Computational Applied Mathematics

Bachelor's Degree in Aeronautical and Space Sciences

Emanuel A. R. Camacho

emanuel.camacho@iseclisboa.pt
earc@earc96.com

Instituto Superior de Educação e Ciências (ISEC Lisboa)

Latest Update: *15th December, 2025, 20:38*

Bibliography

- Ward C, David K, Numerical Mathematics and Computing, Sixth edition, Brooks Cole, USA, 2007.
- S. Chapman. (2019). The Student Edition of Matlab: high-performance numeric computation and visualization software: Version 4, User's guide. New Jersey: Prentice Hall.
- Chapman, Stephen J. (2002). Matlab programming for engineers. 2nd ed. Pacific Grove: Bookware companion series.
- D. Valentine. B. Hahn. (2019). Essential MATLAB for Engineers and Scientists. Academic Press; 7th edition.

Computational Applied Mathematics

Computational Applied Mathematics

- 45 hours
- 3 hours per week (classes of 3 hours)
  - First part: 1h:15min
  - Break: 30min
  - Second part: 1h:15min
- Passive and Active Learning

Computational Applied Mathematics is a field that applies advanced mathematical models, computational methods, and high-performance computing to solve complex real-world problems in science, engineering, and other subject areas.

Evaluation

Computational Applied Mathematics (100% [20/20])

Frequencies (100% [20/20])

- Frequency 1 (50% [10/20]) (17/11/2025)
- Frequency 2 (50% [10/20]) (21/01/2026)

or

Exam (100% [20/20])

- Exam (100% [20/20])

There is a minimum of [5/20] values for any component of the evaluation. [10/20] is required to pass.

Table of Contents

1. Outline
2. Taylor Series & Numerical Differentiation
3. Zeros, Maximums and Minimums of Functions
4. Linear and Non-Linear Systems of Equations
5. Interpolation & Approximation
6. Numerical Integration
7. Ordinary Differential Equations

Outline

1. Outline
2. Taylor Series & Numerical Differentiation
3. Zeros, Maximums and Minimums of Functions
4. Linear and Non-Linear Systems of Equations
5. Interpolation & Approximation
6. Numerical Integration
7. Ordinary Differential Equations

Table of Contents

1. Outline
2. Taylor Series & Numerical Differentiation
3. Zeros, Maximums and Minimums of Functions
4. Linear and Non-Linear Systems of Equations
5. Interpolation & Approximation
6. Numerical Integration
7. Ordinary Differential Equations

Taylor's Theorem for $f(x)$

If the function f possesses continuous derivatives of orders 0, 1, 2, . . . , $(n + 1)$ in a closed interval $I = [a, b]$, then for any c and x in I,

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(c)}{k!} (x - c)^k + E_{n+1}, \tag{1}$$

where the error term E_{n+1} can be given in the form

$$E_{n+1} = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x - c)^{n+1}. \tag{2}$$

Here ξ is a point that lies between c and x and depends on both.

Taylor's Theorem for $f(x+h)$

If the function f possesses continuous derivatives of orders 0, 1, 2, …, $(n+1)$ in a closed interval $I = [a, b]$, then for any c and x in I,

$$f(x+h) = \sum_{k=0}^{n} \frac{f^{(k)}(c)}{k!} h^k + E_{n+1}, \tag{3}$$

where h is any value such that $x+h$ is in I and where

$$E_{n+1} = \frac{f^{(n+1)}(\xi)}{(n+1)!} h^{n+1}. \tag{4}$$

for some ξ between x and $x+h$.

Taylor Series

Pause for example

Taylor Series

Pause for exercices

Exercises

Numerical Differentiation

First-Derivative Formulas via Taylor Series

First Derivative Approximation

$$f'(x) = \frac{1}{h}\left[f(x+h) - f(x)\right] - \frac{h}{2}f''(\xi) \tag{5}$$

$$f'(x) = \frac{1}{h}\left[f(x) - f(x-h)\right] + \frac{h}{2}f''(\xi) \tag{6}$$

$$f'(x) = \frac{1}{2h}\left[f(x+h) - f(x-h)\right] - \frac{h^2}{6}f'''(\xi) \tag{7}$$

$$f'(x) = \frac{1}{2h}\left[f(x-2h) - 4f(x-h) + 3f(x)\right] + \frac{h^2}{3}f'''(\xi) \tag{8}$$

$$f'(x) = \frac{1}{2h}\left[-3f(x) + 4f(x+h) - f(x+2h)\right] + \frac{h^2}{3}f'''(\xi) \tag{9}$$

