System of Linear Equations

In the matrix notation, the system of n linear algebraic equations in n-unknowns:

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned}\right\} \quad \text{——(1)}$$

Can be written as:

$$AX = b \quad \text{——②}$$

Where

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}_{n\times n}, \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{n\times 1}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}_{n\times 1}$$

- The matrix $[A|b]$ of order $n \times (n+1)$ is called the augmented matrix.
- If all b_i's are zero, then the system (① or ②) is said to be homogeneous, and if at least one b_i is non-zero then it is in-homogeneous or non-homogeneous.
- System ② has unique solution if and only if determinant of A (denoted by $|A|$) is non-zero. And the solution is:

$$X = A^{-1}b$$

- The homogeneous system ($b_i = 0$, $i = 1, 2, \cdots n$) possesses only a trivial solution, that is, $x_1 = x_2 = \cdots = x_n = 0$ (when $|A| \neq 0$)

- The methods for solving system of linear equations may broadly be classified into two types:

i) Direct Methods: These methods produce the exact solution after a finite number of steps.

ii) Iterative Methods: These methods give a sequence of approximate solutions, which converges to the exact solution when the number of steps tend to infinity.

Direct Methods

i) Cramer rule
ii) Gauss-elimination method
iii) Gauss-Jordan elimination method

— You are familiar with these from Mathematics-II Course.

Iterative Method

A general linear iterative method for the solution of the system (2) may be defined in the following form:

$$X^{(k+1)} = HX^{(k)} + C, \quad k = 0, 1, 2, \cdots \qquad (3)$$

Where $X^{(k+1)}$ and $X^{(k)}$ are the approximations for X (exact sol^n.) at the $(k+1)^{th}$ and k^{th} iterations, respectively. H is called iteration matrix depending on A, C is a column vector.

When $k \longrightarrow \infty$, $x^{(k)} \longrightarrow X$ (exact sol.^{n}), which is given by

$$X = A^{-1}b \quad \text{——} (4)$$

Therefore, from (3):

$$A^{-1}b = HA^{-1}b + C \quad \text{——} (5)$$

$\therefore$ the column vector C is given by:

$$C = (I - H)A^{-1}b \quad \text{——} (6)$$

where I denotes identity matrix of order $n \times n$.

— We now determine the iteration matrix H and the column vector C for a few well known iteration methods:

Jacobi Iteration Method:

Assume that the quantities a_{ii} $(i = 1, 2, \cdots, n)$ in (2) are pivot elements. The equations (2) may be re-written as:

$$\begin{aligned}
a_{11}x_1 &= -(a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n) + b_1 \\
a_{22}x_2 &= -(a_{21}x_1 + a_{23}x_3 + \cdots + a_{2n}x_n) + b_2 \\
&\vdots \\
a_{nn}x_n &= -(a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{n\,n-1}x_{n-1}) + b_n
\end{aligned} \quad \text{——} (7)$$

Note: pivot must always be non-zero. If not, can be made by row exchange (eq^n. exchange)

The Jacobi (also known as Gauss-Jacobi) iteration method may now be defined as:

$$x_1^{(k+1)} = -\frac{1}{a_{11}}\left(a_{12}x_2^{(k)} + a_{13}x_3^{(k)} + \cdots + a_{1n}x_n^{(k)} - b_1\right)$$

$$x_2^{(k+1)} = -\frac{1}{a_{22}}\left(a_{21}x_1^{(k)} + a_{23}x_3^{(k)} + \cdots + a_{2n}x_n^{(k)} - b_2\right)$$

$$\vdots$$

$$x_n^{(k+1)} = -\frac{1}{a_{nn}}\left(a_{n1}x_1^{(k)} + a_{n2}x_2^{(k)} + \cdots + a_{nn-1}x_{n-1}^{(k)} - b_n\right)$$

— (8)

$k = 0, 1, 2, \cdots$

— Since we replace the complete vector $x^{(k)}$ in the right hand side of (8) at the end of each iteration. This method is also called as Method of Simultaneous Displacement.

Note: Eq. (8) is well-defined as each a_{ii} $(i = 1, 2, \cdots, n)$ is a pivot, hence non-zero.

— Here $k = 0, 1, 2, \cdots$ represent the iteration index, and each value of k will generate one approximation of the solution X

— In the matrix form, Jacobi method can be represented as

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

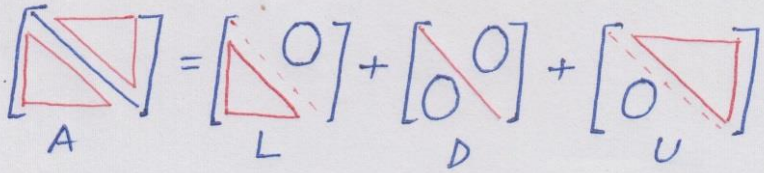
$$= HX^{(k)} + C$$

where $H = -D^{-1}(L+U)$ and $C = D^{-1}b$

Here, L and U are respectively lower and upper triangular matrices (with zero diagonal). And D is the diagonal matrix such that

$$A = L + D + U$$

Note: Any square matrix can be decomposed as $A = L + D + U$:



Example 1: (Jacobi method)

Solve the system of equations:

$$\left.\begin{aligned} 4x_1 + x_2 + x_3 &= 2 \\ x_1 + 5x_2 + 2x_3 &= -6 \\ x_1 + 2x_2 + 3x_3 &= -4 \end{aligned}\right]$$

Matrix notation:

$$\underbrace{\begin{bmatrix} 4 & 1 & 1 \\ 1 & 5 & 2 \\ 1 & 2 & 3 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} 2 \\ -6 \\ -4 \end{bmatrix}}_{b}$$

$$L = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 2 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 3 \end{bmatrix}, \quad U = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix}$$

Iteration matrix: $H = -D^{-1}(L+U)$

$$= -\begin{bmatrix} 4 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 3 \end{bmatrix}^{-1} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 2 \\ 1 & 2 & 0 \end{bmatrix} = -\begin{bmatrix} 1/4 & 0 & 0 \\ 0 & 1/5 & 0 \\ 0 & 0 & 1/3 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 2 \\ 1 & 2 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1/4 & -1/4 \\ -1/5 & 0 & -2/5 \\ -1/3 & -2/3 & 0 \end{bmatrix}$$

Column vector $c = D^{-1}b = \begin{bmatrix} 1/4 & 0 & 0 \\ 0 & 1/5 & 0 \\ 0 & 0 & 1/3 \end{bmatrix} \begin{bmatrix} 2 \\ -6 \\ -4 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -6/5 \\ -4/3 \end{bmatrix}$

$\therefore$ Jacobi iterative method becomes:

$$x^{(k+1)} = \begin{bmatrix} 0 & -1/4 & -1/4 \\ -1/5 & 0 & -2/5 \\ -1/3 & -2/3 & 0 \end{bmatrix} x^{(k)} + \begin{bmatrix} 1/2 \\ -6/5 \\ -4/3 \end{bmatrix}, \quad k = 0, 1, 2, \cdots$$

——— (*)

Note that to start the iterative process given by (*) we need $x^{(0)}$ – known as initial guess.

$$X^{(0)} = \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \\ x_3^{(0)} \end{bmatrix}$$

Choose $X^{(0)} = \begin{bmatrix} 0.5 \\ -0.5 \\ -0.5 \end{bmatrix}$

Then using (*):

$$X^{(1)} = \underbrace{\begin{bmatrix} 0 & -1/4 & -1/4 \\ -1/5 & 0 & -2/5 \\ -1/3 & -2/3 & 0 \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} 0.5 \\ -0.5 \\ -0.5 \end{bmatrix}}_{X^{(0)}} + \underbrace{\begin{bmatrix} 1/2 \\ -6/5 \\ -4/3 \end{bmatrix}}_{C} = \begin{bmatrix} 0.7500 \\ -1.1000 \\ -1.1667 \end{bmatrix}$$

Using $X^{(1)}$, we can get $X^{(2)}$ from (*): $X^{(2)} = \begin{bmatrix} 1.0667 \\ -0.8033 \\ -0.8500 \end{bmatrix}$

Similarly using $X^{(2)}$, we can get $X^{(3)}$ from (*): $X^{(3)} = \begin{bmatrix} 0.9300 \\ -1.0733 \\ -1.1000 \end{bmatrix}$

Likewise, we can continue to compute 4^{th}, 5^{th}, ... approximations to the solution.

