Numerical Analysis

Approximations and Errors

The difference between exact solution and numerical solution is that numerical solution gives the answer with some "error", because numerical solution involved an approximation.

For many engineering problems, we cannot obtain analytical solution; therefore, we cannot compute the errors associated with numerical solution. In professional practice, errors can be costly and sometimes catastrophic. If a structure or device fails, lives can be lost.

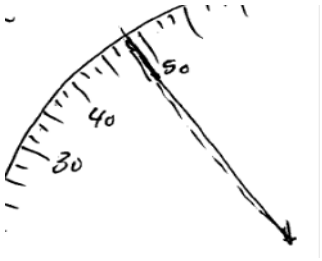
1. **Types of Errors**: There are two major types of errors:

1. **Round off** error: which is due to computers, (e.g. $\pi = 3.14160$ instead of $\pi = 3.14159253589\ldots..$

2. Truncation error is the difference between a truncated value and the actual value. A truncated quantity is represented by a numeral with a fixed number of allowed digits, with any excess digits.

   As an example of truncation error, consider the speed of light in a vacuum. The official value is 299,792,458 meters per second. In scientific (power-of-10) notation, that quantity is expressed as 2.99792458×10^8. Truncating it to two decimal places yields 2.99×10^8. The truncation error is the difference between the actual value and the truncated value, or 0.00792458×10^8. Expressed properly in scientific notation, it is 7.92458×10^5.

1. **Significant digits**: They are the number of digits that can be used with confidence. They correspond to the certain digits plus one estimated digit. The figure shows a car speedometer that reads 48.5 km/h, where (48) are significant digits and (5) is estimated.



2. **Absolute error in algebraic operations**:

1. **Absolute error in summation and subtraction**: If the absolute error (E_a), then if two numbers are added or subtracted, then the magnitude of total absolute error is equal to the sum of individual errors. i.e.
   $E_a = E_{a1} \pm E_{a2} \pm E_{a3} \pm \ldots$
2. **Absolute error in product:** If we have two numbers "A" and "B". Let the absolute error in "A" is "E_a", and absolute error in "B" is "E_b", then

Approximate value of "A" = A + E_a, and
approximate value of "B" = B + E_b
Then abs. error in product (E_{ap}) = (A + E_a) * (B + E_b) - AB
= AB + A*E_b + B*Ea + Ea*E_b – AB
≈ A*E_b + B*Ea

3. **Absolute error in division**: For the same above example, the absolute error will be:

Absolute error in division (E_{ad}) = $\frac{A+Ea}{B+E_b} - \frac{A}{B} \approx \frac{B*Ea-A*E_b}{B^2}$

1. **Error propagation**: When error is introduced in a variable, it propagates in other variables because of computations. This amount of error depends upon the mathematical or numerical operation performed. Consider the function,

$$f(x) = \frac{1}{1-x^2}$$

Let's calculate f(x) for x=0.9. Then exact value will be (5.2631579).
Let's assume that approximate value of (x) is (0.900005,i.e. an error of $5*10^{-6}$). With this value of (x), the approximate value of f(x) will be (5.2634072), so the error

will be (0.000025,i.e. an error of $25*10^{-6}$). This is called error magnification (or error propagation since the error propagate from the 6^{th} digit to the 5^{th} digit.

Under such condition, the numerical method or computation procedure is said to be (numerically unstable). To avoid this instability, the numerical process is rearranged or some other method is used.

2. Solved examples:

Example1.1- Calculate the absolute and relative error in the following cases:

a)True value $= 1 \times 10^{-6}$, Approximate value $= 0.5 \times 10^{-6}$

b)True value $= 1 \times 10^{6}$, Approximate value $= 0.99 \times 10^{6}$

Solution:

1. Absolute error $= |True\ value - Approximate\ value|$

$= 1 \times 10^{-6} - 0.5 \times 10^{-6} = 0.5 \times 10^{-6}$

Relative error(ϵ_r) $\frac{Absolute\ error}{True\ value} = \frac{0.5x10^{-6}}{1x10^{-6}} = 0.5$

Percentage relative error ($\epsilon_r\%$) $= \epsilon_r \times 100\% = 0.5 \times 100 = 50\%$

2. Absolute error $= |True\ value - Approximate\ value|$

$= 1 \times 10^{6} - 0.99 \times 10^{6} = 0.01 \times 10^{6} = 10000$ Relative error (ϵ_r) $= \frac{Absolute\ error}{True\ value} = \frac{10000}{1x10^{6}} = 0.01$

($\epsilon_r\%$) $= \epsilon_r \times 100\% = 0.01 \times 100 = 1\%$

Example1.2- f (x=0.4000) is correct to 4 significant digits, find the relative error.

Solution:

(x) is correct to 4 significant digits, this means there will be error in the (fifth) digit. The maximum value of this error will be: Et = 0.00005 (5 is the max. value of the 5^{th} digit)

Relative error (ϵ_r) $\frac{Absolute\ error}{True\ value} = \frac{0.00005}{0.4000} = 0.000125$

Example1.3-

Find the approximate maximum error in (5.43 x 27.2).

Solution:

Here we have to calculate error in product.

Let A= 5.43 and B= 27.2

The error in A is E_a = 0.005, and error in B is E_b = 0.05

∴ Product absolute error (E_{ap}) = A * E_b + B * E_a

= 5.43 * 0.05 + 27.2 * 0.005 = 0.4075

Example1.4- Determine the value of ($e^{0.5}$) correct to <u>three</u> significant digits using the expansion $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots\ldots$, if true value = 1.648720.

Solution:

No. of terms	Total value	Absolute error	Relative error (ϵ_r)
1	$e^x = e^{0.5} = 1$	1.64872 – 1 = 0.64872	$\frac{0.64872}{1.64872}$= 0.393
2	e^x= 1+x =1+0.5 =1.5	1.64872-1.5 = 0.14872	$\frac{0.14872}{1.64872}$= 0.090
3	e^x=1+x+x^2/2! =1+0.5+$(0.5)^2$/2! = 1.625	1.64872-1.625= 0.02372	$\frac{0.02372}{1.64872}$= 0.01438
4	e^x=1+x+x^2/2! +x^3/3! =1+0.5+$(0.5)^2$/2! + $(0.5)^3$/3! = 1.64583	1.64872-1.645833= 0.002887	$\frac{0.002887}{1.64872}$ ==0.001751
5	e^x = 1.6484375	1.64872-1.6484375= 0.0002825	$\frac{0.0002825}{1.64872}$= 0.0001486

∴ *five terms are needed for determine t□e value of* $e^{0.5}$ *correct for t□ree significant digits*

Example1.5: The quotient $\frac{25.4}{12.37}$ gives the result 2.05335489. Find the maximum error.

Solution: Let a=25.4, then max. absolute error E_a=0.05.

Let b=12.37, then max. absolute error E_b=0.005.

The absolute error in division is given as,

$$E_d = \frac{B * E_a - A * E_b}{b^2}$$

Putting the values in above equation,

$$\varepsilon d = (12.37 * 0.05 - 25.4 * 0.005)/(12.37)^2$$

= 0.003212

Hence the true quotient will have the value of: (2.053±0.003212)

Definition 2.4 (Normalization).

A floating-point number is said to be **normalized** *if either* $d_1 \neq 0$ *or* $d_1 = d_2 = \cdots = d_n = 0$.

Example 2.5. The following are examples of real numbers in the decimal floating point representation.

I. The real number $x = 6.238$ can be represented as $6.238 = (-1)^0 \times 0.6238 \times 10^1$, in which case, we have $s = 0$, $\beta = 10$, $e = 1$, $d_1 = 6$, $d_2 = 2$, $d_3 = 3$ and $d_4 = 8$. Note that this representation is the normalized floating-point representation.

II. The real number $x = -0.0014$ can be represented in the decimal float-point representation as $-0.0014 = (-1)^1 \times 0.0014 \times 10^1$, which is not in the normalized form. But this representation is not in the normalized form. The normalized representation is $x = (-1)^1 \times 0.14 \times 10^{-2}$. □

Definition 2.6 (Overflow and Underflow).

The exponent e *is limited to a range*

$$m < e < M. \tag{2.3}$$

During the calculation, if some computed number has an exponent $e > M$ *then we say, the memory* **overflow** *or if* $e < m$*, we say the memory* **underflow**.

Remark 2.7. In the case of overflow, computer will usually produce meaningless results or simply prints the symbol NaN, which means, the quantity obtained due to such a calculation is 'not a number'. The symbol ∞ is also denoted as NaN on some computers. The underflow is less serious because in this case, a computer will simply consider the number as zero. □

Remark 2.8. The floating-point representation (2.1) of a number has two restrictions, one is the number of digits n in the mantissa and the second is the range of e. The number n is called the **precision** or **length** of the floating point representation. □

Example 2.9. The IEEE (Institute of Electrical and Electronics Engineers) standard for floating-point arithmetic (IEEE 754) is the most widely-used standard for floating-point computation, and is followed by many hardware (CPU and FPU), including intel processors, and software implementations. Many computer languages allow or require that some or all arithmetic be carried out using IEEE 754 formats and operations. The IEEE 754 floating-point representation for a binary number x is given by [1]

$$\mathrm{fl}(x) = (-1)^s \times (1.a_1a_2\cdots a_n)_2 \times 2^e, \tag{2.4}$$

where $a_1, \cdots, a_n$ are either 1 or 0. The IEEE 754 standard always uses binary operations.

The **IEEE single precision** floating-point format uses 4 bytes (32 bits) to store a number. Out of these 32 bits, 24 are allocated for storing mantissa (one binary digit needs 1 bit storage space), 1 bit for s (sign) and remaining 8 bits for the exponent. The storage scheme is given by

$$|(\text{sign})\ b_1\ \ |\ \ (\text{exponent})\ b_2b_3\cdots b_9\ \ |\ \ (\text{mantissa})\ b_{10}b_{11}\cdots b_{32}|$$

Note here that there are only 23 bits used for mantissa. This is because, the digit 1 before the binary point in (2.4) is not stored in the memory and will be inserted at the time of calculation.

Instead of the exponent e, we store the non-negative integer $E = (b_2b_3\cdots b_9)_2$ and define $e = E - 127$. If all b_i's ($i = 2, \cdots, 9$) are zero, then $E = (0)_{10}$ and if all b_i's are 1, then $E = (255)_{10}$. In addition to this, one space corresponding to $e = 128$ (or$E$=255) is reserved for ∞ or NaN depending on whether $b_{10} = \cdots = b_{32} = 0$ or otherwise. Thus, in IEEE 754, we have $-126 \leq e \leq 127$ (note that the range of e is not from -127, because this number is reserved for those numbers not represented otherwise, see Atkinson and Han, 2004, for more details) and one memory space for NaN. The decimal number zero needs a special representation, which is stored as $E = 0$ (ie., $b_2 = \cdots = b_9 = 0$), $b_1 = 0$ and $b_{10} = \cdots = b_{32} = 0$.

[1] Note the difference between the representation given in (2.1) and here. Since, it is a binary representation, the digit before the binary point is always 1 and therefore, this information need not be stored in the computer memory at all. This is the reason why this form of representation rather than (2.1) was prefered.

In the representation (2.4), the value of s is stored in b_1, the positive integer $E = e + 127$ is stored in bits b_2 through b_9. The string of digits $a_1a_2\cdots a_{23}$ are stored in bits b_{10} through b_{32}. The leading binary digit 1 in the mantissa is not stored in the memory. However, this information is inserted into the mantissa when a floating-point number x is brought out of the memory and sent into an arithmetic operation. In the IEEE single precision storage system the overflow occurs for real numbers $|x| > x_{max}$, where

$$x_{max} = 1.11\cdots1 \times 2^{127} \approx 2^{128} \approx 3.40 \times 10^{38}.$$

The **IEEE double precision** floating-point representation of a number has a precision of 53 binary digits and the exponent e is limited by $-1023 \le e \le 1023$. □

2.2 Chopping and Rounding a Number

Any real number x can be represented exactly as

$$x = (-1)^s \times (.d_1d_2\cdots d_nd_{n+1}\cdots)_\beta \times \beta^e, \tag{2.5}$$

with $d_1 \neq 0$ or $d_2 = d_3 = \cdots = 0$, $s = 0$ or 1, and e satisfies (2.3), for which the floating-point form (2.1) is an approximate representation. Let us denote this approximation of x by $\text{fl}(x)$. There are two ways to produce $\text{fl}(x)$ from x as defined below.

Definition 2.10 (Chopped and Rounded Numbers).

The **chopped** *machine approximation of x is given by*

$$\text{fl}(x) = (-1)^s \times (.d_1d_2\cdots d_n)_\beta \times \beta^e. \tag{2.6}$$

The **rounded** *machine approximation of x is given by*

$$\text{fl}(x) = \begin{cases} (-1)^s \times (.d_1d_2\cdots d_n)_\beta \times \beta^e & , \ 0 \le d_{n+1} < \frac{\beta}{2} \\ (-1)^s \times (.d_1d_2\cdots (d_n+1))_\beta \times \beta^e & , \ \frac{\beta}{2} \le d_{n+1} < \beta \end{cases} \tag{2.7}$$

2.3 Different Type of Errors

The approximate representation of a real number obviously differs from the actual number, whose difference is called an **error**.

Definition 2.11 (Errors).

The **error** *in a computed quantity is defined as*

$$\textbf{Error = True Value - Approximate Value}.$$

The **absolute error** *is the absolute value of the error defined above. The* **relative error** *is a measure of the error in relation to the size of the true value as given by*

$$\textbf{Relative Error} = \frac{\textbf{Error}}{\textbf{True Value}}$$

The **percentage error** *is defined as 100 times the relative error.*

The term **truncation error** *is used to denote error, which result from approximating a smooth function by truncating its Taylor series representation to a finite number of terms.*

Example 2.12. A second degree polynomial approximation to

$$f(x) = \sqrt{x+1}, \quad x \in [0,1]$$

using the Taylor series expansion about $x = 0$ is given by

$$f(x) \approx 1 + \frac{x}{2} - \frac{x^2}{8} + \frac{x^3}{16(\sqrt{1+\xi})^5}.$$

Therefore, the truncation error is given by $x^3/(16(\sqrt{1+\xi})^5)$. □

Remark 2.13. Let x_A be the approximation of the real number x. Then

$$E(x_A) := \text{Error}(x_A) = x - x_A. \tag{2.8}$$
$$E_a(x_A) := \text{Absolute Error}(x_A) = |E(x_A)| \tag{2.9}$$
$$E_r(x_A) := \text{Relative Error}(x_A) = \frac{E(x_A)}{x} \tag{2.10}$$

□

Example 2.14. If we denote the relative error in $\text{fl}(x)$ as $\epsilon > 0$, then we have

$$\text{fl}(x) = (1 - \epsilon)x, \tag{2.11}$$

where x is a real number. □

2.4 Loss of Significant Digits

In place of relative error, we often use the concept of **significant digits**.

Definition 2.15 (Significant Digits).
If x_A is an approximation to x, then we say that x_A approximates x to r **significant β-digits** *if*

$$|x - x_A| \leq \frac{1}{2}\beta^{s-r+1} \tag{2.12}$$

with s the largest integer such that $\beta^s \leq |x|$.

Example 2.16. (a) For $x = 1/3$, the approximate number $x_A = 0.333$ has three significant digits, since $|x - x_A| \approx .00033 < 0.0005 = 0.5 \times 10^{-3}$. But $10^{-1} < 0.333\cdots = x$. Therefore, in this case $s = -1$ and hence $r = 3$.
(b) For $x = 0.02138$, the approximate number $x_A = .02144$ has the aboslute error $|x - x_A| \approx .00006 < 0.0005 = 0.5 \times 10^{-3}$. But $10^{-2} < 0.02138 = x$. Therefore, in this case $s = -2$ and therefore, the number x_A has only two significant digits, but not three, with respect to x. □

Remark 2.17. In a very simple way, the number of leading non-zero digits of x_A that are correct relative to the corresponding digits in the true value x is called the **number of significant digits** in x_A. □

The role of significant digits in the numerical calculation is very important in the sense that the loss of significant digits may result in drastic amplification of the relative error.

Example 2.18. Let us consider two real numbers

$$x = 7.6545428 = 0.76545428 \times 10^1, \quad y = 7.6544201 = 0.76544201 \times 10^1.$$

The numbers

$$x_A = 7.6545421 = 0.76545421 \times 10^1, \quad y_A = 7.6544200 = 0.76544200 \times 10^1$$

are approximation to x and y, correct to six and seven significant digits, respectively. In eight-digit floating-point arithmetic,

$$z_A = x_A - y_A = 0.12210000 \times 10^{-3}$$

is the exact difference between x_A and y_A and

$$z = x - y = 0.12270000 \times 10^{-3}$$

is the exact difference between x and y. Therefore,

$$z - z_A = 0.6 \times 10^{-6} < 0.5 \times 10^{-5}$$

and hence z_A has only three significant digits with respect to z as $10^{-3} < z = 0.0001227$. Thus, we started with two approximate numbers x_A and y_A which are correct to six and seven significant digits with respect to x and y respectively, but their difference z_A has only three significant digits with respect to z and hence, there is a loss of significant digits in the process of subtraction. A simple calculation shows that

$$E_r(z_A) \approx 53736 \times E_r(x_A),$$

and similarly for y. Loss of significant digits is therefore dangerous if we wish to minimize the relative error. The loss of significant digits in the process of calculation is refered to as **Loss of Significance**. □

Example 2.19. Consider the function $f(x) = x(\sqrt{x+1} - \sqrt{x})$. On a six-digit decimal calculator, we have $f(100000) = 100$ whereas the true value is 158.113. This makes a drastic error in the calculation. This is the result of the loss of significant digits, which can be seen from the fact that as x increases, the terms $\sqrt{x+1}$ and $\sqrt{x}$ comes closer to each other and therefore loss of significant error in their computed value increases.

Such loss can often be avoided by rewritting the given expression (whenever possible) in such a way that subtraction is avoided. For instance, the defintion of $f(x)$ given in this example can be rewritten as

$$f(x) = \frac{x}{\sqrt{x+1} + \sqrt{x}}.$$

With this new definition, we see that on a six-digit calculator, we have $f(100000) = 158.114000$. □

Example 2.20. When the function $f(x) = 1 - \cos x$ is evaluated in six-decimal-digit arithmetic (say). Since $\cos x \approx 1$ for x near zero, there will be loss of significant digits for x near zero. So, we have to use an alternative formula for $f(x)$ such as

$$f(x) = 1 - \cos x = \frac{1 - \cos^2 x}{1 + \cos x} = \frac{\sin^2 x}{1 + \cos x}$$

which can be evaluated quite accurately for small x. We can also use Taylor's expansion to get an alternative expression for $f(x)$ as

$$f(x) = \frac{x^2}{2} - \frac{x^4}{24} + \cdots = \sum_{n=1}^{2} (-1)^n \frac{x^{2n}}{2n!} + R(x),$$

where

$$R(x) = \frac{x^{2(n+1)}}{2(n+1)!} f^{(2(n+1))}(\xi) = -\frac{x^6}{6!} \cos \xi$$

with ξ very close to zero. □

2.5 Propagation of Error

Once an error is committed, it affects subsequent results as this error propagates through subsequent calculations. We first study how the results are affected by using approximate numbers instead of actual numbers and then will take up function evaluation.

Let x_A and y_A denote the numbers used in the calculation, and let x_T and y_T be the corresponding true values. We will now see how error propagates with the four basic arithmetic operations.

Propagated error in addition and subtraction

Let $x_T = x_A + \epsilon$ and $y_T = y_A + \eta$ are positive numbers. The relative error $E_r(x_A \pm y_A)$ is given by

$$E_r(x_A \pm y_A) = \frac{(x_T \pm y_T) - (x_A \pm y_A)}{x_T \pm y_T} = \frac{(x_T \pm y_T) - (x_T - \epsilon \pm (y_T - \eta))}{x_T \pm y_T} = \frac{\epsilon \pm \eta}{x_T \pm y_T}.$$

This shows that relative error propagate slowly with addition, whereas amplifies drastically with subtraction when $x_T \approx y_T$ as we have witnessed in examples 2.18 and 2.19.

Propagated error in multiplication

The relative error $E_r(x_A \times y_A)$ is given by

$$E_r(x_A \times y_A) = \frac{(x_T \times y_T) - (x_A \times y_A)}{x_T \times y_T} = \frac{(x_T \times y_T) - ((x_T - \epsilon) \times (y_T - \eta))}{x_T \times y_T}$$
$$= \frac{\eta x_T + \epsilon y_T - \epsilon\eta}{x_T \times y_T} = \frac{\epsilon}{x_T} + \frac{\eta}{y_T} - \left(\frac{\epsilon}{x_T}\right)\left(\frac{\eta}{y_T}\right) = E_r(x_A) + E_r(Y_A) - E_r(x_A)E_r(Y_A).$$

This shows that relative error propagate slowly with multiplication.

Propagated error in division

The relative error $E_r(x_A/y_A)$ is given by

$$E_r(x_A/y_A) = \frac{(x_T/y_T) - (x_A/y_A)}{x_T/y_T} = \frac{(x_T/y_T) - ((x_T - \epsilon)/(y_T - \eta))}{x_T/y_T}$$
$$= \frac{x_T(y_T - \eta) - y_T(x_T - \epsilon)}{x_T(y_T - \eta)} = \frac{y_T\epsilon - x_T\eta}{x_T(y_T - \eta)} = \frac{y_T}{y_T - \eta}(E_r(x_A) - E_r(y_A))$$
$$= \frac{1}{1 - E_r(Y_A)}(E_r(x_A) - E_r(y_A)).$$

This shows that relative error propagate slowly with division, unless $E_r(Y_A) \approx 1$. But this is very unlikely because we always expect the error to be very small, ie., very close to zero in which case the right hand side is approximately equal to $E_r(x_A) - E_r(Y_A)$.

Total calculation error

When using floating-point arithmetic on a computer, the calculation of $x_A\omega y_A$ (here ω denotes one of the basic arithmetic operation '+', '−', '×' and '/') involves an additional rounding or chopping error. The computed value of $x_A\omega y_A$ will involve the propagated error plus a rounding or chopping error. To be more precise, let $\hat{\omega}$ denotes the complete operation as carried out on the computer, including any rounding or chopping. Then the **total error** is given by

$$(x_T\omega y_T) - (x_A\hat{\omega}y_A) = [(x_T\omega y_T) - (x_A\omega y_A)] + [(x_A\omega y_A) - (x_A\hat{\omega}y_A)].$$

The first term on the right is the propagated error and the second term is the error due to rounding or chopping the number obtained from the calculation $x_A\omega y_A$.a

Propagated error in function evaluation

Consider evaluating $f(x)$ at the approximate value x_A rather than at x. Then consider how well does $f(x_A)$ approximate $f(x)$? Using the mean-value theorem, we get

$$f(x) - f(x_A) = f'(\xi)(x - x_A),$$

where ξ is an unknown point between x and x_A. The relative error of $f(x)$ with respect to $f(x_A)$ is given by

$$E_r(f(x)) = \frac{f'(\xi)}{f(x)}(x - x_A) = \frac{f'(\xi)}{f(x)}xE_r(x). \tag{2.13}$$

Since x_A and x are assumed to be very close to each other and ξ lies between x and x_A, we make the approximation

$$f(x) - f(x_A) \approx f'(x)(x - x_A) \approx f'(x_A)(x - x_A).$$

Definition 2.21 (Condition number of a function).

The **condition number** *of a funtion f at a point $x = c$ is given by*

$$\left|\frac{f'(c)}{f(c)}c\right| \tag{2.14}$$

Example 2.22. Consider the function $f(x) = \sqrt{x}$, for all $x \in [0, \infty)$. Then

$$f'(x) = \frac{1}{2\sqrt{x}}, \text{ for all } x \in [0, \infty).$$

The condition number of f is

$$\left| \frac{f'(x)}{f(x)} x \right| = \frac{1}{2}, \text{ for all } x \in [0, \infty).$$

From (2.13) we see that taking square roots is a **well-conditioned** process since it actually reduces the relative error. □

Example 2.23. Consider the function

$$f(x) = \frac{10}{1 - x^2}, \text{ for all } x \in \mathbb{R}.$$

Then $f'(x) = 20x/(1 - x^2)^2$, so that

$$\left| \frac{f'(x)}{f(x)} x \right| = \left| \frac{(20x/(1 - x^2)^2)x}{10/(1 - x^2)} \right| = \frac{2x^2}{|1 - x^2|}$$

and this number can be quite large for $|x|$ near 1. Thus, for x near 1 or -1, this function is **ill-conditioned**, as it magnifies the relative error. □

Definition 2.24 (Stability and Instability in Evaluating a Function).

Suppose there are n steps to evaluate a function $f(x)$. Then the total process of evaluating this function is said to have **instability** *if atleast one step is ill-conditioned. If all the steps are well-conditioned, then the process is said to be* **stable**.

Example 2.25. Consider the function

$$f(x) = \sqrt{x + 1} - \sqrt{x}, \text{ for all } x \in [0, \infty).$$

For a sufficiently large x, the condition number of this function is

$$\left| \frac{f'(x)}{f(x)} x \right| = \frac{1}{2} \left| \frac{1/\sqrt{x + 1} - 1/\sqrt{x}x}{\sqrt{x + 1} - \sqrt{x}} \right| = \frac{1}{2} \frac{x}{\sqrt{x + 1}\sqrt{x}} \approx \frac{1}{2},$$

which is quite good. But, if we calculate $f(12345)$ in six digit rounding arithmetic, we find

$$f(12345) = \sqrt{12346} - \sqrt{12345} = 111.113 - 111.108 = 0.005,$$

while, actually, $f(12345) = 0.00450003262627751 \cdots$. The calculated answer has 10% error.

Let us analyze the computational process. It consists of the following four computational steps:

$$x_0 := 12345, \quad x_1 := x_0 + 1, \quad x_2 := \sqrt{x_1}, \quad x_3 := \sqrt{x_0}, \quad x_4 := x_2 - x_3.$$

Now consider the last two step where we already computed x_2 and now going to compute x_3 and finally evaluate the function

$$f_3(t) = x_2 - t.$$

At this step, the condition number for f_3 is given by

$$\left| \frac{f'(t)}{f(t)} t \right| = \left| \frac{t}{x_2 - t} \right|.$$

Thus, f is ill-conditioned when t approaches x_2. For instance, for $t \approx 111.11$, while $x_2 - t \approx 0.005$, the condition number for f_3 is approximately $22,222$ or more than 40,000 times as big as the condition number of f itself. Therefore, the above process of evaluating the function $f(x)$ is **unstable**.

Let us rewrite the same function $f(x)$ as

$$f(x) = \frac{1}{\sqrt{x+1}+\sqrt{x}}.$$

In six digit rounding arithmetic, this gives

$$f(12345) = \frac{1}{\sqrt{12346}+\sqrt{12345}} = \frac{1}{222.221} = 0.0045002,$$

which is in error by only 0.0003%. The computational process is

$$x_0 := 12345, \quad x_1 := x_0 + 1, \quad x_2 := \sqrt{x_1}, \quad x_3 := \sqrt{x_0}, \quad x_4 := x_2 + x_3, \quad x_5 := 1/x_4.$$

It is easy to verify that the condition number of each of the above steps is well-conditioned. For instance, the last step defines $f_3(t) = 1/(x_2 + t)$, and the condition number of this function is approximately,

$$\left|\frac{f'(x)}{f(x)}x\right| = \left|\frac{t}{x_2+t}\right| \approx \frac{1}{2}$$

for t sufficiently close to x_2. Therefore, this process of evaluating $f(x)$ is stable. □

Exercise 2

I. **Floating-Point Representation**

1. Write the storage scheme for the IEEE double precision floating-point representation of a real number with the precision of 53 binary digits. Find the overflow limit (in binary numbers) in this case.
2. In a binary representation, if 2 bytes (ie., $2 \times 8 = 16$ bits) are used to represent a floating-point number with 8 bits used for the exponent. Then, as of IEEE 754 storage format, find the largest binary number that can be represented.

