} r_2$$

The new matrix now becomes:

$$A^{(2)} = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 2 & -4 \\ 0-0 & 6-6 & -5+12 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 2 & -4 \\ 0 & 0 & 7 \end{bmatrix}$$

Now we have found that the lower triangular matrix is:

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

and that the upper triangular matrix is:

$$U = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 2 & -4 \\ 0 & 0 & 7 \end{bmatrix}$$

## Cholesky factorization

When $A \in \mathbb{R}^{n \times n}$ is symmetric and positive definite, it's possible to establish a specialized factorization:

$$A = R^T R$$

Here, $R$ is an upper triangular matrix with positive diagonal elements. This factorization is known as the Cholesky factorization and involves approximately $\frac{1}{3}n^3$ operations. It's noteworthy that, due to symmetry, only the upper part of $A$ is stored, and $R$ can be accommodated in the same memory space.

The elements of $R$ can be computed using the following algorithm:

**Algorithm 10** Cholesky factorization algorithm

$$r_{jj} \leftarrow \sqrt{a_{jj} - \sum_{k=1}^{j-1} r_{kj}^2}$$
$$r_{ij} \leftarrow \frac{1}{r_{ii}} \left( a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj} \right) \qquad i = 1, \ldots, j-1$$

In some instances, it is possible to adjust the matrix $A$ to apply this factorization by utilizing an appropriate matrix $P$:

$$P^T A P = R^T R$$

## The pivoting technique

The pivoting technique allows us to achieve the LU factorization for any non-singular matrix. By performing a suitable row permutation of the original matrix $A$, we can make the entire factorization process feasible, even if the conditions of the proposition are not met, as long as $\det(A) \neq 0$. The decision regarding which row to permute can be made at each step $k$, when a zero diagonal element $a_{kk}^{(k)}$ is generated. Since a row permutation involves changing the pivot element, this method is known as row pivoting. The factorization obtained in this manner preserves the original matrix, albeit with a row permutation. Specifically, we have:

$$PA = LU$$

Here, $P$ is a suitable permutation matrix, initially set as the identity matrix.

Whenever during the procedure, two rows of $A$ are permuted, the same permutation must be applied to the corresponding rows of $P$. In the end, we need to solve the following systems:

$$Ly = PB$$

$$Ux = y$$

In addition to zero pivots, small elements $a_{kk}^{(k)}$ can also pose problems. In such cases, potential round-off errors affecting the coefficients $a_{kj}^{(k)}$ can be significantly amplified. Therefore, it is advisable to implement pivoting at each step of the factorization process by searching among all potential pivot elements $a_{ik}^{(k)}$, with $i = k, \ldots, n$, for the one with the maximum modulus.

**Algorithm 11** Gauss algorithm with pivoting

```
for k = 1, ..., n − 1 do
    find r̄ such that |a_{r̄k}^{(k)}| = max_{r=k,...,n} |a_{rk}^{(k)}|
    exchange row k with row r̄ in both A and P
    for i = k + 1, ..., n do
        l_ik ← a_ik^{(k)} / a_kk^{(k)}
        for j = k + 1, ..., n do
            a_ij^{(k+1)} ← a_ij^{(k)} − l_ik a_kj^{(k)}
        end for
    end for
end for
```

## Linear system solution accuracy

When solving a linear system $Ax = B$ numerically, we are, in fact, seeking the exact solution $\widetilde{x}$ to a perturbed system:

$$(A + \delta A)\,\widetilde{x} = B + \delta b$$

Here, $\delta A$ and $\delta b$ are, respectively, a matrix and a vector that depend on the specific numerical method being employed. Let's begin by considering the case where $\delta A = 0$ and $\delta b \neq 0$, which is simpler than the more general case. Suppose $\delta A \in \mathbb{R}^{n\times n}$ and $\delta b \in \mathbb{R}^n$, with $\|\delta A\|_2 \leq \varepsilon$ and $\|\delta b\|_2 \leq \varepsilon$, where $\varepsilon > 0$. By comparing the previous equation with the general system formula, we can derive the following:

$$A\,(x - \widetilde{x}) = \delta b \Rightarrow \|x - \widetilde{x}\|_2 = \|A^{-1} - \delta b\|_2 \leq \|A^{-1}\|_2 \|\delta b\|_2$$