Numerical Differentiation

Second-Derivative Formulas via Taylor Series

Second Derivative Approximation

$$f''(x) = \frac{1}{h^2}\left[f(x-h) - 2f(x) + f(x+h)\right] - \frac{h^2}{12} f^{(4)}(\xi) \tag{10}$$

$$f''(x) = \frac{1}{h^2}\left[f(x-2h) - 2f(x-h) + f(x)\right] + h f'''(\xi) \tag{11}$$

$$f''(x) = \frac{1}{h^2}\left[f(x) - 2f(x+h) + f(x+2h)\right] - h f'''(\xi) \tag{12}$$

Derivative Approximations

Pause for example

Table of Contents

The Bolzano Theorem

If f is a continuous function on the closed interval $[a, b]$ and $f(a) \cdot f(b) < 0$, then

$$\exists c \in [a, b] : f(c) = 0. \tag{13}$$

The Rolle Theorem

If f is a continuous function on the closed interval $[a, b]$, differentiable in $]a, b[$ and $f(a) = f(b)$, then

$$\exists c \in]a, b[: f'(c) = 0. \tag{14}$$

Bisection Method

1. At each step in this algorithm, we have an interval $[a, b]$ and the values $u = f(a)$ and $v = f(b)$. The numbers u and v satisfy $uv < 0$.
2. Next, we construct the midpoint of the interval, $c = \frac{1}{2}(a+b)$, and compute $w = f(c)$.
3. Compute wu and if:
    - $wu < 0$, we store the value of c in b and w in v.
    - $wu > 0$, we store the value of c in a and w in u.
4. This step can now be repeated until the interval is satisfactorily small, say

$$|b - a| \leq \varepsilon \tag{15}$$

Bisection Method Theorem

If the bisection algorithm is applied to a continuous function f on an interval $[a, b]$, where $f(a)f(b) < 0$, then, after n steps, an approximate root will have been computed with error at most $(b - a)/2^{n+1}$.

If an error tolerance has been prescribed in advance, it is possible to determine the number of steps required by solving the following inequality for n:

$$\frac{b-a}{2^{n+1}} < \varepsilon \tag{16}$$

$$n > \frac{\log(b-a) - \log(2\varepsilon)}{\log 2} \tag{17}$$

Bisection Method

Pause for example

False Position (*Regula Falsi*) Method

Rather than selecting the midpoint of each interval, as observed in the bisection method, this method uses the point where the secant lines intersect the x-axis.

1. At the k^{th} step, it computes

$$c_k = \frac{a_k f(b_k) - b_k f(a_k)}{f(b_k) - f(a_k)} \tag{18}$$

2. Compute $f(a_k)f(c_k)$ and if
   - $f(a_k)f(c_k) > 0$, set $a_{k+1} = c_k$ and $b_{k+1} = b_k$
   - $f(a_k)f(c_k) < 0$, set $a_{k+1} = a_k$ and $b_{k+1} = c_k$.
3. The process is repeated until the root is approximated sufficiently well.

False Position Method

Pause for example

Newton Method (or Newton-Raphson Iteration)

Suppose again that x_0 is an initial approximation to a root of f. We ask: What correction h should be added to x_0 to obtain the root precisely? Obviously, we want

$$f(x_0 + h) = f(x_1) = 0 \tag{19}$$

$$f(x_1) = f(x_0) + (x_1 - x_0)f'(x_0) + ... = 0 \tag{20}$$

$$f(x_0) + hf'(x_0) + ... = 0 \tag{21}$$

Ignoring all but the first two terms in the series

$$f(x_0) + hf'(x_0) = 0 \Rightarrow h = -\frac{f(x_0)}{f'(x_0)} \tag{22}$$

Zeros, Maximums and Minimums of Functions

Zeros

Newton Method (or Newton-Raphson Iteration)

$$h = -\frac{f(x_0)}{f'(x_0)} \tag{23}$$

$$x_1 = x_0 + h = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{24}$$

Recursive Definition

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{25}$$

Newton's Method

Pause for example

Secant Method

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{26}$$

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \tag{27}$$

$$f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} \tag{28}$$

Recursive Definition

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n) \tag{29}$$

Secant Method

Pause for example

Roots of Equations

Pause for exercices

Exercises

Golden Section Search

$$r = \frac{\sqrt{5}-1}{2} \tag{30}$$

$$I_k = [a_k, b_k]$$

$$c_k = b_k - r(b_k - a_k) \quad \text{and} \quad d_k = a_k + r(b_k - a_k), \quad k = 0, 1, \ldots \tag{31}$$