II. **Errors**

3. *The* **machine epsilon** *(also called* **unit round***) of a computer is the smallest positive floating-point number δ such that $fl(1+\delta) > 1$. Thus, for any floating-point number $\hat{\delta} < \delta$, we have $fl(1+\hat{\delta}) = 1$, and $1+\hat{\delta}$ and 1 are identical within the computer's arithmetic.*

   For rounded arithmetic on a binary machine, show that $\delta = 2^{-n}$ is the machine epsilon, where n is the number of digits in the mantissa.
4. If $\text{fl}(x)$ is the machine approximated number of a real number x and ϵ is the corresponding relative error, then show that $\text{fl}(x) = (1-\epsilon)x$.
5. Let x, y and z are the given machine approximated numbers. Show that the relative error in computing $x(y+z)$ is $\epsilon_1 + \epsilon_2 - \epsilon_1\epsilon_2$, where $\epsilon_1 = E_r(\text{fl}(y+z))$ and $\epsilon_2 = E_r(\text{fl}(x\text{fl}(y+z)))$.
6. If the relative error of $\text{fl}(x)$ is ϵ, then show that

$$|\epsilon| \le \beta^{-n+1} \quad \text{(for chopped fl}(x)\text{)}, \quad |\epsilon| \le \frac{1}{2}\beta^{-n+1} \quad \text{(for rounded fl}(x)\text{)},$$

   where β is the radix and n is the number of digits in the machine approximated number.
7. Consider evaluating the integral $I_n = \displaystyle\int_0^1 \frac{x^n}{x+5}dx$ for $n = 0, 1, \cdots, 20$. This can be carried out in two iterative process, namely, (i) $I_n = \frac{1}{n} - 5I_{n-1}$, $I_0 = ln(6/5)$ (called forward iteration) and (ii) $I_{n-1} = \frac{1}{5n} - \frac{1}{5}I_n$, $I_{20} = 7.997522840 \times 10^{-3}$ (called backward iteration). Compute I_n for $n = 0, 1, 2, \cdots, 20$ using both iterative and show that backward iteration gives correct results, whereas forward iteration tends to increase error and gives entirely wrong results. Give reason for why this happens.
8. Find the truncation error around $x = 0$ for the following functions
   (a) $f(x) = \sin x$, (b) $f(x) = \cos x$.
9. Let $x_A = 3.14$ and $y_A = 2.651$ be correctly rounded from x_T and y_T, to the number of decimal digits shown. Find the smallest interval that contains
   (i) x_T, (ii) y_T, (iii) $x_T + y_T$, (iv) $x_T - y_T$, (v) $x_T \times y_T$ and (vi) x_T/y_T.
10. A missile leaves the ground with an initial velocity $\boldsymbol{v}$ forming an angle ϕ with the vertical. The maximum desired altitude is αR where R is the radius of the earth. The laws of mechanics can be used to deduce the relation between the maximum altitude α and the initial angle ϕ, which is given by

$$\sin\phi = (1+\alpha)\sqrt{1 - \frac{\alpha}{1+\alpha}\left(\frac{|\boldsymbol{v}_e|}{|\boldsymbol{v}|}\right)^2},$$

    where $\boldsymbol{v}_e$ = the escape velocity of the missile. It is desired to fire the missile with an angle ϕ and $|\boldsymbol{v}_e|/|\boldsymbol{v}| = 2$ so that the maximum altitude reached by the missile is $0.25R$ (ie., $\alpha = 0.25$). If the maximum altitude reached is within an accuracy of $\pm 2\%$, then determine the range of values of ϕ. [**Hint:** Treat $\sin\phi$ as a function of α and use mean-value theorem]

III. **Loss of Significant Digits and Propagation of Error**

11. For the following numbers x and their corresponding approximations x_A, find the number of significant digits in x_A with respect to x. (a) $x = 451.01$, $x_A = 451.023$,
    (b) $x = -0.04518$, $x_A = -0.045113$, (c) $x = 23.4604$, $x_A = 23.4213$.

12. Show that the function $f(x) = \dfrac{1-\cos x}{x^2}$ leads to unstable computation when $x \approx 0$. Rewrite this function to avoid loss-of-significance when $x \approx 0$. Further check the stablity of $f(x)$ in the equivalent definition of this function in avoiding loss-of-significance error.
13. Let x_A and y_A, the approximation to x and y, respectively, be such that the relative errors $E_r(x)$ and $E_r(y)$ are very much smaller than 1. Then show that (i) $E_r(xy) \approx E_r(x) + E_r(y)$ and (ii) $E_r(x/y) \approx E_r(x) - E_r(y)$. (This shows that relative errors propagate slowly with multiplication and division).
14. The ideal gas law is given by $PV = nRT$, where R is a gas constant given (in MKS system) by $R = 8.3143 + \epsilon$, with $|\epsilon| \leq 0.12 \times 10^{-2}$. By taking $P = V = n = 1$, find a bound for the relative error in computing the temperature T.
15. Find the condition number for the following functions (a) $f(x) = x^2$, (b) $f(x) = \pi^x$, (c) $f(x) = b^x$.
16. Given a value of $x_A = 2.5$ with an error of 0.01. Estimate the resulting error in the function $f(x) = x^3$.
17. Compute and interpret (find whether the funtions are well or ill-conditioned) the condition number for (i) $f(x) = \tan x$, at $x = \frac{\pi}{2} + 0.1\left(\frac{\pi}{2}\right)$. (ii) $f(x) = \tan x$, at $x = \frac{\pi}{2} + 0.01\left(\frac{\pi}{2}\right)$.
18. Let $f(x) = (x-1)(x-2)\cdots(x-8)$. Estimate $f(1+10^{-4})$ using mean-value theorem with $x_T = 1$ and $x_A = 1 + 10^{-4}$.

IV. **Miscellaneous**

19. **Big-oh:** *If $f(h)$ and $g(h)$ are two functions of h, then we say that*

$$f(h) = O(g(h)), \quad \textit{as } h \to 0$$

if there is some constant C such that

$$\left|\frac{f(h)}{g(h)}\right| < C$$

for all h sufficiently small, or equivalently, if we can bound

$$|f(h)| < C|g(h)|$$

for all h sufficiently small. Intuitively, this means that $f(h)$ decays to zero at least as fast as the function $g(h)$.

Little-oh: *We say that*

$$f(h) = o(g(h)), \quad \textit{as } h \to 0 \quad \textit{if} \quad \left|\frac{f(h)}{g(h)}\right| \to 0, \quad \textit{as } h \to 0.$$

Note that this definition is stronger than the "big-oh" statement and means that $f(h)$ decays to zero faster than $g(h)$.
(a) If $f(h) = o(g(h))$, then show that $f(h) = O(g(h))$.
(b) Give an example to show that the converse is not true.
(c) What is meant by $f(h) = o(1)$ and $f(h) = O(1)$?
(d) Give an example of $f(h)$ and $g(h)$ such that $f(h)$ is much bigger than $g(h)$, but still

$$f(h) = O(g(h)) \text{ as } h \to 0.$$

20. Assume that $f(h) = p(h) + O(h^n)$ and $g(h) = q(h) + O(h^m)$, for some positive integers n and m. Find the order of approximation of their sum, ie., find the largest integer r such that

$$f(h) + g(h) = p(h) + q(h) + O(h^r).$$

3

Linear Systems

The most general form of a linear system is

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\cdots \\ &\cdots \\ &\cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned} \tag{3.1}$$

In the matrix notation, we can write this as

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

where A is an $n \times n$ matrix with entries a_{ij}, $\boldsymbol{b} = (b_1, \cdots, b_n)^T$ and $\boldsymbol{x} = (x_1, \cdots, x_n)^T$ are n-dimensional vectors.

Theorem 3.1. *Let n be a positive integer, and let A be given as in (3.1). Then the following statements are equivalent*

I. $\det(A) \neq 0$

II. For each right hand side $\boldsymbol{b}$, the system (3.1) has unique solution $\boldsymbol{x}$.

III. For $\boldsymbol{b} = 0$, the only solution for the system (3.1) is the zero solution.

3.1 Gaussian Elimination

Let us introduce the **Gaussian Elimination** method for $n = 3$. The method for a general $n \times n$ system is similar.

Consider the 3×3 system

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \qquad (E1) \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \qquad (E2) \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3 \qquad (E3) \end{aligned} \tag{3.2}$$

Step 1: Assume that $a_{11} \neq 0$ (otherwise interchange the row for which the coefficient of x_1 is non-zero). Let us eliminate x_1 from (E2) and (E3). For this define

$$m_{21} = \frac{a_{21}}{a_{11}}, \quad m_{31} = \frac{a_{31}}{a_{11}}.$$

Multiply (E1) with m_{21} and subtract with (E2), and multiply (E1) with m_{31} and subtract with (E3) to give

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \qquad (E1)\\ a_{22}^{(2)}x_2 + a_{23}^{(2)}x_3 &= b_2^{(2)} \qquad (E2)\\ a_{32}^{(2)}x_2 + a_{33}^{(2)}x_3 &= b_3^{(2)} \qquad (E3) \end{aligned}$$

The coefficients $a_{ij}^{(2)}$ are defined by

$$\begin{aligned} a_{ij}^{(2)} &= a_{ij} - m_{i1}a_{1j}, \quad i,j = 2,3\\ b_i^{(2)} &= b_i - m_{i1}b_1, \quad i = 2,3 \end{aligned}$$

Step 2: Assume that $a_{22}^{(2)} \neq 0$ and eliminate x_2 from (E3). Define

$$m_{32} = \frac{a_{32}^{(2)}}{a_{22}^{(2)}}.$$

Subtract m_{32} times (E2) from (E3) to get

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \qquad (E1)\\ a_{22}^{(2)}x_2 + a_{23}^{(2)}x_3 &= b_2^{(2)} \qquad (E2)\\ a_{33}^{(3)}x_3 &= b_3^{(3)} \qquad (E3) \end{aligned}$$

The new coefficients are defined by

$$a_{33}^{(3)} = a_{33}^{(2)} - m_{32}a_{23}^{(2)}, \quad b_3^{(3)} = b_3^{(2)} - m_{32}b_2^{(2)}.$$

Step 3: Using back substitution to solve successively for x_3, x_2 and x_1, we get

$$\begin{aligned} x_3 &= \frac{b_3^{(3)}}{a_{33}^{(3)}}\\ x_2 &= \frac{b_2^{(2)} - a_{23}^{(2)}x_3}{a_{22}^{(2)}}\\ x_1 &= \frac{b_1 - a_{12}x_2 - a_{13}x_3}{a_{11}} \end{aligned} \tag{3.3}$$

The algorithm for $n = 3$ is easily extended to a general $n \times n$ non-singular linear system.

Gaussian elimination method is a direct method which solves the linear system exactly. However, sometime, this method fail to give the correct solution as illustrated in the following example.

Example 3.2. When we solve the linear system

$$\begin{aligned} 6x_1 + 2x_2 + 2x_n &= -2\\ 2x_1 + \frac{2}{3}x_2 + \frac{1}{3}x_n &= 1\\ x_1 + 2x_2 - x_n &= 0 \end{aligned}$$

Let us solve this system using Gaussian elimination method on a computer using a floating-point representation with four digits in the mantissa and all operations will be rounded.

The given system is

$$\begin{aligned} 6.000x_1 + 2.000x_2 + 2.000x_n &= -2.000\\ 2.000x_1 + 0.6667x_2 + 0.3333x_n &= 1.000\\ 1.000x_1 + 2.000x_2 - 1.000x_n &= 0.0000 \end{aligned}$$

After eliminating x_1 from the second and third equations, we get (with $m_{21} = 0.3333$, $m_{31} = 0.1667$)

$$\begin{aligned} 6.000x_1 + 2.000x_2 + 2.000x_n &= -2.000\\ 0.000x_1 + 0.0001x_2 - 0.3333x_n &= 1.667\\ 0.000x_1 + 1.667x_2 - 1.333x_n &= 0.3334 \end{aligned} \tag{3.4}$$

After eliminating x_2 from the third equation, we get (with $m_{32} = 16670$)

$$\begin{aligned} 6.000x_1 + 2.000x_2 + 2.000x_n &= -2.000 \\ 0.000x_1 + 0.0001x_2 - 0.3333x_n &= 1.667 \\ 0.000x_1 + 0.0000x_2 + 5555x_n &= -27790 \end{aligned}$$

Using back substitution, we get $x_1 = 1.335$, $x_2 = 0$ and $x_3 = -5.003$, whereas the actual solution is $x_1 = 2.6$, $x_2 = -3.8$ and $x_3 = -5$. The difficulty with this elimination process is that in (4.4), the element in row 2, column 2 should have been zero, but rounding error prevented it and makes the relative error very large. To avoid this, interchange row 2 and 3 in (4.4) and then continue the elimination. The final system is (with $m_{32} = 0.00005999$)

$$\begin{aligned} 6.000x_1 + 2.000x_2 + 2.000x_n &= -2.000 \\ 0.000x_1 + 1.667x_2 - 1.333x_n &= 0.3334 \\ 0.000x_1 + 0.0000x_2 - 0.3332x_n &= 1.667 \end{aligned}$$

with back substitution, we obtain the approximate solution as $x_1 = 2.602$, $x_2 = -3.801$ and D$x_3 = -5.003$. □

Partial Pivoting To avoid the problem presented by the above example, we use the following strategy. At step k, calculate

$$c = \max_{k \le i \le n} |a_{ik}^{(k)}| \tag{3.5}$$

This is the maximum size of the elements in column k of the coefficient matrix of step k, beginning at row k and going downward. If the element $|a_{kk}^{(k)}| < c$, then interchange (Ek) with one of the following equations, to obtain a new equation (Ek) in which $|a_{kk}^{(k)}| = c$. This strategy makes $a_{kk}^{(k)}$ as far away from zero as possible. The element $a_{kk}^{(k)}$ is called the **pivot element** for step k of the elimination, and the process described in this paragraph is called **partial pivoting** or more simply, pivoting.

Operations Count It is important to know the length of a computation and for that reason, we count the number of arithmetic operations involved in Gaussian elimination. Let us divide the count into three parts.

I. **The elimination step.** We now count the additions/subtractions, multiplications and divisions in going from the given system to the triangular system.

Step	Additions/Subtractions	Multiplications	Divisions
1	$(n-1)^2$	$(n-1)^2$	$n-1$
2	$(n-2)^2$	$(n-2)^2$	$n-2$
.	.	.	.
.	.	.	.
.	.	.	.
$n-1$	1	1	1
Total	$\frac{n(n-1)(2n-1)}{6}$	$\frac{n(n-1)(2n-1)}{6}$	$\frac{n(n-1)}{2}$

Here we use the formula

$$\sum_{j=1}^{p} j = \frac{p(p+1)}{2}, \ \sum_{j=1}^{p} j^2 = \frac{p(p+1)(2p+1)}{6}, p \ge 1.$$

II. **Modification of the right side** Proceeding as before, we get
Addition/Subtraction $= (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$
Multiplication/Division $= (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$

III. **The back substitution** Addition/Subtraction $= (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$
Multiplication/Division $= n + (n-1) + \cdots + 1 = \frac{n(n+1)}{2}$

Total number of operations in obtaining x is
Addition/Subtraction $= \frac{n(n-1)(2n-1)}{6} + \frac{n(n-1)}{2} + \frac{n(n-1)}{2} = \frac{n(n-1)(2n+5)}{6}$
Multiplication/Division $= \frac{n(n^2+3n-1)}{3}$

Even if we take only multiplication and division into consideration, we see that for large value of n, the operation count required for Gaussian elimination is about $\frac{1}{3}n^3$. This means that as n doubled, the cost of solving the linear system goes up by a factor of 8. In addition, most of the cost of Gaussian elimination is in the elimination step. For elimination, we have
Multiplication/Division $= \frac{n(n-1)(2n-1)}{6} + \frac{n(n-1)}{2} = \frac{1}{3}(n^3 - n) = \frac{1}{3}n^3(1 - 1/n^2) \approx \frac{1}{3}n^3$,
whereas the remaining steps counts only
Multiplication/Division $= \frac{n(n-1)}{2} + \frac{n(n+1)}{2} = n^2$
Hence, once the elimination part is completed, it is much less expensive to solve the linear system.

3.2 LU Factorization Method

Let $A\boldsymbol{x} = \boldsymbol{b}$ denote the system to be solved with A the $n \times n$ coefficient matrix. In the Gaussian elimination, the linear system was reduced to the upper triangular system $U\boldsymbol{x} = \boldsymbol{g}$ with

$$U = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ . & & \cdots & . \\ . & & \cdots & . \\ . & & \cdots & . \\ 0 & \cdots & 0 & u_{nn} \end{bmatrix}$$

and $u_{ij} = a_{ij}^{(i)}$. Introduce an auxiliary lower triangular matrix L based on the multipliers m_{ij} as

$$L = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ m_{21} & 1 & \cdots & 0 \\ . & & \cdots & . \\ . & & \cdots & . \\ . & & \cdots & . \\ m_{n1} & \cdots & m_{nn-1} & 1 \end{bmatrix}$$

The relationship of the matrices L and U to the original matrix A is given by the following theorem.

Theorem 3.3. *Let A be a non-singular matrix, and let L and U be defined as above. Then if U is produced without pivoting as in the Gaussian elimimation, then*

$$LU = A$$

and this is called the LU factorization of A.

LU factorization leads to another perspective on Gaussian elimination. Since $LU = A$, the linear system $A\boldsymbol{x} = \boldsymbol{b}$ can be re-written as

$$LU\boldsymbol{x} = \boldsymbol{b}.$$

And this is equivalent to solving the two systems

$$L\boldsymbol{g} = \boldsymbol{b}, \quad U\boldsymbol{x} = \boldsymbol{g} \tag{3.6}$$

The first system is the lower tirangular system

$$\begin{aligned} g_1 &= b_1 \\ m_{21}g_1 + g_2 &= b_2 \\ &. \\ &. \\ &. \\ m_{n1}g_1 + m_{n2}g_2 + \cdots + m_{nn-1}g_{n-1} + g_n &= b_n \end{aligned}$$

Once $\boldsymbol{g}$ is obtained by forward substitution from this system the upper triangular system $U\boldsymbol{x} = \boldsymbol{g}$ can be solved using back substitution. Thus once the factorization $A = LU$ is done, the solution of the linear system $A\boldsymbol{x} = \boldsymbol{b}$ is reduced to solving two triangular systems where the computational cost is reduced drastically in the situation when the system is to be solved for a fixed A but for various $\boldsymbol{b}$.

Rather than constructing L and U by using the elimination steps, it is possible to solve directly for these matrices. Let us illustrate the direct computation of L and U in the case of $n = 3$. Write $A = LU$ as

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ m_{21} & 1 & 0 \\ m_{31} & m_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \tag{3.7}$$

The right hand matrix multiplication implies

$$\begin{aligned} a_{11} = u_{11}, a_{12} = u_{12}, a_{13} = u_{13}, \\ a_{21} = m_{21}u_{11}, a_{31} = m_{31}u_{11}. \end{aligned} \tag{3.8}$$

These gives first column of L and the first row of U. Next multiply row 2 of L times columns 2 and 3 of U, to obtain

$$a_{22} = m_{21}u_{12} + u_{22}, \quad a_{23} = m_{21}u_{13} + u_{23} \tag{3.9}$$

These can be solved for u_{22} and u_{23}. Next multiply row 3 of L to obtain

$$m_{31}u_{12} + m_{32}u_{22} = a_{32}, \quad m_{31}u_{13} + m_{32}u_{23} + u_{33} = a_{33} \tag{3.10}$$

These equations yield values for m_{32} and u_{33}, completing the construction of L and U. In this process, we must have $u_{11} \neq 0$, $u_{22} \neq 0$ in order to solve for L.

Note that in general the diagonal elements of L need not be 1. The above procedure of LU decomposition is called **Doolittle's method**.

Example 3.4. Let

$$A = \begin{bmatrix} 1 & 1 & -1 \\ 1 & 2 & -2 \\ -2 & 1 & 1 \end{bmatrix}$$

Using (3.8), we get

$$u_{11} = 1, \ u_{12} = 1, \ \ u_{13} = -1, \ \ m_{21} = \frac{a_{21}}{u_{11}} = 1, m_{31} = \frac{a_{31}}{u_{11}} = -2$$

Using (3.9) and (3.10),

$$\begin{aligned} u_{22} &= a_{22} - m_{21}u_{12} = 2 - 1 \times 1 = 1 \\ u_{23} &= a_{23} - m_{21}u_{13} = -2 - 1 \times (-1) = -1 \\ m_{32} &= (a_{32} - m_{31}u_{12})/u_{22} = (1 - (-2) \times 1)/1 = 3 \\ u_{33} &= a_{33} - m_{31}u_{13} - m_{32}u_{23} = 1 - (-2) \times (-1) - 3 \times (-1) = 2 \end{aligned}$$

Thus,

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

Taking $\boldsymbol{b} = (1, 1, 1)$, we now solve the system $A\boldsymbol{x} = \boldsymbol{b}$ using LU factorization, with the matrix A given above. As discussed above, first we have to solve the lower triangular system

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 3 & 1 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

Forward substitution yields $g_1 = 1, g_2 = 0, g_3 = 3$. Keeping the vector $\boldsymbol{g} = (1, 0, 3)$ as the right hand side, we now solve the upper triangular system

$$\begin{bmatrix} 1 & 1 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 3 \end{bmatrix}.$$

Backward substitution yields $x_1 = 1, x_2 = 3/2, x_3 = 3/2$.. □

3.3 Error in Solving Linear Systems

In computing solution for a linear system using Gaussian elimination, we have seen the propagation of rounding error, which can lead to entirely wrong solution. In this section, we introduce some method to obtain errors prediction and ways to correct them inorder to minimize the error in the computed solution.

Let $\boldsymbol{x}_A$ denote the computed solution using some method. Define

$$\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}_A \tag{3.11}$$

This vector is called the **residual** vector in the approximation of $\boldsymbol{b}$ by $A\boldsymbol{x}_A$. Since $\boldsymbol{b} = A\boldsymbol{x}$, we have

$$\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}_A = A\boldsymbol{x} - A\boldsymbol{x}_A = A(\boldsymbol{x} - \boldsymbol{x}_A).$$

If we denote the error $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_A$, then the above identity can be written as

$$A\boldsymbol{e} = \boldsymbol{r} \tag{3.12}$$

This shows that the error $\boldsymbol{e}$ satisfies a linear system with the same coefficient matrix A as in the orginal system $A\boldsymbol{x} = \boldsymbol{b}$.

There is an obvious difficulty in implementing this procedure on a computer. Since $\boldsymbol{b}$ and $A\boldsymbol{x}_A$ are very close to each other, the computation of $\boldsymbol{r}$ involves loss of significant digits which leads to a very high relative error. To avoid an incorrect residual $\boldsymbol{r}$, the calculation of (3.11) should be carried out in a higher-precision (say if $\boldsymbol{b}$ and $A\boldsymbol{x}_A$ are calculated in single-precision, then $\boldsymbol{r}$ can be computed in double-precision and then rounded back to single precision).

Example 3.5. Consider the system

$$\begin{aligned} 0.729x_1 + 0.81x_2 + 0.9x_3 &= 0.6867 \\ x_1 + x_2 + x_3 &= 0.8338 \\ 1.331x_1 + 1.210x_2 + 1.100x_3 &= 1.000 \end{aligned}$$

As before, we use a four digit decimal-machine with rounding. The true solution of this system is

$$x_1 = 0.2245,\ x_2 = 0.2814,\ x_3 = 0.3279$$

correct rounded to four digits. We consider the solution of the system by Gaussian elimination without pivoting. This leads to the answers

$$x_1 \approx 0.2251, \quad x_2 \approx 0.2790, \quad x_3 \approx 0.3295.$$

Using 8 digit floating point decimal arithmetic, with rounding, we get the residual as

$$\boldsymbol{r} = (0.00006210, 0.0002000, 0.0003519)^T.$$

Solving the linear system $A\boldsymbol{e} = \boldsymbol{r}$, we obtain the approximation to the error

$$\boldsymbol{e}_A = [-0.0004471, 0.002150, -0.001504]^T.$$

Compare this to the true error

$$\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_A = [-0.0007, 0.0024, -0.0016]^T$$

Thus $\boldsymbol{e}_A$ gives a firly good idea of the size of the error $\boldsymbol{e}$ in the computed solution $\boldsymbol{x}_A$. □

The Residual Correction Method:

Step 1: Let $\boldsymbol{x}_0 = \boldsymbol{x}_A$ be the initially computed value for the solution of the system $A\boldsymbol{x} = \boldsymbol{b}$, generally obtained by using Gaussian elimination. Define

$$\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0.$$

The error defined by $\boldsymbol{e}_0 = \boldsymbol{x} - \boldsymbol{x}_0$ is obtained (approximately) by solving the system

$$A\boldsymbol{e}_0 = \boldsymbol{r}_0$$

using Gaussian elimination.

Step 2: Define

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \boldsymbol{e}_0$$

and repeat step 1 to calculate

$$\boldsymbol{r}_1 = \boldsymbol{b} - A\boldsymbol{x}_1, \quad \boldsymbol{x}_2 = \boldsymbol{x}_1 + \boldsymbol{e}_1$$

where $\boldsymbol{e}_1 = \boldsymbol{x} - \boldsymbol{x}_1$ is the approximate solution of the system $A\boldsymbol{e}_1 = \boldsymbol{r}_1$.

Continue this process untill there is no further decrease in the size of $\boldsymbol{e}_k$, $k \geq 0$. □

Example 3.6. Use a computer with four digit floating-point decimal arithmetic with rounding, and use Gaussian elimination with pivoting, the system to be solved is

$$\begin{aligned} x_1 + 0.5x_2 + 0.3333x_3 &= 1 \\ 0.5x_1 + 0.3333x_2 + 0.25x_3 &= 0 \\ 0.3333x_1 + 0.25x_2 + 0.2x_3 &= 0 \end{aligned}$$

The true solution rounded to four digits is $\boldsymbol{x}_2 = (9.062,\ -36.32,\ 30.30)^T$. Using the Residual correction method, we have

$$\begin{aligned} \boldsymbol{x}_0 &= (8.968,\ -35.77,\ 29.77)^T \\ \boldsymbol{r}_0 &= (-0.005341,\ -0.004359,\ -0.0005344)^T \\ \boldsymbol{e}_0 &= (0.09216,\ -0.5442,\ 0.5239)^T \\ \boldsymbol{x}_1 &= (9.060,\ -36.31,\ 30.29)^T \\ \boldsymbol{r}_1 &= (-0.0006570,\ -0.0003770,\ -0.0001980)^T \\ \boldsymbol{e}_2 &= (0.001707,\ -0.01300,\ 0.01241)^T \\ \boldsymbol{x}_2 &= (9.062,\ -36.32,\ 30.30)^T \end{aligned}$$

3.4 Matrix Norm

A useful notion of measuring a vector (in general a matrix) is the well-known **norms**

Definition 3.7 (Vector Norm).

A **vector norm** *on* $\mathbb{R}^n$ *is a function from* $\mathbb{R}^n$ *to* $[0, \infty)$ *denoted by* $\|\cdot\|$ *that satisfies the following properties:*
For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$,

I. $\|\boldsymbol{x}\| \geq 0$

II. $\|\boldsymbol{x}\| = 0$ *if and only if* $\boldsymbol{x} = 0$

III. $\|\alpha\boldsymbol{x}\| = |\alpha|\|\boldsymbol{x}\|$

IV. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$

Example 3.8. Some examples of vector norm are given here.

I. The **Euclidean norm** is defined as

$$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{j=1}^{n} x_i^2}. \tag{3.13}$$

II. The **maximum norm** (similar to the infinite norm defined in section 2.4) is defined as

$$\|\boldsymbol{x}\|_\infty := \max_{1\le i\le n} |x_i|, \boldsymbol{x} = (x_1, \cdots, x_n). \tag{3.14}$$

Definition 3.9 (Matrix Norm).

A **matrix norm** *on the set of all $n \times n$ matrices is a real-valued function, $||\cdot||$, defined on this set, satisfying for all $n \times n$ matrices A and B and all real numbers α:*

I. $||A|| \ge 0$*;*

II. $||A|| = 0$*, if and only if A is a zero matrix;*

III. $||\alpha A|| = |\alpha|\ ||A||$*;*

IV. $||A + B|| \le ||A|| + ||B||$*;*

Definition 3.10 (Natural or Induced Matrix Norm).

If $||\cdot||$ is a vector norm on $\mathbb{R}^n$, then

$$||A|| = \max_{||\boldsymbol{x}||=1} ||A\boldsymbol{x}||$$

is a matrix norm and is called the **natural** *or* **induced** *matrix norm associated with the vector norm.*

Remark 3.11. In this course, all matrix norms will be assumed to be natural matrix norms.

For any $\boldsymbol{z} \neq 0$, we have $\boldsymbol{x} = \boldsymbol{z}/||\boldsymbol{z}||$ as a unit vector. Hence

$$\max_{||\boldsymbol{x}||=1} ||A\boldsymbol{x}|| = \max_{||\boldsymbol{z}||\neq 0} \left\| A\left(\frac{\boldsymbol{z}}{||\boldsymbol{z}||}\right)\right\| = \max_{||\boldsymbol{z}||\neq 0} \frac{||A\boldsymbol{z}||}{||\boldsymbol{z}||},$$

and we can alternatively write

$$\|A\| = \max_{\boldsymbol{z}\neq 0} \frac{\|A\boldsymbol{z}\|}{\|\boldsymbol{z}\|} \tag{3.15}$$

Lemma 3.12. *For any $n \times n$ matrices A and B, and $\boldsymbol{x} \in \mathbb{R}^n$, we have*

I. $\|A\boldsymbol{x}\| \le \|A\|\|\boldsymbol{x}\|$

II. $\|AB\| \le \|A\|\|B\|$

For any $n \times n$ matrix A the **maximum row norm** is defined as

$$\|A\| := \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}|. \tag{3.16}$$

It can be shown (proof is omitted here) that the maximum row norm is induced by the maximum norm defined in (3.14). The Eulidean norm (3.13) induces the matrix norm (proof is omitted here)

$$\|A\|_2 = \sqrt{r_\sigma(A^T A)}, \tag{3.17}$$

where

$$r_\sigma(A) = \max_{\lambda\in\sigma(A)} |\lambda|$$

with $\sigma(A)$ being the set of all eigenvalues of A, called the **spectrum** of A.

