الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة محمد خيضر بسكرة
كلية الهندسة المعمارية، العمران، الهندسة المدنية والري

# Lectures in Numerical Methods

**Dedicated for 2nd year bachelor students in Civil Engineering, Hydraulics, Public Works and all other specialties related to Science and Technology domain**

*Prepared by: Dr Mahdi Abdeddaim*

*Associate professor Rank A at the département of Civil Engineering and Hydraulics*

*Faculty of Architecture, Urbanism, Civil Engineering and Hydraulics*

Academic year 2025/2026

## Foreword

This course will provide you with essential computational techniques for solving mathematical problems that are often too complex for analytical solutions. Through the combination of lectures and tutorials, you will develop both the theoretical understanding and practical skills needed to implement numerical algorithms in engineering and scientific contexts.

The course will cover the following topics in sequence:

- Resolution of non-linear equations
- Polynomial interpolation
- Function approximation
- Numerical integration
- Resolution of differential equations
- Resolution of linear and nonlinear systems of equations

Each topic will include both lectures to introduce concepts and tutorials to strengthen your problem-solving ability through worked examples and practice.

All lecture notes, tutorial sheets, and supporting materials will be available on this Moodle page. Please check regularly for announcements and updates. Active participation in tutorials is strongly encouraged to reinforce your learning.

I look forward to an engaging and productive semester with you all.

## Table of Content


**General Introduction** I

**Chapter I: Resolution of nonlinear equations** 01

**I-1 Introduction** 01

**I-2 Error Analysis** 01

**I-3 Nonlinear equations** 02

**I-4 Bisection method** 02

**I-5 Fixed point method** 05

**I-6 Newton-Raphson method** 06

**I-7 Conclusion** 07

**Chapter II: Polynomial interpolation** 08

**II-1 Introduction** 08

**I-2 Lagrange interpolation method** 09

**I-3 Newton interpolation method** 12

**I-4 Conclusion** 15

**Chapter III: Function approximation** 16

**III-1 Introduction** 16

**III-2 Approximation and error** 16

**III-3 Quadratic mean** 17

**III-4 Quadratic mean for function** 17

**III-5 Inner product and orthogonality** 18

**III-6 Orthogonal systems** 18

**III-7 Pseudo-orthogonal systems** 19

**III-8 Orthogonal polynomials** 21

**III-9 Trigonometric approximation** 25

**III-10 Conclusion** 28

**Chapter IV: Numerical integration** 29

**IV.1 Introduction** 29

**IV.2 Trapezoidal method** 30

**IV.3 Composite trapezoidal method** 33
**IV.4 Simple 1/3 Simpson method** 35
**IV.5 Composite 1/3 Simpson method** 37
**IV.5 Conclusion**

**Chapter V: Ordinary differential equations**
**V.1 Introduction** 45
**V.2 Euler's method** 46
**V.3 Enhanced Euler's method** 47
**V.4 Runger-Kutta method** 48
**V.5 Conclusion** 49

**Chapter VI: Direction resolution methods for linear equations systems**
**VI.1 Introduction to linear systems** 50
**VI.2 Elementary operations on matrices lines** 52
**VI.3 Gauss elimination** 54
**VI.4 Lower-Upper (LU) decomposition** 58
**VI.5 'Crout' decomposition** 58
**IV.6 Conclusion** 61

**Chapter VII: Approximative resolution methods for linear equations systems**
**VII.1 Introduction** 62
**VII.2 Jacobian method** 64
**VII.3 Gauss-Seidel method** 66
**VII.4 Relaxation method** 69
**VII.5 Conclusion** 74

**General Conclusion** 75
**Reference list for further lectures** 76

## General Introduction

Numerical methods form an essential part of modern scientific computation. In many real problems arising in physics, engineering, computer science, and applied mathematics, exact analytical solutions are either impossible to obtain or too complicated to use in practice. Numerical methods provide systematic procedures to construct approximate solutions that are sufficiently accurate for practical purposes. They allow us to handle large systems of equations, complex integrals, and differential equations that cannot be solved with elementary techniques alone.

This handout in numerical methods is designed for second-year bachelor students in the Science and Technology common branch. It builds on the basic concepts of analysis and linear algebra studied in the first year, and shows how these ideas can be used algorithmically. Instead of focusing only on formulas, the course emphasizes procedures: how to approximate, how to iterate, how to control and estimate errors, and how to interpret numerical results. Each method is presented with its underlying idea, its mathematical justification at an accessible level, and several worked examples to illustrate how it is applied in practice.

The topics covered include numerical integration, approximation methods for functions, and iterative methods for solving linear systems, among others. For each topic, the goal is not only to learn a new formula, but also to understand when and why that formula works, its advantages and limitations, and how its parameters (such as step size or relaxation factor) influence accuracy and efficiency. Exercises are chosen to strengthen both computational skills and conceptual understanding.

By the end of the course, students should be able to select appropriate numerical techniques for simple problems, implement them by hand for small examples, and interpret their results critically. This will prepare them for more advanced courses where numerical methods are implemented on computers and used to solve realistic engineering and scientific problems.

# Chapter I: Resolution of non-linear equations

## 1. Error analysis

- Analytical resolution of mathematical problems usually results in precise and unique results even though different methods are used.
- Numerical analysis uses a multitude of resolution techniques, which results in different algorithms.
- These algorithms depend on certain parameters which influence the precision of the result.
- An important part of numerical analysis consists of containing the error's effects.

*Three types of errors can be distinguished:*

a) Modeling errors.
b) Representation errors on computers.
c) Truncation errors, for example, those resulting from the simplification of Taylor's development.

*Definition of absolute error:*

$$\Delta x = \left| x - x^* \right| \tag{1.1}$$

- $x$ is a given number, $x^*$ is the approximation of $x$.

*Note that a small absolute error signifies that the approximation is close to the exact value of $x$.*

*Definition of relative error:*

$$\Delta x = \frac{\left| x - x^* \right|}{\left| x \right|} = \frac{\left| \Delta x \right|}{x} \tag{1.2}$$

*Note that by multiplying by 100, the relative error in percentage can be obtained.*

In practice, it is difficult to evaluate absolute and relative errors since the exact value of $x$ is unknown (for example, when trying to find the roots of a high-order function).

In this case, the upper bound is used.

$$\left| x - x^* \right| \le \Delta x \tag{1.3}$$

Which can also be written

$$x^* - \Delta x \le x \le x^* + \Delta x \quad (1.4)$$

The value of $x$ is estimated based on the value of $x^*$ with an incertitude of $\Delta x$ from either side.

*Note that the absolute error $\Delta x$ can be defined based on the wanted precision.*

- ✓ The absolute error gives a quantitative measure of the error.
- ✓ The relative error gives a measure to the importance of the error.

Example:

A stopwatch with a precision of $1/10[s]$ $(0.1[\text{s}])$, is the measurement error important?

a) In the case of a marathon with a duration of 2 hours and 20 minutes, the relative error is:

$$\frac{0.1}{2\times 60\times 60 + 20\times 60} = 0.0000119$$

b) In the case of a 100 meters race which lasts about 10 seconds, the relative error is:

$$\frac{0.1}{10.0} = 0.01,\ 0.01\times 100 = 1\% \text{ of the race time}$$

## 2. Non-linear equations

It is usually required to solve algebraic equations of the form $f(x) = 0$; a value $x$ solution of $f(x) = 0$ is denoted root of the equation $f(x)$ and noted "$r$".

- It is demonstrated that no formula exists for finding the roots of polynomials having an order higher or equal to five. *Galois* demonstrated that such formulas don't exist.

For high-order polynomials, it is necessary to resort to numerical methods.

## 3. Bisection method

**The idea:** from either side of a root, the function $f(x)$ changes its sign and passes from positive to negative and vice-versa.

**Attention:** It is not always the case; the function can be tangent to the x-axis.

**In general:**

Let $[x_1, x_2]$ an interval having a change of sign, which means: $f(x_1) \times f(x_2) \prec 0$.

A middle value of the interval is defined as $x_m = \dfrac{x_1 + x_2}{2}$.

- $x_m$ is situated in the middle of the interval.

Two new intervals can now be defined as $[x_1, x_m]$ and $[x_m, x_2]$.

Now, it is enough to determine which of the two intervals possess a change of sign.

This operation is renewed until a precise solution is enclosed into an interval.

Note that the relative error in this method is given by the following formula: $E_r(x) = \dfrac{|x_2 - x_1|}{2|x_m|}$

stopping criteria $\varepsilon \lll 0$ can be defined to refine and reduce the error. The iterations will be stopped if the following condition is satisfied $\dfrac{|x_2 - x_1|}{2|x_m|} \prec \varepsilon$.

Example 1:

The function $f(x) = x^2 - 2 = 0$ expresses a change of its sign in the interval $[1,2]$ as shown in Figure 1.1. It is obvious that the root in the interval $[1,2]$ is $\sqrt{2}$. In this example, an approximation of the root will be calculated using the bisection method.

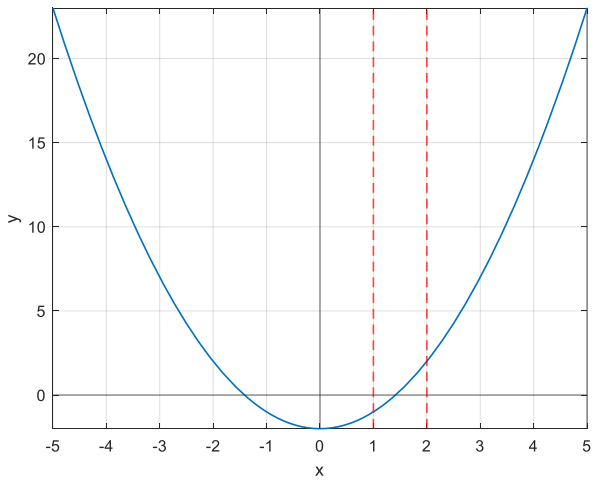



Figure 1.1. Graphical representation of the function $f(x) = x^2 - 2 = 0$

The following table details the iterations performed to obtain an error less than a stopping criterion $\varepsilon \prec 10^{-3}$

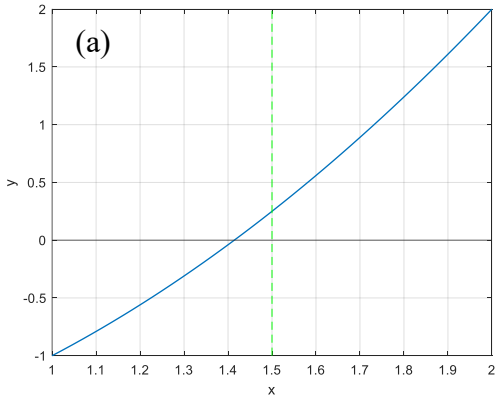
Table 1.1: Iteration for the bisection method.

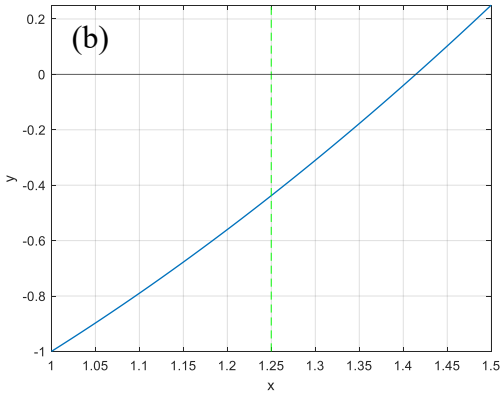
| N° | $x_1$ | $x_m$ | $x_2$ | $f(x_1)$ | $f(x_m)$ | $f(x_2)$ | $E_r(x)$ | $\%E_r(x)$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1,5 | 2 | -1 | 0,25 | 2 | 0,333333 | 33,333 |
| 2 | 1 | 1,25 | 1,50 | -1 | -0,4375 | 0,25 | 0,2 | 20 |
| 3 | 1,25 | 1,375 | 1,5 | -0,4375 | -0,109375 | 0,25 | 0,090909 | 9,091 |
| 4 | 1,375 | 1,4375 | 1,5 | -0,109375 | 0,06640625 | 0,25 | 0,043478 | 4,348 |
| 5 | 1,375 | 1,40625 | 1,4375 | -0,109375 | -0,022460938 | 0,066406250 | 0,022222 | 2,222 |
| 6 | 1,40625 | 1,421875 | 1,4375 | -0,022460938 | 0,021728516 | 0,066406250 | 0,010989 | 1,099 |
| 7 | 1,40625 | 1,4140625 | 1,421875 | -0,022460938 | -0,000427246 | 0,021728516 | 0,005525 | 0,552 |
| 8 | 1,4140625 | 1,41796875 | 1,421875 | -0,000427246 | 0,010635376 | 0,021728516 | 0,002755 | 0,275 |
| 9 | 1,4140625 | 1,416015625 | 1,417968750 | -0,000427246 | 0,005100250 | 0,010635376 | 0,001379 | 0,138 |
| 10 | 1,4140625 | 1,415039063 | 1,416015625 | -0,000427246 | 0,002335548 | 0,005100250 | 0,000690 | 0,069 |

It is worth noticing that there exists a simple method to calculate the required number of iterations to obtain an error less than the stopping criteria.

$$n > \frac{\ln\left(\frac{L}{\varepsilon}\right)}{\ln 2} \tag{1.5}$$

Herein, $n$ is the maximum number of iterations, while $L = x_2 - x_1$ is the length of the initial interval.



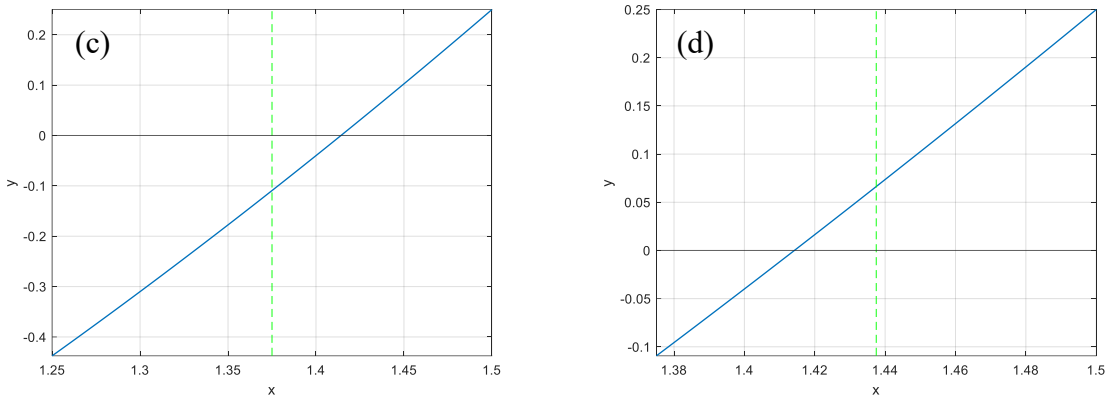



Figure 1.2. Graphical representation of the first four iterations of example 1.