$$f(c_k) < f(d_k) \Rightarrow I_{k+1} = [a_k, d_k] \tag{32}$$

$$f(c_k) > f(d_k) \Rightarrow I_{k+1} = [c_k, b_k] \tag{33}$$

$$E_a(x_k) \leq (1-r)(b_k - a_k) \tag{34}$$

Golden Section Search

Pause for example

Quadratic Interpolation Method

$$f(x) = f(x^*) + f'(x^*)(x - x^*) + \frac{1}{2}f''(x^*)(x - x^*)^2 + \cdots \tag{35}$$

$$f(x) = f(x^*) + \frac{1}{2}f''(x^*)(x - x^*)^2 + \cdots \tag{36}$$

$$x_{k+1} = \frac{f(x_{k-2})(x_{k-1}^2 - x_k^2) + f(x_{k-1})(x_k^2 - x_{k-2}^2) + f(x_k)(x_{k-2}^2 - x_{k-1}^2)}{2f(x_{k-2})(x_{k-1} - x_k) + 2f(x_{k-1})(x_k - x_{k-2}) + 2f(x_k)(x_{k-2} - x_{k-1})} \tag{37}$$

$$k = 2, 3, \ldots$$

Quadratic Interpolation Method

Pause for example

Newton's Method

As seen before, the Newton's method can be used to locate the roots of equations using

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{38}$$

The same recurrence relation can be applied to find a minimum by locating the zero of the first derivative rather than the zero of the function itself.

Newton's Method for Minimization

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)} \tag{39}$$

Newton's Method

Pause for example

Minimization of Functions

Pause for exercices

Exercises

Table of Contents

1. Outline
2. Taylor Series & Numerical Differentiation
3. Zeros, Maximums and Minimums of Functions
4. **Linear and Non-Linear Systems of Equations**
5. Interpolation & Approximation
6. Numerical Integration
7. Ordinary Differential Equations

Gauss Elimination

- **Naive Gaussian Elimination -** Naive Gaussian Elimination is a method for solving a system of linear equations by transforming the coefficient matrix into an upper triangular form using forward elimination, and then solving for the unknowns using back substitution.
- **Gaussian Elimination with Partial Pivoting -** Gaussian elimination with partial pivoting selects the pivot row to be the one with the maximum pivot entry in absolute value from those in the leading column of the reduced submatrix. Two rows are interchanged to move the designated row into the pivot row position.
- **Gaussian Elimination with Complete Partial Pivoting -** Gaussian elimination with complete pivoting selects the pivot entry as the maximum pivot entry from all entries in the submatrix. (This complicates things because some of the unknowns are rearranged.) Two rows and two columns are interchanged to accomplish this.

Gauss Elimination

$$Ax = b$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & a_{i3} & \cdots & a_{in} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{bmatrix}$$

Gauss Elimination

The basic forward elimination procedure using equation k to operate on equations $k+1, k+2, \ldots, n$ is

$$\begin{cases} a_{ij} \leftarrow a_{ij} - \left(\dfrac{a_{ik}}{a_{kk}}\right) a_{kj} & (k \leq j \leq n) \\ b_i \leftarrow b_i - \left(\dfrac{a_{ik}}{a_{kk}}\right) b_k & \end{cases}$$

with $a_{kk} \neq 0$.

Gauss Elimination

The basic back substitution starts by solving the nth equation for x_n as

$$x_n = \frac{b_n}{a_{nn}}$$

We continue working upward, recovering each x_i by the formula

$$x_i = \frac{1}{a_{ii}} \left(b_i - \sum_{j=i+1}^{n} a_{ij} x_j \right) \qquad (i = n-1,\ n-2,\ \ldots,\ 1)$$

Gauss Elimination

Pause for example

Gauss-Jordan Elimination

Used to solve systems of linear equations and to find the inverse of any invertible matrix

$$[A|I] = \left[\begin{array}{cccc|cccc} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & 0 & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 & 0 & \ddots & 0 \\ a_{n1} & a_{n2} & \cdots & a_{nn} & 0 & 0 & 0 & 1 \end{array}\right]$$

$$\downarrow$$

$$[I|A^{-1}] = \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\ 0 & 1 & 0 & 0 & \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2n} \\ 0 & 0 & \ddots & 0 & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & \alpha_{n1} & \alpha_{n2} & \cdots & \alpha_{nn} \end{array}\right]$$