Example 3.13. If we take

$$A = \begin{bmatrix} 1 & 1 & -1 \\ 1 & 2 & -2 \\ -2 & 1 & 1 \end{bmatrix},$$

then

$$\sum_{j=1}^{3} |a_{1j}| = |1| + |1| + |-1| = 3,$$
$$\sum_{j=1}^{3} |a_{2j}| = |1| + |2| + |-2| = 5,$$
$$\sum_{j=1}^{3} |a_{3j}| = |-2| + |1| + |1| = 4.$$

Therefore, the maximum row norm of the given matrix A is 5.

On the other hand, the eigenvalues of $A^T A$ are $\lambda_1 \approx 0.0616$, $\lambda_2 \approx 5.0256$ and $\lambda_3 \approx 12.9128$. Thus, $\|A\|_2 \approx \sqrt{12.9128} \approx 3.5934$. □

Theorem 3.14. *Let A be nonsingular. Then, the solution $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ of the systems $A\boldsymbol{x} = \boldsymbol{b}_1$ and $A\boldsymbol{x} = \boldsymbol{b}_2$, respectively, satisfy*

$$\frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|}{\|\boldsymbol{x}_1\|} \leq \|A\|\|A^{-1}\| \frac{\|\boldsymbol{b}_1 - \boldsymbol{b}_2\|}{\|\boldsymbol{b}_1\|} \tag{3.18}$$

Proof. Subtracting $A\boldsymbol{x}_2 = \boldsymbol{b}_2$ from $A\boldsymbol{x}_1 = \boldsymbol{b}_1$, we get

$$A(\boldsymbol{x}_1 - \boldsymbol{x}_2) = \boldsymbol{b}_1 - \boldsymbol{b}_2 \quad \text{or} \quad \boldsymbol{x}_1 - \boldsymbol{x}_2 = A^{-1}(\boldsymbol{b}_1 - \boldsymbol{b}_2).$$

Using the above lemma, we get

$$\|\boldsymbol{x}_1 - \boldsymbol{x}_2\| = \|A^{-1}(\boldsymbol{b}_1 - \boldsymbol{b}_2)\| \leq \|A^{-1}\|\|\boldsymbol{b}_1 - \boldsymbol{b}_2\|.$$

Dividing by $\|\boldsymbol{x}_1\|$, we obtain

$$\frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|}{\|\boldsymbol{x}_1\|} \leq \|A^{-1}\| \frac{\|\boldsymbol{b}_1 - \boldsymbol{b}_2\|}{\|\boldsymbol{x}_1\|} = \|A\|\|A^{-1}\| \frac{\|\boldsymbol{b}_1 - \boldsymbol{b}_2\|}{\|A\|\|\boldsymbol{x}_1\|}.$$

But $\|\boldsymbol{b}\| = \|A\boldsymbol{x}\| \leq \|A\|\|\boldsymbol{x}\|$. Using this inequality, we get the desired result. □

The multiplying coefficient $\|A\|\|A^{-1}\|$ is interesting. It depends entirely on the matrix in the problem and not on the right-side vector, yet it shows up as an amplifier to the relative change in the RHS vector.

Definition 3.15 (Condition Number).

For a given non-singular matrix $A \in \mathbb{R}^{n\times n}$ and a given matrix norm $\|\cdot\|$, the condition number of A with respect to the given norm is defined by

$$\kappa(A) := \|A\|\|A^{-1}\| \tag{3.19}$$

When the condition number of a matrix is very large, even a small variation in the RHS vector can lead to a drastic variation in the solution. Such matrices are called **ill-conditioned** matrices. The matrices with small condition number are called **well-conditioned** matrices.

Example 3.16. A well-known example of an ill-conditioned matrix is the **Hilbert matrix**

$$H_n = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \cdot & & & \cdots & \\ \cdot & & & \cdots & \\ \cdot & & & \cdots & \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+1} & \cdots & \frac{1}{2n-1} \end{bmatrix} \tag{3.20}$$

For $n = 4$, we have

$$\kappa(H_4) = \|H_4\|\|H_4^{-1}\| = \frac{25}{12}13620 \approx 28000$$

which may be taken as an ill-conditioned matrix. □

Ill-conditioned matrices are very rare in applications. However, discretization of many partial differential equations leads to moderately ill-conditioned linear systems. For this reason, it is best to use linear equation solvers that have some way to detect ill-conditioning, if possible. Otherwise, the error can be computed explicitely as described in section 3.3 to ensure the accuracy in the computed solution.

The following example show how a small variation in the RHS vector lead to a big difference in the solution.

Example 3.17. The linear system

$$\begin{aligned} 5x_1 + 7x_2 &= 0.7 \\ 7x_1 + 10x_2 &= 1 \end{aligned}$$

has the solution $x_1 = 0, x_2 = 0.1$. Let us denote this by $\boldsymbol{x}_T = (0, 0.1)$. The perturbed system

$$\begin{aligned} 5x_1 + 7x_2 &= 0.69 \\ 7x_1 + 10x_2 &= 1.01 \end{aligned}$$

has the solution $x_1 = -0.17, x_2 = 0.22$, which we denote by $\boldsymbol{x}_A = (-0.17, 0.22)$. The relative error between the solutions of the above systems in the maximum vector norm is given by

$$\frac{\|\boldsymbol{x}_T - \boldsymbol{x}_A\|_\infty}{\|\boldsymbol{x}_T\|_\infty} = 1.7,$$

which is too high. On the other hand, the condition number of the coefficient matrix of the above system is 289, and the relative error between the right hand side vectors in the maximum norm is 0.01. Thus, the right hand side of the inequality (3.18) is 2.89, which obviously satisfies this inequality. □

Theorem 3.18. *Let $A \in \mathbb{R}^{n\times n}$ be non-singular. Then, for any singular $n \times n$ matrix B, we have*

$$\frac{1}{\kappa(A)} \leq \frac{\|A - B\|}{\|A\|}. \tag{3.21}$$

Proof. We have

$$\begin{aligned} \frac{1}{\kappa(A)} &= \frac{1}{\|A\|\|A^{-1}\|} \\ &= \frac{1}{\|A\|}\left(\frac{1}{\max\limits_{\boldsymbol{x}\neq 0}\dfrac{\|A^{-1}\boldsymbol{x}\|}{\|\boldsymbol{x}\|}}\right) \\ &\leq \frac{1}{\|A\|}\left(\frac{1}{\dfrac{\|A^{-1}\boldsymbol{y}\|}{\|\boldsymbol{y}\|}}\right) \end{aligned}$$

where $\boldsymbol{y}$ is arbitrary. Now take $\boldsymbol{y} = A\boldsymbol{z}$. Then we get

$$\frac{1}{\kappa(A)} \leq \frac{1}{\|A\|}\left(\frac{\|A\boldsymbol{z}\|}{\|\boldsymbol{z}\|}\right),$$

where $\boldsymbol{z}$ is arbitrary. Let $\boldsymbol{z}$ be such that $B\boldsymbol{z} = 0$ (this is possible since B is singular), we get

$$\frac{1}{\kappa(A)} \leq \frac{\|(A-B)\boldsymbol{z}\|}{\|A\|\|\boldsymbol{z}\|} \leq \frac{\|(A-B)\|\|\boldsymbol{z}|}{\|A\|\|\boldsymbol{z}\|} = \frac{\|(A-B)\|}{\|A\|},$$

and we are done. □

The importance of this result is that it tells us that if A is close to a singular matrix, then the reciprocal of the condition number will be near to zero, ie., $\kappa(A)$ itself will be large.

3.5 Iterative Methods

The $n \times n$ linear system can also be solved using iterative procedures. The most fundamental iterative method is the Jacobi iterative method, which we will explain in the case of 3×3 system of linear equations.

Consider the 3×3 system

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3 \end{aligned}$$

When the diagonal elements of this system are non-zero, we can rewrite the above equation as

$$\begin{aligned} x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3) \\ x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3) \\ x_3 &= \frac{1}{a_{33}}(b_3 - a_{31}x_1 - a_{32}x_2) \end{aligned}$$

Let $\boldsymbol{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)})$ be an initial guess to the true solution $\boldsymbol{x}$, then define an iteration sequence:

$$\begin{aligned} x_1^{(m+1)} &= \frac{1}{a_{11}}(b_1 - a_{12}x_2^{(m)} - a_{13}x_3^{(m)}) \\ x_2^{(m+1)} &= \frac{1}{a_{22}}(b_2 - a_{21}x_1^{(m)} - a_{23}x_3^{(m)}) \\ x_3^{(m+1)} &= \frac{1}{a_{33}}(b_3 - a_{31}x_1^{(m)} - a_{32}x_2^{(m)}) \end{aligned}$$

for $m = 0, 1, 2, \cdots$. This is called the **Jacobi Iteration method**.

A modified version of Jacobi method is the **Gauss-Seidel method** and is given by

$$\begin{aligned} x_1^{(m+1)} &= \frac{1}{a_{11}}(b_1 - a_{12}x_2^{(m)} - a_{13}x_3^{(m)}) \\ x_2^{(m+1)} &= \frac{1}{a_{22}}(b_2 - a_{21}x_1^{(m+1)} - a_{23}x_3^{(m)}) \\ x_3^{(m+1)} &= \frac{1}{a_{33}}(b_3 - a_{31}x_1^{(m+1)} - a_{32}x_2^{(m+1)}) \end{aligned}$$

Note that the Jacobi method is of the form

$$N\boldsymbol{x}^{(m+1)} = \boldsymbol{b} + U\boldsymbol{x}^{(m)}$$

where

$$N = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ . & . & \cdots & . \\ . & . & \cdots & . \\ . & . & \cdots & . \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix}$$

and $U = N - A$. For Gauss-Seidel method, we have

$$N = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & 0 \\ . & . & \cdots & . \\ . & . & \cdots & . \\ . & . & \cdots & . \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

with $U = N - A$.

A general linear iterative method for the solution of the system of linear system of equations $A\boldsymbol{x} = \boldsymbol{b}$ may be defined in the form

$$\boldsymbol{x}^{(m+1)} = B\boldsymbol{x}^{(m)} + \boldsymbol{c}, \quad m = 1, 2, \cdots . \tag{3.22}$$

In this case of Jacobi and Gauss-Seidel methods, we have $B = N^{-1}U$ and $\boldsymbol{c} = N^{-1}\boldsymbol{b}$.

Note that the true solution satisfies the equation

$$\boldsymbol{x} = B\boldsymbol{x} + \boldsymbol{c}$$

and therefore, the error $e^{(m)} = \boldsymbol{x} - \boldsymbol{x}^{(m)}$ satisfies the system

$$\boldsymbol{e}^{(m+1)} = B\boldsymbol{e}^{(m)}.$$

On taking norm, we get

$$\|\boldsymbol{e}^{(m+1)}\| = \|B\boldsymbol{e}^{(m)}\| \leq \|B\|\|\boldsymbol{e}^{(m)}\| \leq \cdots \leq \|B\|^{m+1}\|\boldsymbol{e}^{(0)}\|.$$

Thus, when $\|B\| < 1$, the iteration method always converges for any initial guess.

Definition 3.19 (Diagonally Dominant Matrices). *A matrix A is said to be* **diagonally dominant** *if it satisfies the inequality*

$$\sum_{j=1, j\neq i}^{n} |a_{ij}| < |a_{ii}|, \quad i = 1, 2, \cdots, n.$$

In the case of Jacobi method, we have

$$x_i^{(m+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(m)} \right), \quad i = 1, \cdots, n \quad m \geq 0 \tag{3.23}$$

Therefore, each component of the error satisfies

$$e_i^{(m+1)} = -\sum_{j=1, j\neq i}^{n} \frac{a_{ij}}{a_{ii}} e_j^{(m)}, \quad i = 1, \cdots, n \quad m \geq 0.$$

which gives

$$|e_i^{(m+1)}| \leq \sum_{j=1, j\neq i}^{n} \left| \frac{a_{ij}}{a_{ii}} \right| \|\boldsymbol{e}^{(m)}\|_\infty.$$

Define

$$\mu = \max_{1\leq i\leq n} \sum_{j=1, j\neq i}^{n} \left| \frac{a_{ij}}{a_{ii}} \right|. \tag{3.24}$$

Then

$$|e_i^{(m+1)}| \leq \mu \|\boldsymbol{e}^{(m)}\|_\infty,$$

which is true for all $i = 1, 2, \cdots, n$. Therefore, we have

$$\|\boldsymbol{e}^{(m+1)}\|_\infty \leq \mu \|\boldsymbol{e}^{(m)}\|_\infty.$$

For $\mu < 1$, ie., when the matrix A is diagonally dominant, then Jacobi method converges. Note that the converse is not true. That is, the Jacobi method may converge for A not diagonally dominant.

We will now prove that the Gauss-Seidal method converges if the given matrix A is diagonally dominant. The Gauss-Seidal method reads

$$x_i^{(m+1)} = \frac{1}{a_{ii}} \left\{ b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(m+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(m)} \right\}, \quad i = 1, 2, \cdots, n. \tag{3.25}$$

Therefore, the error in each component is given by

$$e_i^{(m+1)} = -\sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} e_j^{(m+1)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} e_j^{(m)}, \quad i = 1, 2, \cdots, n. \tag{3.26}$$

Define

$$\alpha_i = \sum_{j=1}^{i-1} \left| \frac{a_{ij}}{a_{ii}} \right|, \quad \beta_i = \sum_{j=i+1}^{n} \left| \frac{a_{ij}}{a_{ii}} \right|, \quad i = 1, 2, \cdots, n,$$

with $\alpha_1 = \beta_1 = 0$. Note that μ given in (3.24) can be written as

$$\mu = \max_{1 \le i \le n} (\alpha_i + \beta_i)$$

and since A is assumed to be diagonally dominant, we have $\mu < 1$. Now

$$|e_i^{(m+1)}| \le \alpha_i \|\boldsymbol{e}^{(m+1)}\|_\infty + \beta_i \|\boldsymbol{e}^{(m)}\|_\infty, \quad i = 1, 2, \cdots, n. \tag{3.27}$$

Let k be such that

$$\|\boldsymbol{e}^{(m+1)}\|_\infty = |e_k^{(m+1)}|.$$

Then with $i = k$ in (3.27),

$$\|\boldsymbol{e}^{(m+1)}\|_\infty \le \alpha_k \|\boldsymbol{e}^{(m+1)}\|_\infty + \beta_k \|\boldsymbol{e}^{(m)}\|_\infty.$$

Since $\mu < 1$, we have $\alpha_k < 1$ and therefore the above inequality give

$$\|\boldsymbol{e}^{(m+1)}\|_\infty \le \frac{\beta_k}{1 - \alpha_k} \|\boldsymbol{e}^{(m)}\|_\infty.$$

Define

$$\eta = \max_{1 \le i \le n} \frac{\beta_k}{1 - \alpha_k}. \tag{3.28}$$

Then the above inequality takes the form

$$\|\boldsymbol{e}^{(m+1)}\|_\infty \le \eta \|\boldsymbol{e}^{(m)}\|_\infty. \tag{3.29}$$

Since for each i,

$$(\alpha_i + \beta_i) - \frac{\beta_i}{1 - \alpha_i} = \frac{\alpha_i [1 - (\alpha_i + \beta_i)]}{1 - \alpha_i} \ge \frac{\alpha_i}{1 - \alpha_i} [1 - \mu] \ge 0,$$

we have

$$\eta \le \mu < 1. \tag{3.30}$$

Thus, Gauss-Seidal method converges more faster than the Jacobi method and also when the given matrix is diagonally dominant, then the Gauss-Seidal method converges.

3.6 Eigenvalue Problem: The Power Method

Power method is normally used to determine the largest eigenvalue (in magnitude) and the corresponding eigenvector of the system

$$A\boldsymbol{x} = \lambda \boldsymbol{x}.$$

Let $\lambda_1, \lambda_2, \cdots, \lambda_n$ be the eigenvalues of A such that

$$|\lambda_1| > |\lambda_2| \ge |\lambda_3| \ge \cdots \ge |\lambda_n| \tag{3.31}$$

and further assume that the corresponding eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n$ forms a basis for $\mathbb{R}^n$. Therefore, any vector $\boldsymbol{v} \in \mathbb{R}^n$ can be written as

$$\boldsymbol{v} = c_1 \boldsymbol{v}_1 + c_2 \boldsymbol{v}_2 + \cdots + c_n \boldsymbol{v}_n.$$

Premultiplying by A and substituting $A\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$, $i = 1, \cdots, n$, we get

$$\begin{aligned} A\boldsymbol{v} &= c_1\lambda_1\boldsymbol{v}_1 + \cdots + c_n\lambda_n\boldsymbol{v}_n \\ &= \lambda_1 \left(c_1\boldsymbol{v}_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)\boldsymbol{v}_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)\boldsymbol{v}_n \right) \end{aligned}$$

Premultiplying by A again and simplying, we get

$$\begin{aligned} A^2\boldsymbol{v} &= \lambda_1^2 \left(c_1\boldsymbol{v}_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^2 \boldsymbol{v}_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^2 \boldsymbol{v}_n \right) \\ &\cdots \\ &\cdots \\ &\cdots \\ A^k\boldsymbol{v} &= \lambda_1^k \left(c_1\boldsymbol{v}_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k \boldsymbol{v}_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k \boldsymbol{v}_n \right) \end{aligned}$$

Using the assumption (3.31), we can see that

$$\left|\frac{\lambda_k}{\lambda_1}\right| < 1, \quad k = 2, \cdots, n.$$

Therefore, we have

$$\lim_{k\to\infty} \frac{A^k\boldsymbol{v}}{\lambda_1^k} = c_1\boldsymbol{v}_1. \tag{3.32}$$

For $c_1 \neq 0$, the RHS is a scalar multiple of the eigenvector. Also, from the above expression for $A^k\boldsymbol{v}$, we get

$$\lim_{k\to\infty} \frac{(A^{k+1}\boldsymbol{v})_i}{(A^k\boldsymbol{v})_i} = \lambda_1, \tag{3.33}$$

where i denotes a component of the corresponding vectors.

The power method is based on the results (3.32) and (3.33).

Algorithm 3.20. Choose an arbitrary initial guess $\boldsymbol{x}^{(0)}$. For $k = 1, 2, \cdots$
Step 1 Compute $\boldsymbol{y}^{(k)} = A\boldsymbol{x}^{(k-1)}$
Step 2 Take $\mu_k = y_i^{(k)}$, where $\|\boldsymbol{y}^{(k)}\|_\infty = |y_i^{(k)}|$,
Step 3 Set $\boldsymbol{x}^{(k)} = \frac{\boldsymbol{y}^{(k)}}{\mu_k}$.
Step 4 If $\|\boldsymbol{x}^{(k-1)} - \boldsymbol{x}^{(k)}\|_\infty > \epsilon$, go to step 1.
For some pre-assigned positive quantity ϵ.

Let us now study the convergence of this method.

Theorem 3.21 (Power method).
Let A be an non-singular $n \times n$ matrix with the following conditions:

I. A has n linearly independent eigenvectors, $\boldsymbol{v}_i$, $i = 1, \cdots, n$.

II. The eigenvalues λ_i satisfy

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

III. The vector $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$ is such that

$$\boldsymbol{x}^{(0)} = \sum_{j=1}^{n} c_j\boldsymbol{v}_j, \quad c_1 \neq 0.$$

Then the power method converges in the sense that there exists constants C_1 and C_2 such that

$$\|\boldsymbol{x}^{(k)} - K\boldsymbol{v}_1\| \leq C_1 \left|\frac{\lambda_2}{\lambda_1}\right|^k, \quad \textit{for some } K \neq 0$$

and

$$|\lambda_1 - \mu_k| \leq C_1 \left|\frac{\lambda_2}{\lambda_1}\right|^k.$$

Proof. From the definition of $\boldsymbol{x}^{(k)}$, we have

$$\boldsymbol{x}^{(k)} = \frac{A\boldsymbol{x}^{(k-1)}}{\mu_k} = \frac{A\boldsymbol{y}^{(k-1)}}{\mu_k\mu_{k-1}} = \frac{AA\boldsymbol{x}^{(k-2)}}{\mu_k\mu_{k-1}} = \frac{A^2\boldsymbol{x}^{(k-2)}}{\mu_k\mu_{k-1}} = \cdots = \frac{A^k\boldsymbol{x}^{(0)}}{\mu_k\mu_{k-1}\cdots\mu_1}.$$

Therefore, we have

$$\boldsymbol{x}^{(k)} = m_k A^k \boldsymbol{x}^{(0)},$$

where $m_k = 1/(\mu_1\mu_2\cdots\mu_k)$. But, $\boldsymbol{x}^{(0)} = \sum_{j=1}^{n} c_j\boldsymbol{v}_j, \quad c_1 \neq 0$. Therefore

$$\boldsymbol{x}^{(k)} = m_k\lambda_1^k \left(c_1\boldsymbol{v}_1 + \sum_{j=2}^{n} c_j \left(\frac{\lambda_j}{\lambda_1}\right)^k \boldsymbol{v}_j\right).$$

Taking maximum norm on both sides and noting that $\|\boldsymbol{x}^{(k)}\|_\infty = 1$, we get

$$1 = |m_k\lambda_1^k| \left\|c_1\boldsymbol{v}_1 + \sum_{j=2}^{n} c_j \left(\frac{\lambda_j}{\lambda_1}\right)^k \boldsymbol{v}_j\right\|_\infty.$$

This implies on taking limit,

$$|\lim_{k\to\infty} m_k\lambda_1^k| = \frac{1}{|c_1|\|\boldsymbol{v}_1\|_\infty} < \infty.$$

This is equivalent to

$$\lim_{k\to\infty} m_k\lambda_1^k = \pm\frac{1}{c_1\|\boldsymbol{v}_1\|_\infty} < \infty.$$

Finally,

$$\lim_{k\to\infty} \boldsymbol{x}^{(k)} = \lim_{k\to\infty} m_k\lambda_1^k.c_1\boldsymbol{v}_1 = K\boldsymbol{v}_1$$

Moreover,

$$\|\boldsymbol{x}^{(k)} - K_k\boldsymbol{v}_1\|_\infty = \left\|m_k\lambda_1^k\sum_{j=2}^{n} c_j \left(\frac{\lambda_j}{\lambda_1}\right)^k \boldsymbol{v}_j\right\|_\infty \leq C\left|\frac{\lambda_2}{\lambda_1}\right|^k.$$

For eigenvalue,

$$\mu_k\boldsymbol{x}^{(k)} = \boldsymbol{y}^{(k)}.$$

Therefore,

$$\mu_k = \frac{\boldsymbol{y}_i^{(k)}}{\boldsymbol{x}_i^{(k)}} = \frac{(A\boldsymbol{x}^{(k-1)})_i}{(\boldsymbol{x}^{(k)})_i}$$

Taking limit, we have

$$\lim_{k\to\infty} \mu_k = \frac{A(K\boldsymbol{v}_1)_i}{K(\boldsymbol{v}_1)_i} = \frac{\lambda(\boldsymbol{v}_1)_i}{(\boldsymbol{v}_1)_i} = \lambda_1.$$

which gives the desired result. □

Example 3.22. Consider the matrix

$$A = \begin{bmatrix} 3 & 0 & 0 \\ -4 & 6 & 2 \\ 16 & -15 & -5 \end{bmatrix}$$

The eigenvalues of this matrix are $\lambda_1 = 3$, $\lambda_2 = 1$ and $\lambda_3 = 0$. The corresponding eigen vectors are $\boldsymbol{X}_1 = (1, 0, 2)^T$, $\boldsymbol{X}_2 = (0, 2, -5)^T$ and $\boldsymbol{X}_3 = (0, 1, -3)^T$.

Initial Guess 1: Let us take $\boldsymbol{x}_0 = (1, 0.5, 0.25)^T$. The power method gives the following:
Iteration No: 1

$$\begin{aligned} \boldsymbol{y}_1 &= A\boldsymbol{x}_0 = (3.000000, -0.500000, 7.250000,)^T \\ \mu_1 &= 7.250000 \\ \boldsymbol{x}_1 &= \frac{\boldsymbol{y}_1}{\mu_1} = (0.413793, -0.068966, 1.000000,)^T \end{aligned}$$

Iteration No: 2

$$\begin{aligned} \boldsymbol{y}_2 &= A\boldsymbol{x}_1 = (1.241379, -0.068966, 2.655172,)^T \\ \mu_2 &= 2.655172 \\ \boldsymbol{x}_2 &= \frac{\boldsymbol{y}_2}{\mu_2} = (0.467532, -0.025974, 1.000000,)^T \end{aligned}$$

Iteration No: 3

$$\begin{aligned} \boldsymbol{y}_3 &= A\boldsymbol{x}_2 = (1.402597, -0.025974, 2.870130,)^T \\ \mu_3 &= 2.870130 \\ \boldsymbol{x}_3 &= \frac{\boldsymbol{y}_3}{\mu_3} = (0.488688, -0.009050, 1.000000,)^T \end{aligned}$$

Iteration No: 4

$$\begin{aligned} \boldsymbol{y}_4 &= A\boldsymbol{x}_3 = (1.466063, -0.009050, 2.954751,)^T \\ \mu_4 &= 2.954751 \\ \boldsymbol{x}_4 &= \frac{\boldsymbol{y}_4}{\mu_4} = (0.496172, -0.003063, 1.000000,)^T \end{aligned}$$

Iteration No: 5

$$\begin{aligned} \boldsymbol{y}_5 &= A\boldsymbol{x}_4 = (1.488515, -0.003063, 2.984686,)^T \\ \mu_5 &= 2.984686 \\ \boldsymbol{x}_5 &= \frac{\boldsymbol{y}_5}{\mu_5} = (0.498717, -0.001026, 1.000000,)^T \end{aligned}$$

Iteration No: 6

$$\begin{aligned} \boldsymbol{y}_6 &= A\boldsymbol{x}_5 = (1.496152, -0.001026, 2.994869,)^T \\ \mu_6 &= 2.994869 \\ \boldsymbol{x}_6 &= \frac{\boldsymbol{y}_6}{\mu_6} = (0.499572, -0.000343, 1.000000,)^T \end{aligned}$$

Iteration No: 7

$$\begin{aligned} \boldsymbol{y}_7 &= A\boldsymbol{x}_6 = (1.498715, -0.000343, 2.998287,)^T \\ \mu_7 &= 2.998287 \\ \boldsymbol{x}_7 &= \frac{\boldsymbol{y}_7}{\mu_7} = (0.499857, -0.000114, 1.000000,)^T \end{aligned}$$

Iteration No: 8

$$\boldsymbol{y}_8 = A\boldsymbol{x}_7 = (1.499571, -0.000114, 2.999429,)^T$$
$$\mu_8 = 2.999429$$
$$\boldsymbol{x}_8 = \frac{\boldsymbol{y}_8}{\mu_8} = (0.499952, -0.000038, 1.000000,)^T$$

Iteration No: 9

$$\boldsymbol{y}_9 = A\boldsymbol{x}_8 = (1.499857, -0.000038, 2.999809,)^T$$
$$\mu_9 = 2.999809$$
$$\boldsymbol{x}_9 = \frac{\boldsymbol{y}_9}{\mu_9} = (0.499984, -0.000013, 1.000000,)^T$$

Iteration No: 10

$$\boldsymbol{y}_10 = A\boldsymbol{x}_9 = (1.499952, -0.000013, 2.999936,)^T$$
$$\mu_10 = 2.999936$$
$$\boldsymbol{x}_10 = \frac{\boldsymbol{y}_10}{\mu_10} = (0.499995, -0.000004, 1.000000,)^T$$

Initial Guess 2: Let us take $\boldsymbol{x}_0 = (0, 0.5, 0.25)^T$. The power method gives the following:
Iteration No: 1

$$\boldsymbol{y}_1 = A\boldsymbol{x}_0 = (0.000000, 3.500000, -8.750000,)^T$$
$$\mu_1 = 8.750000$$
$$\boldsymbol{x}_1 = \frac{\boldsymbol{y}_1}{\mu_1} = (0.000000, 0.400000, -1.000000,)^T$$

Iteration No: 2

$$\boldsymbol{y}_2 = A\boldsymbol{x}_1 = (0.000000, 0.400000, -1.000000,)^T$$
$$\mu_2 = 1.000000$$
$$\boldsymbol{x}_2 = \frac{\boldsymbol{y}_2}{\mu_2} = (0.000000, 0.400000, -1.000000,)^T$$

Iteration No: 3

$$\boldsymbol{y}_3 = A\boldsymbol{x}_2 = (0.000000, 0.400000, -1.000000,)^T$$
$$\mu_3 = 1.000000$$
$$\boldsymbol{x}_3 = \frac{\boldsymbol{y}_3}{\mu_3} = (0.000000, 0.400000, -1.000000,)^T$$

Iteration No: 4

$$\boldsymbol{y}_4 = A\boldsymbol{x}_3 = (0.000000, 0.400000, -1.000000,)^T$$
$$\mu_4 = 1.000000$$
$$\boldsymbol{x}_4 = \frac{\boldsymbol{y}_4}{\mu_4} = (0.000000, 0.400000, -1.000000,)^T$$

Note that in the second initial guess, the first coordinate is zero and therefore, c_1 in the power method is zero. This makes the iteration to converge to λ_2, which is the next dominant eigenvalue. □

3.7 Gerschgorin's Theorem

An important tool in eigenvalue approximation is the ability to localize the eigenvalues, and the most important tool in eigenvalue localization is Gerschgorin's theorem.