***Important Remarks***

1- ***The convergence of the bisection method is not very fast, but it is ensured starting from the moment a change of sign occurs in a given interval. (A closed method that operates in a closed interval).***
2- ***The bisection method has some pathological cases that can be denoted as follows:***

*a) It may fail to find a solution if the function is tangent to the x-axis.*
*b) Even number of roots in a given interval will not lead to a change of sign, yet two roots are in the given interval.*
*c) The interval contains an odd number of roots; the sign will change, but it will be difficult to contain the roots.*

These cases can be easily avoided by graphically illustrating the function $f(x)$ in the interval of interest.

## 4. Fixed point method

### *4.1 Definition*

A fixed point of a function $g(\mathrm{x})$ is a value $x$ which remain invariable for this function. Any function can have a fixed point if it cuts the first bisector. Therefore, any solution of the form:

$$x = g(x) \tag{1.6}$$

It is a fixed point of the function $g(\mathrm{x})$. In other terms, a function that returns the same value is similar to $(y = x)$, the intersection between $g(x)$ and $(y = x)$ is being sought.

- A simple algorithm exists that allows one to determine fixed points. In fact, it is enough to perform iterations as follows:

$$\begin{cases} x_0 \text{ given} \\ x_{n+1} = g(x_n) \end{cases} \tag{1.7}$$

Starting from a value considered initial $x_0$.

This algorithm is interesting because it is general, and its convergence is relatively easy to analyse.

- Non-linear equations of the form $f(x)=0$ can be solved using the fixed-point algorithm. For this purpose, the function $f(x)=0$ must be transformed into an equivalent problem (auxiliary function) of the form $x=g(x)$.

**Attention:** there is an infinite way to transform a function into other auxiliary functions.

Example:

Using the fixed-point theory, find the two roots of the function $x^2-2x-3=0$

Note that the relative error in this method is given by the following formula: $E_r(x)=\dfrac{|x_{n+1}-x_n|}{|x_{n+1}|}$

stopping criteria $\varepsilon \lll 0$ can be defined to refine and reduce the error. The iterations will be stopped if the following condition is satisfied $\dfrac{|x_{n+1}-x_n|}{|x_{n+1}|} \prec \varepsilon$.

## 5. Newton-Raphson method

The Newton-Raphson method is among the most used methods for solving non-linear equations. The Newton-Raphson method is based on the usage of Taylor series development.

Assume an equation to be solved has the form $f(x)=0$

Based on an initial value $x_0$ (initial value of the solution), a correction $\delta x$ is being searched, such as:

$$f(x_0+\delta x)=0 \tag{1.8}$$

Using Taylor's development around $x=x_0$ the following equation is obtained.

$$f(x_0)+\frac{f'(x_0).(\delta x)^1}{1!}+\frac{f''(x_0)(\delta x)^2}{2!}+\frac{f'''(x_0)(\delta x)^3}{3!}+\ldots\ldots \quad (1.9)$$

It is sufficient to neglect the terms having an order superior or equal to 2 in $\delta x$ to obtain the following equation.

$$f(x_0)+f'(x_0).(\delta x)\approx 0 \quad (1.10)$$

The searched term can be easily isolated. Hence: $\delta x=-\frac{f(x_0)}{f'(x_0)}$

The correction $\delta x$ is, in principle, the quantity that should be added to $x_0$ to cancel the function $f(x)$. Due to the negligence of terms having an order superior or equal to 2 in Taylor's development, this correction is perfect. Therefore, further iterations are required, such as:

$$x_1=x_0+\delta x \quad (1.11)$$

The process is repeated while trying to correct the new value $x_1$ with a new quantity $\delta x$ thus:

$$x_{n+1}=x_n-\delta x \Rightarrow x_{n+1}=x_n-\frac{f(x_n)}{f'(x_n)} \quad (1.12)$$

Note that the relative error in this method is given by the following formula: $E_r(x)=\frac{|x_{n+1}-x_n|}{|x_{n+1}|}$ stopping criteria $\varepsilon \lll 0$ can be defined to refine and reduce the error. The iterations will be stopped if the following condition is satisfied $\frac{|x_{n+1}-x_n|}{|x_{n+1}|}\prec\varepsilon$.

## Chapter II: Polynomial interpolation

### 1. Introduction &Problem definition

The following problem is posed: based on a function $f(x)$ known in only $(n+1)$ points of the form $((x_i, f(x_i)$ for $i = 0,1,2,3\cdots n)$. Is it possible de construct an approximation of the function $f(x)$ for any given value of $x$ ?

The following figure resumes the situation

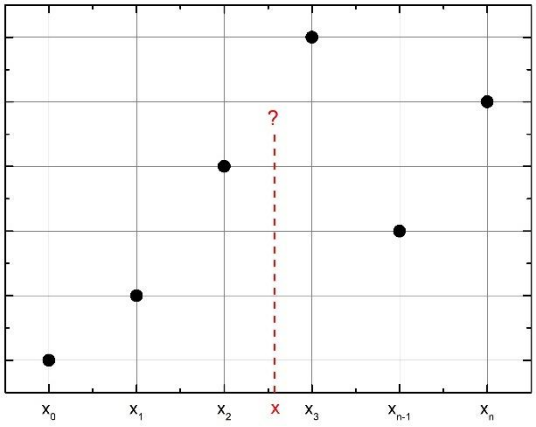



Figure 2.1: Interpolation problem graphical-wise

The idea of the solution is relatively simple. It is sufficient to construct a polynomial with a sufficient high order such as it curve passes through the collocation points.

**The polynomial:** a polynomial of degree $n$ has the following general form

$$P_n(x) = a_0x^0 + a_1x^1 + a_2x^2 + a_3x^3 + \cdots a_nx^n \ \ (a_n \neq 0) \tag{2.1}$$

- This polynomial has exactly $n$ roots (real or complex), $r$ is a root if $P_n(r) = 0$.

**Theorem:** Only and only one polynomial of degree $n$ passes through $(n+1)$ points of collocation $((x_i, f(x_i)$ for $i = 0,1,2,3\cdots n)$.

**Demonstration:** The existence of two different polynomials both of degree $n$ is supposed, these polynomials are denoted $g(x)$ and $q(x)$, respectively. Both these polynomials are supposed to pass through the $(n+1)$ points of collocation. The difference between these two polynomials is considered such as $P(x) = g(x) - q(x)$.

$P(x)$ is a polynomial of degree $n$ at most. This polynomial verifies the following equation

$$P(x_i) = g(x_i) - q(x_i) = 0 \tag{2.2}$$

The equation is valid for $i$ varying from $0$ to $n$. Hence, the polynomial $P(x)$ will have $(n+1)$ roots which is impossible.

Once the unicity of the interpolation polynomial is established. Its existence remains to be addressed; this can be accomplished using diverse numerical methods.

## 2. Lagrange interpolation

Lagrange interpolation supposes that it is known how to build $(n+1)$ polynomials $L_i(x)$ having $n$ degree and satisfying the following conditions:

$$\begin{cases} L_i(x_i) = 1 \ \ \forall\ i \\ L_i(x_j) = 0 \ \ \forall\ j \neq i \end{cases} \tag{2.3}$$

This means that the polynomial $L_i(x)$ of degree $n$ will take the value "1" for $(x_i)$ and cancels for all other collocation points. In these conditions the function $L(x)$ is defined by

$$L(x) = \sum_{i=0}^{n} f(x_i).L_i(x) \tag{2.4}$$

$L(x)$ is a polynomial of degree $n$ since each of the $L_i(x)$ is of degree $n$. Further, the polynomial $L(x)$ passes by all the $(n+1)$ collocation points. Therefor it is the searched polynomial.

### *2.1 Polynomial of degree "1"*

In this part, the methodology for determining a degree "1" polynomial will be detailed, the curve of this polynomial is a straight line that passes through the two points of collocation $(x_0, f(x_0))$ and $(x_1, f(x_1))$.

Two polynomials $L_0(x)$ and $L_1(x)$ should be constructed, these two polynomials should verify the following conditions:

$$\begin{cases} L_0(x_0) = 1 \\ L_0(x_1) = 0 \end{cases} \qquad \begin{cases} L_1(x_0) = 0 \\ L_1(x_1) = 1 \end{cases}$$

Hence, the polynomial $L_0(x)$ should cancel at $x = x_1$, thus

$$(x - x_1) \Rightarrow \text{ which cancels at } x = x_1 \tag{2.5}$$

When $x = x_0$ the term proposed in the equation (2.5) becomes $(x_0 - x_1)$ in order to obtain a "1" value when $x = x_0$ which will satisfy the first condition of the $L_0(x)$ polynomial, the appropriate division can be used. Therefor:

$$L_0(x) = \frac{(x - x_1)}{(x_0 - x_1)} \tag{2.6}$$

similar reasoning for $L_1(x)$ gives:

$$L_1(x) = \frac{(x - x_0)}{(x_1 - x_0)} \tag{2.7}$$

Therefor the polynomial $L(x)$ is

$$L(x) = f(x_0).\frac{(x - x_1)}{(x_0 - x_1)} + f(x_1).\frac{(x - x_0)}{(x_1 - x_0)} = f(x_0).L_0(x) + f(x_1).L_1(x) \tag{2.8}$$

### *2.2 Polynomial of degree "2"*

In this part, the methodology for constructing a second order polynomial passing through three points namely $(x_0, f(x_0)), (x_1, f(x_1))$ and $(x_2, f(x_2))$ is detailed. Following the same logic for construction a first order polynomial. Three functions $L_i(x)$ should be constructed, these functions should verify the following conditions:

$$\begin{cases} L_0(x_0) = 1 \\ L_0(x_1) = 0 \\ L_0(x_2) = 0 \end{cases} \qquad \begin{cases} L_1(x_0) = 0 \\ L_1(x_1) = 1 \\ L_1(x_2) = 0 \end{cases} \qquad \begin{cases} L_2(x_0) = 0 \\ L_2(x_1) = 0 \\ L_2(x_2) = 1 \end{cases}$$

Hence, the polynomial $L_0(x)$ should cancel at $x = x_1$ and $x = x_2$, thus

$$(x - x_1).(x - x_2) \Rightarrow \text{ which cancels at } x = x_1 \text{ and } x = x_2 \tag{2.9}$$

When $x = x_0$ the term proposed in the equation (2.9) becomes $(x_0 - x_1).(x_0 - x_2)$ in order to obtain a "1" value when $x = x_0$ which will satisfy the first condition of the $L_0(x)$ polynomial, the appropriate division can be used. Therefor:

$$L_0(x) = \frac{(x - x_1).(x - x_2)}{(x_0 - x_1).(x_0 - x_2)} \tag{2.10}$$

similar reasoning for $L_1(x)$ and $L_2(x)$ gives:

$$L_1(x) = \frac{(x - x_0).(x - x_2)}{(x_1 - x_0).(x_1 - x_2)} \tag{2.11}$$

$$L_2(x) = \frac{(x - x_0).(x - x_1)}{(x_2 - x_0).(x_2 - x_1)} \tag{2.12}$$

### *2.3 Polynomial of degree "n"*

The general case should be analyzed similarly to the 1st and 2nd degrees polynomials interpolations. The function $L_0(x)$ should cancels for $x = x_1, x_2, x_3 \cdots x_n$. Thus, the following function should be introduced:

$$(x - x_1).(x - x_2).(x - x_3) \cdots (x - x_n) \tag{2.13}$$

The function in equation (2.13) becomes as follow when $x = x_0$:

$$(x_0 - x_1).(x_0 - x_2).(x_0 - x_3) \cdots (x_0 - x_n) \tag{2.14}$$

After the appropriate division, the $L_0(x)$ is obtained such as

$$L_0(x) = \frac{(x - x_1).(x - x_2).(x - x_3) \cdots (x - x_n)}{(x_0 - x_1).(x_0 - x_2).(x_0 - x_3) \cdots (x_0 - x_n)} \tag{2.15}$$

Therefore, it is observed that there are $n$ factors of the form $(x - x_i)$ in this expression, there for $L_0(x)$ is a polynomial of degree $n$.

The function $L_1(x)$ is constructed such as:

$$L_1(x) = \frac{(x - x_0).(x - x_2).(x - x_3) \cdots (x - x_n)}{(x_1 - x_0).(x_1 - x_2).(x_1 - x_3) \cdots (x_1 - x_n)} \tag{2.16}$$

The absence of the term $(x - x_1)$ from the numerator is observed.

The general expression for any function $L_i(x)$ is

$$L_i(x) = \frac{(x - x_0) \cdots (x - x_{i-1}).(x - x_{i+1}) \cdots (x - x_n)}{(x_i - x_0) \cdots (x_i - x_{i-1}).(x_i - x_{i+1}) \cdots (x_i - x_n)} \tag{2.17}$$

In equation (2.17) only the term $(x - x_i)$ is absent from the numerator.

$L_i(x)$ is a polynomial of degree $n$ which is equal to "1" for $(x = x_i)$ and which cancels at all the other collocation points.