Gauss-Jordan Elimination

Pause for example

Linear System of Equations

Pause for exercices

Exercises

LU Decomposition

$$Ax = b$$

$$LUx = b$$

$$Lz = b$$

$$Ux = z$$

Cholesky Decomposition

If A is a real, symmetric ($A = A^T$), and positive definite matrix ($x^T Ax > 0$), then it has a unique factorization $A = LL^T$, in which L is lower triangular with a positive diagonal.

$$\ell_{11} = \sqrt{a_{11}}$$

$$\ell_{1j} = \frac{a_{1j}}{\ell_{11}}, \quad j = 2, \ldots, n$$

$$\ell_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} u_{kj}^2}, \quad j = 2, \ldots, n$$

$$\ell_{ij} = \frac{1}{\ell_{ii}} \left(\sum_{k=1}^{i-1} u_{ki} u_{kj} \right), \quad i = 2, \ldots, n, \quad j = i+1, \ldots, n \quad u_{ij} = 0, i > j$$

Matrix Decomposition

Pause for example

Vector Norms

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \qquad \|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2} \qquad \|x\|_\infty = \max_{1\leq i\leq n} |x_i|$$

Matrix Norms

$$\|A\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} |a_{ij}| \qquad \|A\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}^2} \qquad \|A\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}|$$

Vector and Matrix Norms

Pause for example

Condition Number and Ill-Conditioning

An important quantity that has some influence in the numerical solution of a linear system $Ax = b$ is the condition number, which is defined as

$$\kappa(A) = \|A\|_2 \left\|A^{-1}\right\|_2$$

If the linear system is sensitive to perturbations in the elements of A, or to perturbations of the components of b, then this fact is reflected in A having a large condition number. In such a case, the matrix A is said to be ill-conditioned. Briefly, the larger the condition number, the more ill-conditioned the system.

Jacobi Method

$$x_i^{(k)} = \left[-\sum_{\substack{j=1 \\ j \neq i}}^{n} \left(\frac{a_{ij}}{a_{ii}} \right) x_j^{(k-1)} + \left(\frac{b_i}{a_{ii}} \right) \right] \qquad (1 \leq i \leq n) \tag{40}$$

Convergence Theorem

If A is diagonally dominant, then the Jacobi method converges for any starting vector $x^{(0)}$.

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$$

Jacobi Method

Pause for example

Gauss-Seidel Method

$$x_i^{(k)} = \left[-\sum_{\substack{j=1 \\ j<i}}^{n} \left(\frac{a_{ij}}{a_{ii}}\right) x_j^{(k)} - \sum_{\substack{j=1 \\ j>i}}^{n} \left(\frac{a_{ij}}{a_{ii}}\right) x_j^{(k-1)} + \left(\frac{b_i}{a_{ii}}\right) \right] \tag{41}$$

Convergence Theorem

If A is diagonally dominant, then the Gauss-Seidel method converges for any starting vector $x^{(0)}$.

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$$

Gauss-Seidel Method

Pause for example

Newton's Method

$$\begin{cases} f_1(x_1, x_2, \ldots, x_N) = 0 \\ \quad\vdots \\ f_N(x_1, x_2, \ldots, x_N) = 0 \end{cases} \tag{42}$$

Using vector notation, we can write this system in a more elegant form

$$\mathbf{F}(\mathbf{X}) = \mathbf{0} \tag{43}$$

by defining column vectors

$$\mathbf{F} = [f_1, f_2, \cdots, f_N]^T \tag{44}$$

$$\mathbf{X} = [x_1, x_2, \cdots, x_N]^T \tag{45}$$

Newton's Method

The extension of Newton's method for nonlinear systems is

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]^{-1}\mathbf{F}\left(\mathbf{X}^{(k)}\right), \tag{46}$$

where $\mathbf{F}'\left(\mathbf{X}^{(k)}\right)$ is the **Jacobian matrix**. In practice one solves the Jacobian linear system

$$\left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]\mathbf{H}^{(k)} = -\mathbf{F}\left(\mathbf{X}^{(k)}\right)$$

using Gaussian elimination and then finds the next iterate from the equation

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + \mathbf{H}^{(k)} \tag{47}$$

Jacobian Matrix

$$\mathbf{F}'(\mathbf{X}) = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_2} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \dfrac{\partial f_2}{\partial x_1} & \dfrac{\partial f_2}{\partial x_2} & \cdots & \dfrac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial f_n}{\partial x_1} & \dfrac{\partial f_n}{\partial x_2} & \cdots & \dfrac{\partial f_n}{\partial x_n} \end{bmatrix}$$