Theorem 3.23 (Gerschgorin).
Let $A \in \mathbb{R}^{n\times n}$ be given, and define the quantities

$$r_i = \sum_{j=1, j\neq i}^{n} |a_{ij}|,$$

$$D_i = \{z \in C / |z - a_{ii}| \leq r_i\}.$$

Then every eigenvalue of A lies in the union of the disks D_i, that is,

$$\lambda_k \in \cup_{i=1}^{n} D_i$$

for all $k = 1, 2, \cdots, n$. Moreover, if any collection of p disks is disjoint from the other $n - p$ disks, then we know that exactly p eigenvalues are contained in the union of the set of p disks, and exactly $n - p$ eigenvalues are contained in the set of $n - p$ disks.

Example 3.24. Consider the matrix

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

Center of the disks: $a_{11} = 2,\ a_{22} = 2,\ a_{33} = 2$. The disks are concentric.
Radius of the disks: $r_1 = 1,\ r_2 = 2,\ r_3 = 1$.
The eigenvalues are $\lambda_1 = 3.1414$, $\lambda_2 = 2$, $\lambda_3 = 0.5859$. □

Example 3.25. Consider the matrix

$$A = \begin{bmatrix} 0 & 2 & 0 \\ 2 & 7 & 1 \\ 0 & 1 & 4 \end{bmatrix}$$

Center of the disks: $a_{11} = 0,\ a_{22} = 7,\ a_{33} = 4$.
Radius of the disks: $r_1 = 2,\ r_2 = 3,\ r_3 = 1$.
The eigenvalues are $\lambda_1 = 0.158197$, $\lambda_2 = 3.39573$, $\lambda_3 = 7.446072$. □

Exercise 3

I. **Direct Methods**

1. Given the linear system $2x_1 - 6\alpha x_2 = 3, \;\; 3\alpha x_1 - x_2 = \frac{3}{2}$.
(a) Find value(s) of α for which the system has no solution. (b) Find value(s) of α for which the system has infinitely many solutions. (c) Assuming a unique solution exists for a given α, find the solution.

2. Use Gaussian elimination method (both with and without pivoting) to find the solution of the following systems:
(i) $6x_1 + 2x_2 + 2x_3 = -2, \;\; 2x_1 + 0.6667x_2 + 0.3333x_3 = 1, \;\; x_1 + 2x_2 - x_3 = 0$
Answer: $x_1 = 2.599928$, $x_2 =$ -3.799904, $x_3 =$ -4.999880, Number of Pivoting = 1.
(ii) $0.729x_1 + 0.81x_2 + 0.9x_3 = 0.6867, \;\; x_1 + x_2 + x_3 = 0.8338, \;\; 1.331x_1 + 1.21x_2 + 1.1x_3 = 1$
Answer: $x_1 = 0.224545$, $x_2 = 0.281364$, $x_3 = 0.327891$, Number of Pivoting = 2.
(iii) $x_1 - x_2 + 3x_3 = 2, \;\; 3x_1 - 3x_2 + x_3 = -1, \;\; x_1 + x_2 = 3.$
Answer: $x_1 = 1.187500$, $x_2 = 1.812500$, $x_3 = 0.875000$, Number of Pivoting = 2.

3. Solve the system $0.004x_1 + x_2 = 1, \;\; x_1 + x_2 = 2$ (i) exactly, (ii) by Gaussian elimination using a two digit rounding calculator, and (iii) interchanging the equations and then solving by Gaussian elimination using a two digit rounding calculator.

4. Solve the following system by Gaussian elimination, first without row interchanges and then with row interchanges, using four-digit rounding arithmetic:

$$x + 592y = 437, \;\; 592x + 4308y = 2251.$$

5. Solve the system $0.5x_1 - x_2 = -9.5, \;\; 1.02x_1 - 2x_2 = -18.8$ using Gaussian elimination method. Solve the same system with a_{11} modified slightly to 0.52 (instead of 0.5). In both the cases, use rounding upto 5 digits after decimal point. Obtain the residual error in each case.

6. For an ϵ with absolute value very much smaller than 1, solve the linear system

$$x_1 + x_2 + x_3 = 6, \;\; 3x_1 + (3+\epsilon)x_2 + 4x_3 = 20, \;\; 2x_1 + x_2 + 3x_3 = 13$$

using Gaussian elimination method both with and without partial pivoting. Obtain the residual error in each case on a computer for which the ϵ is an unit round.

7. In the $n \times n$ system of linear equations

$$a_{11}x_1 + \cdots + a_{1n}x_n = b_1, \quad \cdots, \quad a_{n1}x_1 + \cdots + a_{nn}x_n = b_n$$

let $a_{ij} = 0$ whenever $i - j \geq 2$. Write out the general form of this system. Use Gaussian elimination to solve it, taking advantage of the elements that are known to be zero. Do an operations count in this case.

8. Obtain the LU factorization of the matrix

$$\begin{bmatrix} 4 & 1 & 1 \\ 1 & 4 & -2 \\ 3 & 2 & -4 \end{bmatrix}$$

Use this factorization to solve the system with $\boldsymbol{b} = (4, 4, 6)^T$.

9. Show that the following matrix cannot be written in the LU factorization form: $\begin{bmatrix} 1 & 2 & 6 \\ 4 & 8 & -1 \\ -2 & 3 & 5 \end{bmatrix}$

10. Show that the matrix

$$\begin{bmatrix} 2 & 2 & 1 \\ 1 & 1 & 1 \\ 3 & 2 & 1 \end{bmatrix}$$

is invertible but has no LU factorization. Do a suitable interchange of rows and/or columns to get an invertible matrix, which has LU factorization.

II. Errors and Matrix Norm

11. Use the Gaussian elimination method with rounding upto 5 digits after decimal point to solve the system $0.52x_1 - x_2 = -9.5,\ \ 1.02x_1 - 2x_2 = -18.8$. Use residual corrector algorithm to improve the solution till the error vector becomes zero.

12. Solve the system $x_1 + 1.001x_2 = 2.001,\quad x_1 + x_2 = 2$ (i) Compute the residual $\boldsymbol{r} = A\boldsymbol{y} - \boldsymbol{b}$ for $\boldsymbol{y} = (2, 0)^T$. (ii) Compute the relative error of $\boldsymbol{y}$ with respect to the exact solution $\boldsymbol{x}$ of the above system (use Euclidean norm in $\mathbb{R}^2$ defined by $||\boldsymbol{x}|| = \sqrt{x_1^2 + x_2^2}$).

13. For any $n \times n$ matrices A and B, and $\boldsymbol{x} \in \mathbb{R}^n$, show that
    i. $\|A\boldsymbol{x}\| \leq \|A\|\|\boldsymbol{x}\|$
    ii. $\|AB\| \leq \|A\|\|B\|$
    where the matrix norm is the induced norm obtained from the corresponding vector norm.

14. Solve the system
$$5x_1 + 7x_2 = b_1$$
$$7x_1 + 10x_2 = b_2$$
    using Gaussian elimination method to obtain the solution $\boldsymbol{x}_1$ when $\boldsymbol{b}_T = (b_1, b_2) = (0.7, 1)$. Also solve the above system with $\boldsymbol{b}_A = (b_1, b_2) = (0.69, 1.01)$ using the same method to obtain the solution $\boldsymbol{x}_2$. Show that
$$\frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2}{\|\boldsymbol{x}_1\|_2} \leq \|A\|_2\|A^{-1}\|_2\frac{\|\boldsymbol{b}_T - \boldsymbol{b}_A\|_2}{\|\boldsymbol{b}_T\|_2}$$
    where A is the 2×2 coefficient matrix of the above system and the norm in the above inequality is the Eucledian norm for vector and the corresponding induced norm for the matrix.

15. Show by an example that $||\cdot||_M$ defined by $||A||_M = \max\limits_{1\leq i,j\leq n}|a_{ij}|$, does not define an induced matrix norm.

16. Show that $\kappa(A) \geq 1$ for any $n \times n$ non-singular matrix A.

17. For any two $n \times n$ non-singular matrices A and B, show that $\kappa(AB) \leq \kappa(A)\kappa(B)$.

18. Let $A(\alpha) = \begin{bmatrix} 0.1\alpha & 0.1\alpha \\ 1.0 & 2.5 \end{bmatrix}$. Determine α such that the condition number of $A(\alpha)$ is minimized. Use the maximum row norm.

19. Estimate the effect of a disturbance on the right hand side vector $\boldsymbol{b}$ by adding $(\epsilon_1, \epsilon_2)^T$ to $\boldsymbol{b}$, where $|\epsilon_1|,\ |\epsilon_2| \leq 10^{-4}$, when the system of equations is given by $x_1 + 2x_2 = 5,\quad 2x_1 - x_2 = 0$ (use maximum norm for vectors and maximum row norm for matrices).

20. Find a function $C(\epsilon) > 0$ such that $C(\epsilon) \leq \kappa(A)$ using the maximum row norm, when
$$A = \begin{bmatrix} 1 & -1 & 1 \\ -1 & \epsilon & \epsilon \\ 1 & \epsilon & \epsilon \end{bmatrix}$$

III. Iteration Method

21. Find the $n \times n$ matrix B and the n-dimensional vector $\boldsymbol{c}$ such that the Gauss-Seidal method can be written in the form
$$\boldsymbol{x}^{(k+1)} = B\boldsymbol{x}^{(k)} + \boldsymbol{c}, \quad k = 1, 2, \cdots$$

22. Show that the Gauss-Seidal method converges if the coefficient matrix is diagonally dominant.

23. Study the convergence of the Jacobi and the Gauss-Seidel method for the following systems by starting with $\boldsymbol{x}_0 = (0, 0, 0)^T$ and performing three iterations:
    (i)$5x_1 + 2x_2 + x_3 = 0.12,\ 1.75x_1 + 7x_2 + 0.5x_3 = 0.1,\ x_1 + 0.2x_2 + 4.5x_3 = 0.5.$
    (ii)$x_1 - 2x_2 + 2x_3 = 1,\ -x_1 + x_2 - x_3 = 1,\ -2x_1 - 2x_2 + x_3 = 1.$
    (iii)$x_1 + x_2 + 10x_3 = -1,\ 2x_1 + 3x_2 + 5x_3 = -6,\ 3x_1 + 2x_2 - 3x_3 = 4.$
    Check the convergence by obtaining the maximum norm of the residual vector.

24. Use Jacobi method to perform 3 iterations with $\boldsymbol{x}^{(0)} = (0, 0, 0)$ to get $\boldsymbol{x}^{(1)}$, $\boldsymbol{x}^{(2)}$ and $\boldsymbol{x}^{(3)}$ for the system

$$-x_1 + 5x_2 - 2x_3 = 3, \quad x_1 + x_2 - 4x_3 = -9, \quad 4x_1 - x_2 + 2x_3 = 8$$

Compute the maximum norm of the residual error $\boldsymbol{r}_1$, $\boldsymbol{r}_2$ and $\boldsymbol{r}_3$ in $\boldsymbol{x}^{(1)}$, $\boldsymbol{x}^{(2)}$ and $\boldsymbol{x}^{(3)}$, respectively, obtained above. (Observe that the maximum norm of the residual errors increase. Infact, the Jacobi iterative sequence diverges in this case). Interchange the rows suitably in the above system so that the Jacobi iterative sequence converges. Justify your answer without calculating the Jacobi iterations.

25. Study the convergence of the Jacobi and the Gauss-Seidel method for the following system by starting with $\boldsymbol{x}_0 = (0, 0, 0)^T$ and performing 20 iterations (using computer):
$x_1 + 0.5x_2 + 0.5x_3 = 1,\ 0.5x_1 + 1x_2 + 0.5x_3 = 8,\ 0.5x_1 + 0.5x_2 + x_3 = 1.$
Check the convergence by obtaining the maximum norm of the residual vector.

26. For an iterative method $\boldsymbol{x}^{(k)} = B\boldsymbol{x}^{(k-1)} + \boldsymbol{c}$ with an appropriate choice of $\boldsymbol{x}_0$, show that the error $\boldsymbol{e}^{(k)}$ has the estimate

$$\|\boldsymbol{e}^{(k)}\| \leq \frac{\|B\|^{k+1}}{1 - \|B\|}\|\boldsymbol{c}\|.$$

Use this estimate to find the number of iterations needed to compute the solution of the system

$$10x_1 - x_2 + 2x_3 - 3x_4 = 0, \quad x_1 + 10x_2 - x_3 + 2x_4 = 5,$$
$$2x_2 + 3x_2 + 20x_3 - x_4 = -10, \quad 3x_1 + 2x_2 + x_3 + 20x_4 = 15$$

using Jacobi method with absolute error within 10^{-4} and $\boldsymbol{x}^{(0)} = \boldsymbol{c}$ (use maximum norm for vectors and maximum row norm for matrices). **Hint:** In class, we have proved $\|\boldsymbol{e}^{(k)}\| \leq \|B\|^k\|\boldsymbol{e}^{(0)}\|$. But $\|\boldsymbol{e}^{(0)}\| = \|\boldsymbol{x} - \boldsymbol{x}^{(0)}\| \leq \|\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)}\| + \|B\|\|\boldsymbol{x} - \boldsymbol{x}^{(0)}\|$. In this inequality, solve for $\|\boldsymbol{x} - \boldsymbol{x}^{(0)}\|$ and substitute on the RHS of the first inequality to get $\|\boldsymbol{e}^{(k)}\| \leq \dfrac{\|B\|^k}{1 - \|B\|}\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)}\|$. Finally, take $\boldsymbol{x}^{(0)} = \boldsymbol{c}$ to get the desired result.

27. Let $\boldsymbol{x}$ be the solution of the system $A\boldsymbol{x} = \boldsymbol{b}$. Show that the following statements are equivalent:

i. the iterative method

$$\boldsymbol{x}^{(k+1)} = B\boldsymbol{x}^{(k)} + \boldsymbol{c}, \quad k = 1, 2, \cdots$$

is convergent (ie., for any $\boldsymbol{x}^{(0)}$, we have $\boldsymbol{x}^{(k)} \to \boldsymbol{x}$ as $k \to \infty$.

ii. the spectral radius $r_\sigma(B) < 1$.

iii. there exists a induced matrix norm $\|\cdot\|$ such that $\|B\| < 1$.

Hint: Show that (i)⇒(ii)⇒(iii)⇒(i). To prove (i)⇒(ii), first show that $B^{(k)}\boldsymbol{y} \to 0$ as $k \to \infty$, for an arbitrary vector $\boldsymbol{y}$. Then replace this arbitrary vector by an eigen vector of B. In proving (ii)⇒(iii), use the following result (which you don't need to prove): *Let A be a given $n \times n$ matrix and let $\epsilon > 0$. Then there exists an induced matrix norm $\|\cdot\|$ such that $\|A\| \leq r_\sigma(A) + \epsilon$.*

IV. Eigenvalue Problem

28. Let A be an non-singular $n \times n$ matrix with the condition that the eigenvalues λ_i of A satisfy

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$$

and has n linearly independent eigenvectors, $\boldsymbol{v}_i$, $i = 1, \cdots, n$. Let the vector $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$ is such that

$$\boldsymbol{x}^{(0)} = \sum_{j=1}^{n} c_j \boldsymbol{v}_j, \quad c_1 \neq 0.$$

Then find a constant $C > 0$ such that

$$|\lambda_1 - \mu_k| \leq C \left|\frac{\lambda_2}{\lambda_1}\right|^k,$$

where μ_k is as defined in the power method and $k = 1, 2, \cdots$.

29. The matrix

$$A = \begin{bmatrix} 0.7825 & 0.8154 & -0.1897 \\ -0.3676 & 2.2462 & -0.0573 \\ -0.1838 & 0.1231 & 1.9714 \end{bmatrix}$$

has eigenvalues $\lambda_1 = 2$, $\lambda = 2$ and $\lambda_3 = 1$. Does the power method converge for the above matrix? Justify your answer. Perform 5 iterations starting from the initial guess $\boldsymbol{x}^{(0)} = (1, 3, 6)$ to verify your answer.

30. The matrix
$$A = \begin{bmatrix} 2 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}$$
has eigenvalues $\lambda_1 = 2$, $\lambda = 1$ and $\lambda_3 = 1$ and the corresponding eigen vectors may be taken as $\boldsymbol{v}_1 = (1, 2, 3)^T$, $\boldsymbol{v}_2 = (0, 1, 2)^T$ and $\boldsymbol{v}_3 = (0, 2, 1)^T$. Perform 3 iterations to find the eigenvalue and the corresponding eigen vector to which the power method converge when we start the iteration with the initial guess $\boldsymbol{x}^{(0)} = (0, 0.5, 0.75)^T$. Without performing the iteration, find the eigenvalue and the corresponding eigen vector to which the power method converge when we start the iteration with the initial guess $\boldsymbol{x}^{(0)} = (0.001, 0.5, 0.75)^T$. Justify your answer.

31. The matrix
$$A = \begin{bmatrix} 5.4 & 0 & 0 \\ -113.0233 & -0.5388 & -0.6461 \\ -46.0567 & -6.4358 & -0.9612 \end{bmatrix}$$
has eigenvalues $\lambda_1 = 5.4$, $\lambda = 1.3$ and $\lambda_3 = -2.8$ with corresponding eigen vectors $\boldsymbol{v}_1 = (0.2, -4.1, 2.7)^T$, $\boldsymbol{v}_2 = (0, 1.3, -3.7)^T$ and $\boldsymbol{v}_3 = (0, 2.6, 9.1)^T$. To which eigenvalue and the corresponding eigen vector does the power method converge if we start with the initial guess $\boldsymbol{x}^{(0)} = (0, 1, 1)$? Justify your answer.

32. Use Gerschgorin's theorem to the following matrix and determine the intervals in which the eigenvalues lie.
$$A = \begin{bmatrix} 0.5 & 0 & 0.2 \\ 0 & 3.15 & -1 \\ 0.57 & 0 & -7.43 \end{bmatrix}$$
Can power method be used for this matrix? Justify your answer. Use Power method to compute the eigenvalue which is largest in the absolute value and the corresponding eigenvector each of the above matrix.

V. **Computer Program**

33. Write a computer program (in any programming language that you know) to compute an eigenvalue and the corresponding eigen vector of a given $n \times n$ matrix A.

Use your program for the following matrices. In each case plot a graph with x axis as the number of iterations and y axis as the eigenvalue obtained in that iteration.

i. $A = \begin{bmatrix} 1.2357 & -0.5714 & 0.0024 \\ 0.5029 & -0.0557 & -0.0638 \\ 0.78 & -1.56 & 0.88 \end{bmatrix}$, $\boldsymbol{x}^{(0)} = (1, 1, 1)^T$. Perform 110 iteration.(Eigen values are 0.1, 0.95, 1.01 and the corresponding eigenvectors may be taken as $(1, 2, 3)^T$, $(2, 1, 0)^T$ and $(5, 2, 6)^T$.)

ii. $A = \begin{bmatrix} 0.5029 & 0.0051 & -0.0130 \\ 0.8663 & 2.0160 & -3.8984 \\ 0.5775 & 1.0107 & -2.0989 \end{bmatrix}$, $\boldsymbol{x}^{(0)} = (1, 1, 1)^T$. Perform 50 iteration.(Eigen values are -0.58, 0.5, 0.5 and the corresponding eigenvectors may be taken as $(1, 0.2, 0.3)^T$, $(0.1, 0.2, 0.1)^T$ and $(0.001, 0.3, 0.2)^T$.)

iii. $A = \begin{bmatrix} -0.5088 & -0.0025 & 0.0038 \\ -2.0425 & 0.3050 & 0.4125 \\ -1.3588 & 0.5375 & -0.2263 \end{bmatrix}$, $\boldsymbol{x}^{(0)} = (1, 1, 1)^T$. Perform 70 iteration.(Eigen values are -0.5, -0.51, 0.58 and the corresponding eigenvectors may be taken as $(1, 1, 3)^T$, $(1, 2, 1)^T$ and $(0, 3, 2)^T$.)

iv. $A = \begin{bmatrix} -0.5080 & -0.0040 & 0.0060 \\ -1.8358 & 0.0986 & 0.6186 \\ -1.2212 & 0.4004 & -0.0896 \end{bmatrix}$, $\boldsymbol{x}^{(0)} = (1,1,1)^T$. Perform as many as iterations as you wish.(Eigen values are -0.5, -0.51, 0.511 and the corresponding eigenvectors may be taken as $(1,1,2)^T$, $(1,2,1)^T$ and $(0,3,2)^T$.)

4

Nonlinear Equations

One of the most frequently occuring problems in scientific work is to find the roots of equations of the form

$$f(x) = 0. \tag{4.1}$$

In this chapter, we introduce various iterative methods to obtain an approximate solution for the equation (4.1).

By approximate solution to (4.1) we mean a point x^* for which the function $f(x)$ is very near to zero, ie., $f(x^*) \approx 0$.

In what follows, we always assume that $f(x)$ is continuously differentiable real-valued function of a real variable x. We further assume that the equation (4.1) has only **isolated roots**, that is, for each root of (4.1) there is a neighbourhood which does not contain any other roots of the equation.

The key idea in approximating the isolated real roots of (4.1) consisting of two steps:

I. **Initial guess**: Establishing the smallest possible intervals $[a, b]$ constaining one and only one root of the equation (4.1). Take one point $x_0 \in [a, b]$ as an approximation to the root of (4.1).

II. **Improving the value of the root** If this initial guess x_0 is not in desired accuracy, then devise a method to improve the accuracy.

This process of improving the value of the root is called the *iterative process* and such methods are called **iterative methods**. A general form of an iterative method may be written as

$$x_{n+1} = T(x_n), \quad n = 0, 1, \cdots \tag{4.2}$$

where T is a real-valued function called an **iteration function**. In the process of iterating a solution, we obtain a sequence of numbers $\{x_n\}$ which are expected to converge to the root of (4.1).

Definition 4.1 (Convergence).

A sequence of iterates $\{x_n\}$ is said to converge with **order** $p \geq 1$ *to a point x^* if*

$$|x_{n+1} - x^*| \leq c|x_n - x^*|^p, \quad n \geq 0 \tag{4.3}$$

for some constant $c > 0$.

Remark 4.2. If $p = 1$, the sequence is said to **converge linearly** to x^*, if $p = 2$, the sequence is said to **converge quadratically** and so on. □

4.1 Fixed-Point Iteration Method

The idea of this method is to rewrite the equation (4.1) in the form

$$x = g(x) \tag{4.4}$$

so that any solution of (4.4) ie., any **fixed point** of $g(x)$ is a solution of (4.1).

Example 4.3. The equation $x^2 - x - 2 = 0$ can be written as
1. $x = x^2 - 2$
2. $x = \sqrt{x+2}$
3. $x = 1 + \frac{2}{x}$
and so on. □

The **fixed-point iteration method** is to set an iterative process of the form (4.2) with iteration function $g(x)$. Note that for a given nonlinear equation, this iteration function is not unique. Once the iteration function is chosen, then the method is defined as follows:

Step 1: Choose an initial guess x_0.

Step 2: Define the iteration methods as

$$x_{n+1} = g(x_n), \quad n = 0, 1, \cdots \tag{4.5}$$

The crucial point in this method is to choose a good iteration function $g(x)$. A good iteration function should satisfy the following properties:

I. For the given starting point x_0, the successive approximation x_n given by (4.5) can be calculated.
II. The sequence $x_1, x_2, \cdots$ converges to some point ξ.
III. The limit ξ is a fixed point of $g(x)$, ie., $\xi = g(\xi)$.

The first property is the most needed one as illustrated in the following example.

Example 4.4. Consider the equation $x^2 - x = 0$. We can take $x = \pm\sqrt{x}$ and suppose we define $g(x) = -\sqrt{x}$. Since $g(x)$ is defined only for $x > 0$, we have to choose $x_0 > 0$. For this value of x_0, we have $g(x_0) < 0$ and therefore, x_1 cannot be calculated. □

Therefore, the choice of $g(x)$ has to be made carefully so that the sequece of iterates can be calculated. How to choose such a iteration function $g(x)$? Since, we expect $x = g(x)$, we have to define $g(x)$ in such a way that this value should again belong to the domain of g. That is,

Assumption 1: $a \le g(x) \le b$ for all $a \le x \le b$.

It follows that if $a \le x_0 \le b$, then for all n, $x_n \in [a, b]$ and therefore $x_{n+1} = g(x_n)$ is defined and belongs to $[a, b]$.

Let us now discuss about the point 3. This is a natural expectation since the expression $x = g(x)$, which is the solution of the required equation is precisely the definition of a fixed point. To achieve this, we need $g(x)$ to be a continuous function. For if $x_n \to x^*$ then

$$x^* = \lim_{n\to\infty} x_n = \lim_{n\to\infty} g(x_{n-1}) = g(\lim_{n\to\infty} x_{n-1}) = g(x^*)$$

Therefore, we need

Assumption 2: The function $g(x)$ is continuous.

Let us now discuss point 2. This point is well understood geometrically. The figure (a) and (c) illustrated the convergence of the fixed-point iterations whereas the figures (b) and (d) illustrated the diverging iterations. In this geometrical observation, we see that when $g'(x) < 1$, we have convergence and otherwise, we have divergence. Therefore, we make the asspumption

Assumption 3: The iteration function $g(x)$ is differentiable on $I = [a, b]$. Further, there exists a constant $0 < K < 1$ such that

$$|g'(x)| \le K, \quad x \in I. \tag{4.6}$$

Theorem 4.5. *Assume that $g(x)$ is continuously differentiable on $[a, b]$, and $a \le g(x) \le b$ with*

$$\lambda = \max_{a\le x\le b} |g'(x)| < 1. \tag{4.7}$$

Then

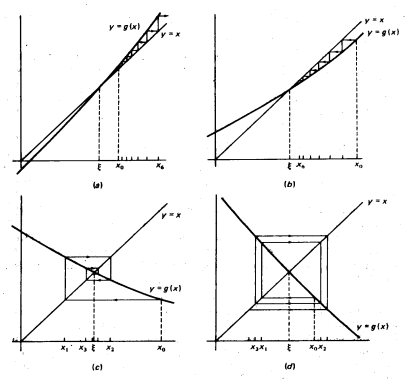


Fig. 4.1. Fixed-point Iteration Procedure.

I. $x = g(x)$ *has a unique solution* x^* *in* I.

II. For any choice of $x_0 \in I$, *with* $x_{n+1} = g(x_n)$, $n = 0, 1, \cdots$,

$$\lim_{n\to\infty} x_n = x^*.$$

III. We further have

$$|x_n - x^*| \le \lambda^n |x_0 - x^*| \le \frac{\lambda^n}{1-\lambda}|x_1 - x_0| \tag{4.8}$$

and

$$\lim_{n\to\infty} \frac{x^* - x_{n+1}}{x^* - x_n} = g'(x^*). \tag{4.9}$$

Proof. Proof for 1 is omitted. To examine the convergence of the iterates x_n, we note that

$$|x^* - x_{n+1}| = |g(x^*) - g(x_n)| \le \lambda |x^* - x_n| \quad \text{(by Mean-value theorem and (4.6))}$$

By induction, we have

$$|x^* - x_{n+1}| \le \lambda^n |x_0 - x^*|, \quad n = 0, 1, \cdots.$$

Since, as $n \to \infty$, $\lambda^n \to 0$, we have $x_n \to x^*$. Further, we have

$$|x_0 - x^*| = |x_0 - x_1 + x_1 - x^*| \le |x_0 - x_1| + |x_1 - x^*| \le \lambda |x_0 - x^*| + |x_0 - x_1|.$$

Then solving for $|x_0 - x^*|$, we get (4.8).