**Theorem:** Given $(n+1)$ points of collocation $(x_i, f(x_i))$ for $i = 1,2,3 \cdots n$ the only interpolation polynomial of degree $n$ that passes through all the collocation points is written as

$$P_n(x) = L(x) = \sum_{i=0}^{n} f(x_i).L_i(x) \tag{2.18}$$

### 3. Newton interpolation (Newton polynomial)

The general form of a polynomial can be written in different ways, using different expressions, Newton suggested the following form for a polynomial of degree $n$

$$\begin{aligned} P_n(x) &= a_0 \\ &+a_1(x-x_0) \\ &+a_2(x-x_0)(x-x_1) \\ &+a_3(x-x_0)(x-x_1)(x-x_2) \\ &+a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) \\ &\vdots \\ &+a_{n-1}(x-x_0)(x-x_1)(x-x_2)\cdots\cdots(x-x_{n-2}) \\ &+a_n(x-x_0)(x-x_1)(x-x_2)\cdots\cdots(x-x_{n-2})(x-x_{n-1}) \end{aligned} \tag{2.19}$$

- It can be observed that the coefficient of "$a_n$" includes "$n$" monomial of the form $(x-x_i)$ and therefore the polynomial is of degree "$n$".
- In order to construct the polynomial, it is required to determine the $(n+1)$ coefficients "$a_i$", such as $P_n(x)$ passes through the $(n+1)$ collocation points $(x_i, f(x_i))$ for $i=1,2,3\cdots n$.
- It is necessary to ensure that:

$$P(x_i) = f(x_i) \text{ for } i=1,2,3\cdots n. \tag{2.20}$$

- All the coefficients $a_i$ cancels for $(x=x_0)$ except the first coefficient $a_0$, it can be easily shown that:

$$P_n(x_0) = a_0 = f(x_0) \tag{2.21}$$

The first coefficient is therefor

$$a_0 = f(x_0) \tag{2.22}$$

- Second, it is necessary to ensure that:

$$P_n(x_1) = a_0 + a_1(x_1 - x_0) = f(x_0) + a_1(x_1 - x_0) = f(x_1) \tag{2.23}$$

From equation (2.23) the value of $a_1$ can be isolated

$$a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \tag{2.24}$$

**Definition:** The first divided differences of the function $f(x)$ can be denoted as:

$$f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \tag{2.25}$$

Therefore, the coefficient $a_1$ can be written

$$a_1 = f\left[x_0, x_1\right] \tag{2.26}$$

The first order polynomial can be written as

$$P_1(x) = f(x_0) + f\left[x_0, x_1\right](x - x_0) \tag{2.27}$$

The third coefficient $a_2$ is determined using the following procedure

$$P_n(x_2) = a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) = f(x_2) \tag{2.28}$$

By replacing $a_0$ and $a_1$ by their known values in equation (2.28), the following equation can be obtained:

$$P_n(x_2) = f(x_0) + f\left[x_0, x_1\right](x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) = f(x_2) \tag{2.29}$$

The unknown coefficient in equation (2.29) is $a_2$, hence

$$a_2 = \frac{1}{(x_2 - x_0)(x_2 - x_1)}.(f(x_2) - f(x_0) - f[x_0, x_1](x_2 - x_0))$$

$$a_2 = \frac{1}{(x_2 - x_0)}.\frac{1}{(x_2 - x_1)}.(f(x_2) - f(x_0) - f[x_0, x_1](x_2 - x_0))$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(\frac{f(x_2) - f(x_0)}{(x_2 - x_1)} - f[x_0, x_1]\frac{(x_2 - x_0)}{(x_2 - x_1)})$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(\frac{f(x_2) - f(x_1) + f(x_1) - f(x_0)}{(x_2 - x_1)} - f[x_0, x_1]\frac{(x_2 - x_0)}{(x_2 - x_1)})$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(\underbrace{\frac{f(x_2) - f(x_1)}{(x_2 - x_1)}}_{Divided\ Difference} + \frac{f(x_1) - f(x_0)}{(x_2 - x_1)}.\frac{(x_1 - x_0)}{(x_1 - x_0)} - f[x_0, x_1]\frac{(x_2 - x_0)}{(x_2 - x_1)})$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(f[x_1, x_2] + \underbrace{\frac{f(x_1) - f(x_0)}{(x_1 - x_0)}}_{Divided\ Difference}.\frac{(x_1 - x_0)}{(x_2 - x_1)} - f[x_0, x_1]\frac{(x_2 - x_0)}{(x_2 - x_1)}) \tag{2.30}$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(f[x_1, x_2] + f[x_0, x_1]\frac{(x_1 - x_0)}{(x_2 - x_1)} - f[x_0, x_1]\frac{(x_2 - x_0)}{(x_2 - x_1)})$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(f[x_1, x_2] + f[x_0, x_1](\frac{(x_1 - x_0)}{(x_2 - x_1)} - \frac{(x_2 - x_0)}{(x_2 - x_1)}))$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(f[x_1, x_2] + f[x_0, x_1](\frac{x_1 - x_0 - x_2 + x_0}{(x_2 - x_1)}))$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(f[x_1, x_2] + f[x_0, x_1](\underbrace{\frac{-(x_2 - x_1)}{(x_2 - x_1)}}_{-1}))$$

$$a_2 = \frac{1}{(x_2 - x_0)}.(f[x_1, x_2] - f[x_0, x_1])$$

Hence, $a_2$ is represented by an expression containing a divided difference of two divided differences.

**Definition:** The second divided differences of the function $f(x)$ are defined based on the first divided differences by the following expression:

$$f[x_i, x_{i+1}, x_{i+2}] = \frac{f[x_{i+1}, x_{i+2}] - f[x_{i+1}, x_i]}{x_{i+2} - x_i} \tag{2.31}$$

Similarly, the $n^{th}$ divided differences of the function $f(x)$ are defined based on the $(n-1)$ divided differences such as:

$$f[x_0, x_1, x_2, \cdots x_n] = \frac{f[x_1, x_2, \cdots x_n] - f[x_0, x_1, \cdots x_{n-1}]}{x_n - x_0} \tag{2.32}$$

Based on equation (2.31), the coefficient $a_2$ can be written as:

$$a_2 = f\left[x_0, x_1, x_2\right] \tag{2.33}$$

It is sufficient to construct a table, known as the divided differences table to build a polynomial such as:

| $x_i$ | $f(x_i)$ | $f\left[x_i, x_{i+1}\right]$ | $f\left[x_i, x_{i+1}, x_{i+2}\right]$ | $f\left[x_i, x_{i+1}, x_{i+2}, x_{i+3}\right]$ |
|---|---|---|---|---|
| $x_0$ | $f(x_0)$ | $f\left[x_0, x_1\right]$ | $f\left[x_0, x_1, x_2\right]$ | $f\left[x_0, x_1, x_2, x_3\right]$ |
| $x_1$ | $f(x_1)$ | $f\left[x_1, x_2\right]$ | $f\left[x_1, x_2, x_3\right]$ | |
| $x_2$ | $f(x_2)$ | $f\left[x_2, x_3\right]$ | | |
| $x_3$ | $f(x_3)$ | | | |

*Note that:*

$$a_0 = f(x_0),\ a_1 = f\left[x_0, x_1\right],\ a_2 = f\left[x_0, x_1, x_2\right],\ a_3 = f\left[x_0, x_1, x_2, x_3\right]$$

These coefficients need to be replaced in the general form of the polynomial presented in equation (2.19).

**Conclusion**

As it has been demonstrated in this chapter there exist a set of methods allowing the construction of a polynomial of order n passing through (n+1) point of collocation. These methods can be very helpful when having a limited number of points to build a function presenting a relation between two different parameters.

# Chapter III: Approximation Methods and Quadratic Mean

## III-1 Introduction

This chapter introduces approximation in the quadratic mean, orthogonal and pseudo-orthogonal systems, approximation by orthogonal polynomials, and trigonometric approximation for second-year bachelor students in the Science and Technology common branch.

Approximation theory studies how a complicated function can be replaced by a simpler one while keeping the error small according to a chosen criterion. In this chapter, the error criterion is mainly the quadratic mean, which leads naturally to least-squares approximation and to orthogonal expansions.

At the end of this chapter, students should be able to:

- Define the quadratic mean and the associated norm for functions.
- Explain the notions of inner product, orthogonality, and orthonormality.
- Determine the best approximation of a function in a finite-dimensional subspace using orthogonal projection.
- Use orthogonal polynomials to build polynomial approximations.
- Write the Fourier coefficients of a periodic function and interpret trigonometric approximation in the mean-square sense.

## III-2 Approximation and error

In applications, an exact expression of a function may be difficult to use, or a physical quantity may only be known approximately. The idea of approximation is then to replace the original function by a simpler expression such as a polynomial or a trigonometric polynomial.

If a function $f$ is approximated by a function $g$, an error must be measured. Several error criteria exist, but this chapter focuses on the quadratic mean because it works well with inner products and orthogonal systems.

For functions defined on an interval $[a, b]$, a natural error quantity is

$$E(f,g) = \int_a^b |f(x) - g(x)|^2 \, dx.$$

A function $g$ is said to be a best approximation of $f$ in a set $K$ if it minimizes this error among all functions in $K$.

### III-3 Quadratic mean

For real numbers $x_1, x_2, \ldots, x_n$, the quadratic mean, also called the root mean square, is defined by

$$Q(x_1, \ldots, x_n) = \sqrt{\frac{1}{n}\sum_{k=1}^{n} x_k^2}.$$

This quantity is always nonnegative and gives more weight to large values than the arithmetic mean does.

#### *Basic example*

Consider the numbers 1, 2, and 3. Their quadratic mean is

$$Q(1,2,3) = \sqrt{\frac{1^2 + 2^2 + 3^2}{3}} = \sqrt{\frac{14}{3}}.$$

Hence, $Q(1,2,3) \approx 2.16$, while the arithmetic mean is 2. This shows that the quadratic mean is slightly larger.

### III-4 Quadratic mean for functions

For a square-integrable function $f$ on $[a, b]$, the quadratic mean is associated with the norm

$$\|f\|_2 = \left(\int_a^b |f(x)|^2 \, dx\right)^{1/2}.$$

Some authors divide by $b - a$ inside the square root to obtain an averaged quantity, but both definitions represent the same geometric idea.

The distance between two functions $f$ and $g$ is then

$$d(f,g) = \|f-g\|_2 = \left(\int_a^b |f(x)-g(x)|^2\,dx\right)^{1/2}.$$

This distance is called the quadratic mean error or the mean-square error.

### III-5 Inner product and orthogonality

The quadratic mean is closely linked to the inner product

$$\langle f,g\rangle = \int_a^b f(x)g(x)\,dx.$$

With this definition,

$$\|f\|_2^2 = \langle f,f\rangle.$$

This relation is the key to the geometry of approximation in function spaces.

Two functions $f$ and $g$ are orthogonal if

$$\langle f,g\rangle = 0.$$

A family $(\varphi_0,\varphi_1,\dots,\varphi_n)$ is orthogonal if every two distinct functions in the family are orthogonal.

If in addition each function has norm 1, then the family is orthonormal.

### Geometric interpretation

In ordinary Euclidean geometry, two vectors are orthogonal when they form a right angle. In the same way, two functions are orthogonal when their inner product is zero.

This analogy allows many geometric ideas from vectors to be used with functions, especially projection onto a subspace.

### III-6 Orthogonal systems

An orthogonal system is a set of nonzero vectors or functions for which all different elements are pairwise orthogonal. Orthogonal systems are important because expansions with respect to them are easier to compute than expansions in a general basis.

If $(u_1, u_2, \dots, u_n)$ is an orthogonal system in an inner product space, then any vector

$$v = \sum_{k=1}^{n} c_k u_k$$

has coefficients given by

$$c_k = \frac{\langle v, u_k \rangle}{\langle u_k, u_k \rangle}.$$

This formula is fundamental in approximation theory.

**Example in $\mathbb{R}^3$**

Consider the standard basis

$$e_1 = (1,0,0), e_2 = (0,1,0), e_3 = (0,0,1).$$

These vectors form an orthonormal basis of $\mathbb{R}^3$. For a vector $v = (a, b, c)$, one has

$$v = ae_1 + be_2 + ce_3,$$

and the coefficients are exactly the inner products $\langle v, e_k \rangle$.

**Orthogonal functions on an interval**

On the interval $[-\pi, \pi]$, the functions $1$, $\cos(nx)$, and $\sin(nx)$ form an orthogonal system for integer $n \geq 1$ with respect to the inner product

$$\langle f, g \rangle = \int_{-\pi}^{\pi} f(x) g(x)\, dx.$$

This orthogonality is the basis of Fourier series.

**III-7 Pseudo-orthogonal systems**

In standard approximation theory, the inner product is positive, meaning that $\langle f, f \rangle > 0$ for every nonzero $f$. There are more general situations where one uses an indefinite form associated with a signature matrix $J$. In that setting, a matrix $Q$ is called pseudo-orthogonal if

$$Q^T J Q = J.$$

This notion generalizes usual orthogonality.

For second-year students, pseudo-orthogonality should be presented only as an extension of the idea of orthogonality. The main results of this chapter rely on ordinary orthogonal systems in positive-definite inner product spaces.

**Best approximation in a subspace**

Let $K$ be a finite-dimensional subspace of an inner product space. The problem is to find, for a given function $f$, an element $p \in K$ such that

$$\|f - p\|_2$$

is as small as possible.

The key theorem states that $p$ is the best approximation of $f$ in $K$ if and only if the error $f - p$ is orthogonal to every element of $K$, that is,

$$\langle f - p, v \rangle = 0 \text{ for all } v \in K.$$

This theorem gives both existence and a practical method for computing the approximation.

**Proof idea**

Suppose $p \in K$ and take any $v \in K$. Then $v - p \in K$. If $f - p$ is orthogonal to $K$, in particular it is orthogonal to $v - p$. Therefore,

$$\|f - v\|_2^2 = \|(f - p) + (p - v)\|_2^2 = \|f - p\|_2^2 + \|p - v\|_2^2.$$

Hence,

$$\|f - v\|_2^2 \geq \|f - p\|_2^2.$$

So $p$ is the best approximation.

This result is the analog of the Pythagorean theorem in function spaces.

**Approximation by an orthogonal basis**

Let $(\varphi_0, \varphi_1, \dots, \varphi_n)$ be an orthogonal basis of a subspace $K$. Then the best approximation of $f$ in $K$ is

$$p_n(x) = \sum_{k=0}^{n} a_k \varphi_k(x),$$

where

$$a_k = \frac{\langle f, \varphi_k \rangle}{\langle \varphi_k, \varphi_k \rangle}.$$

This formula is the main practical advantage of orthogonal systems. If the basis is orthonormal, then the formula simplifies to

$$a_k = \langle f, \varphi_k \rangle.$$

This simplification explains why orthonormal systems are especially useful.