Note: Exact soln. for this system is: $x_1 = 1$, $x_2 = -1$, $x_3 = -1$

□

Gauss – Seidel Iterative method

Gauss-Seidel method can be obtained from Jacobi method as:

– Use on the right hand side of ⑧, all the available values from the present iteration. We write the Gauss-Seidel method as:

$$\left.\begin{aligned}
x_1^{(k+1)} &= -\frac{1}{a_{11}}\left(a_{12}x_2^{(k)} + a_{13}x_3^{(k)} + \cdots + a_{1n}x_n^{(k)}\right) + \frac{b_1}{a_{11}} \\
x_2^{(k+1)} &= -\frac{1}{a_{22}}\left(a_{21}x_1^{(k+1)} + a_{23}x_3^{(k)} + \cdots + a_{2n}x_n^{(k)}\right) + \frac{b_2}{a_{22}} \\
&\vdots \\
x_n^{(k+1)} &= -\frac{1}{a_{nn}}\left(a_{n1}x_1^{(k+1)} + a_{n2}x_2^{(k+1)} + \cdots + a_{n\,n-1}x_{n-1}^{(k+1)}\right) + \frac{b_n}{a_{nn}}
\end{aligned}\right\} \quad (10)$$

which may be re-arranged in the form:

$$\left.\begin{aligned}
a_{11}x_1^{(k+1)} &= -\sum_{i=2}^{n} a_{1i}x_i^{(k)} + b_1 \\
a_{21}x_1^{(k+1)} + a_{22}x_2^{(k+1)} &= -\sum_{i=3}^{n} a_{2i}x_i^{(k)} + b_2 \\
&\vdots \\
a_{n1}x_1^{(k+1)} + \cdots + a_{nn}x_n^{(k+1)} &= b_n
\end{aligned}\right\} \quad (11)$$

– Since we replace the vector $X^{(k)}$ in the right hand side of ⑧ element by element, thus, this method is also called the method of successive displacement.

– In matrix notation, (11) becomes:

$$(D+L)X^{(k+1)} = -UX^{(k)} + b$$

OR $$X^{(k+1)} = -(D+L)^{-1}U X^{(k)} + (D+L)^{-1}b$$

$$= HX^{(k)} + C, \quad k = 0, 1, 2, \cdots \qquad \text{——(12)}$$

where, $H = -(D+L)^{-1}U$ and $C = (D+L)^{-1}b$

Example 2: (Gauss-Seidel method):

Solve the system of equations:

$$\left.\begin{aligned} 2x_1 - x_2 + 0\cdot x_3 &= 7 \\ -x_1 + 2x_2 - x_3 &= 1 \\ 0x_1 - x_2 + 2x_3 &= 1 \end{aligned}\right\}$$

Use initial guess: $X^{(0)} = [0,0,0]^T$

$$A = \begin{bmatrix} 2 & 1 & 0 \\ -1 & 2 & -1 \\ 0 & 1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 7 \\ 1 \\ 1 \end{bmatrix}$$

$$(D+L) = \begin{bmatrix} 2 & 0 & 0 \\ -1 & 2 & 0 \\ 0 & 1 & 2 \end{bmatrix}, \quad U = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

Gauss-Seidel method is

$$X^{(k+1)} = HX^{(k)} + C, \quad \text{where } H = -(D+L)^{-1}U, \; C = (D+L)^{-1}b$$

$$k = 0, 1, 2, \cdots$$

$$(D+L)^{-1} = \begin{bmatrix} 1/2 & 0 & 0 \\ 1/4 & 1/2 & 0 \\ 1/8 & 1/4 & 1/2 \end{bmatrix}, \quad (D+L)^{-1}U = \begin{bmatrix} 1/2 & 0 & 0 \\ 1/4 & 1/2 & 0 \\ 1/8 & 1/4 & 1/2 \end{bmatrix}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1/2 & 0 \\ 0 & -1/4 & -1/2 \\ 0 & -1/8 & -1/4 \end{bmatrix}$$

$$(D+L)^{-1}b = \begin{bmatrix} 1/2 & 0 & 0 \\ 1/4 & 1/2 & 0 \\ 1/8 & 1/4 & 1/2 \end{bmatrix}\begin{bmatrix} 7 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 7/2 \\ 9/4 \\ 13/8 \end{bmatrix}$$

$$\therefore \quad X^{(k+1)} = \begin{bmatrix} 0 & 1/2 & 0 \\ 0 & 1/4 & 1/2 \\ 0 & 1/8 & 1/4 \end{bmatrix} X^{(k)} + \begin{bmatrix} 7/2 \\ 9/4 \\ 13/8 \end{bmatrix}, \quad X^{(0)} = [0,0,0]^T$$

$$k = 0, 1, 2, \cdots$$

Hence, we can calculate approximations as:

$$x^{(1)} = \begin{bmatrix} 0 & 1/2 & 0 \\ 0 & 1/4 & 1/2 \\ 0 & 1/8 & 1/4 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 7/2 \\ 9/4 \\ 13/8 \end{bmatrix} = \begin{bmatrix} 3.5000 \\ 3.6250 \\ 2.3123 \end{bmatrix}$$

$$x^{(2)} = \text{---} = \begin{bmatrix} 4.6250 \\ 3.6250 \\ 2.3125 \end{bmatrix}$$

$$x^{(3)} = \text{---} = \begin{bmatrix} 5.3125 \\ 4.3125 \\ 2.6563 \end{bmatrix}$$

and so on.

Note: Exact soln. is $x = [6, 5, 3]^T$ □

System of Linear Equations (Contd.)

So far we have discussed about two numerical methods to solve system of linear equations of the form $Ax=b$. The basic idea of iterative method is to construct a sequence of vectors $X^{(k)}$ that enjoy the property of convergence, that is:

$$X = \lim_{k \to \infty} X^{(k)} \qquad \text{——— (1)}$$

where X is the exact solution to $Ax=b$ and $X^{(k)}$ is the k^{th} approximation of the exact solution X using an iterative method.

- The iterative process is stopped at the minimum value of n such that $|X^{(n)} - X| < \varepsilon$. ——— (2)

where ε is a fixed tolerance (given by the user)

- However, in many engineering problems, exact sol^n. is obviously not available, it is necessary to introduce suitable stopping Criteria to monitor the convergence of the iterative process. (This will be discussed later)

- Next, we will establish the convergence criteria for Jacobi and Gauss-Seidel methods.

Note: In case if two (or more) iterative methods converge, the betterness of an iterative method will be decided by the Rate of Convergence (that is how fast approximations approach to the exact sol^n as in Eq. (1)).

Take the linear iterative process as:

$$X^{(k+1)} = HX^{(k)} + C \quad , \quad k = 0, 1, 2, \cdots \qquad \text{—— (3)}$$

for $Ax = b$. Here X is the exact solution of $Ax = b$.

$\therefore$ X satisfy (3) $\Rightarrow$ $X = HX + C$ —— (4)

Subtracting (4) from (3):

$$\Rightarrow X^{(k+1)} - X = H\left(X^{(k)} - X\right) \qquad \text{—— (5)}$$

Substitute $X^{(k)} - X = \varepsilon^{(k)}$, where $\varepsilon^{(k)}$ denotes error at k^{th} approximation. Equation (5) becomes:

$$\varepsilon^{(k+1)} = H\varepsilon^{(k)} \quad , \quad k = 0, 1, 2, \cdots \qquad \text{—— (6)} \quad \text{(Recurrence relation)}$$

Note that:

$$\varepsilon^{(k)} = H\varepsilon^{(k-1)} = H\left(H\varepsilon^{(k-2)}\right) = H^2\varepsilon^{(k-2)} = H^2\left(H\varepsilon^{(k-3)}\right)$$
$$= H^3\varepsilon^{(k-3)} = \cdots = H^k\varepsilon^{(0)}$$

Where $\varepsilon^{(0)} = X^{(0)} - X$, that $\varepsilon^{(0)}$ denotes error in initial guess

($X^{(0)}$ ↓ initial guess)

$$\varepsilon^{(k)} = H^k\varepsilon^{(0)} \quad , \quad k = 0, 1, 2, \cdots \qquad \text{—— (7)}$$

Note: Here we assumed that the iteration matrix H remains constant for each iteration — STATIONARY method

From (7), it can be noted that:

$$\varepsilon^{(k)} \longrightarrow 0 \quad \text{if and only if} \quad H^k \longrightarrow 0 \quad \text{as} \quad k \longrightarrow \infty$$

- That is error in k^{th} approximation approaches to zero as $k \longrightarrow \infty$ if and only if $H^k \longrightarrow 0$ as $k \longrightarrow \infty$.

Note: $\varepsilon^{(0)}$ is finite, as this is error in initial guess

- Since, H is a matrix and k is iteration index (+ve integer), therefore, it is not easy to compute H^k (Think of case when k is large). Hence, we will derive some other criteria to establish the convergence of linear iterative methods.

Before the convergence results, let's have few definitions:

Spectral Radius

The largest eigenvalue (in magnitude) is called the spectral radius of a matrix A. Spectral radius of a matrix A is denoted by $\rho(A)$

For example: If $A_{3\times3}$ have eigenvalues as: $\lambda_1 = 1, \lambda_2 = -5, \lambda_3 = 3$

then $\rho(A) = 5$

Diagonal Dominance

If $A = [a_{ij}]_{n\times n}$, then A is diagonally dominant if

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

That is absolute value of the diagonal element in each row is greater than or equal to absolute sum of non-diagonal element in that row

and if

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

then A is said to be <u>strictly diagonally dominant</u>.

For example: $A = \begin{bmatrix} 5 & -1 & 2 \\ 3 & -6 & -1 \\ 2 & -5 & 8 \end{bmatrix}$ is strictly diagonally dominant.

<u>Result 1</u> (Necessary and Sufficient condition for Convergence)

A necessary and sufficient condition for convergence of an iterative method of the form:

$$X^{(k+1)} = H X^{(k)} + C, \quad k = 0, 1, 2, \cdots$$

is that the eigenvalues of the iteration matrix H satisfy

$$\boxed{\rho(H) < 1}$$

<u>Result 2</u> (Sufficient condition for the convergence of Jacobi method)

If A is strictly diagonally dominant, then the Jacobi iteration scheme converges for any choice of initial guess.

