Numerical Calculus

Pătrulescu Flavius

Technical University of Cluj-Napoca

2024

References

1. R.L. Burden, D.J. Faires, A.M. Burden, *Numerical Analysis*, 10th Ed., Cengage Learning, Boston, 2022.
2. J.C. Butcher, *Numerical Methods for Ordinary Differential Equations*, 2nd Ed., John Wiley & Sons Ltd., Chichester, 2008.
3. G. Dahlquist, A. Björk, *Numerical Methods in Scientific Computing*, Vol. I, SIAM, Philadelphia, 2008.
4. M.T. Heath, *Scientific Computing*, 2nd Ed., SIAM, Philadelphia, 2002.
5. A. Quarteroni, F. Saleri, P. Gervasio, *Scientific Computing with Matlab and Octave*, 4th Ed., Springer-Verlag, Berlin Heidelberg, 2014.
6. L.F. Shampine, R.C. Allen, S. Pruess, *Fundamental of Numerical Computing*, John Wiley & Sons Inc., New York, 1997.
7. E. Süli, D. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, Cambridge, 2003.

Numerical Calculus (Heath, 2002)

Most of the problems of continuous mathematics (for example, almost any problem involving derivatives, integrals, or nonlinearities) cannot be solved exactly, even in principle, in a finite number of steps and thus must be solved by a iterative process (theoretically infinite) that ultimately converges to a solution. In practice, of course, one does not iterate forever, but only until the answer is approximately correct, *close enough* to the desired result for practical purposes. Thus, one of the most important aspects of scientific computing is finding rapidly convergent iterative algorithms and assessing the accuracy of the resulting approximation.

General Strategy (Heath, 2002)

In seeking a solution to a given computational problem, a basic general strategy is to replace a difficult problem with an easier one that has the same solution, or at least a closely related solution. Examples of this approach include

Replacing infinite processes with finite processes, such as replacing integrals or infinite series with finite sums, or derivatives with finite difference quotients.

Replacement of general matrices with matrices of a simpler form.

Replacing complicated functions with simple functions, such as polynomials.

Replacing nonlinear problems with linear problems.

Replacing differential equations with algebraic equations.

Replacing high-order systems with low-order systems.

Replacing infinite-dimensional spaces with finite-dimensional spaces

Objectives of Numerical Analysis (Burden *et al.*, 2022)

Numerical Analysis is the branch of mathematics that studies algorithms and methods for solving mathematical problems in numerical form.
The two objectives of numerical analysis:

1. Find an approximation to the solution of a given problem
2. Determine the bound for the accuracy of the approximation

Algorithm (Burden *et al.*,2022)

Algorithm = a procedure that describes a finite or infinite sequence of steps to be performed in a specified order.

The object of the algorithm is to implement a procedure to solve a problem or approximate a solution to the problem.

Stable algorithm: small changes in the initial data produce correspondingly small changes in the final results

Conditionally stable: stable only for certain choice of initial data

Absolute and Relative Errors (Burden *et al.*, 2022; Shampine *et al.*, 1997)

Let $\|\cdot\|$ a given norm in a vector space $\mathcal{V}$. We suppose that $\boldsymbol{x}^*$ is a numerical approximation of the exact value $\boldsymbol{x}$.

Actual error: $\boldsymbol{x} - \boldsymbol{x}^*$

Absolute error: $\|\boldsymbol{x} - \boldsymbol{x}^*\|$

Relative error: $\frac{\|\boldsymbol{x}-\boldsymbol{x}^*\|}{\|\boldsymbol{x}\|}$ provided that $\boldsymbol{x} \neq \mathbf{0}_{\mathcal{V}}$

Relative error is not defined if the true value is zero. In the arithmetic of computers, relative error is the more natural concept, but absolute error may be preferable when studying quantities that are close to zero.

Characterizing Algorithms (Burden *et al.*,2022)

Suppose that E_0 denotes an error introduced at some stage in the calculations and E_n represents the magnitude of the error after n subsequent operations.

> If $E_n \approx CnE_0$, where C is a constant independent of n, then the growth of error is said to be *linear*.
>
> If $E_n \approx C^n E_0$, for some $C > 1$, then the growth of error is called *exponential*.

Linear growth of error is usually unavoidable, and when C and E_0 are small, the results are generally acceptable. Exponential growth of error should be avoided because the term C^n becomes large for even relatively small values of n. This leads to unacceptable inaccuracies, regardless of the size E_0

Computational cost (Quarteroni *et al.*, 2014)

The *computational cost* of an algorithm is the number of floating point operations that are required for its execution. Often, the speed of a computer is measured by the maximum number of floating-point operations which the computer can execute in one second (*flops*).

In general, the exact knowledge of the number of operations required by a given algorithm is not essential.