We can also establish that:

$$\|Ax\|_2 = \|B\|_2 \Rightarrow \|A\|_2\|x\|_2 \geq \|Ax\|_2 = \|B\|_2$$

By dividing these two equations, we derive:

$$\frac{\|\widetilde{x} - x\|_2}{\|x\|_2} \leq \frac{\|A\|_2 \, \|A^{-1}\|_2 \, \|\delta b\|_2}{\|B\|_2} = k_2(A)\frac{\|\delta b\|_2}{\|B\|_2}$$

Since the matrix is symmetric positive definite we have that:

$$k_2(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$$

If the value of the constant $k_2(A)$ is excessively high (on the order of 1000), we need to correct the result using the residual correction algorithm.

**Algorithm 12** Residual correction algorithm

1: solve $Ax = B$
2: $r \leftarrow B - Ax$
3: solve $\boldsymbol{Ad} = r$
4: $x \leftarrow x + \boldsymbol{d}$

## 3.3 Iterative methods

A common approach to formulate an iterative method involves a matrix $A$ being split into:

$$A = P - (P - A)$$

Here, $P$ represents a suitable non-singular matrix referred to as the preconditioner of $A$.

We will denote with $\rho(B)$ the spectral radius of $B$, that is, the maximum modulus of eigenvalues of $B$. If $B$ is symmetric positive definite then $\rho(B)$ coincides with the largest eigenvalue of $B$.

### Jacobi method

If the diagonal entries of matrix $A$ are non-zero, we can set $P = D$, where $D$ is the diagonal matrix containing the diagonal entries of $A$. Consequently, we have:

$$Dx^{(k+1)} = B - (A - D)\, x^{(k)} \qquad k \geq 0$$

Alternatively, when considering individual components, the Jacobi method can be expressed as follows:

**Algorithm 13** Jabobi method algorithm

1: $x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)} \right) \qquad i = 1, \ldots, n$

The stopping criteria for the Jacobi method include:

$$\frac{\left\|x^{(k+1)} - x^{(k)}\right\|_2}{\|x^{(k)}\|_2} \leq \varepsilon \quad \vee \quad \frac{\left\|B - Ax^{(k+1)}\right\|_2}{\|B\|_2} \leq \varepsilon_r$$

**Proposition**

If the matrix $A \in \mathbb{R}^{n\times n}$ is strictly diagonally dominant by row, then the Jacobi method is guaranteed to converge.

## Gauss-Seidel method

The Gauss-Seidel method iteratively computes the result using the following formula:

$$(D - E)\, x^{(k+1)} = B + Fx^{(k)}$$

Here, $-F$ represents the upper triangular part of the matrix, excluding the diagonal, $-E$ is the lower triangular part of the matrix, excluding the diagonal, and $D$ is the diagonal matrix containing the diagonal entries of matrix $A$.

**Algorithm 14** Gauss-Seidel algorithm

1: $x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right) \qquad i = 1, \ldots, n$

**Proposition**

Let $A \in \mathbb{R}^{n\times n}$ be a tridiagonal non-singular matrix whose diagonal elements are all non-null. Then the Jacobi method and the Gauss-Seidel method are either both divergent or both convergent. In the latter case, the Gauss-Seidel method is faster than Jacobi's.

It's worth noting that when $P = D - E$, it is referred to as the Gauss-Seidel method, whereas when $P = D - F$, it is known as the inverted Gauss-Seidel method.