<u>Result 3</u> (Sufficient condition for the convergence of Gauss-Seided method)

If A is strictly diagonally dominant, then the Gauss-Seidel iteration method converges for any choice of initial guess $(X^{(0)})$.

<u>Note</u>: Each result have a very nice mathematical proof. (Not included here).

Rate of convergence: The rate of convergence of an iterative method is given by:

$$R = -\log_{10}\left(\rho(H)\right) \quad \text{OR} \quad R = -\ln\left(\rho(H)\right) \qquad \text{——} (8)$$

where, $\rho(H)$ is the spectral radius of H.

Note: In (8) we are using the final result. To derive (8) you need a mathematical proof!

Remarks:

a.) We have a relation between the spectral radii of iteration matrices of Jacobi and Gauss-Seided methods as

$$\rho(H_{GS}) = \left[\rho(H_J)\right]^2$$

Note: I will prove this relation in the next week

where H_{GS} – iteration matrix for Gauss-Seided

H_J – iteration matrix for Jacobi

Now if (say) $\rho(H_J) = 0.5 \Rightarrow \rho(H_{GS}) = 0.25$ — Now relate this with Result 1 (OR Eq. 8) you can conclude here that if both the methods converges, then rate of convergence of G-S method is more rapid than Jacobi method.

b.) Next assume if $\rho(H_J) = 1.5 \Rightarrow \rho(H_{GS}) = 2.25$ — Now using Result 1 you can conclude that if both the methods diverges (Result 1), then G-S method diverges more rapidly as compared to Jacobi method (Eq. 8).

□

Successive Over-Relaxation Method

This method is a generalization of Gauss-Seided method.

Gauss-Seidel method for $AX = b$ is given as:

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

$$i = 1, 2, \cdots, n$$

Successive relaxation method is obtained from (1) by introducing a relaxation parameter ω as:

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}\, x_j^{(k)}\right] + (1-\omega)\, x_i^{(k)} \quad \text{---(2)}$$

$$i = 1, 2, \cdots, n$$

Equation (2) can be written in vector/matrix form as:

$$X^{(k+1)} = \omega\left[D^{-1}b - D^{-1}L\,X^{(k+1)} - D^{-1}U\,X^{(k)}\right] + (1-\omega)\,I\,X^{(k)}$$

OR,

$$\left(I + \omega D^{-1}L\right)X^{(k+1)} = \left[(1-\omega)I - \omega D^{-1}U\right]X^{(k)} + \omega D^{-1}b$$

OR,

$$\boxed{X^{(k+1)} = (D+\omega L)^{-1}\left[(1-\omega)D - \omega U\right]X^{(k)} + \omega(D+\omega L)^{-1}b} \quad \text{---(3)}$$

Now, comparing (3) with general iterative form that is

$$X^{(k+1)} = HX^{(k)} + C, \quad k = 0, 1, 2, \cdots$$

For Successive Relaxation method we have:

$$H_{SR} = (D+\omega L)^{-1}\left[(1-\omega)D - \omega U\right]$$

$$C = \omega(D+\omega L)^{-1} b$$

This method is consistent for any $\omega \neq 0$ and for $\omega = 1$ it coincides with Gauss-Seidel method. In particular, if $0 < \omega < 1$ the method is called under-relaxation, while if $\omega > 1$ it is called over-relaxation.

Q: How to determine the optimum value ω_{opt} for which the convergence rate is the highest?

To answer this question, consider the iteration matrices of Jacobi and ~~Seidel method~~ Successive relaxation methods:

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

$$H_{SR} = (D+\omega L)^{-1}\left[(1-\omega)D - \omega U\right]$$

Let μ be an eigenvalue of H_J and λ be an eigen value of H_{SR}.

The relation between λ and μ is:

$$\mu = \frac{\lambda + \omega - 1}{\lambda^{1/2}\omega} \qquad \text{——— (4)}$$

$$\Rightarrow \lambda - \mu\omega\lambda^{1/2} + (\omega - 1) = 0 \quad \text{(which is quadratic in } \lambda^{1/2})$$

$$\therefore \lambda^{1/2} = \frac{1}{2}\left[\mu\omega \pm \sqrt{\mu^2\omega^2 - 4(\omega-1)}\right]$$

$$= \frac{1}{2}\left[\mu\omega\right] \pm \frac{\mu}{2}\sqrt{(\omega-\omega_1)(\omega-\omega_2)}$$

where $\omega_{1,2} = \frac{2}{\mu^2}\left[1 \mp \sqrt{1-\mu^2}\right]$

When $\omega = \omega_1$, $\rho(H_{SR})$ is smallest (implying fastest convergence), hence the optimal relaxation parameter is given as:

$$\omega_{opt} = \frac{2}{\mu^2}\left[1 - \sqrt{1-\mu^2}\right] = \frac{2}{1+\sqrt{1-\mu^2}} \qquad \text{——— (5)}$$

Also, when $\omega = \omega_1$, we get

$$\lambda = \frac{1}{4}\mu^2\omega^2 = \omega - 1$$

For convergence, we require

$$|\lambda| < 1 \Rightarrow |\omega - 1| < 1$$

OR $\quad 0 < \omega < 2$

For $0 < \omega < 1$ — Under Relaxation.
And $1 \leq \omega < 2$ — Over Relaxation.

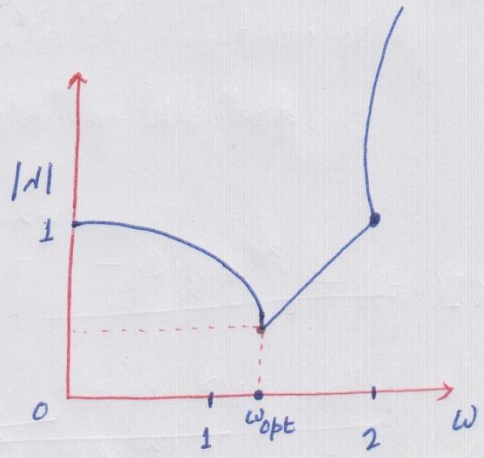


Fig: $|\lambda|$ Vs. ω

The rate of convergence for SR method is:

$$R_{SR} = -\ln(\omega_{opt} - 1) \qquad \text{—— ⑥}$$

Note that if $\omega = 1$ (where SR becomes GS) from ④, we have:

$$\mu = \lambda^{1/2} \quad \text{OR} \quad \lambda = \mu^2$$

where μ is the arbitrary eigenvalue of H_J and λ is the eigenvalue of H_{GS}.

$$\Rightarrow \quad \rho(H_{GS}) = [\rho(H_J)]^2 \qquad \text{—— ⑦}$$

This is the same equation as given in Remarks of Lecture-2

$\therefore$ We can derive ⑦ if we can derive ④.

Also note from ⑤ that ω_{opt} is real if $|\mu| < 1$. Therefore, the necessary condition for the SR method to converge is that the corresponding Jacobi method is convergent.

Example 1: (To find ω_{opt})

$$\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \qquad \text{—— ⑧}$$

Q. Find ω_{opt}.

Ans: Use $\omega_{opt} = \dfrac{2}{1 + \sqrt{1 - \mu^2}}$, where μ is the eigenvalue/spectral radius of Jacobi method for system (⑧)

Example 2: (SR method)

$$\begin{bmatrix} -2 & -1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \\ 1 \end{bmatrix} \quad \text{---------- (**)}$$

Set up the SR method. Find optimal relaxation parameter (ω_{opt}) and the rate of convergence. Perform three iterations of the SR method. Take initial guess as $X^{(0)} = [0, 0, 0]^T$

Ans: To find ω_{opt}: First find H_J and then get spectral radius of H_J, say μ.

Then $$\omega_{opt} = \frac{2}{\mu^2}\left(1 - \sqrt{1-\mu^2}\right)$$

Eigenvalues of H_J: $\lambda_J = 0, \pm\frac{1}{\sqrt{2}}$

Take $\mu = +\frac{1}{\sqrt{2}}$ ↓ Largest (in magnitude)

To find rate of convergence: when $\omega = \omega_{opt}$

$$\rho(H_{SR}) = \omega_{opt} - 1 = 0.1715 \text{ (Check it!)}$$

Rate of convergence $R_{SR} = -\ln(0.1715) = \cdots$

To start the iterations: Put $\omega = \omega_{opt} = 0.1715$ in (3) (SR iterations) and get

$$\begin{cases} X^{(k+1)} = H_{SR} X^{(k)} + C, \quad k = 0, 1, 2, \cdots \\ H_{SR} = (D + \omega L)^{-1}\left[(1-\omega)D - \omega U\right] \\ C = \omega (D + \omega L)^{-1} b \end{cases}$$

Initial guess (for $k=0$) is given.

— Compute $X^{(1)}$, $X^{(2)}$ and $X^{(3)}$ (Three iterations)

— Complete it. Exercise. □

System of Non-Linear Equations

So far, we have discussed about the numerical solution of system of linear equations. Now, we will discuss about the numerical solution to system of non-linear equations.

- As solving non-linear equations is much more challenging as compared to the linear equations.