Now we will prove the rate of convergence (4.9). From Mean-value theorem

$$x^* - x_{n+1} = g(x^*) - g(x_n) = g'(\xi_n)(x^* - x_n), \quad n = 0, 1, \cdots.$$

with ξ_n an unknown point between x^* and x_n. Since $x_n \to x^*$, we must have $\xi_n \to x^*$ and therefore,

$$\lim_{n\to\infty} \frac{x^* - x_{n+1}}{x^* - x_n} = \lim_{n\to\infty} g'(\xi_n) = g'(x^*).$$

This completes the proof. □

Example 4.6. Consider the equation $\sin x + x^2 - 1 = 0$. Take the initial interval as $[0, 1]$. There are three possible choices for the iteration function, namely,

I. $g_1(x) = \sin^{-1}(1 - x^2)$,
II. $g_2(x) = -\sqrt{1 - \sin x}$,
III. $g_3(x) = \sqrt{1 - \sin x}$,

Here we have $g_1'(x) = \frac{-2}{\sqrt{2-x^2}}$. We can see that $|g_1'(x)| > 1$. Taking $x_0 = 0.8$ and denoting the absolute error as ϵ, we have

n	$g_1(x)$	ϵ
0	0.368268	0.268465
1	1.043914	0.407181
2	-0.089877	0.726610
3	1.443606	0.806873

The sequence of iterations is diverging as expected.

If we take $g_2(x)$, clearly the assumption 1 is violated and therefore is not suitable for the iteration process. Let us take $g_3(x)$. Here, we have $g_3'(x) = \frac{-\cos x}{\sqrt{1-\sin x}}$. Therefore,

$$|g_3'(x)| = \frac{\sqrt{1-\sin^2 x}}{2\sqrt{1-\sin x}} = \frac{\sqrt{1+\sin x}}{2} \leq \frac{1}{\sqrt{2}} < 1.$$

Taking $x_0 = 0.8$ and denoting the absolute error as ϵ, we have

n	$g_3(x)$	ϵ
0	0.531643	0.105090
1	0.702175	0.065442
2	0.595080	0.041653
3	0.662891	0.026158

The sequence is converging. □

When to stop the iteration?

Assume a positive number ϵ which is very small. Then, one of the following conditions may be used:

Condition 1: After each iteration check the inequality

$$|x_n - x_{n-k}| < \epsilon$$

for some fixed positive integer k. If this inequality is satisfied, the iteration can be stopped.

Condition 2: Another condition may be to check

$$|f(x_n)| < \epsilon.$$

This error is sometime called the **residual error** for the equation $f(x) = 0$.

4.2 Bisection Method

Assume that $f(x)$ is continuous on a given interval $[a, b]$ and that is also satisfies $f(a)f(b) < 0$ with $f(a) \neq 0$ and $f(b) \neq 0$. Using the intermediate value theorem, we can see that the function $f(x)$ has atleast one root in $[a, b]$. We assume that there is only one root for the equation (4.1) in the interval $[a, b]$. The Bisection includes the following steps:

Step 1: Given an initial interval $[a_0, b_0]$, set $n = 0$.
Step 2: Define $c_{n+1} = (a_n + b_n)/2$, the midpoint of the interval $[a_n, b_n]$.
Step 3:
If $f(a_n)f(c_{n+1}) = 0$, then $x^* = c_{n+1}$ is the root.
If $f(a_n)f(c_{n+1}) < 0$, then take $a_{n+1} = a_n$, $b_{n+1} = c_{n+1}$ and the root $x^* \in [a_{n+1}, b_{n+1}]$.
If $f(a_n)f(c_{n+1}) > 0$, then take $a_{n+1} = c_{n+1}$, $b_{n+1} = b_n$ and the root $x^* \in [a_{n+1}, b_{n+1}]$.
Step 4: If the root is not obtained in step 3, then find the length of the new reduced interval $[a_{n+1}, b_{n+1}]$. If the length of the interval $b_{n+1} - a_{n+1}$ is less than a prescribed positive number ϵ, then take the mid point of this interval ($x^* = (b_{n+1} + a_{n+1})/2$) as the approximate root of the equation (4.1), otherwise go to step 2.

The following theorem gives the convergence and error for the bisection method.

Theorem 4.7 (Convergence and Error of Bisection Method).
Let $[a_0, b_0] = [a, b]$ be the initial interval, with $f(a)f(b) < 0$. Define the approximate root as $x_n = (b_{n-1} + a_{n-1})/2$. Then there exists a root $x^* \in [a, b]$ such that

$$|x_n - x^*| \leq \left(\frac{1}{2}\right)^n (b - a). \tag{4.10}$$

Moveover, to achieve accuracy of $|x_n - x^*| \leq \epsilon$, it suffices to take

$$n \geq \frac{\log(b - a) - \log \epsilon}{\log 2}. \tag{4.11}$$

Proof. It is obvious that

$$b_n - a_n = \frac{1}{2}(b_{n-1} - a_{n-1}),$$

which implies that

$$b_n - a_n = \left(\frac{1}{2}\right)^n (b_0 - a_0).$$

Therefore,

$$|x_n - x^*| \leq \frac{1}{2}(b_{n-1} - a_{n-1}) = \frac{1}{2}\left(\frac{1}{2}\right)^{n-1} (b_0 - a_0) = \left(\frac{1}{2}\right)^n (b_0 - a_0),$$

which proves the estimate. To obtain the bound, we observe that

$$\left(\frac{1}{2}\right)^n (b - a) \leq \epsilon.$$

Taking log on both sides, we get the desired bound. □

Example 4.8. Consider the equation $\sin x + x^2 - 1 = 0$. Take the initial interval as $[0, 1]$. That is $a_0 = 0$, $b_0 = 1$. If the permissible absolute error is 0.125, ie. $|x_n - x^*| \leq 0.125$, then by (4.11), we must perform atleast

$$n \geq \frac{\log(1) - \log(0.125)}{\log 2} = 3$$

number of iterations. Let us perform the iterations.

$$a_0 = 0, b_0 = 1; c_1 = 0.5, f(c_1) = -0.27 < 0 \Rightarrow a_1 = 0.5, b_1 = 1.$$
$$a_1 = 0.5, b_1 = 1; c_2 = 0.75, f(c_2) = 0.24 > 0 \Rightarrow a_2 = 0.5, b_2 = 0.75.$$
$$a_2 = 0.5, b_2 = 0.75; c_3 = 0.625, f(c_3) = -0.024 < 0 \Rightarrow a_3 = 0.625, b_3 = 0.75.$$

Since $|a_1 - b_1| = 0.125$ and $|x_3 - x^*| \leq |a_1 - b_1| = 0.125$, we can stop the iteration here. We may take the approximate solution for the equation as $x^* \approx 0.6875$. The true value is $x^* \approx 0.636733$. Therefore, the absolute error is 0.05. □

4.3 Secant Method

Secant method is one of the most efficient method among all **regula-falsi** methods. Let us first explain the regula-falsi method and given the modification in this method which leads to **secant method**.

The regula-falsi method is closely related to the bisection method introduced in section 5.2. Recall the bisection method is to subdivide the interval $[a, b]$ in which the root lies into two parts, take the part of the interval which still holds the root and discard the other part of the interval. Although the bisection method always converges to the solution, the convergence is sometime very slow in the sense that if the root is very close to one of the boundary points (ie.,. a and b) of the interval. In such a situation, instead of taking the midpoint of the interval, we take the weighted average of $f(x)$ given by

$$w = \frac{f(b)a - f(a)b}{f(b) - f(a)}$$

Example 4.9. Consider the equation $f(x) := x^3 - x - 1 = 0$. Clearly, $f(1) = -1 < 0$ and $f(2) = 5 > 0$. Thus, we can take the initial interval for the bisection method as $[1, 2]$. But here we observe that $f(1)$ is more close to 0 than $f(2)$. So, it is very likely that the root x^* of the given equation is closer to $x = 1$ than $x = 2$. Rather, the weighted of $f(x)$ is

$$w = \frac{5 \times 1 + 1 \times 2}{6} = 1.16666 \cdots .$$

Now $f(w) = -0.578703 \cdots < 0 < 5 = f(2)$. Repeating this process once again, we get

$$w = \frac{5 \times (1.1666 \cdots) + (0.578703 \cdots) \times 2}{5.578703 \cdots} = 1.253112 \cdots$$

from which we have $f(w) = -0.285363 \cdots < 0 < 5 = f(2)$.

Such an algorithm is called the **regula-falsi method**. The algorithm is as follows

Step 1: Given an initial interval $[a_0, b_0]$, set $n = 0$.
Step 2: Define

$$w_{n+1} = \frac{f(b_n)a_n - f(a_n)b_n}{f(b_n) - f(a_n)}. \tag{4.12}$$

Step 3:
If $f(a_n)f(w_{n+1}) = 0$, then $x^* = c_{n+1}$ is the root.
If $f(a_n)f(w_{n+1}) < 0$, then take $a_{n+1} = a_n$, $b_{n+1} = w_{n+1}$ and the root $x^* \in [a_{n+1}, b_{n+1}]$.
If $f(a_n)f(w_{n+1}) > 0$, then take $a_{n+1} = w_{n+1}$, $b_{n+1} = b_n$ and the root $x^* \in [a_{n+1}, b_{n+1}]$.
Step 4: If the root is not obtained in step 3, then check the condition

$$|f(w_{n+1})| < \epsilon$$

for some pre-assigned positive quantity ϵ. If the condition is satisfied, then take the weight of the next iteration as the approximate root of the equation (4.1). If this condition is not satisfied, then repeate the step 2.

Note that the weighted average is the point at which the secant joining the points $(a, f(a))$ and $(b, f(b))$ intersects the x-axis. Let us derive this weighted average now. The secant line is given by

$$s(x) = \frac{f(a)(x - b) + f(b)(a - x)}{a - b} = \frac{(f(a) - f(b))x + f(b)a - f(a)b}{a - b}.$$

The slope of this line is

$$s'(x) = \frac{f(a) - f(b)}{a - b}.$$

On the other hand, if w is the point of intersection of the secant with x-axis, then the line joining $(w, 0)$ and $(b, f(b))$ is given by

$$l(x) = \frac{f(b)(w - x)}{w - b},$$

whose slope is

$$l'(x) = \frac{-f(b)}{w - b}.$$

Equating these slopes, we get

$$\frac{f(a) - f(b)}{a - b} = \frac{-f(b)}{w - b} \Rightarrow w = \frac{f(b)a - f(a)b}{f(b) - f(a)}$$

as expected.

The regula-falsi method can be improved in several ways. The popular one is the **secant method**. Given initial values x_0 and x_1 (not necessarily on the either side of the root) the iteration for secant method is given by

$$x_{n+1} = \frac{f(x_n)x_{n-1} - f(x_{n-1})x_n}{f(x_n) - f(x_{n-1})}. \tag{4.13}$$

This expression can also be written as

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \tag{4.12a}$$

Example 4.10. Consider the equation $\sin x + x^2 - 1 = 0$. Let $x_0 = 0$, $x_1 = 1$. Then the iterations from the secant method are given by

n	x_n	ϵ
2	0.543044	0.093689
3	0.626623	0.010110
4	0.637072	0.000339
5	0.636732	0.000001

Recall that the exact solution is $x^* \approx 0.636733$. Obviously, the secant method is much faster than both bisection and fixed-point iteration methods. □

The order of convergence of secant method is

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^r} = \left|\frac{f''(x^*)}{2f'(x^*)}\right|^{r-1}. \tag{4.14}$$

where $r = (\sqrt{5}+1)/2 \approx 1.62$.

4.4 Newton-Raphson Method

If $f(x)$ is differentiable, then on replacing in (4.12a) the slope of the secant by the slope of the tangent at x_n, one gets the iteration formula

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{4.15}$$

of **Newton-Raphson Method**.

Example 4.11. Consider the equation $\sin x + x^2 - 1 = 0$. Let $x_0 = 1$. Then the iterations from the Newton-Raphson method gives

n	x_n	ϵ
1	0.668752	0.032019
2	0.637068	0.000335
3	0.636733	0.000000

Recall that the exact solution is $x^* \approx 0.636733$. Obviously, the Newton-Raphson method is much faster than both bisection and fixed-point iteration methods. □

Remark 4.12. We will derive analytically the Newton-Raphson method. The Taylor polynomial of degree $n = 1$ with remainder is given by

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{(x - x_0)^2}{2!}f''(\xi),$$

where ξ lies somewhere between x_0 and x. Substituting $x = x^*$ into the above equation, we get

$$0 = f(x_0) + f'(x_0)(x^* - x_0) + \frac{(x^* - x_0)^2}{2!}f''(\xi).$$

When x_0 is very close to x^*, then the last term in the above equation is smaller when compared to the other two terms on the RHS and therefore, can be neglected. The remaining terms read

$$f(x_0) + f'(x_0)(x^* - x_0) \approx 0.$$

Solving for x^*, and using the notation x_1 for this new approximate solution, we get

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

When x_{n-1} is used in place of x_0, we get the general formula (4.15). □

Remark 4.13. Let us define

$$g(x) = x - \frac{f(x)}{f'(x)}. \tag{4.16}$$

Since $f(x^*) = 0$, it is easy to see that $g(x^*) = x^*$ and therefore finding root for the equation $f(x) = 0$ using Newton-Raphson method is equivalent to finding the fixed point of the function $g(x)$. □

Theorem 4.14. *Assume that $f \in C^2[a, b]$ and there exists a number $x^* \in [a, b]$, where $f(x^*) = 0$. If $f'(x^*) \neq 0$, then there exists a $\delta > 0$ such that the sequence $\{x_n\}$ defined by the iteration (4.15) for $n = 1, 2, \cdots$ will converge to x^* for any initial approximation $x_0 \in [x^* - \delta, x^* + \delta]$.*

Further, we have

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = \frac{|f''(x^*)|}{2|f'(x^*)|}. \tag{4.17}$$

Proof. Consider the fixed-point iteration function $g(x)$ defined by (4.16). Now,

$$g'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)f''(x)}{(f'(x))^2}.$$

By hypothesis, $f(x^*) = 0$ and therefore $g'(x^*) = 0$. Since $g(x)$ is continuous, it is possible to find a $\delta > 0$ so that $|g'(x)| < 1$ for all $x \in (x^* - \delta, x^* + \delta)$. Therefore, a sufficient condition for the initial guess x_0 to give a convergent sequence is that $x_0 \in (x^* - \delta, x^* + \delta)$. and that δ be chosen so that

$$\frac{|f(x)f''(x)|}{|f'(x)|^2} < 1 \tag{4.18}$$

for all $x \in (x^* - \delta, x^* + \delta)$.

By Taylor's theorem, we have

$$f(x^*) = f(x_n) + (x^* - x_n)f'(x_n) + \frac{(x^* - x_n)^2}{2!}f''(\xi_n).$$

with ξ_n between x^* and x_n. Note that $f(x^*) = 0$ by assumption and then divide $f'(x_n)$ to obtain

$$\begin{aligned} 0 &= \frac{f(x_n)}{f'(x_n)} + x^* - x_n + (x^* - x_n)^2 \frac{f''(\xi_n)}{2f'(x_n)} \\ &= x_n - x_{n+1} + x^* - x_n + (x^* - x_n)^2 \frac{f''(\xi_n)}{2f'(x_n)} \end{aligned}$$

By taking limit $n \to \infty$, we get the result. □

To examine the order of convergence of the Newton-Raphson method, we need the following definition.

Example 4.15 (Quadratic convergence at an isolated root). Start with $x_0 = -2.4$ and use Newton-Raphson iteration to find the root $x^* = -2.0$ of the polynomial $f(x) = x^3 - 3x + 2$. The iteration formula is

$$x_{k+1} = g(x_k) = \frac{2x_k^3 - 2}{3x_k^2 - 3}.$$

Verify that $|x^* - x_{n+1}|/|x^* - x_n|^2 \approx 2/3$. □

Pitfalls:

I. If $f'(x_n) = 0$ for some n, the method can no longer be applied.

II. If $f(x)$ has no real root, then there is no indication by the method and the iteration may simply oscillates. For example compute the Newton-Raphson iteration for $f(x) = x^2 - 4x + 5$.

III. If the equation $f(x) = 0$ has more than one root and we are specific about capturing a particular root (say the smallest positive root). Then we have to be careful in choosing the initial guess. If the initial guess is far away from the expected root, then there is a danger that the iteration converges to another root of the equation. This usually happens when the slope $f'(x_0)$ is small and the tangent line to the curve $y = f(x)$ is nearly horizontal.

For example, if $f(x) = \cos x$ and we seek the root $p = \pi/2$ and start with $p_o = 3$, calculation reveals that $x_1 = -4.01525$, $x_2 = -4.85266$ and so on and the iteration converges to $x = -4.71238898 \approx -3\pi/2$.

IV. Suppose that $f(x)$ is positive and monotone decreasing on an unbounded interval $[a, \infty)$ and $x_0 > a$. Then the sequence might diverge.

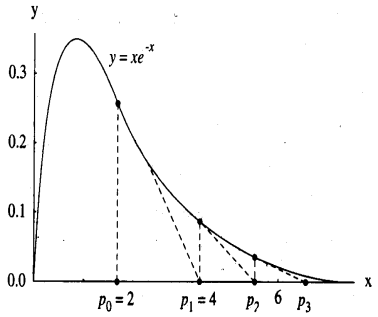


Fig. 4.2. Newton-Raphson Method for $f(x) = xe^{-x}$.

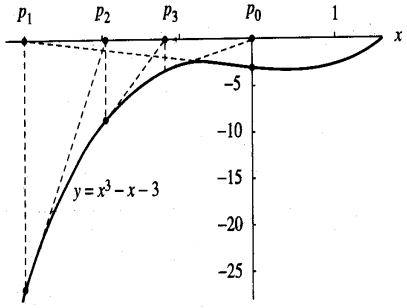


Fig. 4.3. Newton-Raphson Method for $f(x) = x^3 - x - 3$.

For example, if $f(x) = xe^{-x}$ and $x_0 = 2$, then

$$x_1 = 4.0, \quad x_2 = 5.333333..., \quad \cdots, p_{15} = 19.72354..., \cdots .$$

and the sequence diverges to $+\infty$. This particular function has another suprising problem. The value of $f(x)$ goes to zero rapidly as x gets large, for example $f(x_{15}) = 0.0000000536$, and it is possible that p_15 could be mistaken for a root (as per the residual error).

V. The method can stuck in a cycle. For example $f(x) = x^3 - x - 3$ and the initial approximation is $x_0 = 0$. Then the sequence is

$$x_1 = -3.00, \quad x_2 = -1.961538, \quad x_3 = -1.147176, \quad x_4 = -0.006579,$$

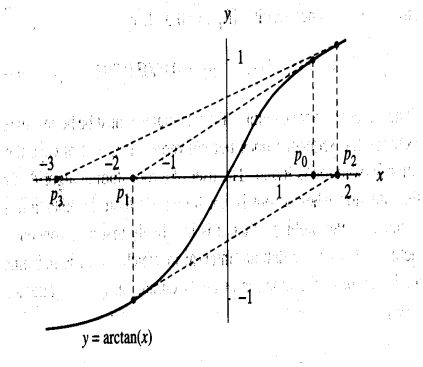


Fig. 4.4. Newton-Raphson Method for $f(x) = \tan^{-1}(x)$.

$$x_5 = -3.000389, \quad x_6 = -1.961818, \quad x_7 = -1.147430, \cdots$$

and we are stuck in a cycle where $x_{n+4} \approx x_k$ for $k = 0, 1, \cdots$. But if we start with a value x_0 sufficiently close with the root $x^* \approx 1.6717$, then the convergence is obtained (check!!!).

VI. When $|g'(x)| \geq 1$ on an interval containing the root x^*, there is a chance of divergent oscillation.

For example, let $f(x) = \tan^{-1}(x)$. The funtion $g(x) = x-(1+x^2)\tan^{-1}(x)$ and $g'(x) = -2x\tan^{-1}(x)$. If we start with the value $x_0 = 1.45$, then

$$x_1 = -1.55 - 26, \quad x_2 = 1.845932, \quad x_3 = -2.88911 \cdots .$$

But if we start with $x_0 = 0.5$, then the iteration converges to the root $x = 0$.

4.5 System of Nonlinear Equations

Let us present the theory for two equations and the theory for any finite number of equation can be done in a similar way. Consider the system of two nonlinear equations

$$f_1(x_1, x_2) = 0, \quad f_2(x_1, x_2) = 0. \tag{4.19}$$

In vector notation, we write as

$$\boldsymbol{f}(\boldsymbol{x}) = 0, \quad \boldsymbol{x} = (x_1, x_2)^T, \quad \boldsymbol{f}(\boldsymbol{x}) = (f_1(x_1, x_2), f_2(x_1, x_2))^T.$$

We assume that this system admits an isolated root $\boldsymbol{x}^* = (x_1^*, x_2^*)^T$.

For fixed point iteration method, we define the iterative sequence as

$$x_{1,n+1} = g_1(x_{1,n}, x_{2,n}), \quad x_{2,n+1} = g_2(x_{1,n}, x_{2,n}), \tag{4.20}$$

where g_1 and g_2 are iterative functions. In vector notation, we write this as

$$\boldsymbol{x}_{n+1} = \boldsymbol{g}(\boldsymbol{x}_n), \quad n = 0, 1, \cdots .$$

with $\boldsymbol{x}_n = (x_{1,n}, x_{2,n})^T$ and $\boldsymbol{g}(\boldsymbol{x}) = (g_1(x_1, x_2), g_2(x_1, x_2))^T$. Convergence of the fixed point iteration method depends on the choice of the iterative function $\boldsymbol{g}$.

To analyze the convergence of (4.20) use the following identities

$$x_1^* = g_1(x_1^*, x_2^*), \quad x_2^* = g_2^*(x_1^*, x_2^*), \tag{4.21}$$

where $\boldsymbol{x}^* = (x_1^*, x_2^*)$ is an isolated root of (4.19). The Taylor formula gives

$$g_i(x_1^*, x_2^*) = g_i(x_{1,n}, x_{2,n}) + \frac{\partial g_i(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_1}(x_1^* - x_{1,n}) + \frac{\partial g_i(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_2}(x_2^* - x_{2,n}), \quad i = 1, 2,$$

where the vector $(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})$ lie on the line segment joining $\boldsymbol{x}^*$ and $\boldsymbol{x}_n$. From (4.21) and (4.20), we have

$$x_i^* - x_{i,n+1} = \frac{\partial g_i(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_1}(x_1^* - x_{1,n}) + \frac{\partial g_i(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_2}(x_2^* - x_{2,n}), \quad i = 1, 2.$$

In matrix notation, we have

$$\begin{pmatrix} x_1^* - x_{i,n+1} \\ x_2^* - x_{i,n+1} \end{pmatrix} = \begin{pmatrix} \frac{\partial g_1(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_1} & \frac{\partial g_1(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_2} \\ \frac{\partial g_2(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_1} & \frac{\partial g_2(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_2} \end{pmatrix} \begin{pmatrix} x_1^* - x_{i,n} \\ x_2^* - x_{i,n} \end{pmatrix}.$$

We denote the 2×2 matrix on the RHS of the above equation as

$$G_n = \begin{pmatrix} \frac{\partial g_1(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_1} & \frac{\partial g_1(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_2} \\ \frac{\partial g_2(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_1} & \frac{\partial g_2(\xi_{1,n}^{(i)}, \xi_{2,n}^{(i)})}{\partial x_2} \end{pmatrix}$$

and recall that this matrix resumbles the Jacobian matrix of the function $\boldsymbol{g} = (g_1, g_2)$ given by

$$G(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial g_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial g_1(\boldsymbol{x})}{\partial x_2} \\ \frac{\partial g_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial g_2(\boldsymbol{x})}{\partial x_2} \end{pmatrix}.$$

In matrix notation, we can write the above equation as

$$\boldsymbol{x}^* - \boldsymbol{x}_{n+1} = G_n(\boldsymbol{x}^* - \boldsymbol{x}_n).$$

We state the following convergence theorem without proof.

Theorem 4.16. *Let D be a closed, bounded and convex set in the plane (we say D is convex if for any two points in D, the line segment joining them is also in D). Assume that the components of $\boldsymbol{g}(\boldsymbol{x})$ are continuously differentiable at all points of D, and further assume*
(a) $\boldsymbol{g}(D) \subset D$,
(b) $\lambda = \max_{\boldsymbol{x} \in D} \|G(\boldsymbol{x})\|_\infty < 1$.
Then

I. $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{x})$ has a unique solution $\boldsymbol{x}^* \in D$.

II. For any initial point $\boldsymbol{x}_0 \in D$, the iteration

$$\boldsymbol{x}_{n+1} = \boldsymbol{g}(\boldsymbol{x}_n)$$

converges to $\boldsymbol{x}^* \in D$.

III. $\|\boldsymbol{x}^* - \boldsymbol{x}_{n+1}\| \leq (\|G(\boldsymbol{x}^*)\|_\infty + \epsilon_n) \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty$ with $\epsilon \to 0$ as $n \to \infty$.

Proof: Omitted.

We will now see how to choose $\boldsymbol{g}$ for a given system of nonlinear equations (4.19), so as to have a faster convergence?

Let A be a constant non-singular matrix of order 2×2. We rewrite (4.19) as

$$\boldsymbol{x} = \boldsymbol{x} + A\boldsymbol{f}(\boldsymbol{x}) =: \boldsymbol{g}(\boldsymbol{x}).$$

The Jacobian matrix of $\boldsymbol{g}(\boldsymbol{x})$ is

$$G(\boldsymbol{x}) = I + AF(\boldsymbol{x}),$$

where $F(\boldsymbol{x})$ is the Jacobian matrix of $\boldsymbol{f}(\boldsymbol{x})$ given by

$$F(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_2} \\ \frac{\partial f_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_2(\boldsymbol{x})}{\partial x_2} \end{pmatrix}.$$

Choose A such that

$$\|G(\boldsymbol{x})\|_\infty < 1, \quad \boldsymbol{x} \in D.$$

Practically this may not be possible. So, for a given $\boldsymbol{x}_0$ choose A such that

$$\|G(\boldsymbol{x}_0)\|_\infty < 1.$$

For rapid convergence, we can choose A such that

$$\|G(\boldsymbol{x}_0)\|_\infty = 0,$$

for sufficiently close $\boldsymbol{x}_0$ to $\boldsymbol{x}^*$. This is equivalent to taking A as

$$A = -(F(\boldsymbol{x}_0))^{-1}.$$

More rapid convergence is obtained when we choose

$$A = -(F(\boldsymbol{x}_n))^{-1}.$$

The respective method is the well-known **Newton's method** given by

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - (F(\boldsymbol{x}_n))^{-1} f(\boldsymbol{x}_n), \quad n = 0, 1, \cdots \tag{4.22}$$

Example 4.17. Consider solving the system

$$\begin{aligned} f_1 &= 3x_1^2 + 4x_2^2 - 1 = 0, \\ f_2 &= x_2^3 - 8x_1^3 - 1 = 0. \end{aligned}$$

with $\boldsymbol{x}_0 = (-0.5, 0.25)$. The Jacobian of the given system is

$$F(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_2} \\ \frac{\partial f_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_2(\boldsymbol{x})}{\partial x_2} \end{pmatrix} = \begin{pmatrix} 6x_1 & 8x_2 \\ -24x_1^2 & 3x_2^2 \end{pmatrix}$$

$$F^{-1}(\boldsymbol{x}) = \frac{1}{192x_1 + 18x_2} \begin{pmatrix} \frac{3x_2}{x_1} & -\frac{8}{x_1} \\ \frac{24x_1}{x_2} & \frac{6}{x_2} \end{pmatrix}$$

Put $\boldsymbol{x} = (x_1, x_2) = (-0.5, 0.25) =: \boldsymbol{x}_0$, we get

$$F^{-1}(\boldsymbol{x}_0) = \begin{pmatrix} 0.0164 & -0.1749 \\ 0.5246 & -0.2623 \end{pmatrix}$$

The fixed point iteration is given by

$$\begin{pmatrix} x_{1,n+1} \\ x_{2,n+1} \end{pmatrix} = \begin{pmatrix} x_{1,n} \\ x_{2,n} \end{pmatrix} - \begin{pmatrix} 0.0164 & -0.1749 \\ 0.5246 & -0.2623 \end{pmatrix} \begin{pmatrix} 3x_{1,n}^2 + 4x_{2,n}^2 - 1 \\ x_{2,n}^3 - 8x_{1,n}^3 - 1 \end{pmatrix}$$

For the first iteration, we have

$$\begin{pmatrix} x_{1,1} \\ x_{2,1} \end{pmatrix} = \begin{pmatrix} -0.5 \\ 0.25 \end{pmatrix} - \begin{pmatrix} 0.0164 & -0.1749 \\ 0.5246 & -0.2623 \end{pmatrix} \begin{pmatrix} 0 \\ 0.0156 \end{pmatrix} = \begin{pmatrix} -0.4973 \\ 0.2541 \end{pmatrix}$$

and so on.