## III-8 Orthogonal polynomials

A sequence of polynomials $(P_0, P_1, P_2, \dots)$ is called orthogonal on an interval $[a, b]$ with respect to a weight function $w(x) > 0$ if

$$\int_a^b w(x) P_m(x) P_n(x)\, dx = 0 \text{ for } m \neq n.$$

Orthogonal polynomials are used because they provide natural bases for polynomial approximation.

### Legendre polynomials

One important family is the Legendre polynomials, which are orthogonal on $[-1,1]$ with weight $w(x) = 1$.

The first few Legendre polynomials are

$$P_0(x) = 1,$$

$$P_1(x) = x,$$

$$P_2(x) = \frac{1}{2}(3x^2 - 1).$$

These polynomials satisfy

$$\int_{-1}^{1} P_m(x)P_n(x)\,dx = 0 \text{ for } m \neq n.$$

This orthogonality makes them well adapted to approximation on $[-1,1]$.

**Chebyshev polynomials**

Another classical family is the Chebyshev polynomials, orthogonal on $[-1,1]$ with weight $(1 - x^2)^{-1/2}$.

They play an important role in interpolation and in minimizing the maximum error, although this chapter focuses mainly on mean-square approximation.

**Gram-Schmidt process for polynomials**

Starting from the linearly independent family

$$1,$$

one can construct an orthogonal family by the Gram-Schmidt process.

For instance,

$$p_0(x) = 1.$$

Then

$$p_1(x) = x - \frac{\langle x, p_0\rangle}{\langle p_0, p_0\rangle} p_0(x).$$

Similarly,

$$p_2(x) = x^2 - \frac{\langle x^2, p_0\rangle}{\langle p_0, p_0\rangle} p_0(x) - \frac{\langle x^2, p_1\rangle}{\langle p_1, p_1\rangle} p_1(x).$$

By repeating the process, one obtains a sequence of orthogonal polynomials.

**Worked example on** $[-1,1]$

Take the inner product

$$\langle f, g\rangle = \int_{-1}^{1} f(x)g(x)\,dx.$$

Start with $p_0(x) = 1$. Then

$$\langle x, p_0\rangle = \int_{-1}^{1} x\,dx = 0.$$

Hence,

$$p_1(x) = x.$$

Now compute $p_2$. First,

$$\langle x^2, p_0\rangle = \int_{-1}^{1} x^2\,dx = \frac{2}{3}.$$

Also,

$$\langle p_0, p_0\rangle = \int_{-1}^{1} 1\,dx = 2.$$

Thus,

$$\frac{\langle x^2, p_0\rangle}{\langle p_0, p_0\rangle} = \frac{1}{3}.$$

Next,

$$\langle x^2, p_1\rangle = \int_{-1}^{1} x^3\,dx = 0.$$

Therefore,

$$p_2(x) = x^2 - \frac{1}{3}.$$

Up to a constant factor, this is the same as the Legendre polynomial $P_2(x) = \frac{1}{2}(3x^2 - 1)$.

**Approximation by orthogonal polynomials**

Suppose that $f$ is to be approximated on $[a,b]$ by a polynomial of degree at most $n$. If $(P_0, P_1, \dots, P_n)$ is an orthogonal polynomial basis, the best approximation is

$$p_n(x) = \sum_{k=0}^{n} a_k P_k(x),$$

with

$$a_k = \frac{\int_a^b w(x) f(x) P_k(x)\, dx}{\int_a^b w(x) P_k(x)^2\, dx}.$$

This formula comes directly from orthogonal projection.

**Worked example: approximation of $f(x) = x^2$ by degree 1 polynomials**

Approximate $f(x) = x^2$ on $[-1,1]$ by a polynomial of degree at most 1 using the orthogonal basis $P_0(x) = 1$, $P_1(x) = x$.

Write

$$p_1(x) = a_0 P_0(x) + a_1 P_1(x) = a_0 + a_1 x.$$

The coefficients are

$$a_0 = \frac{\int_{-1}^{1} x^2 \cdot 1\, dx}{\int_{-1}^{1} 1\, dx} = \frac{2/3}{2} = \frac{1}{3},$$

and

$$a_1 = \frac{\int_{-1}^{1} x^2 \cdot x\, dx}{\int_{-1}^{1} x^2\, dx} = \frac{0}{2/3} = 0.$$

Hence the best degree 1 approximation is

$$p_1(x) = \frac{1}{3}.$$

This result is reasonable because $x^2$ is even, while $x$ is odd, so the odd component must vanish by symmetry.

**Worked example: approximation of $f(x) = e^x$ by degree 2 polynomials**

To approximate $f(x) = e^x$ on $[-1,1]$, one may use the first Legendre polynomials. The approximation is written as

$$p_2(x) = a_0P_0(x) + a_1P_1(x) + a_2P_2(x),$$

where each coefficient is obtained by projection onto the corresponding orthogonal polynomial.

This example is useful in class because it shows how a non-polynomial function can be approximated efficiently by only a few polynomial terms.

**Discrete least-squares approximation**

In practice, a function is often known only at points $x_1, x_2, \dots, x_N$. If one wants to approximate these data by a polynomial $p$, a common criterion is to minimize

$$E = \sum_{k=1}^{N} (f(x_k) - p(x_k))^2.$$

This is the discrete version of quadratic mean approximation.

**Example with three points**

Suppose the data points are $(-1,1)$, $(0,0)$, and $(1,1)$. To approximate them by a constant $p(x) = c$, minimize

$$E(c) = (1-c)^2 + (0-c)^2 + (1-c)^2.$$

Expanding gives

$$E(c) = 2(1-c)^2 + c^2 = 2 - 4c + 3c^2.$$

Differentiating,

$$E'(c) = -4 + 6c.$$

Hence the minimum occurs at

$$c = \frac{2}{3}.$$

So the best constant discrete least-squares approximation is $p(x) = \frac{2}{3}$.

This section connects approximation theory to data fitting and numerical analysis.

## III-9 Trigonometric approximation

Trigonometric approximation replaces a periodic function by a finite sum of sines and cosines. It is particularly suitable for periodic phenomena such as sound, waves, and vibrations.

On $[-\pi, \pi]$, consider the orthogonal system

$$1,$$

with inner product

$$\langle f, g \rangle = \int_{-\pi}^{\pi} f(x)g(x)\, dx.$$

The orthogonality relations are the basis of Fourier approximation.

### Fourier coefficients

A trigonometric polynomial of degree $N$ has the form

$$S_N(x) = \frac{a_0}{2} + \sum_{n=1}^{N} (a_n \cos(nx) + b_n \sin(nx)).$$

The coefficients that give the best approximation in the quadratic mean are

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(nx)\, dx,$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(nx)\, dx.$$

These coefficients are obtained exactly as in the case of orthogonal polynomials: by projection onto an orthogonal basis.

### Orthogonality relations

For integers $m, n \geq 1$, the following relations hold:

$$\int_{-\pi}^{\pi} \cos(mx)\cos(nx)\,dx = 0 \text{ if } m \neq n,$$

$$\int_{-\pi}^{\pi} \sin(mx)\sin(nx)\,dx = 0 \text{ if } m \neq n,$$

and

$$\int_{-\pi}^{\pi} \sin(mx)\cos(nx)\,dx = 0.$$

These formulas justify the computation of Fourier coefficients term by term.

**Worked example: Fourier approximation of $f(x) = x$**

Consider the function $f(x) = x$ on $[-\pi, \pi]$, extended periodically. Since $f$ is odd, all cosine coefficients vanish and only sine terms remain.

The coefficient $b_n$ is

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} x\sin(nx)\,dx.$$

A standard computation gives

$$b_n = \frac{2(-1)^{n+1}}{n}.$$

Therefore, the Fourier series begins with

$$S_N(x) = 2\sin x - \sin 2x + \frac{2}{3}\sin 3x - \cdots.$$

This example shows that a nontrivial function can be approximated by simple trigonometric functions.

**Comparison of polynomial and trigonometric approximation**

Polynomial approximation and trigonometric approximation rely on the same general principle: orthogonal projection in a space with inner product.

The main difference lies in the choice of basis. On a finite interval, orthogonal polynomials are well adapted to non-periodic problems, while trigonometric functions are especially adapted to periodic functions.

| Aspect | Orthogonal polynomials | Trigonometric approximation |
|---|---|---|
| Typical interval | $[-1,1]$ or general $[a,b]$ | $[-\pi,\pi]$ for periodic functions |
| Basis elements | Polynomials $P_n(x)$ | $\cos(nx), \sin(nx)$ |
| Main use | Approximation on finite intervals [ | Approximation of periodic phenomena |
| Coefficients | Obtained by projection formulas | Fourier coefficients |

## III-10 Conclusion

Approximation in the quadratic mean is based on a simple but powerful idea: the best approximation is obtained by orthogonal projection.

This idea unifies polynomial approximation, approximation by orthogonal polynomials, and trigonometric approximation. It also provides a bridge between algebra, analysis, and many applications in science and engineering.

## Chapter IV Numerical integration

### Overview

This chapter introduces numerical integration for second-year bachelor students in the Science and Technology common branch, with emphasis on the simple trapezoid method, composite trapezoid method, simple Simpson's 1/3 method, and composite Simpson's 1/3 method.

Numerical integration is used when a definite integral cannot be evaluated exactly or when the integrand is known only through measured or tabulated values. The main idea is to replace the given function by a simpler function whose integral is easy to compute, then use that integral as an approximation of the original one.

### Learning objectives

At the end of this chapter, students should be able to:

- Explain why numerical integration is needed in mathematics, physics, and engineering.
- Derive and apply the simple trapezoid rule.
- Apply the composite trapezoid rule on a partition of the interval.
- Derive and use Simpson's 1/3 rule as a quadratic approximation formula.
- Apply the composite Simpson's 1/3 rule and state the condition that the number of subintervals must be even.
- Compare the accuracy of the trapezoid and Simpson methods through worked examples and error terms.

### General introduction

For a continuous function $f$ on an interval $[a, b]$, the definite integral

$$I = \int_a^b f(x)\, dx$$

represents the signed area under the curve. In many situations, this integral is difficult or impossible to compute by elementary antiderivative techniques.

This difficulty appears in at least three common situations:

- The antiderivative of the function is not known in elementary form, for example for functions such as $e^{-x^2}$.
- The function is known only by measurements or a table of values.
- A fast approximate value is needed in computation, simulation, or engineering design.

The oldest and simplest idea is to approximate the region under the curve by elementary geometric shapes. Rectangle rules use constant approximations, the trapezoid rule uses a linear approximation, and Simpson's rule uses a quadratic approximation.

**Partition of the interval**

To apply numerical integration on $[a, b]$, the interval is often divided into $n$ subintervals of equal length If

$$h = \frac{b-a}{n},$$

then the nodes are

$$x_0 = a, x_1 = a + h, x_2 = a + 2h, \dots, x_n = b.$$

The values of the function at these nodes are denoted by

$$y_k = f(x_k), k = 0,1,2, \dots, n.$$

These notations are used throughout the chapter.

**Simple trapezoid method**

**Principle**

The simple trapezoid rule uses only the two endpoints $a$ and $b$. The function $f(x)$ is replaced on $[a, b]$ by the straight line joining the points $(a, f(a))$ and $(b, f(b))$.

The area under this straight line is the area of a trapezoid, so the approximation becomes

$$\int_a^b f(x)\, dx \approx \frac{b-a}{2}[f(a) + f(b)].$$

This is called the simple trapezoid formula.

**Derivation**

The linear interpolating polynomial through $(a, f(a))$ and $(b, f(b))$ is used in place of $f$.Integrating that polynomial over $[a, b]$ gives the trapezoid formula directly.

Another geometric derivation is possible: the bases of the trapezoid are $f(a)$ and $f(b)$, and its height is $b - a$. Therefore, its area is

$$A = \frac{\text{sum of parallel sides}}{2} \times \text{height} = \frac{f(a) + f(b)}{2}(b - a).$$

Hence,

$$T = \frac{b - a}{2}[f(a) + f(b)].$$

**Error term**

If $f$ has a continuous second derivative on $[a, b]$, then the error in the simple trapezoid rule is of the form

$$E_T = -\frac{(b - a)^3}{12} f''(\xi), a < \xi < b.$$

This shows that the error depends on the curvature of the function. When the graph is nearly linear on the interval, the trapezoid rule is usually accurate.

**Worked example 1**

Approximate

$$I = \int_0^1 x^2 \, dx$$

by the simple trapezoid rule.

Here,

$$a = 0, b = 1, f(x) = x^2,$$

so

$$f(0) = 0, f(1) = 1.$$

Then

$$T = \frac{1-0}{2}[0+1] = \frac{1}{2}.$$

The exact value is

$$I = \int_0^1 x^2\, dx = \frac{1}{3}.$$

Thus the absolute error is

$$|I - T| = \left|\frac{1}{3} - \frac{1}{2}\right| = \frac{1}{6}.$$

This example shows that the trapezoid rule is only a rough approximation when the function is curved.

**Worked example 2**

Approximate

$$I = \int_1^2 \ln\, x\, dx$$

by the simple trapezoid rule.

Since

$$f(1) = \ln\, 1 = 0, f(2) = \ln\, 2,$$

we obtain

$$T = \frac{2-1}{2}(0 + \ln\, 2) = \frac{\ln\, 2}{2}.$$

Using $\ln\, 2 \approx 0.6931$,

$$T \approx 0.3466.$$

The exact value is

$$I = [x\ln\ x - x]_1^2 = 2\ln\ 2 - 1 \approx 0.3862.$$

So the trapezoid approximation is close, but still underestimates the true integral.

**Composite trapezoid method**

**Principle**

The simple trapezoid rule can be improved by dividing the interval into several equal subintervals and applying the trapezoid rule on each one. The results are then added to approximate the whole integral.