- Therefore, first we will consider a single non-linear equation, then slowly will move towards system of non-linear equations (that is two or more number of equations)

- Examples of non-linear equations (Transcendental and Polynomials)

  i) $y = f(x) = \cos x - xe^{-x}$

  ii) $y = f(x) = x^3 - 5x + 1$ □

- We will consider the real valued functions, that is,

  $f : \mathbb{R} \longrightarrow \mathbb{R}$, where $\mathbb{R} = (-\infty, \infty)$ — set of real numbers.

- Solving a non-linear equation means we have to find a scalar α such that $f(\alpha) = 0$ (That is α satisfies $f(x)$). Then α is called solution or zero or root of $f(x)$.

- Methods for the numerical approximation of a zero of $f(x)$ are usually iterative. Thus the aim is to generate a

sequence of approximations (similar to linear case) $\{x^{(k)}\}_{k=0}^{\infty}$ (2)

such that

$$\boxed{\lim_{k\to\infty} x^{(k)} = \alpha} \quad \text{---(1)}$$

, where α is the exact sol^n of $f(x)$.

— Geometric Approach to Rootfinding (of a non-linear function)

Here, we introduce following methods for finding roots:

i) The bisection method
ii) The Chord method
iii) The secant method
iv) The false position (or Regula Falsi) method
v) The Newton's method.

Growing complexity of the algorithms.

The Bisection method:

In the bisection method, the only information that is being used is the sign of the function $f(x)$ at the end of any bisection (sub-interval).

The bisection method is based on the following property:

Property 1 (IVT Theorem) Given a continuous function $f: [a,b] \to \mathbb{R}$, such that $f(a)f(b)<0$, then there exists $\alpha \in (a,b)$ (That is $a<\alpha<b$) such that $f(\alpha)=0$.

Note: IVT – Intermediate Value Theorem
– You know it from

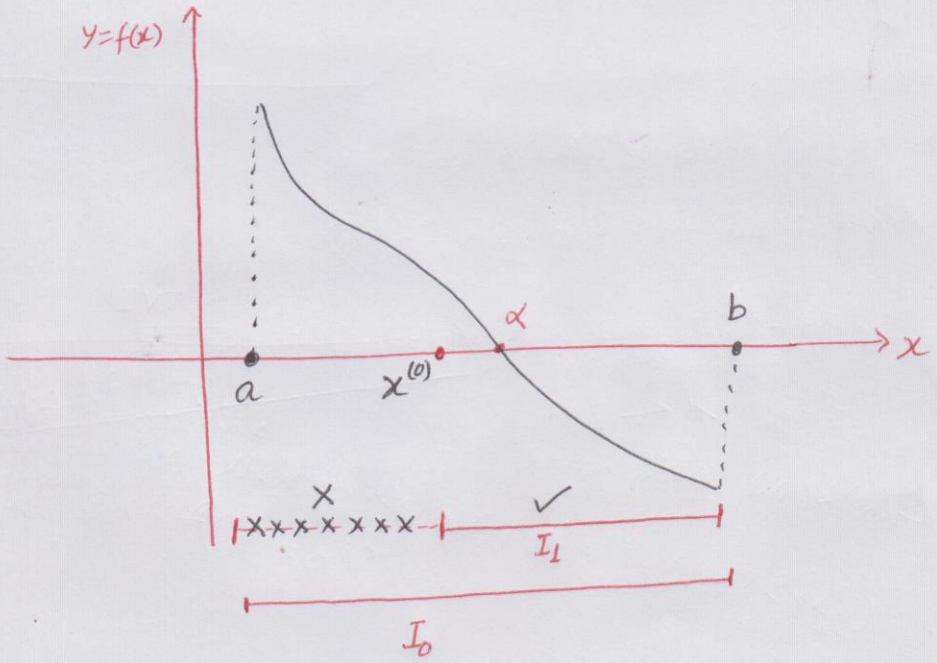
- Starting from $I_0 = [a,b]$, the bisection method generates a sequence of subintervals $I_k = [a^{(k)}, b^{(k)}]$, $k \geq 0$ with

$$I_k \subset I_{k-1}, \quad k \geq 1$$

and enjoys the property that $f(a^{(k)}) f(b^{(k)}) < 0$.

- Precisely, we set $a^{(0)} = a$, $b^{(0)} = b$ and $x^{(0)} = (a^{(0)} + b^{(0)})/2$, then, for $k \geq 0$:

$$\left.\begin{array}{l} \text{set } a^{(k+1)} = a^{(k)}, \; b^{(k+1)} = x^{(k)}, \text{ if } f(x^{(k)}) f(a^{(k)}) < 0 \\ \text{set } a^{(k+1)} = x^{(k)}, \; b^{(k+1)} = b^{(k)}, \text{ if } f(x^{(k)}) f(b^{(k)}) < 0 \\ \text{finally, set } x^{(k+1)} = \left(a^{(k+1)} + b^{(k+1)}\right)/2 \end{array}\right] \text{—②}$$



— The bisection iteration terminates at the m^{th} step for which

$$|x^{(m)} - \alpha| \leq |I_m| \leq \varepsilon$$

where ε is a fixed tolerance and $|I_m|$ is the length of I_m. As for the speed of convergence of the bisection method, note that $|I_0| = b-a$, while

$$|I_k| = I_0/2^k = \frac{(b-a)}{2^k}, \quad k \geq 0 \qquad \text{—— (3)}$$

Denoting $e^{(k)} = |x^{(k)} - \alpha|$ — Absolute error at k^{th} iteration, from (3) it follows that

$$|e^{(k)}| \leq \frac{(b-a)}{2^k}, \; k \geq 0 \text{ which implies}$$

$$\lim_{k \to \infty} |e^{(k)}| = 0 \iff x^{(k)} \longrightarrow \alpha \text{ as } k \to \infty.$$

∴ The bisection method is convergent. To get the value of m (for which $|x^{(m)} - \alpha| \leq \varepsilon$, we must take

$$m \geq \log_2(b-a) - \log_2(\varepsilon) = \frac{\log\left((b-a)/\varepsilon\right)}{\log(2)} \simeq \frac{\log\left((b-a)/\varepsilon\right)}{0.6931} \qquad (4*)$$

Since m is an (+ve) integer, we take m as the next nearest integer.

- This singles out the bisection method. The (only) drawback of this method is its slow convergence.

- The Methods of Chord, Secant and Regula Falsi and Newton's Method :

- To devise algorithms with better convergence properties than the bisection method, it is necessary to include information from the values attained by f and, possibly, also by its derivative f' (if f is differentiable) or by a suitable approximation.

- Let us expand f in a Taylor series around α and truncate the expansion at the first order, as:

$$f(\alpha) = 0 = f(x) + (\alpha - x) f'(\xi) \qquad \text{——— (5)}$$

where ξ lies between α and x.

- Equation (5) prompts the following iterative method:

For any $k \geq 0$, given $x^{(k)}$, determine $x^{(k+1)}$ by solving equation

$$\boxed{f(x^{(k)}) + (x^{(k+1)} - x^{(k)}) q_k = 0}$$

where q_k is a suitable approximation of $f'(x^{(k)})$.

— The methods described here amounts to find the intersection between x-axis and the straight line of slope q_k passing through the point $\left(x^{(k)}, f(x^{(k)})\right)$.

— Therefore, we have:

$$x^{(k+1)} = x^{(k)} - q_k^{-1} f(x^{(k)}), \quad k = 0, 1, 2, \cdots$$

Q: What are the possibilities for q_k?

— We consider <u>four</u> particular choices of q_k.

A) <u>The Chord Method</u>:

Here, we let

$$q = q_k = \frac{f(b) - f(a)}{b - a}, \text{ for all } k \geq 0$$

∴ we have:

$$\boxed{x^{(k+1)} = x^{(k)} - \left(\frac{b-a}{f(b) - f(a)}\right) f(x^{(k)}), \quad k = 0, 1, 2, \cdots} \quad \text{—— (6)}$$

Note that to start (6), <u>one</u> initial guess $x^{(0)}$ is required.

B) <u>The Secant Method</u>

In this case, take

$$q_k = \frac{f(x^{(k)}) - f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}, \quad k = 0, 1, 2, \cdots \quad \text{—— (7)}$$

∴ We have

$$x^{(k+1)} = x^{(k)} - \left(\frac{x^{(k)} - x^{(k-1)}}{f(x^{(k)}) - f(x^{(k-1)})} \right) f(x^{(k)}), \quad k = 0, 1, 2, \cdots$$

——(8)

Note that to start iteration with (8), we need <u>two</u> initial guesses.