Non-Linear Systems of Equations

Pause for example

Non-Linear Systems of Equations

Pause for exercices

Exercises

Table of Contents

1. Outline
2. Taylor Series & Numerical Differentiation
3. Zeros, Maximums and Minimums of Functions
4. Linear and Non-Linear Systems of Equations
5. **Interpolation & Approximation**
6. Numerical Integration
7. Ordinary Differential Equations

Linear Interpolation

x	x_0	x_1	$\cdots$	x_n
y	y_0	y_1	$\cdots$	y_n

$$p(x) = \left(\frac{x - x_1}{x_0 - x_1}\right) y_0 + \left(\frac{x - x_0}{x_1 - x_0}\right) y_1 \tag{48}$$

$$= y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0}\right)(x - x_0) \tag{49}$$

Lagrange Polynomial

$$p_n(x) = \sum_{i=0}^{n} \ell_i(x) f(x_i) \tag{50}$$

where

$$\ell_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \qquad (0 \leq i \leq n) \tag{51}$$

$$\ell_i(x) = \left(\frac{x - x_0}{x_i - x_0}\right)\left(\frac{x - x_1}{x_i - x_1}\right)\cdots\left(\frac{x - x_{i-1}}{x_i - x_{i-1}}\right)\left(\frac{x - x_{i+1}}{x_i - x_{i+1}}\right)\cdots\left(\frac{x - x_n}{x_i - x_n}\right) \tag{52}$$

Newton Polynomial

$$p_n(x) = \sum_{i=0}^{n} a_i \pi_i(x) \tag{53}$$

where

$$\pi_i(x) = \prod_{j=0}^{i-1} (x - x_j) \tag{54}$$

$$p(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0) \cdots (x - x_{n-1}) \tag{55}$$

Divided Differences

$$a_n = f\left[x_0, x_1, \ldots, x_n\right] \tag{56}$$

where $f\left[x_0, x_1, \ldots, x_n\right]$ is called the divided difference of order n.

$$f\left[x_0, x_1, \ldots, x_n\right] = \sum_{j=0}^{n} \frac{f(x_j)}{\displaystyle\prod_{\substack{k=0\\k\neq j}}^{n} (x_j - x_k)} \tag{57}$$

Divided Differences

$$f\left[x_0, x_1, \ldots, x_n\right] = \sum_{j=0}^{n} \frac{f(x_j)}{\displaystyle\prod_{\substack{k=0\\k\neq j}}^{n} (x_j - x_k)} \tag{58}$$

$$a_0 = f[x_0] = f(x_0) \tag{59}$$

$$a_1 = f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \tag{60}$$

$$a_2 = f[x_0, x_1, x_2] = \frac{\dfrac{f(x_2) - f(x_1)}{x_2 - x_1} - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0} \tag{61}$$

Natural Cubic Spline

$$S(x) = \begin{cases} S_0(x) & (t_0 \leq x \leq t_1) \\ S_1(x) & (t_1 \leq x \leq t_2) \\ \vdots & \vdots \\ S_{n-1}(x) & (t_{n-1} \leq x \leq t_n) \end{cases} \tag{62}$$

$$S(t_i) = y_i \qquad (0 \leq i \leq n) \tag{63}$$

$$\lim_{x \to t_i^-} S^{(k)}(t_i) = \lim_{x \to t_i^+} S^{(k)}(t_i) \qquad (k = 0, 1, 2) \tag{64}$$

$$S''(t_0) = S''(t_n) = 0 \tag{65}$$

Natural Cubic Spline

$$\begin{bmatrix} 1 & 0 & & & & \\ h_0 & u_1 & h_1 & & & \\ & h_1 & u_2 & h_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & h_{n-2} & u_{n-1} & h_{n-1} \\ & & & & 0 & 1 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_{n-1} \\ z_n \end{bmatrix} = \begin{bmatrix} 0 \\ v_1 \\ v_2 \\ \vdots \\ v_{n-1} \\ 0 \end{bmatrix} \tag{66}$$

$$h_i = t_{i+1} - t_i \tag{67}$$

$$u_i = 2(h_{i-1} + h_i) \tag{68}$$

$$v_i = 6(b_i - b_{i-1}) \tag{69}$$

$$b_i = \frac{1}{h_i}(y_{i+1} - y_i) \tag{70}$$

Natural Cubic Spline

$$
\begin{aligned}
S_i(x) = \frac{z_{i+1}}{6h_i}(x - t_i)^3 + \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \\
+ \left(\frac{y_{i+1}}{h_i} - \frac{h_i}{6} z_{i+1}\right)(x - t_i) + \left(\frac{y_i}{h_i} - \frac{h_i}{6} z_i\right)(t_{i+1} - x) \qquad (71)
\end{aligned}
$$

$$
h_i = t_{i+1} - t_i \qquad (72)
$$

Method of Least Squares

$$f(x) \approx p(x) \tag{73}$$

$$p(x) = \sum_{i=0}^{n} a_i \varphi_i(x) \tag{74}$$

where

$$\{\varphi_0(x), \varphi_1(x), \dots, \varphi_n(x)\} \tag{75}$$

is a set of basis functions.