4.6 Unconstrained Optimization

Optimization refers to finding the maximum or minimum of a continuous function $f(x_1, x_2, \cdots, x_n)$.

A point $\boldsymbol{x}^*$ is called a strict local minimum of f if $f(\boldsymbol{x}) > f(\boldsymbol{x}^*)$ in a small neighborhood of $\boldsymbol{x}^*$. We restrict ourselves in finding local minimum of $f(\boldsymbol{x})$.

A necessary condition for $\boldsymbol{x}^*$ to be a strict local minimum is that

$$\frac{\partial f(\boldsymbol{x})}{\partial x_i} = 0, \quad i = 1, 2, \cdots, n.$$

Thus, the nonlinear system

$$\frac{\partial f(\boldsymbol{x})}{\partial x_i} = 0, \quad i = 1, 2, \cdots, n$$

can be solved and each calculated solution can be checked as to whether it is a local maximum or minimum or neither.

In the gradiant notation, this system can be written as

$$\nabla f(\boldsymbol{x}) = 0. \tag{4.23}$$

where

$$\nabla f(\boldsymbol{x}) = \left(\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_n} \right)^T.$$

To solve the system (4.23), Newton's method can be used. The Newton's method leads to

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - H(\boldsymbol{x}_n)^{-1} \nabla f(\boldsymbol{x}_n), \quad n = 0, 1, 2, \cdots,$$

where H is the **Hessian matrix** of f given by

$$H(\boldsymbol{x})_{ij} = \frac{\partial^2 f(\boldsymbol{x})}{\partial x_i \partial x_j}, \quad 1 \leq i, j \leq n. \tag{4.24}$$

Note that if $\boldsymbol{x}^*$ is strict local minimum of f, then Taylor formula can be used to show that $H(\boldsymbol{x}^*)$ is non-singular and therefore H is non-singular in a small neighborhood of $\boldsymbol{x}^*$.

Example 4.18. Given $f(x_1, x_2) = x_1^3 + 4x_1 x_2^2 + x_1 - x_2$. To find a point at which this function attains its maximum or minimum, we have to solve the system (4.23). Here

$$\frac{\partial f}{\partial x_1} = 3x_1^2 + 4x_2^2 + 1,$$
$$\frac{\partial f}{\partial x_2} = 8x_1 x_2 - 1.$$

Therefore, the required system of equations is

$$3x_1^2 + 4x_2^2 + 1 = 0 \tag{4.25}$$
$$8x_1 x_2 - 1 = 0 \tag{4.26}$$

To form the Newton's method for the above system of equations, we need the inverse of the Hessian matrix of f given by

$$H(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial^2 f_1(\boldsymbol{x})}{\partial^2 x_1} & \frac{\partial^2 f_1(\boldsymbol{x})}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f_2(\boldsymbol{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 f_2(\boldsymbol{x})}{\partial^2 x_2} \end{pmatrix} = \begin{pmatrix} 6x_1 & 8x_2 \\ 8x_2 & 8x_1 \end{pmatrix}.$$

Inverse of this matrix is given by

$$H^{-1}(\boldsymbol{x}) = \frac{1}{8(3x_1^2 - 4x_2^2)} \begin{pmatrix} 4x_1 & -4x_2 \\ -4x_2 & 3x_1 \end{pmatrix}.$$

Thus the Newton's method for finding the maximum or minimum for the given function f takes the form

$$\begin{pmatrix} x_{1,n+1} \\ x_{2,n+1} \end{pmatrix} = \begin{pmatrix} x_{1,n} \\ x_{2,n} \end{pmatrix} - \frac{1}{8(3x_{1,n}^2 - 4x_{2,n}^2)} \begin{pmatrix} 4x_{1,n} & -4x_{2,n} \\ -4x_{2,n} & 3x_{1,n} \end{pmatrix} \begin{pmatrix} 3x_{1,n}^2 + 4x_{2,n}^2 + 1 \\ 8x_{1,n} x_{2,n} - 1 \end{pmatrix}, \quad n = 0, 1, \cdots$$

When the initial guess $\boldsymbol{x}_0 = (x_{1,0}, x_{2,0})^T$ is given, the above iteration for $n = 0, 1, 2, \cdots$ can be computed. □

Exercise 4

I. **Fixed-Point Iteration Method**

1. Let $f(x) = 0$ be a nonlinear equation for which the sequence $\{x_n\}$, generated by an appropriate fixed-point iteration method, converges to a limit x^*. Under what condition on the iteration function does this limit x^* be a solution to the nonlinear equation $f(x) = 0$? Prove it.
2. For each of the following equations, find the correct iteration function that converges to the desired solution:
   (a) $x - \tan x = 0$, (b) $e^{-x} - \cos x = 0$.
   Study geometrically how the iterations behave with different iteration functions.
3. Show that $g(x) = \pi + \frac{1}{2}\sin(x/2)$ has a unique fixed point on $[0, 2\pi]$. Use fixed-point iteration method with g as the iteration function and $x_0 = 0$ to find an approximate solution for the equaton $\frac{1}{2}\sin(x/2) - x + \pi = 0$. Stop the iteration when the residual error is less than 10^{-4}.
4. If α and β be the roots of $x^2 + ax + b = 0$. If the iterations
$$x_{n+1} = -\frac{ax_n + b}{x_n} \text{ and } x_{n+1} = -\frac{b}{x_n + a}$$
converges, then show that they converge to α and β, respectively, if $|\alpha| > |\beta|$.
5. Let $\{x_n\} \subset [a, b]$ be a sequence generated by a fixed point iteration method with continuous iteration function $g(x)$. If this sequence converges to x^*, then show that
$$|x_{n+1} - x^*| \leq \frac{\lambda}{1 - \lambda}|x_{n+1} - x_n|,$$
where $\lambda := \max\limits_{x \in [a,b]} |g'(x)|$. (This enables us to use $|x_{n+1} - x_n|$ to decide when to stop iterating.)
6. Give reason for why the sequence $x_{n+1} = 1 - 0.9x_n^2$, with initial guess $x_0 = 0$, does not converge to any solution of the quadratic equation $0.9x^2 + x - 1 = 0$? [Hint: Observe what happens after 25 iterations]
7. Let x^* be the smallest positive root of the equation $20x^3 - 20x^2 - 25x + 4 = 0$. If the fixed-point iteration method is used in solving this equation with the iteration function $g(x) = x^3 - x^2 - \frac{x}{4} + \frac{1}{5}$ for all $x \in [0, 1]$ and $x_0 = 0$, then find the number of iterations n required in such a way that $|x^* - x_n| < 10^{-3}$.

II. **Bisection Method**

7. Find the number of iterations to be performed in the bisection method to obtain a root of the equation
$$2x^6 - 5x^4 + 2 = 0$$
in the interval $[0, 1]$ with absolute error $\epsilon \leq 10^{-3}$. Find the approximation solution.
8. Find the approximate solution of the equation $x \sin x - 1 = 0$ (sine is calculated in radians) in the interval $[0, 2]$ using Bisection method. Obtain the number of iterations to be performed to obtain a solution whose absolute error is less than 10^{-3}.
9. Find the root of the equation $10^x + x - 4 = 0$ correct to four significant digits by the bisection method.

III. **Secant and Newton-Raphson Method**

10. Let x^* be the point of intersection of the circle
$$(x + 1)^2 + (y - 2)^2 = 16$$
and the positive x-axis. Choose a value ξ with $0.5 < \xi < 3$, such that the iterative sequence generated by the secant method (with circle function values taken in the fourth quadrant) fails to converge to x^* when started with the initial guess $x_0 = 0.5$ and $x_1 = \xi$. Explain geometrically why secant method failed to converge with your choice of the initial guess (x_0, x_1).
11. Given the following equations:
    (a) $x^4 - x - 10 = 0$, (b) $x - e^{-x} = 0$.

Determine the initial approximations for finding the smallest positive root. Use these to find the roots upto a desired accuracy with secant and Newton-Raphson methods.

12. Find the iterative method based on Newton-Raphson method for finding $\sqrt{N}$ and $N^{1/3}$, where N is a positive real number. Apply the methods to $N = 18$ to obtain the results correct to two significan digits.

13. Find the iterative method based on the Newton-Raphson method for approximating the root of the equation $\sin x = 0$ in the interval $(-\pi/2, \pi/2)$.
Let $\alpha \in (-\pi/2, \pi/2)$ and $\alpha \neq 0$ be such that if the above iterative process is started with the initial guess $x_0 = \alpha$, then the iteration becomes a cycle in the sense that $x_{n+2} = x_n$, for $n = 0, 1, \cdots$.
Find a non-linear equation $g(x) = 0$ whose solution is α.
Starting with the initial guess $x_0 = \alpha$, write the first five iterations using Newton-Raphson method for the equation $\sin x = 0$.
Starting with the initial guess $x_0 = 1$, perform five iterations using Newton-Raphson method for the equation $g(x) = 0$ to find an approximate value of α.

14. Let $\{x_n\}_{n=1}^{\infty}$ be the iterative sequence generated by the Newton-Raphson method in finding the root of the equation $e^{-ax} = x$, where a in the range $0 < a \leq 1$. If x^* denoted the exact root of this equation and $x_0 > 0$, then show that

$$|x^* - x_{n+1}| \leq \frac{1}{2}(x^* - x_n)^2.$$

15. Consider the equation $x \sin x - 1 = 0$. Choose an initial guess $x_0 > 1$ such that the Newton-Raphson method converges to the solution x^* of this equation such that $-10 < x^* < -9$. Compute four iterations and give an approximate value of this x^*. For the same equation, choose another initial guess $x_0 > 1$ such that the Newton-Raphson method converges to the smallest positive root of this equation. Compute four iterations and give an approximate value of this smallest positive root.

16. Give an initial guess x_0 for which the Newton-Raphson method fails to obtain the real root for the equation $\frac{1}{3}x^3 - x^2 + x + 1 = 0$. Give reason for why it failed.

17. Can Newton-Raphson method be used to solve $f(x) = 0$ if
(i) $f(x) = x^2 - 14x + 50$?
(ii) $f(x) = x^{1/3}$?
(iii) $f(x) = (x-3)^{1/2}$ with $x_0 = 4$?
Give reasons.

18. Consider the distribution function for the random variable X given by

$$F(x) = 1 - e^{-\frac{x}{(x-1)^2}}, \quad 0 \leq x \leq 1.$$

Use Newton-Raphson method to find a value of $0 \leq x \leq 1$ such that $P(X > x) = \sin y$, where $y = x^2$. Here P denotes the probability. (**Note:** A distribution function F of a random variable X is defined for any real number x as $F(x) = P(X \leq x)$. Therefore, the required value of x is precisely a solution of the nonlinear equation obtained using the fact that $P(X > x) = 1 - P(X \leq x)$.)

IV. System of Nonlinear Equations

19. Using Newton's method to obtain a root for the following nonlinear systems:
(i) $x_1^2 + x_2^2 - 2x_1 - 2x_2 + 1 = 0, \quad x_1 + x_2 - 2x_1x_2 = 0.$
(ii) $4x_1^2 + x_2^2 - 4 = 0, \quad x_1 + x_2 - \sin(x_1 - x_2) = 0.$

20. Use Newton's method to find the minimum value of the function $f(\boldsymbol{x}) = x_1^4 + x_1x_2 + (1 + x_2)^2$.

5

Interpolation by Polynomials

Suppose that a function $f(x)$ is not defined explicitely, but its value at some finite number of points $\{x_i, i = 1, 2, \cdots, n\}$ is given. The interest is to find the value of f at some point x lying between x_j and x_k, for some $j, k = 1, 2, \cdots, n$. This can be obtained by first approximating f by a known function and then finding the value of this approximate function at the point x. Such a process is called the **interpolation**. The interpolating function is usually chosen from a restricted class of functions, namely, polynomials. In this chapter, we study the methods of interpolating a function. In section 2.1, we introduce Lagrange interpolation. Section 2.2 introduces the notion of divided difference and Newton divided difference formula. The error analysis of the interpolation is studied in section 2.3. The advanced interpolation is presented in the final section.

5.1 Lagrange Interpolation

The basic interpolation problem can be posed in one of two ways:

I. Given a set of **nodes** $\{x_i/\ i = 0, 1, \cdots, n\}$ and corresponding data values $\{y_i/\ i = 0, 1, \cdots, n\}$, find the polynomial $p_n(x)$ of degree less than or equal to n, such that

$$p_n(x_i) = y_i, \quad i = 0, 1, \cdots n.$$

II. Given a set of **nodes** $\{x_i/\ i = 0, 1, \cdots, n\}$ and a continuous function $f(x)$, find the polynomial $p_n(x)$ of degree less than or equal to n, such that

$$p_n(x_i) = f(x_i), \quad i = 0, 1, \cdots n.$$

Note that in the first problem we are trying to fit a polynomial to the data, and in the second case, we are trying to approximate a given function with the interpolating polynomial. Note that the first problem can be viewed as a particular case of the second.

Theorem 5.1 (Lagrange Interpolation Formula).
Let x_0, x_1, $\cdots$, $x_n \in I = [a, b]$ be $n + 1$ distinct nodes and let $f(x)$ be a continuous real-valued function defined on I. Then, there exists a unique polynomial p_n of degree $\leq n$ (called **Lagrange Formula for Interpolating Polynomial***), given by*

$$p_n(x) = \sum_{k=0}^{n} f(x_k) l_k(x), \quad l_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}, k = 0, \cdots, n \tag{5.1}$$

such that

$$p_n(x_i) = f(x_i), \quad i = 0, 1, \cdots, n. \tag{5.2}$$

The function $l_k(x)$ is called the **Lagrange multiplier**.

Proof: Clearly p_n defined by (5.1) is a polynomial of degree $\leq n$ that satisfies (5.2). All that remains is to show the uniqueness of the polynomial. To this end, assume that there exists another interpolating polynomial $q(x)$ of degree $\leq n$ that satisfies (5.2) and define

$$r(x) = p_n(x) - q(x).$$

Since both p_n and q are polynomials of degree less than or equal to n, so is their difference. However, we must note that

$$r(x_i) = p_n(x_i) - q(x_i) = f(x_i) - f(x_i) = 0$$

for each node point x_i, $i = 0, 1, \cdots, n$. Thus, we have a polynomial of degree less than or equal to n that has $n+1$ roots. The only such polynomial is the zero polynomial, ie., $r(x) = 0$ or $p_n(x) = q(x)$ and thus $p_n(x)$ is unique. □

Example 5.2. Consider the case $n = 1$ in which case we have two distinct points x_0 and x_1. Then

$$l_0(x) = \frac{x - x_1}{x_0 - x_1}, \quad l_1(x) = \frac{x - x_0}{x_1 - x_0}$$

and

$$\begin{aligned} p_1(x) &= f(x_0)l_0(x) + f(x_1)l_1(x) \\ &= f(x_0)\frac{x - x_1}{x_0 - x_1} + f(x_1)\frac{x - x_0}{x_1 - x_0} \\ &= \frac{f(x_0)(x - x_1) - f(x_1)(x - x_0)}{x_0 - x_1} \\ &= f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0). \end{aligned} \tag{5.3}$$

This is the familiar case of **linear interpolation**. □

Example 5.3. To obtain an estimate of $e^{0.826}$ using the function values

$$e^{0.82} \approx 2.270500, \quad e^{0.83} \approx 2.293319.$$

Denote $x_0 = 0.82$, $f(x_0) = 2.270500$, $x_1 = 0.83$ and $f(x_1) = 2.293319$, and apply the the formula (5.3) to get

$$p_1(x) = 2.270500 + \frac{2.293319 - 2.270500}{0.83 - 0.82}(x - 0.82) = 2.2819x + 0.399342.$$

In particular, taking $x = 0.826$, we get

$$p_1(0.826) \approx 2.2841914.$$

The true value is

$$e^{0.826} \approx 2.2841638,$$

to eight significant digits.

Note that if we use quadratic interpolation with an additional node $x_2 = 0.84$ and $f(x_2) = 2.316367$, then the approximation value is

$$p_2(0.826) \approx 2.2841639,$$

which is more accurate than the linear interpolation. □

Remark 5.4. The above example gives us a feeling that if we increase the degree of the interpolating polynomial, the polynamial approximates the orginal function more accurately. But this is not in general true as we will see in example 2.12. □

Remark 5.5. Although the Lagrange interpolation formula gives the existence and uniqueness of a polynomial interpolation for a given function, the main disadvantage is that in calculating the polynomial $p_k(x)$, no advantage can be taken of the fact that one already has $p_{k-1}(x)$ available. Thus, it is very expensive to go for Lagrange interpolation when it is not known apriori the minimal degree of the polynomial to get the best approximation to a given function. □

5.2 Newton Interpolation and Divide Differences

In the previous section, we have seen that in the Lagrange formula of interpolating polynomial for a function, if we decide to add a point to the set of nodes to increase the accuracy, we have to completely recompute all of the $l_i(x)$ functions. In other words, we cannot express p_{n+1} in terms of p_n, using Lagrange formula. An alternate form of the polynomial, known as the Newton form, avoids this problem, and allows us to easily write p_{n+1} in terms of p_n.

The idea behind the Newton formula of the interpolating polynomial is to write $p_n(x)$ in the form (called **Newton form**)

$$p_n(x) = A_0 + A_1(x - x_0) + A_2(x - x_0)(x - x_1) + \cdots + A_n(x - x_0)\cdots(x - x_{n-1}) \tag{5.4}$$

where the coefficients A_i, $i = 0, 1, \cdots, n$ are to be obtained. From the interpolation condition that this polynomial agrees with the function value at the node points, we get

$$A_0 = p_n(x_0) = f(x_0).$$

For $x = x_1$, we have

$$A_1 = \frac{p_n(x_1) - A_0}{x_1 - x_0} = \frac{f(x_1) - f(x_0)}{x_1 - x_0} := f[x_0, x_1].$$

For $x = x_2$, we have

$$A_2 = \frac{p_n(x_2) - p_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} = \frac{f(x_2) - p_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} := f[x_0, x_1, x_2].$$

In this way we can obtain all the coefficients.

The advantage in this form is that if p_n is already calculated, then p_{n+1} can be written as

$$p_{n+1}(x) = p_n(x) + A_{n+1}(x - x_0)\cdots(x - x_n).$$

This also shows that the coefficient A_{n+1} in the Newton form (5.4) for the interpolating polynomial is the leading coefficient, ie., the coefficient of x^{n+1}, in the polynomial p_{n+1} of degree $\leq n + 1$ which agree with $f(x)$ at $x_0, \cdots, x_{n+1}$. We summarize this in the following theorem.

Theorem 5.6 (Newton Interpolation Formula).
Let p_n be the polynomial that interpolates a continuous function $f(x)$ at $(n + 1)$ distinct nodes $x_i \in I$, for $i = 0, 1, \cdots, n$. Then the polynomial p_{n+1} that interpolates f at $(n + 2)$ distinct nodes $x_i \in I$, for $i = 0, 1, \cdots, n + 1$ is given by

$$p_{n+1}(x) = p_n(x) + f[x_0, x_1, \cdots, x_{n+1}]w_n(x) \tag{5.5}$$

where

$$f[x_0, x_1, \cdots, x_{n+1}] = \frac{f(x_{n+1}) - p_n(x_{n+1})}{w_n(x_{n+1})}, \quad f[x_0] = f(x_0) \tag{5.6}$$

is called the $(n + 1)th$ **divided difference** *of $f(x)$ at points x_0, x_1, $\cdots$, x_{n+1} with*

$$w_n(x) = \prod_{i=0}^{n}(x - x_i). \tag{5.7}$$

The formula (5.5) is called the **Newton Formula for Interpolating Polynomial**.

Proof. Since we know that the interpolation polynomial is unique, all we have to do is to show that p_{n+1}, as given in (5.5), satisfies the interpolation conditions by assuming that p_n indeed satisfies this condition.

For $0 \leq k \leq n$, we have $w_n(x_k) = 0$ and so we have

$$p_{n+1}(x_k) = p_n(x_k) + f[x_0, x_1, \cdots, x_{n+1}]w_n(x_k) = p_n(x_k) = f(x_k).$$

Hence, p_{n+1} interpolates all but the last point. To check for x_{n+1}, we observe

$$\begin{aligned} p_{n+1}(x_{n+1}) &= p_n(x_{n+1}) + f[x_0, x_1, \cdots, x_{n+1}]w_n(x_{n+1}) \\ &= p_n(x_{n+1}) + f(x_{n+1}) - p_n(x_{n+1}) \\ &= f(x_{n+1}). \end{aligned}$$

Thus p_{n+1} interpolates $f(x)$ at all the nodes. Moreover, it clearly is a polynomial of degree less than or equal to $n+1$, and so we are done. □

Example 5.7. As a continuation of example 2.2, let us try to contruct the linear interpolating polynomial of a function $f(x)$ in the Newton form. In this case, the interpolating polynomial is given by

$$p_1(x) = p_0(x) + f[x_0, x_1]w_1(x) = f[x_0] + f[x_0, x_1](x - x_0),$$

where

$$f[x_0] = f(x_0), \quad f[x_0, x_1] = \frac{f(x_0) - f(x_1)}{x_0 - x_1} \tag{5.8}$$

are the **zeroth** and **first order divided differences**, respectively. □

Algorithm 5.8 (Construction of Divided Difference).

```
input: n,x(i),y(i) (i= 0,1,2, ... ,n)
a(0) = y(0)
for k=1 to n do
    p = 0
    w = 1
    for j = 0 to k-1 do
        p = p + a(j) * w
        w = w * (x(k) - x(j))
    end for
    a(k) = (y(k) - p)/w
end for
output: a(k) (k= 0,1,2, ... ,n)
```

An alternate way of deriving the divided difference coefficients is by means of a **divided difference table**.

The divided difference table is constructed by obtaining higher order divided differences recursively using lower order divided differences. The **second order divided difference** is given by (using (5.6) and (5.8))

$$\begin{aligned} f[x_0, x_1, x_2] &= \frac{f(x_2) - p_1(x_2)}{(x_2 - x_0)(x_2 - x_1)} \\ &= \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)} - \frac{p(x_0)}{(x_2 - x_0)(x_2 - x_1)} - \frac{f[x_0, x_1]w_0(x_2)}{(x_2 - x_0)(x_2 - x_1)} \\ &= \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)} - \frac{f(x_0)}{(x_2 - x_0)(x_2 - x_1)} - \frac{f(x_1) - f(x_0)}{(x_1 - x_0)(x_2 - x_1)} \\ &= \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)} + \frac{f(x_0)}{(x_1 - x_0)(x_2 - x_0)} - \frac{f(x_1)}{(x_1 - x_0)(x_2 - x_1)} \\ &= \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)} + \frac{f(x_0)}{(x_1 - x_0)(x_2 - x_0)} \\ &\qquad -f(x_1)\left(\frac{1}{(x_2 - x_0)(x_2 - x_1)} + \frac{1}{(x_1 - x_0)(x_2 - x_0)}\right) \\ &= \frac{f(x_2) - f(x_1)}{(x_2 - x_0)(x_2 - x_1)} - \frac{f(x_1) - f(x_0)}{(x_1 - x_0)(x_2 - x_0)}. \end{aligned}$$

Therefore,

$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}. \tag{5.9}$$

Similarly, we can derive the **third order divided difference**

$$f[x_0, x_1, x_2, x_3] = \frac{f[x_1, x_2, x_3] - f[x_0, x_1, x_2]}{x_3 - x_0}. \tag{5.10}$$

In general, the nth **order divided difference** formula, sometime called **Newton divided difference** is defined as

$$f[x_0, x_1, \cdots, x_n] = \frac{f[x_1, x_2, \cdots, x_n] - f[x_0, x_1, \cdots, x_{n-1}]}{x_n - x_0} \tag{5.11}$$

A simple way to generate divided difference for Newton interpolation formula (5.5) may be through the divided difference table shown in table 1.

x_i	$f[\cdot] = f(\cdot)$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$				
		$f[x_0, x_1]$			
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$		
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$	
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$		$f[x_0, x_1, x_2, x_3, x_4]$
		$f[x_2, x_3]$		$f[x_1, x_2, x_3, x_4]$	
x_3	$f[x_3]$		$f[x_2, x_3, x_4]$		
		$f[x_3, x_4]$			
x_4	$f[x_4]$				

Table 1. *Divided-Difference Table*

Let the nodes x_0, x_1, $\cdots$, x_n be equally spaces, that is, $x_i = x_0 + ih$, $i = 0, 1, \cdots, n$. Define the difference operator

$$\Delta f(x_i) = f(x_i + h) - f(x_i) =: f_{i+1} - f_i \tag{5.12}$$

Repeated application of the difference operators lead to the followoing higher order differences

$$\Delta^n f(x_i) = \Delta^{n-1} f_{i+1} - \Delta^{n-1} f_i, \tag{5.13}$$

The Newton divided difference can be written in the above notation as

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{h} = \frac{1}{h}\Delta f_0$$
$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = \frac{\frac{1}{h}\Delta f_1 - \frac{1}{h}\Delta f_0}{2h} = \frac{1}{2!h^2}\Delta^2 f_0$$

By induction, we can show that

$$f[x_0, x_1, \cdots, x_n] = \frac{1}{n!h^n}\Delta^n f_0 \tag{5.14}$$

The Newton's interpolation formula (5.5) for equally spaced nodes with step size h is thus given by

$$p_n(x) = \sum_{k=0}^{n} \frac{1}{k!h^k}(\Delta^k f_0)\ w_k(x) \tag{5.15}$$

5.3 Error in Polynomial Interpolation

Let $f(x)$ be defined on an interval $I = [a, b]$. How good a polynomial $p_n(x)$ of degree $\leq n$ interpolates the function $f(x)$ at $n + 1$ nodes x_0, x_1, $\cdots$, x_n in I? This question leads to the analysis of **interpolation error** $e_n(x)$ of $p_n(x)$ given by

$$e_n(x) = f(x) - p_n(x). \tag{5.16}$$

The following theorem provides a formula for the interpolation error.

Theorem 5.9 (Polynomial Interpolation Error Formula).
Let $f \in C^{n+1}([a,b])$ and let the distinct nodes $x_0, x_1, \cdots, x_n$ be in $[a,b]$. Then, for each $\bar{x} \in I$ with $\bar{x} \neq x_i$ $(i = 0, 1, \cdots, n)$, there is a $\xi \in (a,b)$ such that

$$e_n(\bar{x}) = \frac{w_n(\bar{x})}{(n+1)!} f^{(n+1)}(\xi), \tag{5.17}$$

where $w_n(x)$ is given in (5.7).