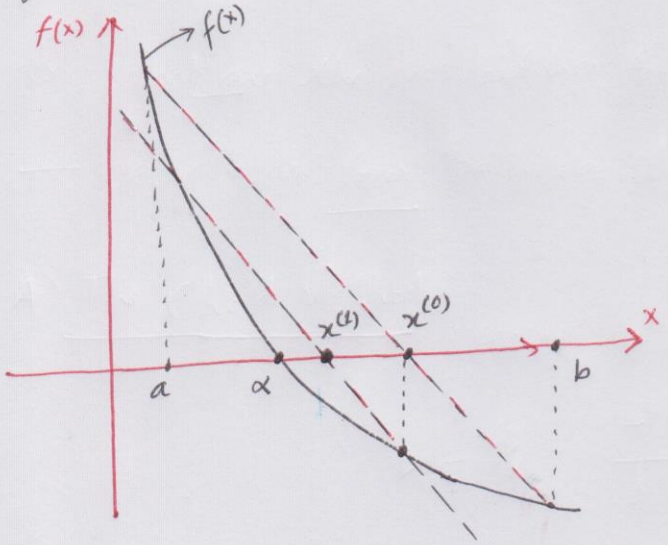


Fig1: Chord Method

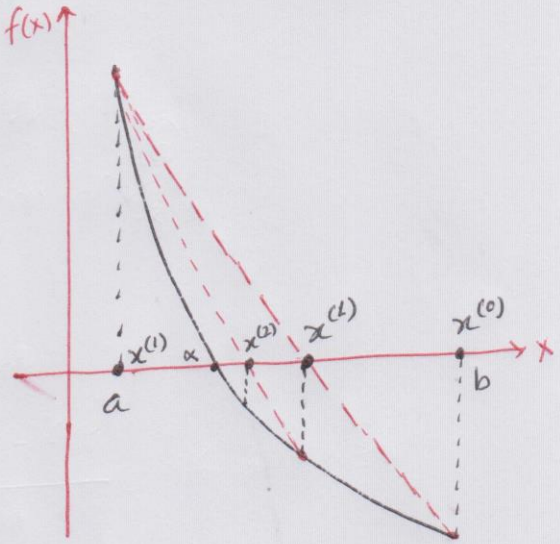


Fig2: Secant Method

(Here we need two initial guesses. Take guesses as:

$x^{(0)} = b \quad x^{(1)} = a$

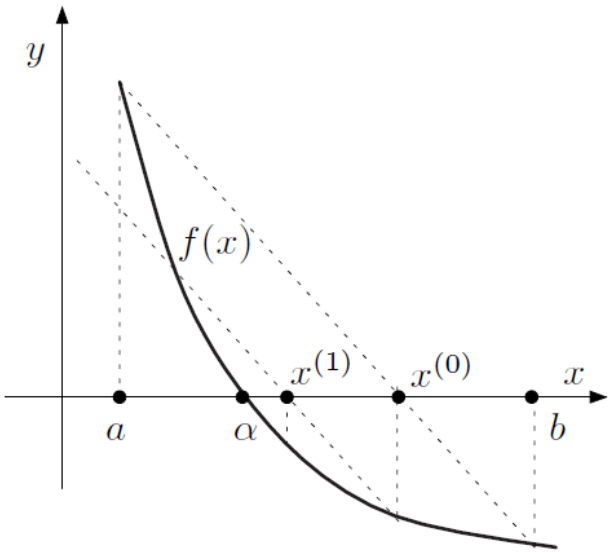


Fig. 1: Chord Method

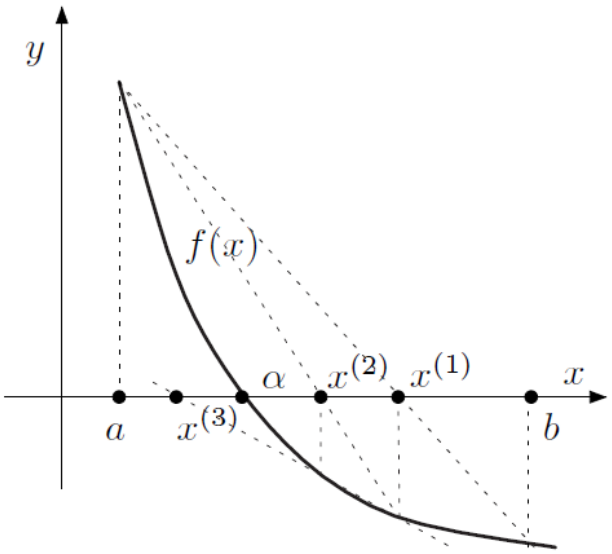


Fig. 2: Secant Methods

System of Non-Linear Equations

The Regula Falsi (or false position) method

- Variant of the Secant method.
- Here, instead of taking the secant line through the points $\left(x^{(k)}, f(x^{(k)})\right)$ and $\left(x^{(k-1)}, f(x^{(k-1)})\right)$, we take the one through $\left(x^{(k)}, f(x^{(k)})\right)$ and $\left(x^{(k')}, f(x^{(k')})\right)$.

where k' denotes the maximum index less than k such that

$$f(x^{(k)}) \cdot f(x^{(k')}) < 0.$$

Therefore, once two initial guesses $x^{(0)}$ and $x^{(1)}$ have been found such that

$$f(x^{(0)}) \cdot f(x^{(1)}) < 0,$$

we let

$$x^{(k+1)} = x^{(k)} - \left[\frac{x^{(k)} - x^{(k')}}{f(x^{(k)}) - f(x^{(k')})}\right] f(x^{(k)}), \quad k \geq 0 \qquad \text{---------- (1)}$$

- If error tolerance (ε) is given, then the iterative process (1) terminates at the m^{th}-step such that

$$|f(x^{(m)})| < \varepsilon$$

- Regula Falsi method has the same complexity as the Secant method
- However, unlike the Secant method, the iterates generated by Regula falsi are all contained with in the starting interval $[x^{(0)}, x^{(1)}]$.

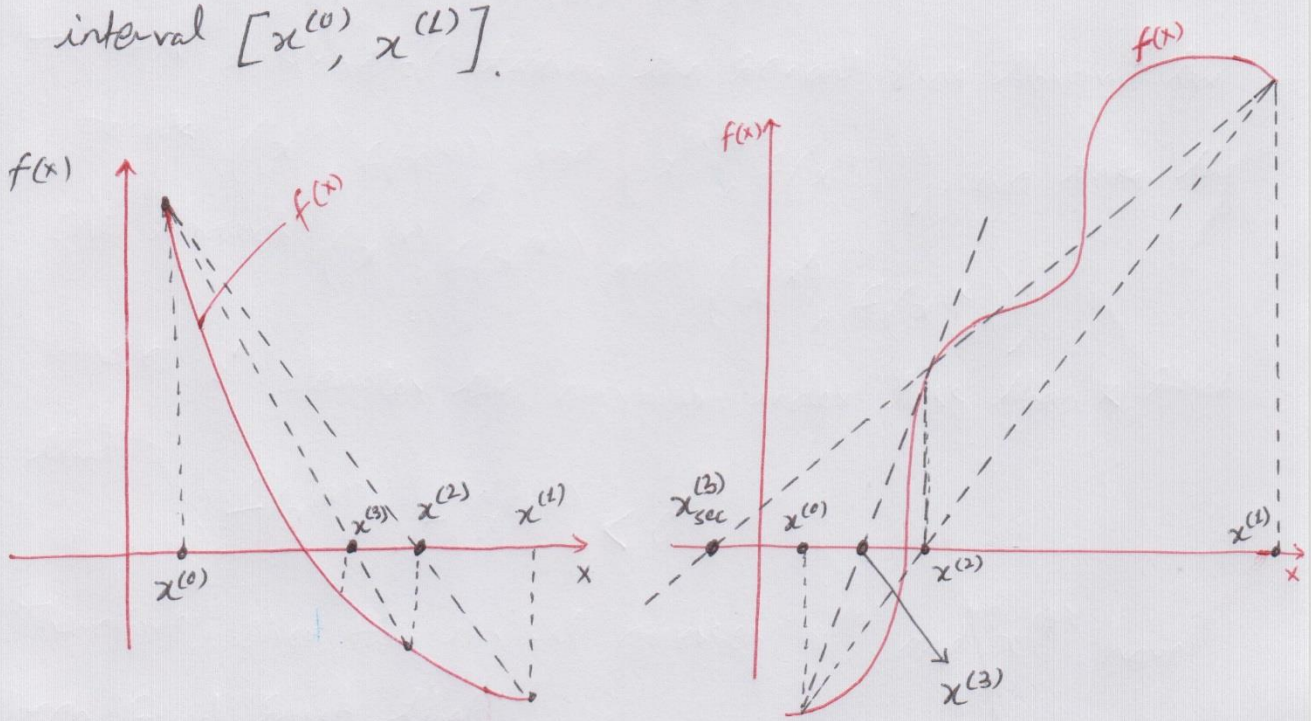


Fig 1: First two step of the Regula Falsi method for two different functions

- On the right side frame, we can notice that initial guesses are chosen as: $x^{(0)}$ and $x^{(1)}$.
- Note that $x^{(2)}$ computed by the Secant method coincides with Regula Falsi method, however, $x^{(3)}$ computed by secant (that is $x^{(3)}_{sec}$) falls outside the searching interval $[x^{(0)}, x^{(1)}]$.

Note: Regula Falsi and bisection methods can be regarded as globally convergent method.

Newton's (Newton-Raphson) method :

Assuming that f is continuously differentiable on a interval I (that is $f \in C^1(I)$) and $f'(\alpha) \neq 0$ (that is α is a simple root of f).

If we take

$$q_k = f'(x^{(k)}) \quad \text{for all } k \geq 0$$

and if one initial guess $x^{(0)}$ is given. Then the Newton's (or N-R) method is :

$$\boxed{x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \quad k \geq 0} \qquad \text{——(2)}$$

— At k^{th} iteration, Newton's method requires two functional evaluations : $f(x^{(k)})$ and $f'(x^{(k)})$.

— (You will see later) that increasing computational cost will be compensated for by a higher order of convergence.