Computational cost (Quarteroni *et al.*, 2014)

It is useful to determine its order of magnitude as a function of a parameter n which is related to the problem dimension. An algorithm has

-*constant complexity* (it requires a number of operations independent of n): $O(1)$

-*linear complexity*: $O(n)$

-*polynomial complexity*: $O(n^p)$, $p \in \mathbb{N}^*$

-*exponential complexity*: $O(c^n)$, $c > 1$

-*factorial complexity*: $O(n!)$

We recall that the symbol $O(n^p)$ means *it behaves, for large n, like a constant times $O(n^p)$*.

Matix-vector product (Quarteroni *et al.*, 2014)

$$A \in \mathcal{M}_n(\mathbb{R}),\ \boldsymbol{v} \in \mathbb{R}^n \Rightarrow A\boldsymbol{v} \in \mathbb{R}^n$$

The jth component of the product $A\boldsymbol{v}$ is given by

$$(A\boldsymbol{v})_j = a_{j1}v_1 + \ldots + a_{jn}v_n = \sum_{k=1}^{n} a_{jk}v_k$$

and requires n multiplications and $(n-1)$ additions.
We need $n(2n-1)$ operations to compute all components. Thus this algorithm requires $O(n^2)$ operations, so it has a *quadratic complexity* with respect to the parameter n.

Matrix Multiplication (Quarteroni *et al.*, 2014)

$$A, B \in \mathcal{M}_n(\mathbb{R}) \Rightarrow C = A \cdot B \in \mathcal{M}_n(\mathbb{R})$$

More exactly,

$$A = (a_{ij}),\ B = (b_{ij}) \Rightarrow C = (c_{ij}),\ \forall\, i, j = \overline{1, n}$$

where

$$c_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{ij}$$

This algorithm would require $O(n^3)$ operations to compute the product of two square matrices of order n. However, there is an algorithm, due to Strassen, which requires *only* $O(n^{\log_2 7})$ operations and another, due to Winograd and Coppersmith, requiring $O(n^{2.376})$ operations.

Determinant of a square matrix (Quarteroni *et al.*, 2014)

The determinant of a square matrix is defined by the following recursive formula (*Laplace rule*):

$$\det(A) = \begin{cases} a_{11}, & \text{if } n = 1 \\ \displaystyle\sum_{j=1}^{n} a_{ij}\Delta_{ij}, & \text{for } n > 1, \forall\, i = 1, \ldots, n \end{cases}$$

where $\Delta_{ij} = (-1)^{i+j}\det(A_{ij})$ and A_{ij} is the matrix obtained by eliminating the ith row and jth column from matrix A. Moreover, $\det(A_{ij})$ is called the *cofactor* of a_{ij}.

Determinant of a square matrix (Süli and Mayers, 2003)

Assume that $n \geqslant 2$, and denote by d_n the number of arithmetic operations that are required to calculate $\det(A)$ for $A \in \mathcal{M}_n(\mathbb{R})$. For a 2×2 matrix, $\det(A) = a_{11}a_{22} - a_{12}a_{21}$ and we have $d_2 = 3$ (2 multiplications and 1 subtraction). For general case

$$d_n = n(d_{n-1} + 1) + n - 1, \text{ for } n \geqslant 3.$$

We consider $d_n = c_n n!$ and we obtain

$$c_n = c_{n-1} + \frac{2}{(n-1)!} - \frac{1}{n!}, \; n \geqslant 3, \quad c_2 = \frac{3}{2}$$

We deduce that

$$c_p = -\frac{1}{p!} + \sum_{n=0}^{p-1} \frac{1}{n!}, \; p \geqslant 3 \Rightarrow \lim_{p \to \infty} c_p = e$$

The algorithm has a factorial complexity with respect to n, $O(en!)$.

Evaluation of a polynomial (Quarteroni *et al.*, 2014)

Standard form: $P_n(x) = a_n x^n + \ldots + a_1 x + a_0$
We need $(n-1)$ multiplications to compute recursively the powers $x^2, \ldots, x^n$ by $x^i = x^{i-1} \cdot x$ and n multiplications to obtain the terms $a_i x^i$. The standard form requires n sums and $2n-1$ multiplications to evaluate P_n.

Nested multiplications

$$P_n(x) = ((\ldots(a_n x + a_{n-1})x + a_{n-2})x + \ldots + a_2)x + a_1)x + a_0$$

can be written as a recurrence relation

$$p_{new} := p_{old} \cdot x + a_i; p_{old} := p_{new}$$

Nested form requires n sums and n multiplications to evaluate P_n.

Kepler's equation (Dahlquist and Björk, 2008)

The Cartesian coordinates of a planet in elliptic orbit at time t are equal to $(ea\sin(x), ea\cos(x))$, where a is the semi-major axis, and e is the eccentricity of the ellipse. Using Kepler's laws of planetary motion it can be shown that the angle x, called the *eccentric anomaly*, satisfies Kepler's equation

$$x - e\sin(x) = M$$

where $0 < |e| < 1$, $M = \frac{2\pi}{T}$.