Proof. Let $p_{n+1}(x)$ be the polynomial of degree $\leq n+1$ which interpolates $f(x)$ at $n+2$ nodes $x_0, x_1, \cdots, x_n$ and $\bar{x}$. Then $p_{n+1}(\bar{x}) = f(\bar{x})$. From (5.5), we have

$$p_{n+1}(x) = p_n(x) + f[x_0, \cdots, x_n, \bar{x}] w_n(x).$$

It follows that

$$f(\bar{x}) = p_{n+1}(\bar{x}) = p_n(\bar{x}) + f[x_0, \cdots, x_n, \bar{x}] w_n(\bar{x}).$$

Therefore, we have

$$e_n(\bar{x}) = f[x_0, \cdots, x_n, \bar{x}] w_n(\bar{x}). \tag{5.18}$$

For any $t \in I$, $t \neq x_i$ $(i = 0, 1, \cdots, n)$, define the function

$$G(x) = e_n(x) - \frac{w_n(x)}{w_n(t)} e_n(t).$$

Then, for $i = 0, 1, \cdots, n$,

$$G(x_i) = e_n(x_i) - \frac{w_n(x_i)}{w_n(t)} e_n(t) = 0$$

and

$$G(t) = e_n(t) - e_n(t) = 0.$$

Thus, G has $n+2$ distinct zeros in I. Using the mean value theore, G' has atleast $n+1$ distinct zeros. Inductively, $G^{(j)}(x)$ has $n+2-j$ zeros in I, for $j = 0, 1, \cdots, n+1$. Let ξ be a zero of $G^{(n+1)}(x)$,

$$G^{(n+1)}(\xi) = 0.$$

Since $e_n^{(n+1)}(x) = f^{(n+1)}(x)$ and $w_n^{(n+1)}(x) = (n+1)!$, we obtain

$$G^{(n+1)}(x) = f^{(n+1)}(x) - \frac{(n+1)!}{w_n(t)} e_n(t).$$

Substituting $x = \xi$ and solving for $e_n(t)$,

$$e_n(t) = \frac{w_n(t)}{(n+1)!} \cdot f^{(n+1)}(\xi).$$

Taking $t = \bar{x}$, we get the desired result. □

Definition 5.10 (Infinity Norm).
If f is continuous on a closed interval $I = [a,b]$, then the **infinity norm** *of f denoted as $\|f\|_{\infty,I}$ is defined as*

$$\|f\|_{\infty,I} = \max_{x \in I} |f(x)|. \tag{5.19}$$

Example 5.11. Let us find a bound for the error in linear interpolation given in example 2.5. The linear interpolating polynomial for $f(x)$ at x_0 and x_1 is given by

$$p_1(x) = p_0(x) + f[x_0, x_1] w_1(x) = f(x_0) + f[x_0, x_1](x - x_0),$$

where $f[x_0, x_1]$ is given by (5.8). Therefore, the error $e_1(x)$ is given by (by (5.17))

$$e_1(x) = \frac{(x-x_0)(x-x_1)}{2} \cdot f''(\xi),$$

where ξ depends on x. If $x \in I = [x_0, x_1]$, then $\xi \in (x_0, x_1)$. Therefore,

$$|e_1(x)| \leq |(x-x_0)(x-x_1)| \, \frac{\|f''\|_{\infty,I}}{2}.$$

Note that the maximum value of $|(x-x_0)(x-x_1)|$ for all $x \in [x_0, x_1]$ occurs at $x = (x_0+x_1)/2$ and therefore, we have

$$|(x-x_0)(x-x_1)| \leq \frac{(x_1-x_0)^2}{4}.$$

Using this inequality, we get the bound for error $e_1(x)$ as

$$|e_1(x)| \leq (x_1-x_0)^2 \frac{\|f''\|_{\infty,I}}{8},$$

for all $x \in [x_0, x_1]$, which further implies

$$\|e_1\|_{\infty,I} \leq (x_1-x_0)^2 \frac{\|f''\|_{\infty,I}}{8}.$$

□

Quite often, the polynomial interpolation that we compute is based on the function data subjected to rounding error. Let us denote the approximate value of $f(x_k)$ by $\tilde{f}(x_k)$ for each node point x_k, $k = 0, 1, \cdots, n$. Then the corresonding polynomial interpolation using Lagrange formula gives

$$\tilde{p}_n(x) = \sum_{k=0}^{n} \tilde{f}(x_k) l_k(x)$$

and we want to estimate the total error, which is given by

$$f(x) - \tilde{p}_n(x) = (f(x) - p_n(x)) + (p_n(x) - \tilde{p}_n(x)), \tag{5.20}$$

where the first term on the right hand side the error due to polynomial interpolation whose formula is given by (5.17) and the second term is the error due to rounding.

We now turn our attention to analyze the error due to rounding. Let

$$f(x_k) - \tilde{f}(x_k) = \epsilon_k \text{ and } ||\epsilon||_\infty = \max\{|\epsilon_k| / k = 0, 1, \cdots, n\},$$

then we have

$$\begin{aligned} |p_n(x) - \tilde{p}_n(x)| &= \left| \sum_{k=0}^{n} (f(x_k) - \tilde{f}(x_k)) l_k(x) \right| \\ &\leq ||\epsilon||_\infty \sum_{k=0}^{n} ||l_k||_\infty \end{aligned}$$

Although the error due to rounding looks bounded, the sum on the right hand side can grow quite large as n increases, especially, when the nodes are equally spaced as we will study now.

Assume that the nodes are equidistant on the interval $[a, b]$, with $x_0 = a$ and $x_n = b$, and $x_{k+1} - x_j = h$ for all $k = 0, 1, \cdots, n-1$. We write

$$x_k = a + kh, \quad k = 0, 1, \cdots, n, \text{ and } x = a + \eta h, 0 \leq \eta \leq n.$$

Therefore,

$$l_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i} = \prod_{i=0, i\neq k}^{n} \frac{\eta - i}{i - k}, k = 0, \cdots, n$$

Hence, the Lagrange multipliers are not dependent on the choice of a, b or h. They depend entirely on n, η (which depends on x) and the distribution of the nodes. The figure 2.1 shows the function

$$l(x) = \sum_{k=0}^{n} |l_k(x)|$$

for various values of n and figure 2.2 shows the n in the x-axis and the function

$$M_n = \sum_{k=0}^{n} ||l_k||_\infty$$

in the y-axis. In fact, this behavior of the Legrange multiplier can also be analyzed theoretically, but this is outside the scope of the present course.

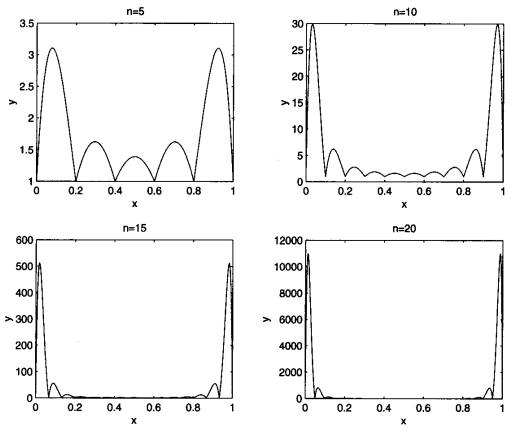


Fig. 5.1. $y = \sum_{k=0}^{n} |l_k(x)|$.

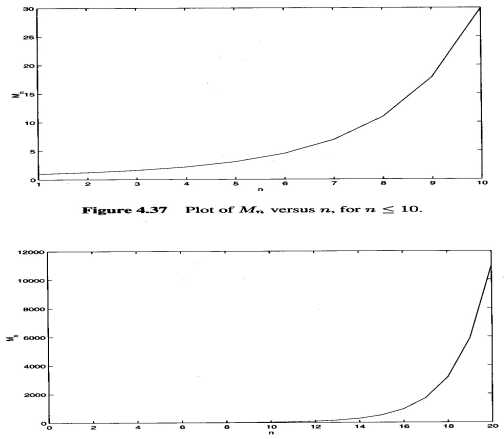


Figure 4.37 Plot of M_n versus n, for $n \leq 10$.

Fig. 5.2. $y = \sum_{k=0}^{n} ||l_k||_\infty$.

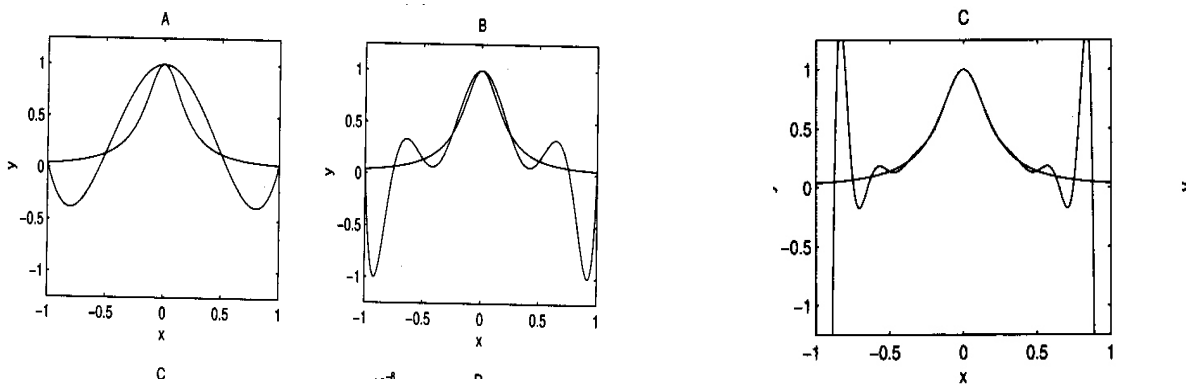


Fig. 5.3. Interpolation polynomial for $f(x) = 1/(1 + 25x^2)$ for $n = 4$, $n = 6$ and $n = 8$ respectively.

With this knowledge we now take the equation (5.20) which gives

$$||f - \tilde{p}||_\infty \leq ||f - p_n||_\infty + ||p_n - \tilde{p}||_\infty n \leq ||f - p_n||_\infty + ||\epsilon||_\infty M_n. \tag{5.21}$$

As it is clear from the figure 2.2 that M_n increases exponentially with respect to n, although we have a very small value for the rounding error $||\epsilon||_\infty$, a large enough n can bring in a significantly large error in the interpolated polynomial as illustrated in the example.

Example 5.12. Consider the function $f(x) = 1/(1 + 25x^2)$. The polynomial interpolation with $n = 4$, $n = 6$ and $n = 8$ are depicted in figure 2.3. □

The above example shows that the polynomial interpolation of higher degree suffers very badly due to rounding error. However, this is not true for any function as the exponential function gets better approximation as the degree of polynomial increases. A more deeper analysis is required to understand the reason behind the behavior of rounding error in polynomial interpolation. But this is outside the scope of this course and therefore is omitted.

5.4 Piecewise Linear and Cubic Spline Interpolation

Quite often polynomial interpolation will be unsatisfactory as an approximation tool. This is true if we insist on letting the order of the polynomial get larger and larger. However, if we keep the order of the polynomial fixed, and use different polynomial over different intervals, with the length of the intervals getting smaller and smaller, then interpolation can be very accurate and powerful approximation tool.

Let us start with linear interpolation over an interval $I = [a, b]$ which leads to

$$p_1(x) = f(a) + f[a, b](x - a) = f(a) + \frac{f(b) - f(a)}{b - a}(x - a) = \frac{x - b}{a - b}f(a) + \frac{x - a}{b - a}f(b).$$

With the nodes $x_0 = a$, $x_2 = b$ and $x_0 < x_1 < x_2$, we can obtain a quadratic interpolation polynomial as discussed in the previous sections. Instead, we can interpolate the function $f(x)$ as two piece of linear polynomials, one in $[x_0, x_1]$ and another one in $[x_1, x_2]$. Such polynomials are defined as

$$p_{1,1}(x) = \frac{x - x_1}{x_0 - x_1}f(x_0) + \frac{x - x_0}{x_1 - x_0}f(x_1), \quad p_{1,2}(x) = \frac{x - x_2}{x_1 - x_2}f(x_1) + \frac{x - x_1}{x_2 - x_1}f(x_2)$$

and the interpolating polynomial is given by

$$P(x) = \begin{cases} p_{1,1}(x) \ , x \in [x_0, x_1] \\ p_{1,2}(x) \ , x \in [x_1, x_2]. \end{cases}$$

Note that $P(x)$ is a continuous function in $[a, b]$, which interpolates $f(x)$ and is linear in $[a, x_1]$ and $[x_1, b]$. Such a polynomial is called **piecewise linear polynomial**. Although piecewise linear interpolation is continuous, it is not differentiable at the nodes and also, it makes a poor approximation to $f(x)$. We wish to find an interpolation function that is smooth and does a better approximation to $f(x)$. This can be achieved by **spline interpolation**.

Definition 5.13 (Spline Function).

A **spline function** *of degree d with nodes x_i, $i = 0, 1, \cdots, n$ is a function $s(x)$ with the properties*

I. On each subinterval $[x_{i-1}, x_i]$, $i = 1, 2, \cdots, n$, $s(x)$ is a polynomial of degree $\leq d$.

II. The interpolation condition $s(x_i) = f(x_i)$, $i = 0, 1, \cdots, n$ is satisfied.

III. $s(x)$ and its first $(d - 1)$ derivatives are continous on $[a, b]$.

We shall now study how we can obtain the interpolation of a function $f(x)$ as spline functions instead of polynomials. For the sake of simplicity, we restrict only to cubic splines. The construction of the spline interpolation $s(x)$ of a function $f(x)$ is as follows:

Step 1: Let us denote by $M_1, \cdots, M_n$,

$$M_i = s''(x_i), \quad i = 0, 1, \cdots, n$$

and first obtain $s(x)$ in terms of M_i's which are unknowns.

Step 2: Since $s(x)$ is cubic on each $[x_{i-1}, x_i]$, the function $s''(x)$ is linear on the interval such that

$$s''(x_{i-1}) = M_{i-1}, \quad s''(x_i) = M_i.$$

Therefore, it is given by

$$s''(x) = \frac{(x_i - x)M_{i-1} + (x - x_{i-1})M_i}{x_i - x_{i-1}}, \quad x_{i-1} \leq x \leq x_i \tag{5.22}$$

Integrating (5.22) two times with respect to x, we get

$$s(x) = \frac{(x_i - x)^3 M_{i-1}}{6(x_i - x_{i-1})} + \frac{(x - x_{i-1})^3 M_i}{6(x_i - x_{i-1})} + K_1 x + K_2,$$

where K_1 and K_2 are integrating constants to be determined by using the conditions $s(x_{i-1}) = f(x_{i-1})$ and $s(x_i) = f(x_i)$. We have

$$K_1 = \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} - \frac{(M_i - M_{i-1})(x_i - x_{i-1})}{6}$$
$$K_2 = \frac{x_i f(x_{i-1}) - x_{i-1} f(x_i)}{x_i - x_{i-1}} - \frac{(M_{i-1}x_i - M_i x_{i-1})(x_i - x_{i-1})}{6}$$

Substituting these values in the above equation, we get

$$s(x) = \frac{(x_i - x)^3 M_{i-1} + (x - x_{i-1})^3 M_i}{6(x_i - x_{i-1})} + \frac{(x_i - x)f(x_{i-1}) + (x - x_{i-1})f(x_i)}{x_i - x_{i-1}}$$
$$- \frac{1}{6}(x_i - x_{i-1})[(x_i - x)M_{i-1} + (x - x_{i-1})M_i], \quad x_{i-1} \le x \le x_i \tag{5.23}$$

Formula (5.23) applies to each of the intervals $[x_1, x_2], \cdots, [x_{n-1}, x_n]$. The formulas for adjacent intervals $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$ will agree at their common point $x = x_i$ because of the interpolating condition $s(x_i) = f(x_i)$. This implies that $s(x)$ will be continuous over the entire interval $[a, b]$. Similarly, formula (5.22) for $s''(x)$ implies that it is continuous on $[a, b]$.

Step 3: All that remains is to find the values of M_i for all $i = 0, 1, \cdots, n$. This is obtained by ensuring the continuoity of $s'(x)$ over $[a, b]$, ie., the formula for $s'(x)$ on $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$ are required to give the same value at their common point $x = x_i$, for $i = 1, 2, \cdots, n-1$. After simplification (???), we get the system of linear equations for $i = 1, 2, \cdots n-1$

$$\frac{x_i - x_{i-1}}{6} M_{i-1} + \frac{x_{i+1} - x_{i-1}}{3} M_i + \frac{x_{i+1} - x_i}{6} M_{i+1} = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} - \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}. \tag{5.24}$$

These $n-1$ equations together with the assumption that

$$M_0 = M_n = 0 \tag{5.25}$$

leads to the values of M_0, M_1, $\cdots$, M_n and hence to the interpolation function $s(x)$.

A spline constructed above is called a **natural spline**.

Example 5.14. Calculate the natural cubic spline interpolating the data $\{(1,1), (2, \frac{1}{2}), (3, \frac{1}{3}), (4, \frac{1}{4})\}$. The number of points is $n = 4$ and all $x_i - x_{i-1} = 1$. The system (5.24) together with $M_0 = M_3 = 0$ becomes

$$\frac{2}{3}M_2 + \frac{1}{6}M_3 = \frac{1}{3}, \quad \frac{1}{6}M_1 + \frac{2}{3}M_3 = \frac{1}{12},$$

which gives $M_2 = \frac{1}{2}, \quad M_3 = 0$. Substituting these values into (5.23), we obtain

$$s(x) = \begin{cases} \frac{1}{12}x^3 - \frac{1}{4}x^2 - \frac{1}{3}x + \frac{3}{2} & , 1 \le x \le 2 \\ -\frac{1}{12}x^3 + \frac{3}{4}x^2 - \frac{7}{3}x + \frac{17}{6} & , 2 \le x \le 3 \\ -\frac{1}{12}x + \frac{7}{3} & , 3 \le x \le 4 \end{cases}$$

Remark 5.15. There is a relationship between the degree of spline approximation n (say) and the degree of smoothness, N (say) expected. The degree of the polynomials is related to the number of unknown coefficients ie., the degrees of freedom D_f (say), in the problem, whereas N is related to the number of constraints D_c (say). We expect that the degrees of freedom and the number of constraints have to balance in order for the spline to be well-defined.

Let there be m subintervals, each being the domain of definition for a seperate polynomial of degree n, we have a total of $D_f = m(n+1)$ degrees of freedom. On the other hand, there are $m+1$ interpolation conditions (ie., $s(x_i) = f(x_i)$, $i = 0, 1, \cdots, m$) and $m-1$ interior nodes where continuity of $s(x)$ and its N derivatives are expected to be continuous and thereby, there are $N+1$ continuity conditions imposed on each of $m-1$ interior point. Therefore, $D_c = m + 1 + (m-1)(N+1)$ constraints. If we consider the difference $D_f - D_c$, we get

$$D_f - D_c = m(n+1) - m - 1 - (m-1)(N+1) = mn - m - mN + N = m(n - 1 - N) + N.$$

We can make the first term vanish by setting $n - 1 - N = 0$. This establishes a relationship between the polynomial degree of the spline and smoothness degree. For example, if we consider the cubic spline, we need to have $N = 2$. However, we will not have the number of constraints equal to the number of degrees of freedom, since $D_f - D_c = N$. Thus, we need to add N additional constraints, which in the case of natural cubic spline we have $M_0 = M_3 = 0$. Partly for this reason, odd polynomial order splines are prefered, because if n is odd, then N is even and the additional constraints can be imposed equally at the two endpoints of the interval. □

Exercise 5

I. **Lagrange Interpolation**

1. Obtain Lagrange interpolation formula for equally spaced nodes.
2. Using Lagrange interpolation formula, express the rational function $f(x) = \frac{3x^2+x+1}{(x-1)(x-2)(x-3)}$ as a sum of partial fractions.
3. Construct the Lagrange interpolation polynomial for the function $f(x) = \sin \pi x$, choosing the points $x_0 = 0$, $x_1 = 1/6$, $x_3 = 1/2$. **Answer:** $7/2x - 3x^2$
4. Find a cubic polynomial using Lagrange's formula for the data:

x	-2	-1	1	3
$f(x)$	-1	3	-1	19

   Answer: $p_3(x) = x^3 - 3x + 1$
5. Use Lagrange interpolation formula to find a quadratic polynomial $p_2(x)$ that interpolates the function $f(x) = e^{-x^2}$ at $x_0 = -1$, $x_1 = 0$ and $x_2 = 1$. Further, find the value of $p_2(-0.9)$ with rounding to six decimal places after decimal point and compare the value with the true value $f(-0.9)$ of same figure. Find the percentage error in this calculation.
   Answer: $p_2(x) = 1 - 0.632121x^2$, Error $\approx 9.69\%$
6. Given a table of values of the function $f(x)$

x	321.0	322.8	324.2	325.0
$f(x)$	2.50651	2.50893	2.51081	2.51188

   Compute the value $f(323.5)$. **Answer:** 2.50987
7. Let $p(x)$ be a polynomial of degree $\leq n$. For $n+1$ distinct nodes x_k, $k = 0, 1, \cdots, n$, show that we can write $p(x) = \sum_{k=0}^{n} p(x_k) l_k(x)$.
8. *The functions* $l_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}, k = 0, \cdots, n$ *are the weight polynomials of the corresponding nodes and are often called* **Lagrange multipliers**. Prove that for any $n \geq 1$, $\sum_{k=0}^{n} l_k(t) = 1$. [**Hint:** Use problem 7 with an appropriate polynomial p]
9. Let $x_k \in [a.b]$, $k = 0, 1, \cdots, n$ be $n+1$ distinct nodes and let $f(x)$ be a continuous function on $[a, b]$. Show that for $x \neq x_k$, $k = 0, 1, \cdots, n$, the Lagrange interpolating polynomial can be represented in the form
$$p_n(x) = w(x) \sum_{k=0}^{n} \frac{f(x_k)}{(x - x_k) w'(x_k)}$$
   where $w(x) = (x - x_0)(x - x_1) \cdots (x - x_n)$. Verify the interpolation condition.

II. **Newton Interpolation and Divided Difference**

10. For the function data given in the table below, fit a polynomial using Newton interpolation formula and find the value of $f(2.5)$.

x	-3	-1	0	3	5
$f(x)$	-30	-22	-12	330	3458

    Answer: $p_4(x) = 5x^4 + 9x^3 - 27x^2 - 21x - 12$, $p_4(2.5) = 102.6875$.
11. Calculate the nth divided difference of $f(x) = 1/x$ **Answer:** $(-1)^n/(x_0 x_1 \cdots x_n)$

12. Let x_0, x_1, $\cdots$, x_n be $n+1$ distinct nodes in the closed interval $[a,b]$ and let $f(x)$ be $n+1$ times continuously differentiable function on $[a,b]$. Then,
    i. show that the divided differences are symmetric functions of their arguments, that is, for an arbitrary permutation π of the indices $0,1,\cdots,i$, we have $f[x_0,\cdots,x_i]=f[x_{\pi 0},\cdots,x_{\pi i}]$.
    ii. show that $f[x_0,x_1,\cdots,x_{i-1},x]=f[x_0,x_1,\cdots,x_{i-1},x_i]+f[x_0,x_1,\cdots,x_i,x](x-x_i)$, for each $i=1,\cdots,n$ and for all $x\in[a,b]$.
    iii. show $\dfrac{d}{dx}f[x_0,\cdots,x_{i-1},x]=f[x_0,\cdots,x_{i-1},x,x]$.
13. Let $f(x)$ be a real-valued function defined on $I=[a,b]$ and k times differentiable in (a,b). If x_0, x_1, $\cdots$, x_k are $k+1$ distinct points in $[a,b]$, then show that there exists $\xi\in(a,b)$ such that $f[x_0,\cdots,x_k]=\frac{f^{(k)}(\xi)}{k!}$.

III. **Error in Interpolating Polynomials**

14. Let x_0, x_1, $\cdots$, x_n be $n+1$ distinct nodes where instead of the function values $f(x_i)$, the corresponding approximate values $\tilde{f}(x_i)$ rounded to 5 decimal digits after decimal point. If the Lagrange interpolation polynomial obtained from the approximate values $\tilde{f}(x_i)$ is $\tilde{p}_n(x)$, then show that the error at a fixed point $\tilde{x}$ satisfies the inequality
$$|p_n(\tilde{x})-\tilde{p}_n(\tilde{x})|\le\frac{1}{2}10^{-5}\sum_{k=0}^{n}|l_k(\tilde{x})|,$$
where $p_n(\tilde{x})$ is the Lagrange interpolated polynomial for exact values $f(x_i)$ $(i=0,1,\cdots,n)$.
15. Let $p_1(x)$ be the linear Newton interpolation polynomial for data (6000, 0.33333) and (6001, -0.66667). If the calculation is performed with 5 decimal digit rounding, then show that the process of evaluating $p_1(x)$ in the form $p_1(x)=f(x_0)+\Delta f_0(x-x_0)$ at $x=6000$ and $x=6001$ involves less error than evaluating the same linear polynomial in the form $p_1(x)=\Delta f_0 x+(f(x_0)-\Delta f_0 x_0)=:mx+a$ at these points. Find the percentage error in each case.
16. Let $x_0,x_1,\cdots,x_n$ be distinct real numbers, and let f be a given real-valued function with $n+1$ continuous derivatives on an interval $I=[a,b]$. Let $t\in I$ be such that $t\ne x_i$ for $i=0,\cdots,n$. Then show that there exists an $\xi\in(a,b)$ such that
$$e_n(t):=f(t)-\sum_{k=0}^{n}f(x_k)l_k(t)=\frac{(t-x_0)\cdots(t-x_n)}{(n+1)!}f^{(n+1)}(\xi),$$
where $l_k(t)=\displaystyle\prod_{i=0,i\ne k}^{n}\frac{t-x_i}{x_k-x_i}, k=0,\cdots,n.$
17. Given the square of the integers N and $N+1$, what is the largest error that occurs if linear interpolation is used to approximate $f(x)=x^2$ for $N\le x\le N+1$? **Answer:** 0.25
18. The following table gives the data for $f(x)=\sin x/x^2$.

x	0.1	0.2	0.3	0.4	0.5
$f(x)$	9.9833	4.9667	3.2836	2.4339	1.9177

    Calculate $f(0.25)$ as accurately as the number of figures shown in the table
    (a) by using the data in the table and using Newton's interpolation formula
    (b) by first tabulating $xf(x)$ with rounding the same number of figures as in the table and then using Newton's interpolation formula.
    (c) Find the error in each case and explain the difference between the results in (a) and (b).
    Answer: (a) 3.8647 (b) 3.9585 (c) 0.0469 for (a) and 0.000005625 for (b) (you may perform this calculation with more accurace)
19. Determine the spacing h in a table of equally spaced values of the function $f(x)=\sqrt{x}$ between 1 and 2, so that interpolation with a second-degree polynomial in this table will yield a desired accuracy.

IV. **Cubic Spline Interpolation**

20. Obtain the cubic spline approximation for the function given in the tabular form

x	0	1	2	3
$f(x)$	1	2	33	244

6

Numerical Differentiation and Integration

There are two reasons for approximating derivatives and integrals of a function $f(x)$. One is when the function is very difficult to differentiate or integrate, or only the tabular values are available for the function. Another reason is to obtain solution of a differential or integral equation. In this chapter we introduce some basic methods to approximate derivative and integral of a function either explicitely or by tabulated values.

In section 1, we obtain numerical methods to find derivatives of a function. Rest of the chapter introduce various methods for numerical integration.

6.1 Numerical Differentiation

Numerical differentiation methods are obtained using one of the following three techniques:

I. Methods based on Finite Difference Operators

II. Methods based on Interpolation

III. Methods based on Undetermined Coefficients

We now discuss each of the methods in details.

1. Finite Difference

The most simple way to obtain a numerical method to approximate the derivative of $f(x)$ is using the definition of derivative given by

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

which justifies the usage of the approximation formula

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} =: D_h^+ f(x) \tag{6.1}$$

for a small value of h. $D_h^+ f(x)$ is called a **forward difference formula** for the derivative of $f(x)$ with step size h.

To find a formula for error, we use Taylor's theorem

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2} f''(c)$$

for some c between x and $x+h$. Substituting in the right side of (6.1), we obtain

$$D_h f(x) = \frac{1}{h}\left\{\left[f(x) + hf'(x) + \frac{h^2}{2} f''(c)\right] - f(x)\right\} = f'(x) + \frac{h}{2} f''(c)$$

Therefore, the required error is given by

$$f'(x) - D_h f(x) = -\frac{h}{2} f''(c). \tag{6.2}$$

If we consider the left hand side of (6.2) as a function of h, ie., if $g(h) = f'(x) - D_h f(x)$, then we see that $|g(h)/h| = -\frac{1}{2}|f''(c)|$. If we assume f'' to be bounded by a constant $M > 0$, then we see that $|g(h)/h| \leq M/2$, This shows that when $f \in C^2(I)$ for some closed and bounded interval I, then $g = O(h)$, which we say that the forward difference formula $D_h^+ f(x)$ is of order 1 (order of accuracy).

The derivative of a function f can also be defined as

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

and

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x-h)}{2h}.$$

The first definition gives the **backward difference formula** of order 1 as

$$f'(x) \approx \frac{f(x) - f(x-h)}{h} =: D_h^- f(x). \tag{6.3}$$

The error for this formula can be obtained similar to that of the forward difference formula. The second definition gives the **central difference formula**

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} =: D_h^0 f(x) \tag{6.4}$$

To obtain the error for the central difference formula, we use the Taylor's theorem to obtain

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \frac{h^3}{3!}f'''(c_1)$$

where c_1 lies between x and $x+h$, and

$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2!}f''(x) - \frac{h^3}{3!}f'''(c_2),$$

where c_2 lies between $x-h$ and x. Therefore, we have

$$f(x+h) - f(x-h) = 2hf'(x) + \frac{h^3}{3!}(f'''(c_1) + f'''(c_2)).$$

Since $f'''(x)$ is continuous, by I.4 of tutorial 1, we see that

$$f'''(c_1) + f'''(c_2) = 2f'''(c)$$

where $c \in (x-h, x+h)$. Therefore, we obtain the error formula as

$$f'(x) - D_h^0(f(x)) = -\frac{h^2}{6}f'''(c) \tag{6.5}$$

where c lies between $x-h$ and $x+h$. Clearly, the central difference formula is of second order. Geometrical interpretation of the three primitive difference formulae is shown in figure 3.1.

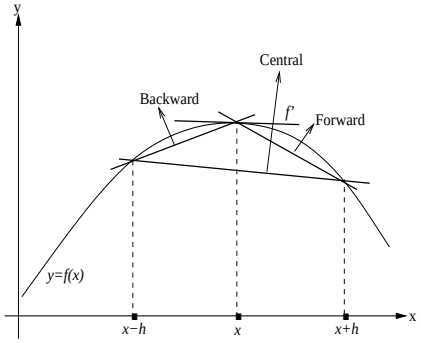


Fig. 6.1. Geometrical interpretation of difference formulae.