$I = [a, b]$

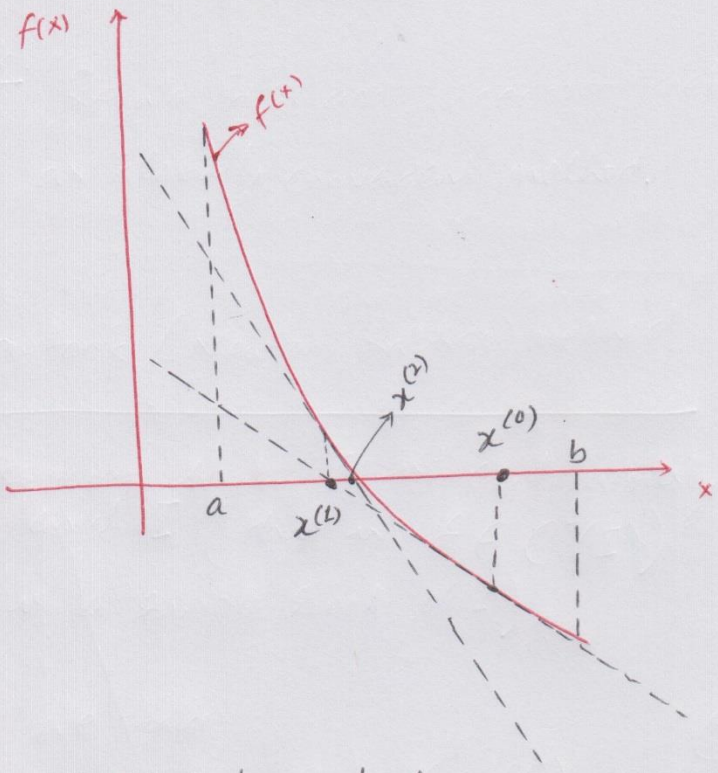


Fig 2: Newton's method.

Rate of Convergence (of iterative methods to find the root of a non-linear equation):

— An iterative method is said to be of order p (or has the rate of convergence p) if p is the largest positive real number for which there exists a finite constant $C\ (\neq 0)$ such that

$$|\varepsilon_{k+1}| \leq C\,|\varepsilon_k|^p$$

where $\varepsilon_k = x_k - \alpha$ is the error in the k^{th} approximation.

Here α is the exact root of $f(x)$.

Rate of convergance of Newton's method:

Newton's Method

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \quad k = 0, 1, 2, 3, \cdots \qquad \text{——(3)}$$

On substituting $x_{(k)} = \alpha + \varepsilon_k$, we get from (3):

$$\varepsilon_{k+1} = \varepsilon_k - \frac{f(\alpha + \varepsilon_k)}{f'(\alpha + \varepsilon_k)}$$

Using Taylor series expansions for f and f':

$$= \varepsilon_k - \frac{\left(\overset{\nearrow 0}{f(\alpha)} + \varepsilon_k f'(\alpha) + \frac{\varepsilon_k^2}{2!} f''(\alpha) + \cdots\right)}{\left(f'(\alpha) + \varepsilon_k f''(\alpha) + \cdots\right)}$$

$$= \varepsilon_k - \left[\varepsilon_k + \frac{1}{2}\frac{f''(\alpha)}{f'(\alpha)}\varepsilon_k^2 + \cdots\right]\left[1 + \frac{f''(\alpha)}{f'(\alpha)}\varepsilon_k + \cdots\right]^{-1}$$

↓

[Use Binomial expansion $(1+x)^n = \sum_{k=0}^{\infty} \binom{n}{k} x^k$]

OR,

$$\varepsilon_{k+1} = \frac{1}{2}\frac{f''(\alpha)}{f'(\alpha)}\varepsilon_k^2 + O(\varepsilon_k^3)$$

On neglecting ε_k^3 and higher powers of ε_k, we get

$$\varepsilon_{k+1} = C\,\varepsilon_k^2 \qquad \text{—— (4)}$$

where $C = \frac{1}{2}\,\frac{f''(\alpha)}{f'(\alpha)}$

Thus (using (4)) we conclude that Newton's method has quadratic convergence (or second order convergence) □

Exercise

1. Show that Regula Falsi method has linear rate of convergence (that is of order 1)

2. Show that Secant Method converges to α with order $p = (1+\sqrt{5})/2 \simeq 1.63$

Note: (Summary of order of convergence)

- Chord and Regula-Falsi have linear convergence ($p=1$)
- Secant has convergence of order $p = 1.63$ (larger than Chord/RF and smaller than N-R)
- Newton's Method has quadratic convergence ($p=2$). □

Multiple Roots

Let's consider a (non-linear) function $f(x)$

— To find the root of $f(x)$, write $f(x)=0$ — (1)

— If we can write (1) as

$$f(x) = (x-\alpha)^m g(x) = 0 \qquad (g(x) \text{ bounded})$$

such that $g(\alpha) \neq 0$, then α is called a multiple root of multiplicity m. In this case:

$$f\Big|_{x=\alpha} = f'\Big|_{x=\alpha} = \cdots = f^{(m-1)}\Big|_{x=\alpha} = 0$$

— If $m=1$, the number α is said to be simple root.

— Example: $f(x) = x^3 - 5x^2 + 7x - 3 = (x-3)(x-1)^2$

So, here $x=1$ is the double root of $f(x)$.
and $x=3$ is a simple root of $f(x)$.

— Multiple roots pose some difficulties for numerical method (many)

i) The function does not change sign at multiple roots!
— can not use root bracketing methods (like bisection and Regula Falsi)

ii) Another possible problem is related to the fact that not only $f(x)$ but also $f'(x)$ goes to zero at the root

— This poses problems for both the Newton's and Secant methods (which both contain the derivative or its estimate in the denominator of their respective formulas)

iii) A very simple way to circumvent these problems is based on the fact that it can be demonstrated theoretically that $f(x)$ will always reach zero before $f'(x)$. Therefore, if a zero check for $f(x)$ is incorporated, then the computation can be terminated before $f'(x)$ reaches to zero.

How to alleviate these problems?

— Following is the modification proposed for Newton's method:

$$x^{(k+1)} = x^{(k)} - m\frac{f(x^{(k)})}{f'(x^{(k)})}, \quad k \geq 0 \qquad \text{—— (2)}$$

where m is the multiplicity of the root.

— The drawback for alternative (2) is that it hinges on forknowledge of the multiplicity of the root!

— Another alternative, is to define a new function $u(x)$ such that-

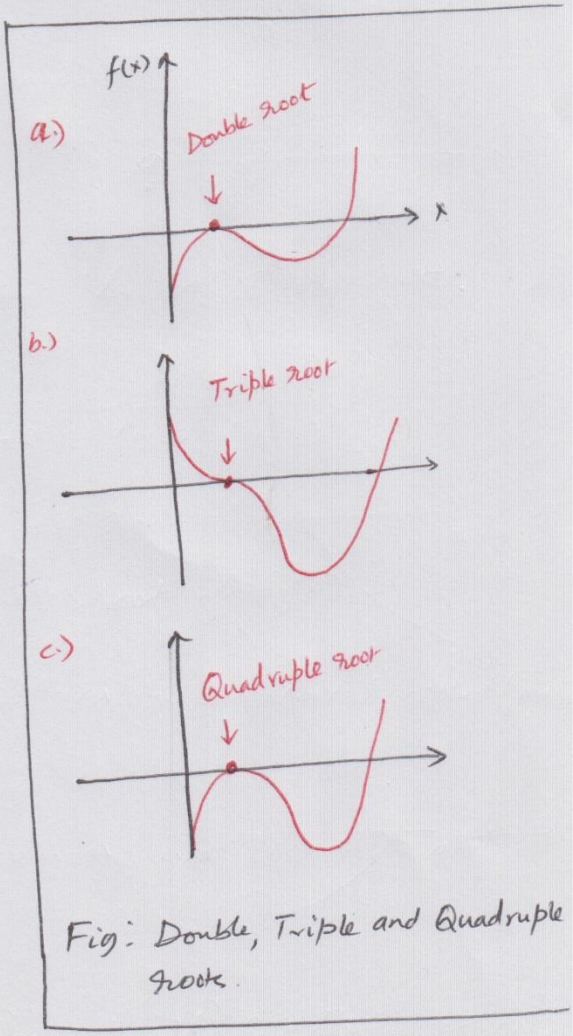


Fig: Double, Triple and Quadruple roots.

$$u(x) = \frac{f(x)}{f'(x)} \qquad \text{—— (3)}$$

— It can be shown that $u(x)$ has roots at all the same locations as the $f(x)$.

Substute (3) into Newton's iterative formula for simple roots (which was $x^{(k+1)} = x^{(k)} - f(x^{(k)})/f'(x^{(k)})$)

$$\Rightarrow x^{(k+1)} = x^{(k)} - \frac{u(x^{(k)})}{u'(x^{(k)})} \qquad \text{—— (4)}$$

Differentiate (3) $\Rightarrow$

$$u'(x) = \frac{f'(x)f'(x) - f(x)f''(x)}{[f'(x)]^2} \qquad \text{—— (5)}$$

Substitute (5) and (3) into (4): we have

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})\, f'(x^{(k)})}{[f'(x^{(k)})]^2 - f(x^{(k)})\, f''(x^{(k)})}, \quad k \geq 0$$

—— (6)

↳ Modified Newton's Method for Multiple Roots.