Let $[a, b]$ be divided into $n$ equal parts of length $h = (b - a)/n$. The composite trapezoid rule is

$$\int_a^b f(x)\,dx \approx \frac{h}{2}\left[y_0 + 2\sum_{k=1}^{n-1} y_k + y_n\right].$$

Equivalently,

$$T_n = \frac{h}{2}[y_0 + 2(y_1 + y_2 + \cdots + y_{n-1}) + y_n].$$

This formula is one of the most important numerical integration formulas.

**Why it is better**

When the interval is subdivided, each small part of the graph is approximated by a short straight segment. If the function is smooth, this piecewise linear approximation is much better than one single secant line over the whole interval.

**Error term**

For a sufficiently smooth function, the global error of the composite trapezoid rule satisfies

$$E_{T_n} = -\frac{(b-a)}{12}h^2 f''(\xi), a < \xi < b.$$

So the error is of order $h^2$. This means that reducing the step size improves the approximation significantly.

**Worked example 3**

Approximate

$$I = \int_0^1 x^2\, dx$$

by the composite trapezoid rule with $n = 4$.

Here,

$$h = \frac{1-0}{4} = \frac{1}{4}.$$

The nodes are

$$x_0 = 0,$$

The function values are

$$y_0 = 0,$$

Then

$$T_4 = \frac{1/4}{2}[0 + 2(0.0625 + 0.25 + 0.5625) + 1].$$

Since

$$0.0625 + 0.25 + 0.5625 = 0.875,$$

we get

$$T_4 = \frac{1}{8}(1 + 1.75) = \frac{1}{8}(2.75) = 0.34375.$$

The exact value is $1/3 \approx 0.33333$, so the error is much smaller than for the simple trapezoid rule.

**Worked example 4**

Approximate

$$I = \int_0^2 (x^3 + x - 1)\, dx$$

by the composite trapezoid rule with $n = 2$.

We have

$$h = \frac{2-0}{2} = 1,$$

and the nodes $x_0 = 0$, $x_1 = 1$, $x_2 = 2$.

The function values are

$$y_0 = -1, y_1 = 1, y_2 = 9.$$

Thus,

$$T_2 = \frac{1}{2}[-1 + 2(1) + 9] = \frac{10}{2} = 5.$$

The exact value is

$$I = \left[\frac{x^4}{4} + \frac{x^2}{2} - x\right]_0^2 = 4 + 2 - 2 = 4.$$

Hence the approximation error is 1.

**Simple Simpson's 1/3 method**

**Principle**

Simpson's 1/3 rule improves the trapezoid idea by replacing the function with a quadratic polynomial instead of a straight line. The rule uses three equally spaced points, usually $x_0 = a$, $x_1 = a + h$, and $x_2 = b = a + 2h$.

The approximation formula is

$$\int_a^b f(x)\, dx \approx \frac{b-a}{6}[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)].$$

If one writes the nodes as $x_0, x_1, x_2$ with $h = (b-a)/2$, the formula becomes

$$S = \frac{h}{3}(y_0 + 4y_1 + y_2).$$

This is Simpson's 1/3 rule.

**Derivation idea**

The function $f$ is approximated by the unique quadratic interpolating polynomial through the three points $(x_0, y_0)$, $(x_1, y_1)$, and $(x_2, y_2)$. Integrating that polynomial over $[a, b]$ gives Simpson's formula.

This rule is exact for every polynomial of degree at most 3, which explains why it is often much more accurate than the trapezoid rule.

**Error term**

If $f$ has a continuous fourth derivative on $[a, b]$, the error of the simple Simpson's 1/3 rule is

$$E_S = -\frac{(b-a)^5}{2880} f^{(4)}(\xi), a < \xi < b.$$

This error is generally much smaller than the trapezoid error when the function is smooth.

**Worked example 5**

Approximate

$$I = \int_0^1 x^2 \, dx$$

using Simpson's 1/3 rule.

Here,

$$a = 0, b = 1, h = \frac{1}{2}.$$

The midpoint is $x_1 = 0.5$, and

$$y_0 = f(0) = 0, y_1 = f(0.5) = 0.25, y_2 = f(1) = 1.$$

Then

$$S = \frac{h}{3}(y_0 + 4y_1 + y_2) = \frac{1/2}{3}(0 + 4 \times 0.25 + 1).$$

Therefore,

$$S = \frac{1}{6}(2) = \frac{1}{3}.$$

This is exactly the true value. This is consistent with the fact that Simpson's rule is exact for quadratic functions.

**Worked example 6**

Approximate

$$I = \int_0^2 (x^3 + x - 1)\, dx$$

using Simpson's 1/3 rule.

We have $a = 0$, $b = 2$, and $h = 1$. Then

$$y_0 = f(0) = -1, y_1 = f(1) = 1, y_2 = f(2) = 9.$$

So

$$S = \frac{1}{3}(-1 + 4 \cdot 1 + 9) = \frac{12}{3} = 4.$$

This matches the exact integral exactly. Again, this is expected because the integrand is a polynomial of degree 3.

**Composite Simpson's 1/3 method**

**Principle**

When the interval is large or the function is not well approximated by one quadratic polynomial, the interval is divided into an even number $n$ of equal subintervals. Simpson's 1/3 rule is then applied pairwise on adjacent subintervals.

The requirement that $n$ must be even is essential because each Simpson panel uses two subintervals.

If $h = (b-a)/n$ and $n$ is even, then the composite Simpson's 1/3 rule is

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[y_0 + y_n + 4\sum_{\text{odd } k} y_k + 2\sum_{\text{even } k,} y_k\right].$$

In expanded form,

$$S_n = \frac{h}{3}[y_0 + y_n + 4(y_1 + y_3 + \cdots + y_{n-1}) + 2(y_2 + y_4 + \cdots + y_{n-2})].$$

This is the composite Simpson's 1/3 formula.

**Error term**

For a smooth function, the global error satisfies

$$E_{S_n} = -\frac{(b-a)}{180}h^4 f^{(4)}(\xi), a < \xi < b.$$

Thus the method is of order $h^4$, which is more accurate than the composite trapezoid rule for the same step size when the function is sufficiently regular.

**Worked example 7**

Approximate

$$I = \int_0^1 x^4\,dx$$

using the composite Simpson's 1/3 rule with $n = 4$.

Here,

$$h = \frac{1}{4}.$$

The nodes are

$$0,$$

The corresponding values are

$$y_0 = 0,$$

$$y_1 = (0.25)^4 = 0.00390625,$$

$$y_2 = (0.5)^4 = 0.0625,$$

$$y_3 = (0.75)^4 = 0.31640625,$$

$$y_4 = 1.$$

Then

$$S_4 = \frac{1/4}{3}[y_0 + y_4 + 4(y_1 + y_3) + 2y_2].$$

So

$$S_4 = \frac{1}{12}[1 + 4(0.00390625 + 0.31640625) + 0.125].$$

Since

$$0.00390625 + 0.31640625 = 0.3203125,$$

we obtain

$$S_4 = \frac{1}{12}[1 + 1.28125 + 0.125] = \frac{2.40625}{12} = 0.2005208333.$$

The exact value is

$$I = \int_0^1 x^4\, dx = \frac{1}{5} = 0.2.$$

So the error is approximately $0.0005208333$, which is very small.

**Worked example 8**

Approximate

$$I = \int_0^2 e^x\, dx$$

with the composite Simpson's 1/3 rule using $n = 4$.

Here,

$$h = \frac{2}{4} = 0.5,$$

and the nodes are

$$0,$$

The corresponding values are approximately

$$y_0 = 1,$$

$$y_1 = e^{0.5} \approx 1.6487,$$

$$y_2 = e \approx 2.7183,$$

$$y_3 = e^{1.5} \approx 4.4817,$$

$$y_4 = e^2 \approx 7.3891.$$

Then

$$S_4 = \frac{0.5}{3}[y_0 + y_4 + 4(y_1 + y_3) + 2y_2].$$

Thus,

$$S_4 = \frac{1}{6}[1 + 7.3891 + 4(1.6487 + 4.4817) + 2(2.7183)].$$

Now

$$1.6487 + 4.4817 = 6.1304,$$

so

$$S_4 = \frac{1}{6}(8.3891 + 24.5216 + 5.4366) = \frac{38.3473}{6} \approx 6.3912.$$

The exact value is

$$I = e^2 - 1 \approx 6.3891.$$

The approximation is therefore very good.

**Comparison of methods**

The four methods studied in this chapter can be compared by their basic idea, their formulas, and their accuracy.

| Method | Approximation idea | Formula | Main condition | Typical error |
|---|---|---|---|---|
| Simple trapezoid | Linear interpolation on one interval | $\frac{b-a}{2}[f(a)+f(b)]$ | None beyond continuity | Depends on $f''$, order $(b-a)^3$ |
| Composite trapezoid | Piecewise linear interpolation | $\frac{h}{2}[y_0+2\sum y_k+y_n]$ | Equal subintervals | Global order $h^2$ |
| Simple Simpson 1/3 | Quadratic interpolation on two subintervals | $\frac{h}{3}(y_0+4y_1+y_2)$ | Three equally spaced points | Depends on $f^{(4)}$ |
| Composite Simpson 1/3 | Piecewise quadratic interpolation | $\frac{h}{3}[y_0+y_n+4\sum y_{odd}+2\sum y_{even}]$ | $n$ even | Global order $h^4$ |

In general, Simpson's 1/3 methods are more accurate than trapezoid methods for smooth functions because quadratic approximations follow the shape of the curve better than straight segments do. However, the trapezoid method is simpler and can still be useful when only endpoint or regularly spaced data are available.

**Practical remarks**

When applying a numerical integration formula, students should always begin by identifying the interval, the number of subintervals, the step size, and the needed function values. A clear table of nodes and function values often prevents arithmetic mistakes.

For composite Simpson's 1/3 rule, the number of subintervals must be even. If this condition is not satisfied, the formula cannot be applied directly in its standard form.

Numerical integration formulas may overestimate or underestimate the true integral depending on the concavity of the curve and the behavior of higher derivatives. This point can be illustrated graphically in class.

**Exercises**

1. Use the simple trapezoid rule to approximate $\int_0^2 x^2\,dx$.
2. Use the composite trapezoid rule with $n = 4$ to approximate $\int_0^2 x^2\,dx$.
3. Apply Simpson's 1/3 rule to $\int_0^2 x^2\,dx$.
4. Apply the composite Simpson's 1/3 rule with $n = 4$ to $\int_0^2 x^4\,dx$.
5. Compare the simple trapezoid rule and Simpson's 1/3 rule for $\int_0^1 e^x\,dx$.
6. State the condition required to apply the composite Simpson's 1/3 rule and explain why it is necessary.
7. For $f(x) = \sin x$, approximate $\int_0^\pi \sin x\,dx$ using the composite trapezoid rule with $n = 4$.
8. For $f(x) = \sin x$, approximate $\int_0^\pi \sin x\,dx$ using the composite Simpson's 1/3 rule with $n = 4$.

**Guided solutions**

**Solution 1**

For $f(x) = x^2$ on [0,2],

$$T = \frac{2-0}{2}[f(0) + f(2)] = 1(0+4) = 4.$$

**Solution 2**

Here $h = 0.5$, and the values are

$$y_0 = 0,$$

Then

$$T_4 = \frac{0.5}{2}[0 + 2(0.25 + 1 + 2.25) + 4] = 0.25(11) = 2.75.$$

The exact value is $8/3 \approx 2.6667$, so this approximation is much better than the simple trapezoid value.

**Solution 3**

With $h = 1$,

$$S = \frac{1}{3}[0 + 4(1) + 4] = \frac{8}{3}.$$

So Simpson's 1/3 rule gives the exact answer for this quadratic function.

**Solution 4**

Students should obtain the node values, substitute them in the composite Simpson formula, and compare the result with the exact value $32/5$.

**Teaching notes**

A good pedagogical sequence is to begin with the geometric idea of area, then move to the trapezoid rule, and finally present Simpson's rule as a more accurate improvement based on quadratic interpolation.

It is helpful to solve at least one polynomial example and one transcendental-function example in class. Polynomial examples show when a method can be exact, while transcendental examples illustrate approximation error.

Students should also be trained to organize calculations in a table with columns for $x_k$, $f(x_k)$, and the appropriate weights. This habit becomes essential when composite formulas are used with many subintervals.

**Conclusion**

Numerical integration provides practical ways to approximate definite integrals when exact integration is difficult or impossible.

The trapezoid rule is based on linear approximation, while Simpson's 1/3 rule is based on quadratic approximation. Their composite versions improve accuracy by subdividing the

interval, and composite Simpson's 1/3 rule is generally the most accurate of the four methods presented here for smooth functions.

# Chapter V: Ordinary differential equations

## 1. Introduction

The general form of a first order differential equation can be defined as follow

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases} \tag{4.1}$$

- $t$ : is an independent variable that represent time is most of the differential equations.
- $y$ : is a dependent variable, it depends on $t$.
- $f$ : is a given function having two variables.
- $y(t_0) = y_0$ : is the initial condition (I.C) it is the state of the solution in given time denoted ($t_0$). Prior to solving the differential equation, the initial condition is the only point where the solution is known.

Example:

Assume the first order differential equation given by:

$$\begin{cases} y'(t) = t \\ y(0) = 1 \end{cases} \tag{4.2}$$

The solution consists of finding the original function $y(t)$ which first derivative give $y'(t) = t$

The classical approach involves the usage of theoretical integration such as:

$$\int y'(t).dt = \int t.dt$$

$$y(t) = \frac{t^2}{2} + c$$

$y(t) = \frac{t^2}{2} + c$ is the general solution that satisfies $y'(t) = t$ for any value of $(c)$.

Using the initial condition given in Equation (4.2) the integration constant $(c)$ is calculated. Hence:

$$y(0) = 1 = c$$

The solution related to the initial conditions given in Equation (4.2) is then:

$$y(t) = \frac{t^2}{2} + 1$$

## 2. Euler's method

- It is the simplest method.
- It has a good graphical interpretation.
- It has a low precision.

Using the general form of a first order differential equation

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases} \tag{4.3}$$

The main objective of this method is to obtain an approximation of the solution at multiple instants $t$ using an iterative method, such as $t = t_1 = t_0 + h$.