Method of Least Squares

$$\begin{bmatrix} \varphi_0(x_0) & \varphi_1(x_0) & \cdots & \varphi_n(x_0) \\ \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_n(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_0(x_m) & \varphi_1(x_m) & \cdots & \varphi_n(x_m) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}$$

$$Ax = b \tag{76}$$

$$A^+ = (A^T A)^{-1} A^T \tag{77}$$

$$x = A^+ b \tag{78}$$

Table of Contents

1. Outline
2. Taylor Series & Numerical Differentiation
3. Zeros, Maximums and Minimums of Functions
4. Linear and Non-Linear Systems of Equations
5. Interpolation & Approximation
6. Numerical Integration
7. Ordinary Differential Equations

Numerical Integration

Closed Newton-Cotes Rules

$$\int_a^b f(x)\ \mathrm{d}x \tag{79}$$

Here, $a = x_0$, $b = x_n$, $h = (b-a)/n$, $x_i = x_0 + ih$, for $i = 0, 1, \ldots, n$, where $h = (b-a)/n$, $f_i = f(x_i)$, and $a = x_0 < \xi < x_n = b$ in the error terms.

Trapezoid Rule:

$$\int_{x_0}^{x_1} f(x)\,dx = \frac{1}{2}h[f_0 + f_1] - \frac{1}{12}h^3 f''(\xi) \tag{80}$$

Simpson's 1/3 Rule:

$$\int_{x_0}^{x_2} f(x)\,dx = \frac{1}{3}h[f_0 + 4f_1 + f_2] - \frac{1}{90}h^5 f^{(4)}(\xi) \tag{81}$$

Simpson's 3/8 Rule:

$$\int_{x_0}^{x_3} f(x)\,dx = \frac{3}{8}h[f_0 + 3f_1 + 3f_2 + f_3] - \frac{3}{80}h^5 f^{(4)}(\xi) \tag{82}$$

Boole's Rule:

$$\int_{x_0}^{x_4} f(x)\,dx = \frac{2}{45}h[7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4] - \frac{8}{945}h^7 f^{(6)}(\xi) \tag{83}$$

Six-Point Newton-Cotes Closed Rule:

$$\int_{x_0}^{x_5} f(x)\,dx = \frac{5}{288}h\left[19f_0 + 75f_1 + 50f_2 + 50f_3 + 75f_4 + 19f_5\right] - \frac{275}{12096}h^7 f^{(6)}(\xi) \tag{84}$$

Numerical Integration

Open Newton-Cotes Rules

$$\int_a^b f(x)\ \mathrm{d}x \tag{85}$$

Here, $a = x_0$, $b = x_n$, $h = (b-a)/n$, $x_i = x_0 + ih$, for $i = 0, 1, \ldots, n$, where $h = (b-a)/n$, $f_i = f(x_i)$, and $a = x_0 < \xi < x_n = b$ in the error terms.

Midpoint Rule:

$$\int_{x_0}^{x_2} f(x)\,dx = 2hf_1 + \frac{1}{24}h^3 f''(\xi) \tag{86}$$

Two-Point Newton-Cotes Open Rule:

$$\int_{x_0}^{x_3} f(x)\,dx = \frac{3}{2}h[f_1 + f_2] + \frac{1}{4}h^3 f''(\xi) \tag{87}$$

Three-Point Newton-Cotes Open Rule:

$$\int_{x_0}^{x_4} f(x)\,dx = \frac{4}{3}h[2f_1 - f_2 + 2f_3] + \frac{28}{90}h^5 f^{(4)}(\xi) \tag{88}$$

Four-Point Newton-Cotes Open Rule:

$$\int_{x_0}^{x_5} f(x)\,dx = \frac{5}{24}h[11f_1 + f_2 + f_3 + 11f_4] + \frac{95}{144}h^5 f^{(4)}(\xi) \tag{89}$$