Exercise:

Use modified Newton's method to evaluate the multiple root of $f(x) = x^3 - 5x^2 + 7x + 3$, with an initial guess $x^{(0)} = 1$.

(Perform THREE iterations using (6))

□

Accuracy and Precision

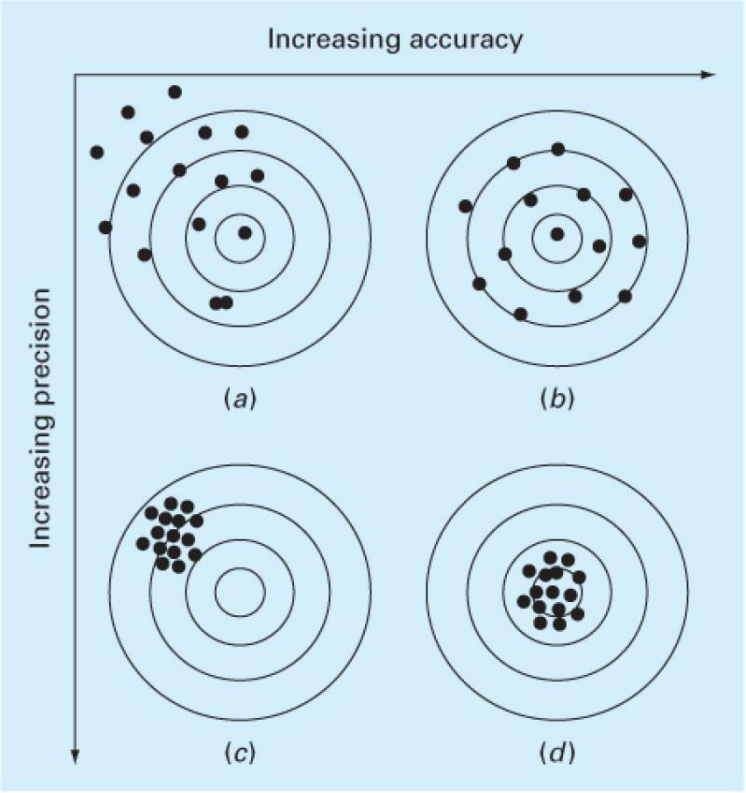
Accuracy:

Accuracy refers to how closely a computed or measured value agrees with the true value.

Precision:

Precision refers to how closely individual computed or measured values agree with each other.

a) inaccurate and imprecise
b) accurate and imprecise
c) inaccurate and precise
d) accurate and precise



Error Definitions (1/2)

Numerical errors arise due to use of approximations to represent exact mathematical operations and quantitates. In general, these include: Truncation errors and Round-off errors.

- Truncation errors: arise when approximations are used to represent exact mathematical procedure.
- Round-off errors: arise when numbers having limited significant figures are used to represent exact numbers.

For both the cases, we have

True value = approximation + error

True error (E_t): the difference between the true value and the approximation

E_t = true value - approximation --------- (1)

Error Definitions (1/2)

Absolute error ($|E_t|$): the absolute difference between the true value and the approximation.

Note: the drawback of (E_t) is that it takes no account of the order of magnitude of the value under examination.

True fractional relative error: the true error divided by the true value

$$\text{True fractional relative error} = \frac{\text{true value} - \text{approximation}}{\text{true value}}$$

Relative error (ε_t): the true fractional relative error expressed as a percentage

$$\varepsilon_t = \frac{\text{true value} - \text{approximation}}{\text{true value}} 100\ \% \qquad \text{-------- (2)}$$

Error Definitions (1/2)

Example-1: Suppose that you have the task of measuring the lengths of a bridge and a rivet and come up with 9999 and 9 cm, respectively. If the true values are 10,000 and 10 cm, respectively, compute (i) the true error and (ii) the true percent relative error for each case.

Solution: E_t = 10,000 – 9999 = 1 cm (for bridge)

E_t = 10 – 9 = 1 cm (for rivet)

(i) The percent relative error for bridge is:

$$\varepsilon_t = \frac{1}{10000}\ 100\% = 0.01\%$$

(ii) The percent relative error for rivet is:

$$\varepsilon_t = \frac{1}{10}\ 100\% = 10\%$$

Error Definitions (2/2)

The previous definitions of error relied on knowing a true value. If that is not the case, approximations can be made to the error as:

The approximate percent relative error can be given as the approximate error divided by the approximation, expressed as a percentage - though this presents the challenge of finding the approximate error!

$$\varepsilon_a = \frac{\text{approximate error}}{\text{approximation}} 100\ \%$$

For iterative processes, the error can be approximated as the difference in values between successive iterations

$$\varepsilon_a = \frac{\text{present approximate} - \text{previous approximation}}{\text{present approximation}} 100\ \% \quad \text{----- (4)}$$

Error Tolerance and Error Estimates

Often, when performing calculations, we may not be concerned with the sign of the error but are interested in whether the absolute value of the percent relative error is lower than a pre-specified tolerance (ε_s).

For such cases, the computation is repeated until $|\varepsilon_a| < \varepsilon_s$

This relationship is referred to as a stopping criterion

Note that for the remainder of our discussions, we almost always employ absolute values when using relative errors

We say that an approximation is correct to at least *n* significant figures (significant digits) if its $|\varepsilon_a|$ is smaller than ε_s that has a value

$$\varepsilon_s = (0.5 \times 10^{2-n})\% \quad \text{------- (5)}$$

Example-2: It is known that the exponential function can be computed using Maclaurin series, as:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!}\,\frac{x^4}{4!} + - - - + \frac{x^n}{n!}$$

Starting with the simplest version, $e^x = 1$, add terms one at a time to estimate $e^{0.5}$. Note that the true value is $e^{0.5} = 1.648721\ldots$Add terms until the absolute value of the ε_a falls below a prescribed ε_s conforming to three significant figures.

Solution:

Using Eq. (5) determine the error criterion that ensures a result is correct to at least three significant figures:

$$\varepsilon_s = (0.5 \times 10^{2-3})\,\% = 0.05\%$$

Thus, we will add terms to the series until ε_a falls below this level.

Terms	Result	ε_t, %	ε_a, %
1	1	39.3	
2	1.5	9.02	33.3
3	1.625	1.44	7.69
4	1.645833333	0.175	1.27
5	1.648437500	0.0172	0.158
6	1.648697917	0.00142	0.0158

Thus, after six terms are included, the approximate error falls below $\varepsilon_s = 0.05\%$, and computations are terminated.

Round-off Errors

Roundoff Errors:

Roundoff errors arise because digital computers cannot represent some quantities exactly. There are two major facets of roundoff errors involved in numerical calculations:

i. Digital computers have size and precision limits on their ability to represent numbers.
ii. Certain numerical manipulations are highly sensitive to roundoff errors.

Computer Representation of Numbers (1/2)

Number System: A convention used for representing quantities.

We are familiar with decimal, or base-10, number system. The base-10 system uses the 10 digits:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

to represent a number.

Numbers on the computer are represented with a binary, or base-2, system (having just 0 and 1)

Floating-Point Representation: Fractional quantities are typically represented in computers using floating point form. Here, the number is expressed as a fraction part, called a mantissa or significand, and an integer part, called exponent or characteristic, as an

Computer Representation of Numbers (2/2)

$$m \cdot b^e$$

Where m = the mantissa, b = the base of the number system being used, and e = the exponent. For example, 156.78 can be represented as 0.15678×10^3 in a floating-point base-10 system.

Note that mantissa is usually normalized if it has leading zero digits. For example, suppose the quantity 1/34 = 0.029411765…was stored in a floating-point base-10 system that allowed only four decimal places to be stored as

$$0.0294 \times 10^0$$

Which can be normalized as, 0.2941×10^{-1}, thus we retain an additional significant figure when the number is stores.

Extended Precision

The most commonly used precision for Computers are:

i. Single precision

ii. Double precision

- - -

Single Precision:

Computers that use IEEE format allow 24 bits to be used for the mantissa, which translates into about seven significant base-10 digits of precision with a range of about 10^{-38} to 10^{39}.

Double Precision:

Here, about fifteen to sixteen significant base-10 digits of precision with a range of approximately 10^{-308} to 10^{308}.

Round-off Errors with Arithmetic Manipulations (1/3)

Roundoff error can happen in several circumstances other than just storing numbers. For example:

i. Large computations: if a process performs a large number of computations, roundoff errors may build up to become significant

```
function sout = sumdemo()
s = 0;
for i = 1 : 10000
   s = s + 0.0001;
% notice that 0.0001 cannot be expressed exactly in base - 2.
end
sout = s;
```

```
function sout = sumdemo()
s = 0;
for i = 1 : 10000
   s = s + 0.0001;
>> format long
>> sumdemo
ans =
    0.99999999999991
```

ii. Adding a Large and a Small Number: Since the small number's mantissa is shifted to the right to be the same scale as the large number, digits are lost

Round-off Errors with Arithmetic Manipulations (2/3)

What if $0.0010 + 4000$ is represented
with 4 - digit mantissa and 1 - digit exponent?

$$\begin{array}{ll} 4.000 & \times 10^3 \\ \underline{0.000001} & \underline{\times 10^3} \\ 4.000001 & \times 10^3 \end{array} \quad \text{(chopped to } 4.000 \times 10^3\text{)}$$

iii. Smearing: Smearing occurs whenever the individual terms in a summation are larger than the summation itself Round-off errors:

$(x + 10^{-20}) - x = 10^{-20}$ mathematically, but
$x = 1$; $(x + 10^{-20}) - x$ gives a 0 in MATLAB!