## Richardson method

The general form of the Richardson method is expressed as:

$$P(x^{(k+1)} - x^{(k)}) = \alpha_k r^{(k)} \qquad k \geq 0$$

In this method, we refer to it as "stationary" when $\alpha_k = \alpha$ for any $k \geq 0$, and "dynamic" when $\alpha_k$ may vary during the iterations. When $P \neq I$, it is known as the preconditioned Richardson method, and the formula becomes:

$$x^{(k+1)} = x^{(k)} + \alpha_k r^{(k)}$$

**Proposition**

Let $A \in \mathbb{R}^{n\times n}$. For any non-singular matrix $P \in \mathbb{R}^{n\times n}$ the stationary Richardson method converges if and only if:

$$|\lambda_i| < \frac{2}{\alpha} \mathrm{Re}\,[\lambda_i] \qquad \forall i = 1, \ldots, n$$

Here, $\lambda_i$ are the eigenvalues of $(P^{-1}A)$. If the latter are all real, it converges if and only if:

$$0 < \alpha\lambda_i < 2 \qquad \forall i = 1, \ldots, n$$

If both $A$ and $P$ are symmetric positive definite matrices, the stationary Richardson method

converges for any possible choice of $x^{(0)}$ if and only if:

$$0 < \alpha < \frac{2}{\lambda_{max}}$$

Here, $\lambda_{max} > 0$ is the maximum eigenvalue of $(P^{-1}A)$. Moreover, the spectral radius $\rho(B_\alpha)$ of the iteration matrix $B_\alpha = I - \alpha P^{-1}A$ is minimized for $\alpha = \alpha_{opt}$, where:

$$\alpha_{opt} = \frac{2}{\lambda_{min} + \lambda_{max}}$$

$\lambda_{min}$ being the smallest eigenvalue of $P^{-1}A$. If I choose $\alpha_{opt}$, then:

$$\left\|e^{(k+1)}\right\|_A = \left\|x^{(k+1)} - x^{(k)}\right\|_A \leq \frac{k_2(P^{-1}A) - 1}{k_2(P^{-1}A) + 1} \left\|e^{(k)}\right\|_A$$

## Gradient method

If matrix $A$ is symmetric positive definite, solving the equation $Ax = B$ is equivalent to finding the unique minimum of the function $\phi(y) = \frac{1}{2}y^T Ay - B^T y$. Given an initial guess $x^{(0)}$, we compute the values as follows:

$$x^{(k+1)} - \alpha_k \nabla\phi(x^{(k)}) = x^{(k)} + \alpha_k r^{(k)}$$

In the preconditioned version, we replace $r^{(k)}$ with the preconditioned residual $Z^{(k)}$, which is the solution of $= Pz^{(k)}r^{(k)}$:

$$x^{(k+1)} - \alpha_k \nabla\phi(x^{(k)}) = x^{(k)} + \alpha_k z^{(k)}$$

Let's define the function $\varphi(\alpha) = \phi(x^{(k)} + \alpha r^{(k)})$. The minimum of this function converges to $\alpha_k$, which is given by:

$$\alpha_k = \frac{\left\|r^{(k)}\right\|^2}{r^{(k)T} A r^{(k)}}$$

The convergence ratio of the gradient method is once again determined by:

$$\left\|e^{(k+1)}\right\|_A \leq \frac{k_2(P^{-1}A) - 1}{k_2(P^{-1}A) + 1} \left\|e^{(k)}\right\|_A$$

## Conjugate gradient method

When both matrices $A$ and $P$ in $\mathbb{R}^{n\times n}$ are symmetric positive definite, the conjugate gradient method can be employed. In exact arithmetic, the conjugate gradient method converges for any initial approximation $x^{(0)}$ within at most $n$ iterations. The stopping criteria for this method include monitoring the residual, controlling the step size, and imposing a maximum of $n$ iterations.

### Generalized minimal residual method

Among the iterative methods applicable to general matrices, the Generalized Minimal Residual (GMRES) stands out as one of the most significant. The primary objective of GMRES is to progressively decrease the norm of the residual at each iteration.