- Before proceeding to the first iteration, it is important to determine on which direction the method should move forward stating from the point $(t_0, y_0)$ in order to obtain the point $(t_1, y_1)$, which is an approximation of the point $(t_1, y(t_1))$.
- It is important to highlight that the equation of the curve $y(t)$ solution of the differential equation is **unknown**. Yet, **the slope** $y'(t)$ in $t = t_0$ is **known**. Thus, the differential equation ensures that:

$$y'(t_0) = f(t_0, y(t_0)) = f(t_0, y_0) \ \left[\text{From the initial conditions}\right] \tag{4.4}$$

- The line passing by the point $(t_0, y_0)$ and having a slope $f(t_0, y_0)$ can be tracked, the equation of this later is noted $d_0(t)$ and given by the following equation:

$$d_0(t) = f(t_0, y_0)(t - t_0) + y_0 \text{ (see the figure 4.1)} \tag{4.5}$$

In $t = t_1$

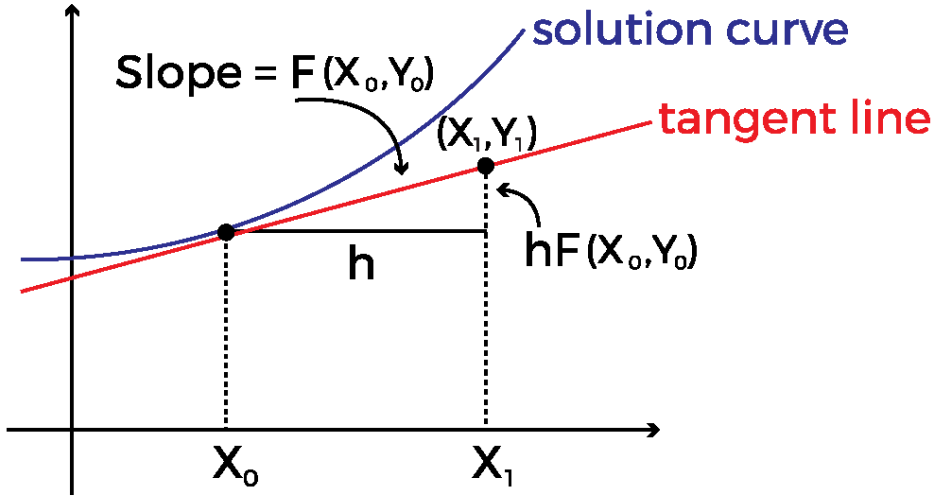



Figure 4.1

## VI. Direction resolution methods for linear equations systems

### 6.1 Introduction to Linear systems

Given the following set of equations:

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \cdots a_{1n}x_n = b_1 \\ &a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \cdots a_{2n}x_n = b_2 \\ &a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \cdots a_{3n}x_n = b_3 \\ &\vdots \qquad\quad \vdots \qquad\qquad\qquad \vdots \\ &a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 \cdots a_{nn}x_n = b_n \end{aligned} \tag{5.1}$$

The set of equations in (3.1) can be rewritten under the following form

$$[A]\{x\} = \{b\} \tag{5.2}$$

Where $[A]$ is the coefficient matrix:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \tag{5.3}$$

The solution of Equation (3.2) can be written as:

$$\{\vec{x}\} = A^{-1}.\{\vec{b}\} \tag{5.4}$$

Knowing that the calculation of the inverse of the matrix $[A]$ can be complicated especially for large dimension matrices.

Example:

$$\begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 8 \\ 11 \end{Bmatrix} \Rightarrow \begin{cases} 2x_1 + 3x_2 = 8 \\ 3x_1 + 4x_2 = 11 \end{cases} \tag{5.5}$$

As it can be seen in Equation (5.5) we can deduce $x_2$ in function of $x_1$ however it is complicated to code in a program**.**

Example:

There exist a few systems that are easy to solve even if they are of large dimensions.

❖ The simpliest case is the diagonal matrix*

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 2 \\ 2 \\ 9 \end{Bmatrix} \Rightarrow \vec{x} = \begin{Bmatrix} 2 \\ 1 \\ 3 \end{Bmatrix} \Rightarrow x_i = \frac{b_i}{a_{ii}} \tag{5.6}$$

*This case can rarely be found in applications

❖ The second simple system is the triangular system which can be lower or upper triangular.

✓ For lower triangular systems all the $a_{ij}$ are null for $i \prec j$

$$\begin{bmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ a_{21} & a_{22} & 0 & \cdots & 0 \\ a_{31} & a_{31} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \tag{5.7}$$

A triangular upper system is the transpose of a triangular lower system.

**« Triangular systems are easy to solve »**

We start from the tip of the triangle then we solve the equations one by one.

Example:

$$\begin{bmatrix} 3 & 0 & 0 \\ 1 & 2 & 0 \\ 3 & 2 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 9 \\ 7 \\ 14 \end{Bmatrix} \tag{5.8}$$

- The first equation:

$$x_1 = \frac{b_1}{a_{11}} = \frac{9}{3} = \boxed{3} \tag{5.9}$$

- The second equation ( $x_1$ is known therefore we can determine $x_2$ )

$$x_2 = \frac{b_2 - a_{21}x_1}{a_{22}} = \frac{7-(1)(3)}{2} = \boxed{2} \tag{5.10}$$

- The third equation ( $x_1$ and $x_2$ are known we can determine $x_3$ )

$$x_3 = \frac{b_3 - a_{31}x_1 - a_{32}x_2}{a_{33}} = \frac{14-(3)(3)-(2)(2)}{1} = \boxed{1} \tag{5.11}$$

**It is important to notice that the diagonal terms for this method should not be null**

- We must transform any system to a triangular one to facilitate its resolution.
- Gauss elimination is a particular case of LU decomposition.

How to proceed to transform any system to a triangular one?

We multiply our system $A\vec{x} = \vec{b}$ by $W \Rightarrow WA\vec{x} = W\vec{b} \Rightarrow$ We can re-multiply by $W^{-1}$

*The matrix $W$ should be reversible in order to calculate $W^{-1}$

**6.2 Elementary operations on matrices lines**

Three (3) operations corresponding to three (3) types of $W$ matrix

1. Replace a line $l_i$ by its multiple $\vec{l}_i \leftarrow \lambda\vec{l}_i$
2. Permutate line $i$ and line $j$ $\vec{l}_i \leftrightarrow \vec{l}_j$
3. Replace a line $l_i$ by a line $l_i$ plus a multiple of a line $j$, $\vec{l}_i \leftarrow \vec{l}_i + \lambda\vec{l}_j$

Examples:

- Multiple of a line (We want to multiply the second line of the matrix A by 3)

$$\underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 1 \\ 5 & 4 & 1 \end{bmatrix}}_{[A]} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \underbrace{\begin{Bmatrix} 6 \\ 11 \\ 10 \end{Bmatrix}}_{\{b\}} \tag{5.12}$$

We multiply the matrix $[A]$ by the matrix $[M]$ (it is important to multiply also the $\{b\}$ vector by the same matrix to keep the equilibrium of equation (5.12).

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{[M]} \begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 1 \\ 5 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 1 & 2 \\ 18 & 12 & 3 \\ 5 & 4 & 1 \end{bmatrix} \tag{5.13}$$

After the multiplication of the vector $\{b\}$ by the matrix $[M]$ we obtain the following equation:

$$\begin{bmatrix} 3 & 1 & 2 \\ 18 & 12 & 3 \\ 5 & 4 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 6 \\ \boxed{33} \\ 10 \end{Bmatrix} \tag{5.14}$$

It is important to note that equations (5.12) and (5.14) have the same solution.

The inverse of $[M]$ is $[M]^{-1}$ where $M(\vec{l}_i \leftarrow (1/\lambda)\vec{l}_i)$

- Line permutation (We want to permute line 2 and 3)

$$\underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 1 \\ 5 & 4 & 1 \end{bmatrix}}_{[A]} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \underbrace{\begin{Bmatrix} 6 \\ 11 \\ 10 \end{Bmatrix}}_{\{b\}} \tag{5.15}$$

We multiply the matrix $[A]$ by the matrix $[P]$ (It is important to also multiply the vector $\{b\}$ by the same matrix to ensure the equilibrium of equation (5.15).

The $[P]$ matrix basically a diagonal unity matrix, we change the ones (1) in the diagonal to allow the permutation. For example in our case, the A(2,2) = 1 becomes A(2,3) = 1, thus the second line of the matrix is permutated with the third line and the A(3,3) = 1 becomes A(3,2) = 1 thus the third line is permutated with the second line.

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{[P]} \underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 1 \\ 5 & 4 & 1 \end{bmatrix}}_{[A]} = \begin{bmatrix} 3 & 1 & 2 \\ 5 & 4 & 1 \\ 6 & 4 & 1 \end{bmatrix} \tag{5.16}$$

After the multiplication of the vector $\{b\}$ by the matrix $[P]$ we obtain the following equation:

$$\begin{bmatrix} 3 & 1 & 2 \\ 5 & 4 & 1 \\ 6 & 4 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 6 \\ 10 \\ 11 \end{Bmatrix} \tag{5.17}$$

It is important to note that equations (5.15) and (5.17) have the same solution.

The inverse of the matrix $[P]$ is $[P]$ it-self.

- Replace a line by itself plus a multiple of another line

$$\begin{matrix} -2 \leftarrow \\ \rightarrow \end{matrix} \underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 1 \\ 5 & 4 & 1 \end{bmatrix}}_{[A]} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \underbrace{\begin{Bmatrix} 6 \\ 11 \\ 10 \end{Bmatrix}}_{\{b\}} \tag{5.18}$$

We want to replace the 2nd line of the [A] matrix by itself minus twice the 1st line of the [A] matrix. For this purpose, we multiply by the matrix $[T]$, the matrix$[T]$ is basically a diagonal unity matrix, We cross the column of the pivot and the line we want to replace to obtain the location of the factor (coefficient) $(\lambda)$ that we have to introduce in the $[T]$ matrix.

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ \boxed{-2} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{[T]} \begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 1 \\ 5 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 1 & 2 \\ \boxed{0} & 2 & -3 \\ 5 & 4 & 1 \end{bmatrix} \tag{5.19}$$

After multiplying the vector $\{b\}$ by the matrix $[T]$ we obtain the following equation:

$$\begin{bmatrix} 3 & 1 & 2 \\ \boxed{0} & 2 & -3 \\ 5 & 4 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 6 \\ -1 \\ 10 \end{Bmatrix} \tag{5.20}$$

It is important to note that equations (5.18) and (5.20) have the same solution.

To find the inverse of $[T]$ we simply remplace $(\lambda)$ by $(-\lambda)$

In equation (5.20) we can notice that we have introduced a zero (0) into the matrix [A], by replacing the third line with itself minus (3/5) of the 1st line we would introduce a zero (0) to the position A(3,1), and so on we can transform any system into a triangular system. This is the basis on which the Gaussian elimination method is based.

### 6.3 Gauss Elimination

#### *6.3.1 The augmented matrix*

Since the operation are going to be made on the $[A]$ matrix and the vector $\{\vec{b}\}$ we write

$$\left[\begin{array}{ccccc|c} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} & b_2 \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} & b_3 \\ \vdots & & & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} & b_n \end{array}\right] \tag{5.21}$$

Gauss used the operations on the matrix lines to introduce zeros bellow the diagonal

Example

We have the following system:

$$\begin{bmatrix} 2 & 1 & 2 \\ 6 & 4 & 0 \\ 8 & 5 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 10 \\ 26 \\ 35 \end{Bmatrix} \Rightarrow \begin{cases} 2x_1 + 1x_2 + 2x_3 = 10 \\ 6x_1 + 4x_2 + 0x_3 = 26 \\ 8x_1 + 5x_2 + 1x_3 = 35 \end{cases} \tag{5.22}$$

For this system the augmented matrix can be written as follow:

$$\left[\begin{array}{ccc|c} 2 & 1 & 2 & 10 \\ 6 & 4 & 0 & 26 \\ 8 & 5 & 1 & 35 \end{array}\right] \tag{5.23}$$

We want to introduce zeros under the diagonal; we will have to eliminate the 6 and 8 located in A(2,1) and A(3,1) respectively. We will rely on the number 2 located in A(1,1) which will serve as a pivot.

$$\begin{array}{l} \leftarrow \\ -8/2 \rightarrow \\ \rightarrow \end{array} \begin{array}{c} \leftarrow \\ -6/2 \rightarrow \\ \end{array} \left[\begin{array}{ccc|c} \boxed{2} & 1 & 2 & 10 \\ \boxed{6} & 4 & 0 & 26 \\ \boxed{8} & 5 & 1 & 35 \end{array}\right] \tag{5.24}$$

To eliminate the 6 we multiply the augmented matrix by the matrix $T_1$ as follow:

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{[T_1]} * \underbrace{\left[\begin{array}{ccc|c} 2 & 1 & 2 & 10 \\ 6 & 4 & 0 & 26 \\ 8 & 5 & 1 & 35 \end{array}\right]}_{[A]} = \underbrace{\left[\begin{array}{ccc|c} 2 & 1 & 2 & 10 \\ 0 & 1 & -6 & -4 \\ 8 & 5 & 1 & 35 \end{array}\right]}_{[T_1]\times[A]} \tag{5.25}$$

To eliminate the 8 we multiply the new augmented matrix shown in equation (5.25) by the matrix $T_2$ as follow:

$$\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\-4&0&1\end{bmatrix}}_{[T_2]}*\underbrace{\left[\begin{array}{ccc|c}2&1&2&10\\0&1&-6&-4\\8&5&1&35\end{array}\right]}_{[T_1]\times[A]}=\underbrace{\left[\begin{array}{ccc|c}2&1&2&10\\0&1&-6&-4\\0&1&-7&-5\end{array}\right]}_{[T_2]\times[T_1]\times[A]} \tag{5.26}$$

The obtained matrix in (5.26) is not triangular, it is necessary to eliminate the 1 located in A(3,2) [3rd line, 2nd column] for this we will use the 1 located in A(2,2) [2nd line, 2nd column] as a pivot as it can be seen in equation (5.27).

$$\begin{matrix}\\-1/1\leftarrow\\1\rightarrow\end{matrix}\left[\begin{array}{ccc|c}2&1&2&10\\0&\boxed{1}&-6&-4\\0&\boxed{1}&-7&-5\end{array}\right] \tag{5.27}$$