Five-Point Newton-Cotes Open Rule:

$$\int_{x_0}^{x_6} f(x)\,dx = \frac{6}{20}h[11f_1 - 14f_2 + 26f_3 - 14f_4 + 11f_5] - \frac{41}{140}h^7 f^{(6)}(\xi) \tag{90}$$

Composite Trapezoidal Rule

$$\int_a^b f(x)\,dx \approx \frac{h}{2}\left[f(a) + 2\sum_{k=1}^{n-1} f(x_k) + f(b)\right] - \frac{b-a}{12}h^2 f''(\xi) \tag{91}$$

Composite Simpson's Rule (Simpson's 1/3 Rule)

For even number of subintervals n

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[f(a) + 4\sum_{k=1,3,5,\ldots}^{n-1} f(x_k) + 2\sum_{k=2,4,6,\ldots}^{n-2} f(x_k) + f(b)\right] - \frac{b-a}{180}h^4 f^{(4)}(\xi) \tag{92}$$

Table of Contents

Ordinary Differential Equations

Initial-Value Problem

In this chapter, we concentrate on one type of differential equation and one type of auxiliary condition: the initial-value problem for a first-order differential equation. The standard form that has been adopted is

$$\begin{cases} x' = f(t, x) \\ x(a) \text{ is given} \end{cases} \tag{93}$$

It is understood that x is a function of t, so the differential equation is written in more detail looks like:

$$\frac{dx(t)}{dt} = f(t, x(t)) \tag{94}$$

Taylor Series Methods

Its principle is to represent the solution of a differential equation locally by a few terms of its Taylor series.

$$x(t+h) = x(t) + h\,x'(t) + \\ + \frac{1}{2!}h^2\,x''(t) + \frac{1}{3!}h^3\,x'''(t) + \frac{1}{4!}h^4\,x^{(4)}(t) + \cdots + \frac{1}{m!}h^m\,x^{(m)}(t) + \cdots \quad (95)$$

For numerical purposes, the Taylor series truncated after $m+1$ terms enables us to compute $x(t+h)$ rather accurately if h is small and if $x(t)$, $x'(t)$, $x''(t), \ldots, x^{(m)}(t)$ are known. When only terms through $h^m x^{(m)}(t)/m!$ are included in the Taylor series, the method that results is called the **Taylor series method of order** m.

Euler Method

The Taylor series method of order 1 is known as **Euler's method**. To find approximate values of the solutions to the initial-value problem

$$\begin{cases} x' = f(t, x(t)) \\ x(a) = x_a \end{cases} \tag{96}$$

over the interval $[a, b]$, the first two terms in the Taylor series (95) are used:

$$x(t+h) \approx x(t) + hx'(t) \tag{97}$$

Hence, the formula

$$x(t+h) = x(t) + hf\big(t, x(t)\big) \tag{98}$$

can be used to step from $t = a$ to $t = b$ with n steps of size $h = (b-a)/n$.

Runge-Kutta Methods

The methods named after Carl Runge and Wilhelm Kutta are designed to imitate the Taylor series method without requiring analytic differentiation of the original differential equation.

The resulting **second-order Runge-Kutta method** is

$$x(t+h) = x(t) + \frac{1}{2}(K_1 + K_2) \tag{99}$$

where

$$\begin{cases} K_1 = hf(t, x) \\ K_2 = hf\left(t+h,\ x+K_1\right) \end{cases} \tag{100}$$

Runge-Kutta Methods

The classical **fourth-order Runge-Kutta method** uses the following formulas:

$$x(t+h) = x(t) + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \tag{101}$$

where

$$\begin{cases} K_1 = hf(t,x) \\ K_2 = hf\left(t + \frac{1}{2}h,\ x + \frac{1}{2}K_1\right) \\ K_3 = hf\left(t + \frac{1}{2}h,\ x + \frac{1}{2}K_2\right) \\ K_4 = hf\left(t + h,\ x + K_3\right) \end{cases} \tag{102}$$

The Adams-Bashforth-Moulton methods are a family of predictor-corrector numerical techniques used for solving ordinary differential equations (ODEs). The Adams-Bashforth method provides an initial guess for the new value. The Adams-Moulton method then corrects this guess to improve accuracy.