However, due to the increasing computational cost with each iteration, it is often more efficient to employ a modified version known as "restarted GMRES".

## 3.4 Overdetermined systems

**Definition**

A linear system $Ax = b$ with $A \in \mathbb{R}^{m \times n}$ is called *overdetermined* if $m > n$.
A linear system $Ax = b$ with $A \in \mathbb{R}^{m \times n}$ is called *underdetermined* if $m < n$.

An overdetermined system $Ax = b$ generally has no solution unless the vector $b$ is an element of range$(A)$, where:

$$\text{range}(A) = \{z \in \mathbb{R}^n | z = Ay \text{ for } y \in \mathbb{R}^n\}$$

In general, for an arbitrary $b$ we can search for a vector $x \in \mathbb{R}^n$ that minimizes the Euclidean norm of the residual, that is:

$$x = \underset{y \in \mathbb{R}^n}{\arg\min} \left( \frac{1}{2} \|r(y)\|^2 \right) = \frac{1}{2} \|b - Ay\|^2 = \Phi(y)$$

To find the minimum of the argument we have to choose an $x$ such that $\nabla\Phi(x) = 0$. By expanding the previous formula we have that:

$$\Phi(y) = \frac{1}{2}$$

# CHAPTER 4

## Approximations of functions and data

# CHAPTER 5

# Definitions

## 5.1 Matrix

**Definition**

A matrix is *diagonally dominant by row* if:

$$|a_{ii}| \geq \sum_{j=1, j\neq i} |a_{ij}| \quad i = 1, \ldots, n$$

A matrix is *strictly diagonally dominant by row* if:

$$|a_{ii}| > \sum_{j=1, j\neq i} |a_{ij}| \quad i = 1, \ldots, n$$

A matrix is *diagonally dominant by column* if:

$$|a_{ii}| \geq \sum_{j=1, j\neq i} |a_{ji}| \quad i = 1, \ldots, n$$

A matrix is *strictly diagonally dominant by column* if:

$$|a_{ii}| > \sum_{j=1, j\neq i} |a_{ji}| \quad i = 1, \ldots, n$$

A matrix $A \in \mathbb{R}^{n\times n}$ is *positive definite* if

$$\forall x \in \mathbb{R}^n, \quad x^T A x > 0$$

A matrix $A \in \mathbb{R}^{n\times n}$ is *semi positive definite* if

$$\forall x \in \mathbb{R}^n, \quad x^T A x \geq 0$$

## 5.2 Norm of vectors

**Definition**

The *p-norm* of a vector $x$ is defined as:

$$\|x\|_p = \sqrt[p]{\sum |x_i|^p}$$

The properties for the vector's norm are:

1. $\|x + y\| \leq \|x\| + \|y\|$
2. $\|\alpha x\| \leq |\alpha| \, \|x\|$

The most used vector's norms are:

- $\|x\|_1 = \sum |x_i|$
- $\|x\|_2 = \|x\| = \sqrt{\sum |x_i|^2}$
- $\|x\|_\infty = \max(x_i)$

## 5.3 Norm of matrices

**Definition**

The *p-norm* of a matrix $A$ is defined as:

$$\|A\|_p = \max_{x \neq 0} \left( \frac{\|A_x\|_p}{\|x\|_p} \right) = \max_{\|x\|_p = 1} \left( \|A_x\|_p \right)$$

The most used matrix's norms are:

- $\|A\|_1 = \max_i \sum_{i=1}^{n} |a_{ij}|$
- $\|A\|_2 = \|A\| = \sqrt{\sum \lambda_{max}(A^T A)}$
- $\|y\|_A = \sqrt{y^T A y}$
- $\|A\|_\infty = \max_j \sum_{j=1}^{n} |a_{ij}|$

**Proposition**

If the matrix $A$ is symmetric positive definite we have that:

$$\|A\|_2 = \lambda_{max}$$