For this we multiply the equation in (5.27) by the matrix $T_3$ as follow:

$$\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\0&-1&1\end{bmatrix}}_{[T_3]}*\underbrace{\left[\begin{array}{ccc|c}2&1&2&10\\0&1&-6&-4\\0&1&-7&-5\end{array}\right]}_{[T_2]\times[T_1]\times[A]}=\underbrace{\left[\begin{array}{ccc|c}2&1&2&10\\0&1&-6&-4\\0&0&-1&-1\end{array}\right]}_{[T_3]\times[T_2]\times[T_1]\times[A]} \tag{5.28}$$

We therefore obtain a upper triangular matrix, to solve this system we only have to do **a triangular ascent**.

$$\begin{aligned}&\text{We find } x_3=-1/-1=\boxed{1}\\&\text{Then } x_2=\frac{-4-(-6)(1)}{1}=\boxed{2}\\&\text{Finally } x_1=\frac{10-(1)(2)-(2)(1)}{2}=\boxed{3}\end{aligned} \tag{3.29}$$

We can resume the operation accomplished on the system in the equation (5.23) by the following writing:

$$\begin{aligned}&T_1[\vec{l}_2\leftarrow\vec{l}_2-(6/2)\vec{l}_1]\\&T_2[\vec{l}_3\leftarrow\vec{l}_3-(8/2)\vec{l}_1]\\&T_3[\vec{l}_3\leftarrow\vec{l}_3-(1/1)\vec{l}_2]\end{aligned} \tag{5.30}$$

The upper triangular matrix obtained in equation (5.28) is denoted $[U]$ from the word « Upper » in English. Thus, we can write

$$[U]=[T_3]\times[T_2]\times[T_1]\times[A] \tag{5.31}$$

An explicit writing of the equation (5.31) will result in the following form:

$$\begin{bmatrix}2&1&2\\0&1&-6\\0&0&-1\end{bmatrix}=\begin{bmatrix}1&0&0\\0&1&0\\0&-1&1\end{bmatrix}\times\begin{bmatrix}1&0&0\\0&1&0\\-4&0&1\end{bmatrix}\times\begin{bmatrix}1&0&0\\-3&1&0\\0&0&1\end{bmatrix}\times\begin{bmatrix}2&1&2\\6&4&8\\8&5&1\end{bmatrix} \tag{5.32}$$

Now, we want to do the inverse operation, which means we want to calculate $[A]$ in function of $[U]$ We only have to divide the equation (5.31) by the matrices $[T_3]$,$[T_2]$ and $[T_1]$ thus we obtain:

$$[A]=[T_1^{-1}]\times[T_2^{-1}]\times[T_3^{-1}]\times[U] \tag{5.33}$$

We know the inverse of the matrix $[T_i]$ (see page 5). It is sufficient to multiply the sign of the coefficient $\lambda$ by a negative sign (-). Thus, we obtain the following writing:

$$\underbrace{\begin{bmatrix}2&1&2\\6&4&0\\8&5&1\end{bmatrix}}_{[A]}=\underbrace{\begin{bmatrix}1&0&0\\3&1&0\\0&0&1\end{bmatrix}}_{[T_1^{-1}]}\times\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\4&0&1\end{bmatrix}}_{[T_2^{-1}]}\times\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\0&1&1\end{bmatrix}}_{[T_3^{-1}]}\times\underbrace{\begin{bmatrix}2&1&2\\0&1&-6\\0&0&-1\end{bmatrix}}_{[U]} \tag{5.34}$$

Or also,

$$\underbrace{\begin{bmatrix}2&1&2\\6&4&0\\8&5&1\end{bmatrix}}_{[A]}=\underbrace{\begin{bmatrix}1&0&0\\3&1&0\\4&1&1\end{bmatrix}}_{\substack{\text{Lower}\\\text{inférieure}}}\times\underbrace{\begin{bmatrix}2&1&2\\0&1&-6\\0&0&1\end{bmatrix}}_{\substack{\text{Upper}\\\text{supérieur}}} \tag{5.35}$$

Thus the matrix $[A]$ can be written in the form on a multiplication of a lower triangular matrix denoted « L » and triangular upper matrix denoted « U » it is the **« LU » decomposition.**

- **This is possible only and only if no line permutation were performed**.

**6.4 Décomposition « LU »**

We were able to write the matrix $[A]$ in the form of a product of two matrices $[A]=[L][U]$

How will this facilitate solving the equation $[A]\{\vec{x}\}=\{\vec{b}\}$ ?

We write:

$$[A]\{\vec{x}\}=[L][U]\{\vec{x}\}=\{\vec{b}\} \tag{5.36}$$

We pose

$$[U]\{\vec{x}\}=\{\vec{y}\} \tag{5.37}$$

The resolution of the linear system is then performed in two steps:

$$\begin{cases}[L]\{\vec{y}\}=\{\vec{b}\}\rightarrow 1\\ [U]\{\vec{x}\}=\{\vec{y}\}\rightarrow 2\end{cases} \tag{5.38}$$

We thus have two triangular systems, we start by solving the first system to obtain the vector $\{\vec{y}\}$ then using the vector $\{\vec{y}\}$ we solve the second system to obtain the solution we were looking for which is the vector $\{\vec{x}\}$

Important remark:

- A matrix $[A]$can be written as the product of two triangular matrices in a multitude of ways.

The “LU” decomposition is not unique, arbitrary choices must be made, one of the choices consists of putting ones (1) on the upper matrix diagonal “U”, this is the “Crout” decomposition.

**6.5 Crout decomposition**

- ✓ We have to determine the coefficients $l_{ij}$ and $u_{ij}$ for the equation (5.39) and this by imposing the diagonal of the « U » matrix contain only ones (1).
- ✓ We take a general case of a (4* 4) matrix

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ l_{31} & l_{32} & l_{33} & 0 \\ l_{41} & l_{42} & l_{43} & l_{44} \end{bmatrix} * \begin{bmatrix} 1 & u_{12} & u_{13} & u_{14} \\ 0 & 1 & u_{23} & u_{24} \\ 0 & 0 & 1 & u_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.39)$$

- In the equation (3.39) we observe that we have 4*4 unknowns
- If we do the L*U multiplication we obtain 4*4 equations to determine the $l_{ij}$ and $u_{ij}$ coefficients.

The resolution will involve the following steps:

***1. Product of all the lines of $L$ by the first column of $U$***

- We immediately observe that:

$$l_{11} = a_{11}\ ,\ l_{21} = a_{21}\ ,\ l_{31} = a_{31}\ ,\ l_{41} = a_{41} \quad (5.40)$$

***2. Product of the fist line of $L$ by all the columns of $U$***

- We do not repeat the first operation since it won't reveal any new unknown coefficient.
- We observe that:

$$l_{11} \times u_{12} + 0 \times 1 + 0 \times 0 + 0 \times 0 = a_{12} \Rightarrow u_{12} = \frac{a_{12}}{l_{11}}\ (l_{11} \neq 0) \quad (5.41)$$

- Similarly, we deduce:

$$u_{13} = \frac{a_{13}}{l_{11}}\ (l_{11} \neq 0) \text{ and } u_{14} = \frac{a_{14}}{l_{11}}\ (l_{11} \neq 0) \quad (5.42)$$

***3. Product of all the lines of $L$ by the second column of $U$***

- We do not repeat the first operation since it won't reveal any new unknown coefficient.
- We observe that:

$$l_{21} \times u_{12} + l_{22} \times 1 + 0 \times 0 + 0 \times 0 = a_{22} \Rightarrow l_{22} = a_{22} - l_{21} \times u_{12} \quad (5.43)$$

- Similarly, we deduce:

$$l_{32} = a_{32} - l_{31} \times u_{12} \text{ and } l_{42} = a_{42} - l_{41} \times u_{12} \quad (5.44)$$

***4. Product of the second line of $L$ by all the columns of $U$***

- We do not repeat the 1st and 2nd operations since it won't reveal any new unknown coefficient.
- We observe that:

$$l_{21}\times u_{13}+l_{22}\times u_{23}+0\times 1+0\times 0=a_{23}\Rightarrow u_{23}=\frac{a_{23}-l_{21}\times u_{13}}{l_{22}}\ (l_{22}\neq 0) \tag{5.45}$$

- Similarly, we deduce:

$$u_{24}=\frac{a_{24}-l_{21}\times u_{14}}{l_{22}}\ (l_{22}\neq 0) \tag{5.46}$$

***5. Product of all the lines of $L$ by the third column of $U$***

- We do not repeat the 1st and 2nd operations since it won't reveal any new unknown coefficient.
- We observe that:

$$l_{31}\times u_{13}+l_{32}\times u_{23}+l_{33}\times 1+0\times 0=a_{33}\Rightarrow l_{33}=a_{33}-l_{31}\times u_{13}-l_{32}\times u_{23} \tag{5.47}$$

- Similarly, we deduce:

$$l_{43}=a_{43}-l_{41}\times u_{13}-l_{42}\times u_{23} \tag{5.48}$$

***6. Product of the third line of $L$ by all the columns of $U$***

- We do not repeat the 1st, 2nd and 3rd operations since it won't reveal any new unknown coefficient.
- We observe that:

$$l_{31}\times u_{14}+l_{32}\times u_{24}+l_{33}\times u_{34}+0\times 1=a_{34}\Rightarrow u_{34}=\frac{a_{34}-l_{31}\times u_{14}-l_{32}\times u_{24}}{l_{33}}\ (l_{33}\neq 0) \tag{5.49}$$

***7. Production of all the line of $L$ by the fourth columns of $U$***

- We do not repeat the 1st, 2nd and 3rd operations since it won't reveal any new unknown coefficient.
- We observe that:

$$l_{41}\times u_{14}+l_{42}\times u_{24}+l_{43}\times u_{34}+l_{44}\times 1=a_{44}\Rightarrow l_{44}=a_{44}-l_{41}\times u_{14}-l_{42}\times u_{24}-l_{43}\times u_{34} \tag{5.50}$$

Following these steps, we were able to determine all the unknown coefficients. The methodology for Crout decomposition consists of alternating between the product of all the line of $L$ by a column of $U$ then the product of a single line of $L$ by all the columns of $U$ .

# Chapter VII: Approximation Methods for Solving Linear Systems

## VII-1 Introduction

This chapter introduces iterative approximation methods for solving linear systems, with focus on the Jacobi method, the Gauss-Seidel method, and the use of relaxation for second-year bachelor students in the Science and Technology common branch.

These methods are especially useful when the system is large, when direct methods are expensive, or when an approximate solution is sufficient.The chapter emphasizes the algorithmic idea of successive approximations, practical computation, and the basic conditions that guarantee convergence.

### Learning objectives

At the end of this chapter, students should be able to:

- Explain why iterative methods are used to solve linear systems.
- Rewrite a linear system in a form suitable for Jacobi and Gauss-Seidel iterations.
- Apply the Jacobi method step by step to a small system.
- Apply the Gauss-Seidel method and explain why it usually converges faster than Jacobi.
- Explain the role of the relaxation parameter in under-relaxation and over-relaxation methods.
- State a basic convergence condition such as strict diagonal dominance.

### Definition

A linear system of $n$ equations in $n$ unknowns can be written in matrix form as

$$Ax = b,$$

where $A$ is an $n \times n$ matrix, $x$ is the unknown vector, and $b$ is the right-hand side vector.

Direct methods such as Gaussian elimination produce the exact solution in a finite number of arithmetic steps, at least in exact arithmetic. Iterative methods follow a different idea: they start from an initial guess and generate a sequence of approximations that ideally converges to the exact solution.

The general form of an iterative method is

$$x^{(k+1)} = Bx^{(k)} + c,$$

where $x^{(k)}$ is the approximation at iteration $k$, $B$ is the iteration matrix, and $c$ is a constant vector. If the sequence converges, then its limit is the solution of the system.

**Why iterative methods are useful**

Iterative methods are attractive for several reasons:

- They are often simpler to implement than full elimination methods for large systems.
- They may require less memory, especially in sparse problems.
- They can provide approximate solutions after only a few iterations.
- They are the basis of many advanced numerical methods used in scientific computing.

For small systems, direct methods are usually faster and more reliable. However, iterative methods are pedagogically important because they introduce the ideas of convergence, error control, and algorithmic approximation.

**Matrix splitting**

To construct stationary iterative methods, the matrix $A$ is often decomposed into three parts:

$$A = D - L - U,$$

where:

- $D$ is the diagonal part of $A$.
- $L$ contains the entries below the diagonal.
- $U$ contains the entries above the diagonal.

This splitting is the basis of Jacobi, Gauss-Seidel, and relaxation methods.

Consider the system

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1,$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2,$$

$$\vdots$$

$$a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n.$$

If all diagonal coefficients are nonzero, each equation can be solved for its corresponding variable. This gives formulas that lead naturally to iterative schemes.

**Jacobi method**

**Principle**

In the Jacobi method, each equation is solved for one variable, and all new values are computed using only the values from the previous iteration.This means the method updates all components simultaneously after completing the full iteration.

For the linear system $Ax = b$, the Jacobi iteration can be written as

$$Dx^{(k+1)} = (L + U)x^{(k)} + b,$$

or equivalently,

$$x^{(k+1)} = D^{-1}(L + U)x^{(k)} + D^{-1}b.$$

The corresponding iteration matrix is

$$B_J = D^{-1}(L + U).$$

This is the standard matrix form of the Jacobi method.

**Component form**

For each $i = 1,2, \dots, n$, the Jacobi update is

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

Every new component at step $k + 1$ depends only on values from step $k$.

**Algorithm**

A practical form of the Jacobi method is:

1. Choose an initial approximation $x^{(0)}$.
2. Compute each component of $x^{(1)}$ from the Jacobi formulas.
3. Repeat until the difference between two successive approximations is smaller than a prescribed tolerance.