M- mean anomaly

T- orbital period.

Plank equation (Quarteroni *et al.*, 2014)

In order to plan a room for infrared beams we are interested in calculating the energy emitted by a black body (that is, an object capable of irradiating in all the spectrum to the ambient temperature) in the (infrared) spectrum comprised between $3\mu m$ and $14\mu m$ wavelength. The solution of this problem is obtained by computing the integral

$$E(T) = 2.39 \cdot 10^{-11} \int_{3\cdot 10^{-4}}^{14\cdot 10^{-4}} \frac{1}{x^5 (e^{\frac{1.432}{TX}} - 1)} \, dx$$

which is the Planck equation for the energy $E(T)$, where x is the wavelength (in cm) and T is the temperature (K) of the black body.

The Kepler problem (Butcher, 2008)

The problem describes the motion of a single planet about a heavy sun. The attraction of the planet on the sun is negligible and the sun will be treated as being stationary. Let (y_1, y_2) denote the rectangular coordinates centred at the sun, specifying the position of the planet. Also let (y_3, y_4) denote the components of velocity in the y_1 and y_2 directions, respectively.

$$\begin{cases} y_1' = y_3 \\ y_2' = y_4 \\ y_3' = -\frac{y_1}{(y_1^2+y_2^2)^{3/2}} \\ y_4' = -\frac{y_2}{(y_1^2+y_2^2)^{3/2}} \end{cases}$$

The solutions of this system are known to be conic sections, that is, ellipses, parabolas or hyperbolas, if we ignore the possibility that the trajectory is a straight line directed either towards or away from the sun.

The Lotka (1925)-Volterra (1926) Problem (predator-prey problem) (Butcher, 2008)

In the modelling of the two-species *predator–prey* problem, differential equation systems of the following type arise:

$$\begin{cases} u' = u(2 - v) \\ v' = v(u - 1) \\ u(0) = u_0, \; v(0) = v_0 \end{cases}$$

where the factors $(2 - v)$ and $(u - 1)$ can be generalized in various ways. The two variables represent the time-dependent populations, of which v is the population of predators which feed on prey whose population is denoted by u.

The simple pendulum (Butcher, 2008)

Formulation as a differential-algebraic equation.

Consider a small mass m attached to a light inelastic string of length l, with the other end attached to the origin of coordinates, which can swing back and forth in a vertical plane. Let y_1, measured in a rightwards direction, and y_2, measured in a downward direction, be the coordinates. Because the string is inelastic, the tension T in the string always matches other forces resolved in the direction of the string so as to guarantee that the length does not change. We denote by y_3 and y_4, respectively, the velocity components in the y_1 and y_2 directions. The motion of the pendulum is governed by the equations

$$\begin{cases} y_1' = y_3 \\ y_2' = y_4 \\ y_3' = -y_1 y_5 \\ y_4' = -y_2 y_5 + 1 \\ y_1^2 + y_2^2 = 1 \end{cases}$$

The simple pendulum (Butcher, 2008)

Formulation as a single second order equation.
Make the substitutions $y_1 = \sin(\theta)$, $y_2 = \cos(\theta)$ we obtain the well-known single-equation formulation of the simple pendulum:

$$\theta'' + \sin(\theta) = 0$$

with initial values $\theta(0) = \Theta_0$, $\theta'(0) = 0$. The period of the pendulum is given by

$$T = 4\int_0^{\frac{\pi}{2}} \frac{1}{\sqrt{1 - \sin^2\phi \sin^2\frac{\Theta_0}{2}}}\,d\phi.$$

Elastic springs (Quarteroni *et al.*, 2014)

Consider the system made of two pointwise bodies P_1 and P_2 of mass m, connected by two springs and free to move along the line joining P_1 and P_2. Let $x_i(t)$ denote the position occupied by P_i at time t for $i = 1, 2$. Then from the second law of dynamics we obtain

$$m\ddot{x}_1 = K(x_2 - x_1) - Kx_1, \; m\ddot{x}_2 = K(x_1 - x_2).$$

where K is the elasticity coefficient of both springs. We are interested in free oscillations whose corresponding solution is $x_i = a_i \sin(\omega t + \phi)$, $i = 1, 2$ with $a_i \neq 0$. In this case we find the *eigenvalues-eigenvectors* problem $A\boldsymbol{a} = \lambda \boldsymbol{a}$

$$\underbrace{\begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix}}_{A=} \underbrace{\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}}_{\boldsymbol{a}=} = \underbrace{\frac{m\omega^2}{K}}_{\lambda=} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}.$$