Example 6.1. To find the value of the derivative of the function given by $f(x) = \sin x$ at $x = 1$ with $h = 0.003906$, we use the three primitive difference formulas. We have

$$f(x-h) = f(0.996094) = 0.839354, \quad f(x) = f(1) = 0.841471, \quad f(x+h) = f(1.003906) = 0.843575.$$

I. Backward difference: $D_h^- f(x) = \frac{f(x)-f(x-h)}{h} = 0.541935$.

II. Central Difference: $D_h^0(f(x)) = \frac{f(x+h)-f(x-h)}{2h} = 0.540303$.

III. Forward Difference: $D_h^+ f(x) = \frac{f(x+h)-f(x)}{h} = 0.538670$.

Note that the exact value is $f'(1) = \cos 1 = 0.540302$.

2. Interpolation

An alternate way to obtain the same difference formulae as obtained above, we can also use the polynomial interpolation introduced in chapter 2. Thus, to calculate $f'(x)$ at some point $x = t$, we use the formula

$$f'(t) \approx p_n'(t),$$

where $p_n(x)$ denotes the interpolation polynomial of $f(x)$ with degree $\leq n$. Many different formulas can be obtained by varying n and by varying the placement of the nodes $x_0, \cdots, x_n$ relative to the point t of interest. For instance, if we take $n = 1$, the linear interpolation polynomial is given by

$$p_1(x) = f(x_0) + f[x_0, x_1](x - x_0).$$

Hence, we may take

$$f'(x) \approx p_1'(x) = f[x_0, x_1]. \tag{6.6}$$

In particular, if we take $x_0 = x$ and $x_1 = x + h$ for a small value h, we obtain the forward difference formula. If we take $x_0 = x - h$ and $x_1 = x$ for small value h, we obtain the backward difference formula. Finally, if we take $x_0 = x - h$ and $x_1 = x + h$, we get the central difference formula.

Theorem 6.2 (Error formula for derivative using polynomial interpolation).
Assume $f(x)$ has $n + 2$ continuous derivatives on an interval $[a, b]$. Let $x_0, x_1, \cdots, x_n$ be $n + 1$ distinct nodes in $[a, b]$, and let t be an arbitrary given point in $[a, b]$. Then

$$f'(t) - p_n'(t) = w_n(t)\frac{f^{(n+2)}(\xi_1)}{(n+2)!} + w_n'(t)\frac{f^{(n+1)}(\xi_2)}{(n+1)!} \tag{6.7}$$

with

$$w_n(t) = \prod_{i=0}^{n}(t - x_i). \tag{6.8}$$

and ξ_1 and ξ_2 are points in between the maximum and minimum of x_0, x_1 $\cdots$, x_n and t.

Proof. By Newton Interpolation formula, we have

$$f(x) = p_n(x) + f[x_0, \cdots, x_n, x]w_n(x),$$

where $p_n(x)$ is the polynomial of degree $\leq n$ which interpolates $f(x)$ at $x_0, \cdots, x_n$. Taking derivative on both sides, we get

$$f'(x) = p_n'(x) + w_n(x)\frac{d}{dx}f[x_0, \cdots, x_n, x] + w_n'(x)f[x_0, \cdots, x_n, x].$$

But we know that

$$\frac{d}{dx}f[x_0, \cdots, x_n, x] = f[x_0, \cdots, x_n, x, x].$$

Therefore, we have

$$f'(x) = p_n'(x) + w_n(x)f[x_0, \cdots, x_n, x, x] + w_n'(x)f[x_0, \cdots, x_n, x].$$

Further, we know

$$f[x_0, \cdots, x_n, x] = \frac{f^{(n+1)}(\xi)}{(n+1)!},\ \xi \in (a, b).$$

Therefore, we get

$$f'(t) - p_n'(t) = w_n(t)\frac{f^{(n+2)}(\xi_1)}{(n+2)!} + w_n'(t)\frac{f^{(n+1)}(\xi_2)}{(n+1)!}$$

which is what we wish to show. □

Higher order differentiation formulas and their error can be obtained similarly.

3. Method of Undetermined Coefficients

Another method to derive formulas for numerical differentiation is called the method of undetermined coefficients. We will illustrate the method by deriving a formula for $f''(x)$.

$$f''(x) \approx D_h^{(2)} f(x) := Af(x+h) + Bf(x) + Cf(x-h) \tag{6.9}$$

with A, B and C unspecified. Replace $f(x+h)$ and $f(x-h)$ by the Taylor expansions

$$f(x \pm h) = f(x) \pm hf'(x) + \frac{h^2}{2}f''(x) \pm \frac{h^3}{6}f^{(3)}(x) + \frac{h^4}{24}f^{(4)}(\xi_\pm),$$

with $x - h \le \xi_- \le x \le \xi_+ \le x + h$. Substitute into (6.9) and rearrange into a polynomial in powers of h:

$$\begin{aligned} Af(x+h) + Bf(x) + Cf(x-h) = (A+B+C)f(x) + h(A-C)f'(x) + \frac{h^2}{2}(A+C)f''(x) \\ + \frac{h^3}{6}(A-C)f'''(x) + \frac{h^4}{24}[Af^{(4)}(\xi_+) + Cf^{(4)}(\xi_-)]. \end{aligned}$$

In order for this is to equal $f''(x)$, we set

$$A + B + C = 0, \quad A - C = 0, \quad A + C = \frac{2}{h^2}.$$

The solution of this system is $A = C = 1/h^2$ and $B = -2/h^2$. This yields the formula

$$D_h^{(2)} f(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} \tag{6.10}$$

The error is given by

$$f''(x) - D_h^{(2)} f(x) = -\frac{h^2}{24}[f^{(4)}(\xi_+) + f^{(4)}(\xi_-)].$$

Using the problem I.4 of tutorial 1, we get

$$f''(x) - D_h^{(2)} f(x) = -\frac{h^2}{12}f^{(4)}(\xi) \tag{6.11}$$

for some $x - h \le \xi \le x + h$.

Remark 6.3. The preceding formulas are useful when deriving methods for solving differential equations, but they can lead to serious errors when applied to function values that are obtained empirically. To illustrate a method for analyzing the effect of such errors, we consider the second derivative approximation (6.10)

$$f''(x_1) \approx D_h^{(2)} f(x_1) = \frac{f(x_2) - 2f(x_1) + f(x_0)}{h^2}$$

with $x_i = x_0 + ih$. Instead of using the exact values $f(x_i)$, we use the appoximate values f_i with

$$f(x_i) = f_i + \epsilon_i, \quad i = 0, 1, 2.$$

The actual numerical derivative computed is

$$\bar{D}_h^{(2)} f(x_1) = \frac{f_2 - 2f_1 + f_0}{h^2}.$$

The error committed is

$$\begin{aligned} f''(x_1) - \bar{D}_h^{(2)} f(x_1) &= f''(x_1) - \frac{f(x_2) - 2f(x_1) + f(x_0)}{h^2} + \frac{\epsilon_2 - 2\epsilon_1 + \epsilon_0}{h^2} \\ &= -\frac{h^2}{12} f^{(4)}(\xi) + \frac{\epsilon_2 - 2\epsilon_1 + \epsilon_0}{h^2}. \end{aligned}$$

Assuming $-E \le \epsilon_i \le E$, we have

$$|f''(x_1) - \bar{D}_h^{(2)} f(x_1)| \le \frac{h^2}{12} |f^{(4)}(\xi)| + \frac{4E}{h^2} \tag{6.12}$$

The last bound would be attainable in many situations. As example of such errors would be rounding errors, with E a bound on their magnitude.

The error bound in (6.12) will initially get smaller as h decreases, but for h sufficiently close to zero, the error will begin to increase again. There is an optimal value of h to minimize the right side of (6.12). □

Example 6.4. In finding $f''(\pi/6)$ for the function $f(x) = \cos x$, if we use the function values f_i by rounding $f(x_i)$ to six significant digits, then

$$|f(x_i) - f_i| \le 0.5 \times 10^{s-6+1}$$

where s is the largest integer such that $10^s \le |f(x_i)|$. Although cosine function varies from 0 to 1, here we assume (as we are interested in the function valued in a neighborhood of $x = \pi/6$), $|f(x_i)| \ge 0.1$. With this assumption, we have $s = -1$ and hence we have

$$|f(x_i) - f_i| \le 0.5 \times 10^{-6}.$$

We now use the formula $\bar{D}_h^{(2)} f(x)$ to approximate $f''(x)$ as given in the above remark. Assume that other than these rounding error, the formula $\bar{D}_h^{(2)} f(x)$ is calculated exactly. Then the total error bound given by (6.12) takes the form

$$|f''(\pi/6) - \bar{D}_h^{(2)} f(\pi/6)| \le \frac{h^2}{12} |f^{(4)}(\xi)| + \frac{4E}{h^2},$$

where $E = 0.5 \times 10^{-6}$ and $\xi \approx \pi/6$. Thus, we have

$$|f''(\pi/6) - \bar{D}_h^{(2)} f(\pi/6)| \le \frac{h^2}{12} \cos\left(\frac{\pi}{6}\right) + \frac{4}{h^2}(0.5 \times 10^{-6}) \approx 0.0722h^2 + \frac{2 \times 10^{-6}}{h^2} =: E(h).$$

The bound $E(h)$ indicates that there is a smallest value of h, call it h^*, below which the error bound will begin to increase. To find it, let $E'(h) = 0$, with its root being h^*. This leads to $h^* \approx 0.0726$. □

6.2 Numerical Integration

In this section we derive and analyze numerical methods for evaluating definite integrals. The problem is to evaluate the number

$$I(f) = \int_a^b f(x)dx. \tag{6.13}$$

Most such integrals cannot be evaluated explicitly, and with many others, it is faster to integrate numerically than explicitly. The approximation of $I(f)$ is usually referred to as **numerical integration** or **quadrature**.

The idea behind numerical integration is to approximate the integrand $f(x)$ to a much simpler function that can be integrated easily. One obvious approximation is the interpolation by polynomials. Thus, we approximate $I(f)$ by $I(p_n)$, where $p_n(x)$ is the polynomial of degree $\leq n$ which agrees with $f(x)$ at the distinct points $x_0, \cdots, x_n$. The approximation is written as

$$I(p_n) = A_0 f(x_0) + A_1 f(x_1) + \cdots + A_n f(x_n).$$

The weights could be calculated as $A_i = I(l_i)$, with $l_i(x)$ the ith Lagrange multiplier.

Assume that the integrand $f(x)$ is sufficiently smooth on some interval $[c, d]$ containing a and b so that we can write

$$f(x) = p_n(x) + f[x_0, \cdots, x_n, x]\phi_n(x),$$

where

$$\phi_n(x) = \prod_{j=0}^{n}(x - x_j).$$

Then the error is given by

$$E(f) = I(f) - I(p_n) = \int_a^b f[x_0, \cdots, x_n, x]\phi_n(x)dx. \tag{6.14}$$

In particular, if $\phi_n(x)$ is of **one sign** on (a, b), then, by the Mean-value theorem for integrals, we have

$$\int_a^b f[x_0, \cdots, x_n, x]\phi_n(x)dx = f[x_0, \cdots, x_n, \xi]\int_a^b \phi_n(x)dx, \text{ for some } \xi \in (a, b). \tag{6.15}$$

If, in addition, $f(x)$ is $n + 1$ times continuously differentiable on (c, d), we get

$$E(f) = \frac{1}{(n+1)!}f^{(n+1)}(\eta)\int_a^b \phi_n(x)dx, \text{ for some } \eta \in (c, d). \tag{6.16}$$

We now consider the case when $n = 0$. Then

$$f(x) = f(x_0) + f[x_0, x](x - x_0).$$

Hence

$$I(p_0) = (b - a)f(x_0).$$

If $x_0 = a$, then this approximation becomes

$$I(f) \approx I_R(f) := (b - a)f(a) \tag{6.17}$$

and is called **rectangle rule**. Since $\phi_0(x) = x - a$, this function is of one sign in (a, b) and therefore, the error E^R of the rectangle rule takes the form

$$E_R(f) = f'(\eta)\int_a^b (x - a)dx = \frac{f'(\eta)(b - a)^2}{2} \tag{6.18}$$

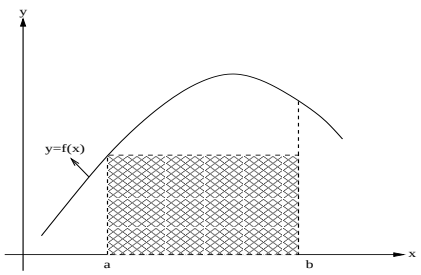


Fig. 6.2. Rectangle Rule.

We now consider the case when $n = 1$. Then

$$f(x) = f(x_0) + f[x_0, x_1](x - x_0) + f[x_0, x_1, x]\phi_1(x).$$

To get $\phi_1(x) = (x - x_0)(x - x_1)$ of one sign on (a, b), we choose $x_0 = a$ and $x_1 = b$. Then we have

$$I(f) = \int_a^b \{f(a) + f[a, b](x - a)\}dx + \frac{1}{2}f''(\eta)\int_a^b (x - a)(x - b)dx$$

or

$$I(f) \approx I_T(f) := \frac{1}{2}(b - a)\{f(a) + f(b)\} \tag{6.19}$$

with the error

$$E_T(f) = -\frac{f''(\eta)(b - a)^3}{12} \text{ some } \eta \in (a, b). \tag{6.20}$$

This rule is called the **Trapezoidal Rule**.

Example 6.5. Approximate the integral

$$I = \int_0^1 \frac{dx}{1 + x}.$$

The true value is $I = \log(2) \approx 0.693147$. Using the trapezoidal rule (6.19), we get

$$I_T = \frac{1}{2}[1 + \frac{1}{2}] = \frac{3}{4} = 0.75.$$

Therefore, the error is $I - I_T \approx -0.0569$. □

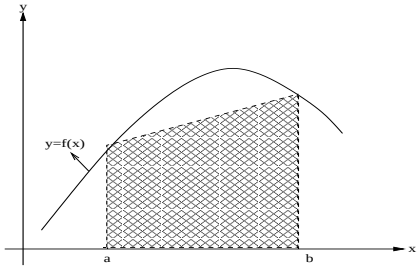


Fig. 6.3. Trapezoidal Rule.

To improve on this approximation, when $f(x)$ is not a nearly linear function on $[a, b]$, break the interval $[a, b]$ into smaller subintervals and apply the Trepezoidal rule (6.19) on each subinterval. We will derive a general formula for this. Let us subdivide the interval $[a, b]$ into n equal subintervals of length

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

with endpoints of the subintervals as

$$x_j = a + jh, \quad j = 0, 1, \cdots, n.$$

Then break the integral into n subintegrals, we get

$$\begin{aligned} I(f) &= \int_a^b f(x)dx \\ &= \int_{x_0}^{x_n} f(x)dx \\ &= \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} f(x)dx. \end{aligned}$$

Approximate each subintegral by Trapezoidal rule (6.19), we get

$$I(f) \approx I_T^n(f) = h\left[\frac{f(x_0)+f(x_1)}{2}\right] + h\left[\frac{f(x_1)+f(x_2)}{2}\right] + \cdots + h\left[\frac{f(x_{n-1})+f(x_n)}{2}\right].$$

The terms on the right can be combined to give the simpler formula

$$I_T^n(f) := h\left[\frac{1}{2}f(x_0) + f(x_1) + f(x_2) + \cdots + f(x_{n-1}) + \frac{1}{2}f(x_n)\right]. \tag{6.21}$$

This rule is called **Composite Trapezoidal rule**.

Example 6.6. Approximate the integral

$$I = \int_0^1 \frac{dx}{1+x}.$$

As we have seen in example 3.3, the true value is $I = \log(2) \approx 0.693147$. Now let us use this piecewise Trapezoidal rule with $n = 2$. Then we have

$$I = \int_0^1 \frac{dx}{1+x} = \int_0^{1/2} \frac{dx}{1+x}dx + \int_{1/2}^1 \frac{dx}{1+x}dx.$$

and therefore we have

$$I_T^2(f) \approx 0.70833.$$

Thus the error is -0.0152. □

We now calculate $I(p_2(x))$ to obtain the formula for the case when $n = 2$. Let us choose $x_0 = a$, $x_1 = (a+b)/2$ and $x_2 = b$. The quadratic interpolating polynomial can be written as

$$p_2(x) = f(a) + f[a,b](x-a) + f\left[a, b, \frac{a+b}{2}\right](x-a)(x-b)$$

Then

$$\int_a^b p_2(x)dx = f(a)(b-a) + f[a,b]\frac{(b-a)^2}{2} - f\left[a, b, \frac{a+b}{2}\right]\frac{(b-a)^3}{6}.$$

Using the symmetry property of divided difference, we can write

$$f\left[a, b, \frac{a+b}{2}\right] = f\left[a, \frac{a+b}{2}, b\right].$$

Therefore, we have

$$\int_a^b p_2(x)dx = f(a)(b-a) + f[a,b]\frac{(b-a)^2}{2} - f\left[a, \frac{a+b}{2}, b\right]\frac{(b-a)^3}{6}.$$

But we have $f[a,b](b-a) = f(b) - f(a)$ and

$$f\left[a, \frac{a+b}{2}, b\right](b-a)^2 = \left(f\left[\frac{a+b}{2}, b\right] - f\left[a, \frac{a+b}{2}\right]\right)(b-a) = 2\left(f(b) - 2f\left(\frac{a+b}{2}\right) - f(a)\right).$$

Using these expression, we get

$$\begin{aligned}\int_a^b p_2(x)dx &= (b-a)\left\{f(a) + \frac{f(b)-f(a)}{2} - \frac{1}{3}\left(f(b) - 2f\left(\frac{a+b}{2}\right) + f(a)\right)\right\} \\ &= \frac{b-a}{6}\left\{f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right\}\end{aligned}$$

We thus arrive at the formula

$$I(f) \approx I_s(f) := \int_a^b p_2(x)dx = \frac{b-a}{6}\left\{f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right\} \tag{6.22}$$

which is the famous **Simpson's Rule**.

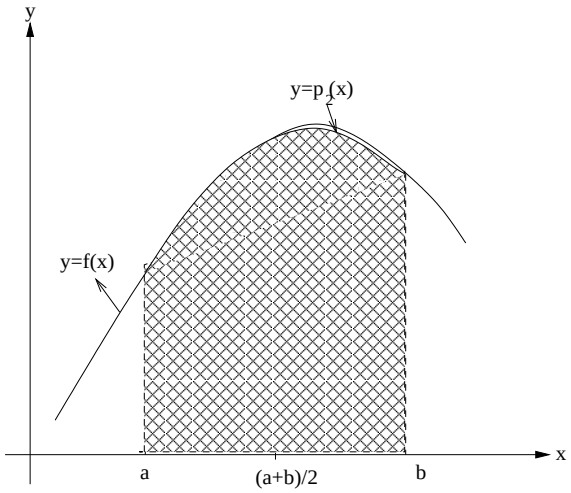


Fig. 6.4. Simpson Rule.

Example 6.7. Approximate the integral

$$I = \int_0^1 \frac{dx}{1+x}.$$

The true value is $I = \log(2) \approx 0.693147$. Using the Simpson's rule (6.22), we get

$$I_s = \frac{1}{6}[1 + \frac{8}{3} + \frac{1}{2}] = \frac{25}{36} \approx 0.694444.$$

Therefore, the error is $I - I_s \approx 0.001297$. □

Let us now obtain the error formula for Simpson's rule. Note that for any distinct nodes x_0, x_1 and x_2 in (a, b), the function $\phi_2(x) = (x - x_0)(x - x_1)(x - x_2)$ is not of one sign on (a, b). Therefore, the idea followed in deriving error formula for Trapezoidal rule cannot be adopted here. Rather, if we choose $x_0 = a$, $x_1 = (a + b)/2$, $x_2 = b$, then one can show by direct integration or by symmetry arguments that

$$\int_a^b \phi_2(x)dx = \int_a^b (x - a)\left(x - \frac{a+b}{2}\right)(x - b)dx = 0.$$

In this special case, if we can choose x_3 in such a way that $\phi_3(x) = (x - x_3)\phi_2(x)$ is of one sign on (a, b) and f is four times continuously differentiable, then we have

$$E_S(f) = -\frac{f^{(4)}(\eta)[(b-a)/2]^5}{90}, \tag{6.23}$$

which follows from the following lemma.

Lemma 6.8. *If ϕ_n is not of one-sign but*

$$\int_a^b \phi_n(x)dx = 0.$$

Further if can choose x_{n+1} in such a way that $\phi_{n+1}(x) = (x - x_{n+1})\phi_n(x)$ is of one-sign on (a, b) and if $f(x)$ is $n + 2$ times continuously differentiable, then

$$E(f) = \frac{1}{(n+2)!} f^{(n+2)}(\eta) \int_a^b \phi_{n+1}(x)dx, \quad \text{for some } \eta \in (c, d). \tag{6.24}$$

Proof. Since

$$f[x_0, \cdots, x_n, x] = f[x_0, \cdots, x_n, x_{n+1}] + f[x_0, \cdots, x_{n+1}, x](x - x_{n+1}),$$

we have from (6.14)

$$E(f) = \int_a^b f[x_0, \cdots, x_n, x_{n+1}]\phi_n(x)dx + \int_a^b f[x_0, \cdots, x_{n+1}, x](x - x_{n+1})\phi_n(x)dx$$

Further since $\int_a^b \phi_n(x)dx = 0$, the first term vanishes and we are left with

$$E(f) = \int_a^b f[x_0, \cdots, x_{n+1}, x](x - x_{n+1})\phi_n(x)dx.$$

Thus, if we choose x_{n+1} in such a way that $\phi_{n+1}(x) = (x - x_{n+1})\phi_n(x)$ is of one-sign on (a, b) and if $f(x)$ is $n + 2$ times continuously differentiable, then using Mean-value theorem for integration, we can arrive at the formula (6.24). □

Let us now derive the **composite Simpson rule**. Taking $a = x_{i-1}$, $b = x_i$, $x_{i-1/2} = (x_i + x_{i-1})/2$ and $x_i - x_{i-1} = h$ in Simpson rule, we get

$$\int_{x_{i-1}}^{x_i} f(x)dx \approx \frac{h}{6}\left\{f(x_{i-1}) + 4f(x_{i-1/2}) + f(x_i)\right\}.$$

Summing for $i = 1, \cdots, N$, we get

$$\int_a^b f(x)dx = \sum_{i=1}^{N} \int_{x_{i-1}}^{x_i} f(x)dx \approx \frac{h}{6}\sum_{i=1}^{N} \left\{f(x_{i-1}) + 4f(x_{i-1/2}) + f(x_i)\right\}.$$

Therefore, the **composite Simpson's rule** takes the form

$$I_s^n(f) = \frac{h}{6}\left[f(x_0) + f(x_N) + 2\sum_{i=1}^{N-1} f(x_i) + 4\sum_{i=1}^{N} f(x_{i-1/2})\right] \tag{6.25}$$

All the rules so far derived can be written in the form

$$I(f) = \int_a^b f(x)dx \approx w_0 f(x_0) + w_1 f(x_1) + \cdots + w_n f(x_n). \tag{6.26}$$

Here w_i are called **weights**, which are non-negative constants. The nodes are picked in such a way that the quadrature rule is exact for polynomials of degree $\leq n$. These methods are refered to **Newton-Conte formula** of order n. But it is possible to make such a rule exact for polynomials of degree $\leq 2n + 1$ by choosing the nodes appropriately. This is the basic idea of **Gaussian rules**.

Let us consider the special case

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=0}^{n} w_i f(x_i) \tag{6.27}$$

The weights w_i and the nodes x_i $(i = 0, \cdots, n)$ are to be chosen in such a way that the error

$$E_n(f) = \int_{-1}^{1} f(x)dx - \sum_{i=0}^{n} w_i f(x_i) \tag{6.28}$$

is zero when $f(x)$ is a polynomial of degree $\leq 2n + 1$. To derive equations for the nodes and weights, we first note that

$$E_n(a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m) = a_0 E_n(1) + a_1 E_n(x) + \cdots + a_n E_n(x^m).$$

Thus, $E_n(f) = 0$ for every polynomial of degree $\leq m$ if and only if $E_n(x^i) = 0$ for $i = 0, 1, \cdots, m$.

Case 1: $n = 0$. Since there are two parameters, namely, w_0 and x_0, we consider the requiring $E_0(1) = E_0(x) = 0$. This gives $\int_{-1}^{1} 1dx - w_0 = 0$, and $\int_{-1}^{1} xdx - w_0x_0 = 0$. These gives $w_0 = 2$ and $x_0 = 0$. Thus, we have the formula

$$\int_{-1}^{1} f(x)dx \approx 2f(0), \tag{6.29}$$

which is the required Gaussian quadrature for $n = 0$.

Case 2: $n = 1$. There are four parameters, w_0, w_1, x_0 and x_1 and thus we put four constraints on these parameters:

$$E_1(x^i) = \int_{-1}^{1} x^i dx - (w_0x_0^i + w_1x_1^i) = 0, \quad i = 0, 1, 2, 3.$$

This gives a system of nonlinear equations

$$w_1 + w_2 = 2, \quad w_1x_1 + w_2x_2 = 0, \quad w_1x_1^2 + w_2x_2^2 = \frac{2}{3}, \quad w_1x_1^3 + w_2x_2^3 = 0$$

The solutions are $w_1 = w_2 = 1$ and $x_1 = -1/\sqrt{3}$ and $x_2 = 1/\sqrt{3}$ which lead to the unique formula

$$\int_{-1}^{1} f(x)dx \approx f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right) =: I_{G1}(f). \tag{6.30}$$

Case 3: General. There are $2(n + 1)$ free parameters x_i and w_i for $i = 0, 1, \cdots, n$. The equations to be solved are $E_n(x^i) = 0, \quad i = 0, 1, \cdots, 2n + 1$ or

$$\sum_{j=0}^{n} w_j x_j^i = \begin{cases} 0, & i = 1, 3, \cdots, 2n + 1 \\ \dfrac{2}{i+1}, & i = 0, 2, \cdots, 2n \end{cases}$$

These are nonlinear equations and their solvability is not at all obvious. But most of the computer softwares will have programs to produce these nodes and weights or to directly perform the numerical integration. There is also another approach to the development of the numerical integration formula (6.26) using the theory of orthogonal polynomials, which is outside the scope of this course.

The formulas constructed above are called the **Gaussian numerical integration formula** or **Gaussian quadrature**. Note that this formula is limited to an integral over $[-1, 1]$. But this limitation can easily be removed by introducing the linear change of variable

$$x = \frac{b + a + t(b - a)}{2}, \quad -1 \le t \le 1. \tag{6.31}$$

Thus, an integral

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

can be transferred to

$$I(f) = \frac{b - a}{2} \int_{-1}^{1} f\left(\frac{b + a + t(b - a)}{2}\right) dt.$$

The following theorem provides the error formula for the Gaussian quadrature.

Example 6.9. Approximate the integral

$$I = \int_0^1 \frac{dx}{1 + x}.$$

Note that the true value is $I = \log(2) \approx 0.693147$. To use the Gaussian quadrature, we first need to make the linear change of variable (6.31) with $a = 0$ and $b = 1$ and we get

$$x = \frac{t}{2}, \quad -1 \le t \le 1.$$

Thus the required integration is

$$I = \int_0^1 \frac{dx}{1+x} = \int_{-1}^1 \frac{dt}{3+t}.$$

We need to take $f(t) = 1/(3+t)$ in the Gaussian quadrature formula (6.30) and we get

$$\int_0^1 \frac{dx}{1+x} = \int_{-1}^1 \frac{dt}{3+t} \approx f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right) \approx 0.692308 \approx I_{G1}(f).$$

Therefore, the error is $I - I_{G1} \approx 0.000839$. □

Definition 6.10 (Degree of Precision).

The degree of precision of a quadrature formula is the positive integer n such that $E(p_k) = 0$ for all polynomials $p_k(x)$ of degree $\leq n$, but for which $E(p_{n+1}) \neq 0$ for some polynomial $p_{n+1}(x)$ of degree $n+1$.

Example 6.11. Let us determine the degree of precision of Simpson rule. It will suffice to apply the rule over the interval $[0, 2]$.

$$\int_0^2 dx = 2 = \frac{2}{6}(1+4+1), \quad \int_0^2 xdx = 2 = \frac{2}{6}(0+4+2), \int_0^2 x^2 dx = \frac{8}{3} = \frac{2}{6}(0+4+4)$$
$$\int_0^2 x^3 dx = 4 = \frac{2}{6}(0+4+8) \int_0^2 x^4 dx = \frac{32}{5} \neq \frac{2}{6}(0+4+16) = \frac{20}{3}.$$

Therefore, the degree of precision is 3. □