Adams-Bashforth-Moulton Methods

- Second-order multistep method

$$\tilde{x}(t+h) = x(t) + \frac{h}{2}\Big(3f(t, x(t)) - f(t-h, x(t-h))\Big) \tag{103}$$

$$x(t+h) = x(t) + \frac{h}{2}\Big(f(t+h, \tilde{x}(t+h)) + f(t, x(t))\Big) \tag{104}$$

Adams-Bashforth-Moulton Methods

- Third-order multistep method

$$\begin{split}\tilde{x}(t+h) = x(t) + \frac{h}{12}\Big(23f(t,x(t)) - 16f(t-h,x(t-2h))\\ + 5f(t-2h,x(t-2h))\Big)\end{split}\tag{105}$$

$$\begin{split}x(t+h) = x(t) + \frac{h}{12}\Big(5f(t+h,\tilde{x}(t+h)) + 8f(t,x(t))\\ - f(t-h,x(t-h))\Big)\end{split}\tag{106}$$

Adams-Bashforth-Moulton Methods

- Fourth-order multistep method

$$\begin{aligned}\tilde{x}(t+h) = x(t) + \frac{h}{24}\Big(&55f(t,x(t)) - 59f(t-h,x(t-h)) \\ &+ 37f(t-2h,x(t-2h)) - 9f(t-3h,x(t-3h))\Big) \qquad (107)\end{aligned}$$

$$\begin{aligned}x(t+h) = x(t) + \frac{h}{24}\Big(&9f(t+h,\tilde{x}(t+h)) + 19f(t,x(t)) \\ &- 5f(t-h,x(t-h)) + f(t-2h,x(t-2h))\Big) \qquad (108)\end{aligned}$$

$$\begin{cases} x_1' = f_1(t, x_1, x_2, \ldots, x_n) \\ x_2' = f_2(t, x_1, x_2, \ldots, x_n) \\ \vdots \\ x_n' = f_n(t, x_1, x_2, \ldots, x_n) \\ x_1(a) = s_1,\ x_2(a) = s_2,\ \ldots,\ x_n(a) = s_n \qquad \text{all given} \end{cases} \tag{109}$$

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \mathbf{X}' = \begin{bmatrix} x_1' \\ x_2' \\ \vdots \\ x_n' \end{bmatrix} \qquad \mathbf{F} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \qquad \mathbf{S} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix} \tag{110}$$

m-order Taylor Series method

$$\mathbf{X}(t+h)=\mathbf{X}(t)+h\mathbf{X}(t)'+\frac{h^2}{2}\mathbf{X}(t)''+\cdots+\frac{h^m}{m!}\mathbf{X}(t)^{(m)} \tag{111}$$

4th-order Runge-Kutta method

$$\mathbf{X}(t+h)=\mathbf{X}(t)+\frac{h}{6}(\mathbf{K}_1+2\mathbf{K}_2+2\mathbf{K}_3+\mathbf{K}_4) \tag{112}$$

$$\begin{cases} \mathbf{K}_1=\mathbf{F}(t,x) \\ \mathbf{K}_2=\mathbf{F}(t+1/2h,\mathbf{X}+1/2h\mathbf{K}_1) \\ \mathbf{K}_3=\mathbf{F}(t+1/2h,\mathbf{X}+1/2h\mathbf{K}_2) \\ \mathbf{K}_4=\mathbf{F}(t+h,\mathbf{X}+h\mathbf{K}_3) \end{cases} \tag{113}$$

4^{th}-order Adams-Bashforth-Moulton method

- Adams-Bashforth method (predictor)

$$\begin{multline}\tilde{\mathbf{X}}(t+h) = \mathbf{X}(t) + \frac{h}{24}\Big(55\mathbf{F}(t,\mathbf{X}(t)) - 59\mathbf{F}(t-h,\mathbf{X}(t-h)) \\ + 37\mathbf{F}(t-2h,\mathbf{X}(t-2h)) - 9\mathbf{F}(t-3h,\mathbf{X}(t-3h))\Big) \tag{114}\end{multline}$$

- Adams-Moulton method (corrector)

$$\begin{multline}\mathbf{X}(t+h) = \mathbf{X}(t) + \frac{h}{24}\Big(9\mathbf{F}(t+h,\tilde{\mathbf{X}}(t+h)) + 19\mathbf{F}(t,\mathbf{X}(t)) \\ - 5\mathbf{F}(t-h,\mathbf{X}(t-h)) + \mathbf{F}(t-2h,\mathbf{X}(t-2h))\Big) \tag{115}\end{multline}$$

Computational Applied Mathematics

Bachelor's Degree in Aeronautical and Space Sciences

Emanuel A. R. Camacho

emanuel.camacho@iseclisboa.pt
earc@earc96.com

Instituto Superior de Educação e Ciências (ISEC Lisboa)

Latest Update: *15th December, 2025, 20:38*