**Worked example 1**

Solve approximately the system

$$10x_1 - x_2 + 2x_3 = 6,$$

$$-x_1 + 11x_2 - x_3 = 25,$$

$$2x_1 - x_2 + 10x_3 = -11$$

by the Jacobi method, starting from

$$x^{(0)} = (0,0,0)^T.$$

The equations are rewritten as

$$x_1 = \frac{1}{10}(6 + x_2 - 2x_3),$$

$$x_2 = \frac{1}{11}(25 + x_1 + x_3),$$

$$x_3 = \frac{1}{10}(-11 - 2x_1 + x_2).$$

Using the initial guess $(0,0,0)$, the first iteration gives

$$x_1^{(1)} = \frac{6}{10} = 0.6,$$

$$x_2^{(1)} = \frac{25}{11} \approx 2.2727,$$

$$x_3^{(1)} = \frac{-11}{10} = -1.1.$$

So

$$x^{(1)} = (0.6, 2.2727, -1.1)^T.$$

For the second iteration,

$$x_1^{(2)} = \frac{1}{10}(6 + 2.2727 - 2(-1.1)) = 1.0473,$$

$$x_2^{(2)} = \frac{1}{11}(25 + 0.6 - 1.1) = 2.2273,$$

$$x_3^{(2)} = \frac{1}{10}(-11 - 2(0.6) + 2.2727) = -0.9927.$$

Thus,

$$x^{(2)} = (1.0473, 2.2273, -0.9927)^T.$$

Continuing the process gives values approaching the exact solution.

**Remarks on Jacobi**

The Jacobi method is conceptually simple and easy to parallelize because all new values are computed independently from the previous vector. However, it often converges more slowly than the Gauss-Seidel method.

**Gauss-Seidel method**

**Principle**

The Gauss-Seidel method improves the Jacobi idea by using the newest available values as soon as they are computed during the same iteration. This usually accelerates convergence.

In matrix form, the method is written as

$$(D - L)x^{(k+1)} = Ux^{(k)} + b,$$

so that

$$x^{(k+1)} = (D - L)^{-1}Ux^{(k)} + (D - L)^{-1}b.$$

The iteration matrix is therefore

$$B_{GS} = (D - L)^{-1}U.$$

This is the standard matrix form of the Gauss-Seidel method.

**Component form**

For each equation, the new value is computed by

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

So the components already updated in the current iteration are reused immediately.

**Worked example 2**

Use Gauss-Seidel on the same system with the same initial guess $x^{(0)} = (0,0,0)^T$.

The formulas are

$$x_1^{(k+1)} = \frac{1}{10}(6 + x_2^{(k)} - 2x_3^{(k)}),$$

$$x_2^{(k+1)} = \frac{1}{11}(25 + x_1^{(k+1)} + x_3^{(k)}),$$

$$x_3^{(k+1)} = \frac{1}{10}(-11 - 2x_1^{(k+1)} + x_2^{(k+1)}).$$

At the first iteration,

$$x_1^{(1)} = 0.6.$$

Then this new value is used immediately:

$$x_2^{(1)} = \frac{1}{11}(25 + 0.6 + 0) = 2.3273.$$

Then both new values are used for the third equation:

$$x_3^{(1)} = \frac{1}{10}(-11 - 2(0.6) + 2.3273) = -0.9873.$$

So

$$x^{(1)} = (0.6,2.3273,-0.9873)^T.$$

For the second iteration,

$$x_1^{(2)} = \frac{1}{10}(6 + 2.3273 - 2(-0.9873)) = 1.0302,$$

$$x_2^{(2)} = \frac{1}{11}(25 + 1.0302 - 0.9873) = 2.2766,$$

$$x_3^{(2)} = \frac{1}{10}(-11 - 2(1.0302) + 2.2766) = -1.0784.$$

Thus,

$$x^{(2)} = (1.0302, 2.2766, -1.0784)^T.$$

The method converges rapidly for this system.

**Why Gauss-Seidel is often faster**

Because Gauss-Seidel uses the newest computed values immediately, it generally reduces the error more efficiently than Jacobi does. In many examples, fewer iterations are needed to reach the same tolerance.

However, unlike Jacobi, the method is sequential, so it is less naturally parallel.

**Convergence of Jacobi and Gauss-Seidel methods**

An iterative method is useful only when the sequence it generates converges. A standard sufficient condition for the convergence of both Jacobi and Gauss-Seidel methods is strict diagonal dominance of the coefficient matrix.

A matrix $A = (a_{ij})$ is strictly diagonally dominant by rows if, for every row $i$,

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|.$$

If this condition holds, then both Jacobi and Gauss-Seidel converge.

**Example of diagonal dominance**

For the matrix

$$A = \begin{pmatrix} 10 & -1 & 2 \\ -1 & 11 & -1 \\ 2 & -1 & 10 \end{pmatrix},$$

one has

$$|10| > |-1| + |2| = 3,$$

$$|11| > |-1| + |-1| = 2,$$

$$|10| > |2| + |-1| = 3.$$

Therefore the matrix is strictly diagonally dominant, so Jacobi and Gauss-Seidel converge for this system.

**Practical stopping criterion**

In computations, the iterations are stopped when the change between two approximations becomes sufficiently small. A common criterion is

$$\|x^{(k+1)} - x^{(k)}\| < \varepsilon,$$

where $\varepsilon$ is a prescribed tolerance. For classroom exercises, one can also stop when all components stabilize to the desired number of decimal places.

**Relaxation methods**

**Idea of relaxation**

Relaxation modifies an iterative method by introducing a parameter $\omega$, called the relaxation factor or relaxation coefficient. The purpose is to improve convergence by either damping or accelerating the update.

There are two main cases:

- Under-relaxation: $0 < \omega < 1$, which makes changes more cautious.
- Over-relaxation: $1 < \omega < 2$, which may accelerate convergence when the choice is suitable.

The most common relaxation method in this context is successive over-relaxation, abbreviated SOR.

**Relaxed update formula**

If $x_i^{(k+1,GS)}$ denotes the Gauss-Seidel update, then the relaxed value is given by

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \omega x_i^{(k+1,GS)}.$$

This formula shows that the new value is a weighted combination of the old value and the Gauss-Seidel correction.

In matrix form, SOR is a modification of Gauss-Seidel obtained by introducing $\omega$ into the iterative scheme.

**Interpretation**

If $\omega = 1$, the SOR method reduces to the ordinary Gauss-Seidel method. If $\omega < 1$, the correction is reduced, which may help stabilize difficult iterations. If $\omega > 1$, the correction is amplified, which can speed up convergence but can also cause divergence if $\omega$ is poorly chosen.

**Worked example 3**

Consider a Gauss-Seidel update that gives, for one component,

$$x_i^{(k)} = 2.000, x_i^{(k+1,GS)} = 1.600.$$

If $\omega = 1.25$, then the relaxed value is

$$x_i^{(k+1)} = (1-1.25)(2.000) + 1.25(1.600).$$

So

$$x_i^{(k+1)} = -0.25 \times 2.000 + 2.000 = 1.500.$$

The method has moved more aggressively than standard Gauss-Seidel, which would have given 1.600.

If instead $\omega = 0.5$, then

$$x_i^{(k+1)} = 0.5(2.000) + 0.5(1.600) = 1.800.$$

This is a more cautious correction.

**Remarks on relaxation**

Relaxation can significantly improve performance, but the choice of $\omega$ is problem dependent. A good value of $\omega$ may speed up convergence, while a poor value may slow it down or destroy convergence entirely.

For an introductory course, it is enough to explain the role of the parameter and show how it modifies the Gauss-Seidel step.

**Worked example 4: comparison on a 2x2 system**

Consider the system

$$4x_1 - x_2 = 3,$$

$$2x_1 + 5x_2 = 7.$$

The exact solution exists and can be used for comparison.

**Jacobi form**

Rewrite the system as

$$x_1 = \frac{3 + x_2}{4},$$

$$x_2 = \frac{7 - 2x_1}{5}.$$

Starting from $(0,0)$, the first Jacobi iteration gives

$$x_1^{(1)} = 0.75, x_2^{(1)} = 1.4.$$

The second iteration gives

$$x_1^{(2)} = \frac{3 + 1.4}{4} = 1.1,$$

$$x_2^{(2)} = \frac{7 - 2(0.75)}{5} = 1.1.$$

So

$$x^{(2)} = (1.1,1.1)^T.$$

**Gauss-Seidel form**

Starting again from $(0,0)$, the first Gauss-Seidel iteration gives

$$x_1^{(1)} = 0.75,$$

then

$$x_2^{(1)} = \frac{7 - 2(0.75)}{5} = 1.1.$$

So

$$x^{(1)} = (0.75,1.1)^T.$$

At the second iteration,

$$x_1^{(2)} = \frac{3 + 1.1}{4} = 1.025,$$

$$x_2^{(2)} = \frac{7 - 2(1.025)}{5} = 0.99.$$

Thus,

$$x^{(2)} = (1.025,0.99)^T.$$

This illustrates how Gauss-Seidel often approaches the solution more effectively than Jacobi.

**Implementation steps**

For classroom and exam problems, students should follow a clear procedure:

1. Rewrite each equation so that the diagonal variable is isolated.
2. Choose an initial approximation, often the zero vector.
3. Apply the chosen iterative formula carefully.
4. Organize results in a table of iterations.
5. Stop when the tolerance is reached or when the requested number of iterations has been performed.

It is also useful to check first whether the matrix is diagonally dominant, because this often gives a quick indication that the process will converge.

**Exercises**

1. Perform two Jacobi iterations for the system

$$5x_1 - x_2 = 4,$$

$$-x_1 + 6x_2 = 5,$$

starting from $(0,0)$.

2. Perform two Gauss-Seidel iterations for the same system.

3. Verify that the matrix

$$\begin{pmatrix} 5 & -1 \\ -1 & 6 \end{pmatrix}$$

is strictly diagonally dominant.

4. Solve approximately by Jacobi the system

$$8x_1 + x_2 - x_3 = 8,$$

$$2x_1 + 10x_2 + x_3 = 13,$$

$$x_1 - x_2 + 9x_3 = 8,$$

starting from the zero vector.

5. Solve the same system by Gauss-Seidel and compare the number of iterations needed to reach two correct decimal places.

6. Explain the meaning of under-relaxation and over-relaxation.

7. If $x_i^{(k)} = 3.2$, $x_i^{(k+1,GS)} = 2.8$, and $\omega = 1.5$, compute the relaxed update.

8. Why is Gauss-Seidel generally faster than Jacobi for many convergent systems?

**Guided solutions**

**Solution 1**

The Jacobi formulas are

$$x_1^{(k+1)} = \frac{4 + x_2^{(k)}}{5}, x_2^{(k+1)} = \frac{5 + x_1^{(k)}}{6}.$$

From $(0,0)$, the first iteration gives

$$x^{(1)} = \left(\frac{4}{5}, \frac{5}{6}\right) \approx (0.8, 0.8333).$$

The second iteration gives

$$x_1^{(2)} = \frac{4 + 0.8333}{5} = 0.9667, x_2^{(2)} = \frac{5 + 0.8}{6} = 0.9667.$$

**Solution 2**

For Gauss-Seidel,

$$x_1^{(k+1)} = \frac{4 + x_2^{(k)}}{5}, x_2^{(k+1)} = \frac{5 + x_1^{(k+1)}}{6}.$$

Starting from $(0,0)$,

$$x_1^{(1)} = 0.8, x_2^{(1)} = \frac{5.8}{6} = 0.9667.$$

Then,

$$x_1^{(2)} = \frac{4 + 0.9667}{5} = 0.9933, x_2^{(2)} = \frac{5 + 0.9933}{6} = 0.9989.$$

This is already very close to the solution.

**Conclusion**

Iterative methods solve linear systems by generating a sequence of approximate solutions rather than eliminating variables in one finite procedure. The Jacobi method is simple and systematic, the Gauss-Seidel method usually converges faster by reusing new values immediately, and relaxation introduces a parameter that can further improve convergence when chosen appropriately.

## General Conclusion

This numerical methods handout has presented a set of fundamental tools for approximating solutions to mathematical problems that are difficult or impossible to treat exactly. Starting from basic ideas such as approximation, iteration, and error, the course has shown how to construct and analyze numerical schemes for integration, function approximation, and the solution of linear systems. Throughout, the emphasis has been on understanding the logic of each method, not simply on memorizing formulas.

A recurring theme is the balance between accuracy, cost, and stability. Methods like the trapezoidal rule and Simpson's rule illustrate how better approximations can be obtained by using more information about the function, while iterative schemes such as Jacobi and Gauss–Seidel show how repeated correction can converge to an exact solution under suitable conditions. In all cases, numerical error and convergence behavior are central: a numerical result is meaningful only when there is a clear idea of how close it is to the true solution and how the algorithm behaves as its parameters are varied.

For second-year students, these methods are an introduction to the broader world of scientific computing. In more advanced studies, the techniques introduced here will be extended, refined, and implemented in software, allowing the treatment of large-scale and complex systems. The concepts of consistency, stability, convergence, and error analysis will remain fundamental. The hope is that this handout has not only provided practical computational tools, but also developed a numerical way of thinking: formulating problems in a discrete setting, designing algorithms, and evaluating their reliability.

Students are encouraged to revisit these notes when they encounter numerical problems in other courses. The methods presented here are not isolated topics; they are common foundations used in mechanics, electromagnetism, fluid dynamics, optimization, data science, and many other fields. A solid understanding of basic numerical methods will be a valuable asset throughout their studies and future professional careers.

*General conclusion*

**References for further readings**


1. C. Brezinski, Introduction à la pratique du calcul numérique, Dunod, Paris 1988.
2. G. Allaire et S.M. Kaber, Algèbre linéaire numérique, Ellipses, 2002.
3. G. Allaire et S.M. Kaber, Introduction à Scilab. Exercices pratiques corrigés d'algèbre linéaire, Ellipses, 2002.
4. G. Christol, A. Cot et C.-M. Marle, Calcul différentiel, Ellipses, 1996.
5. M. Crouzeix et A.-L. Mignot, Analyse numérique des équations différentielles, Masson, 1983.
6. S. Delabrière et M. Postel, Méthodes d'approximation. Équations différentielles. Applications Scilab, Ellipses,
7. 2004.
8. J.-P. Demailly, Analyse numérique et équations différentielles. Presses Universitaires de Grenoble, 1996.
9. E. Hairer, S. P. Norsettet G. Wanner, Solving Ordinary Differential Equations, Springer, 1993.
10. P. G. Ciarlet, Introduction à l'analyse numérique matricielle et à l'optimisation, Masson, Paris, 1982.