Round-off Errors with Arithmetic Manipulations (3/3)

iv. Subtractive Cancellation: This refers to the round-off error induced when subtracting two nearly equal floating-point numbers.

For example, one common instance where this can occur involves finding the roots of a quadratic equation (or parabola) with the quadratic formula:

$$\begin{matrix} x1 \\ x2 \end{matrix} = \frac{-b \mp \sqrt{b^2 - 4ac}}{2a}$$

For cases where $b^2 \gg 4ac$, the difference in the numerator can be very small. In such cases double precision can mitigate the problem. To avoid this, an alternative formula can be used to minimize subtractive cancellation, as:

$$\begin{matrix} x1 \\ x2 \end{matrix} = \frac{-2c}{b \mp \sqrt{b^2 - 4ac}}$$

System of Non-linear Equations

— Numerical Solution of System of non-linear equations

— Let's generalizes to the n-dimensional case to find the zero/root of a function as:

$$F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$$

n is any +ve integer.

— Our problem is to find x^* in $\mathbb{R}^n$ such that $F(x^*) = 0$. ——(1)

For example:

Take the non-linear system

$$\left.\begin{aligned} e^{x_1^2 + x_2^2} - 1 &= 0 \\ e^{x_1^2 - x_2^2} - 1 &= 0 \end{aligned}\right]$$

Here $n = 2$ and

$$F(x) = \left[e^{x_1^2 + x_2^2} - 1,\ e^{x_1^2 - x_2^2} - 1\right]^T$$

OR

$$F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} \text{ where}$$

$$F_1(x) = e^{x_1^2 + x_2^2} - 1$$

$$F_2(x) = e^{x_1^2 - x_2^2} - 1$$

and $$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

For this system there is unique solution $x^* = 0$.

Q: How to extend the iterative methods discussed for 1-dimension to several dimensions?

- Here, we shall always assume that $F \in C^1(D)$ that is

$F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is a continuously differentiable on D, where $D \subseteq \mathbb{R}^n$.

- We denote also by $J_F(x)$ the Jacobian matrix associated with F and evaluated at the point $x = (x_1, x_2, \cdots, x_n)^T$ of $\mathbb{R}^n$, defined as

$$\left(J_F(x)\right)_{ij} = \left(\frac{\partial F_i}{\partial x_j}\right)(x), \quad i, j = 1, 2, \cdots n. \qquad \text{——(2)}$$

OR

$$J_F(x) = \begin{bmatrix} \frac{\partial F_1(x)}{\partial x_1} & \frac{\partial F_1(x)}{\partial x_2} & \cdots & \frac{\partial F_1(x)}{\partial x_n} \\ \frac{\partial F_2(x)}{\partial x_1} & \frac{\partial F_2(x)}{\partial x_2} & \cdots & \frac{\partial F_2(x)}{\partial x_n} \\ \vdots & & & \\ \frac{\partial F_n(x)}{\partial x_1} & \frac{\partial F_n(x)}{\partial x_2} & \cdots & \frac{\partial F_n(x)}{\partial x_n} \end{bmatrix}$$

Newton's Method :

Here, we consider the extension of Newton's method (for simple roots) for scalar equations to the vector case :

- Given $X^{(0)} \in \mathbb{R}^n$, for $k = 0, 1, \cdots$, until convergence

Solve
$$J_F(X^{(k)})\,\delta X^{(k)} = -F(X^{(k)})$$

Set
$$X^{(k+1)} = X^{(k)} + \delta X^{(k)}$$

} (3)

Therefore, at each iterative-step k the solution of a linear system with matrix $J_F(X^{(k)})$ is required.

Example 1:

$$\begin{cases} e^{x_1^2 + x_2^2} - 1 = 0 \\ e^{x_1^2 - x_2^2} - 1 = 0 \end{cases}$$

exact sol^n: $X^* = 0$

- Take $X^{(0)} = (0.1, 0.1)^T$

Then after 15 iterations you will get

$$X^{(15)} = (0.61 \times 10^{-5}, 0.61 \times 10^{-5})$$

- Again, if $X^{(0)} = (10, 10)^T$, then 220 iterations are needed to obtain a solution comparable to previous case.

- And, if $X^{(0)} = (20, 20)^T$, then Newton's method fails to converge.

Remarks:

— Thus, this example points out the high sensitivity of Newton's method on the choice of the initial guess $x^{(0)}$.

— Theoretically, it can be established that Newton's method is quadratically convergent only if $x^{(0)}$ is sufficiently close to the solution x^* and if the Jacobian matrix is non-singular.

— Also note that computation complexity (effort) can be excessively high as n gets large.

Derivation of Newton's method (for n = 2 case):

— Let's derive Eq. ③ for n = 2 case. Therefore, have

$$F = \left(F_1(x_1, x_2),\ F_2(x_1, x_2)\right)^T \quad \text{—— ④}$$

Means we have system of two non-linear equations

$$F_1(x_1, x_2) = 0$$
$$F_2(x_1, x_2) = 0$$

Let $(x_1^{(k)}, x_2^{(k)})$ be a suitable approximation to the root $x^* = (x_1^*, x_2^*)$ of the system ④. Let Δx_1 be an increment in $x_1^{(k)}$ and Δx_2 be an increment in $x_2^{(k)}$ such that $(x_1^{(k)} + \Delta x_1,\ x_2^{(k)} + \Delta x_2)$ is the exact solution of ①, that is:

(5)

$$\left.\begin{aligned} F_1\left(x_1^{(k)}+\Delta x_1,\ x_2^{(k)}+\Delta x_2\right)&=0\\ F_2\left(x_1^{(k)}+\Delta x_1,\ x_2^{(k)}+\Delta x_2\right)&=0 \end{aligned}\right]$$

Using Taylor's series expansion about point $\left(x_1^{(k)}, x_2^{(k)}\right)$, we get

$$F_1\left(x_1^{(k)}, x_2^{(k)}\right)+\left[\Delta x_1\frac{\partial}{\partial x_1}+\Delta x_2\frac{\partial}{\partial x_2}\right]F_1\left(x_1^{(k)}, x_2^{(k)}\right)+\frac{1}{2!}\left[\Delta x_1\frac{\partial}{\partial x_1}+\Delta x_2\frac{\partial}{\partial x_2}\right]^2$$

$$F_1\left(x_1^{(k)}, x_2^{(k)}\right)+\cdots=0$$

$$F_2\left(x_1^{(k)}, x_2^{(k)}\right)+\left[\Delta x_1\frac{\partial}{\partial x_1}+\Delta x_2\frac{\partial}{\partial x_2}\right]F_2\left(x_1^{(k)}, x_2^{(k)}\right)+\frac{1}{2!}\left[\Delta x_1\frac{\partial}{\partial x_1}+\Delta x_2\frac{\partial}{\partial x_2}\right]^2$$

$$F_2\left(x_1^{(k)}, x_2^{(k)}\right)+\cdots=0$$

Neglecting second and higher powers of Δx_1 and Δx_2, we get:

$$\left.\begin{aligned} F_1\left(x_1^{(k)}, x_2^{(k)}\right)+\Delta x_1\frac{\partial F_1\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_1}+\Delta x_2\frac{\partial F_1\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_2}&=0\\ F_2\left(x_1^{(k)}, x_2^{(k)}\right)+\Delta x_1\frac{\partial F_2\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_1}+\Delta x_2\frac{\partial F_2\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_2}&=0 \end{aligned}\right]$$ (5)

Solving (5) for Δx_1 and Δx_2, we get:

$$\begin{bmatrix} \dfrac{\partial F_1\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_1} & \dfrac{\partial F_1\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_2}\\ \dfrac{\partial F_2\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_1} & \dfrac{\partial F_2\left(x_1^{(k)}, x_2^{(k)}\right)}{\partial x_2} \end{bmatrix}\begin{bmatrix}\Delta x_1\\ \Delta x_2\end{bmatrix}=-\begin{bmatrix} F_1\left(x_1^{(k)}, x_2^{(k)}\right)\\ F_2\left(x_1^{(k)}, x_2^{(k)}\right)\end{bmatrix}$$ (6)

OR

$$J_F(x^{(k)})\,\Delta x^{(k)} = -F(x^{(k)})$$

which is nothing but Eq. (3).

The solution of the system (6) is:

$$\Delta x^{(k)} = -\left(J_F(x^{(k)})\right)^{-1} F(x^{(k)})$$

Where $\left(J_F(x^{(k)})\right)^{-1}$ is the inverse of Jacobian matrix $J_F(x^{(k)})$. Then set $x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$

□

Example 2:

Perform three iterations of the Newton's method to solve the system of equations:

$$x_1^2 + x_1 x_2 + x_2^2 = 7$$
$$x_1^3 + x_2^3 = 9$$

Here exact soln:
$x^* = (x_1^*, x_2^*) = (2, 1)$

with $x^{(0)} = (x_1^{(0)}, x_2^{(0)}) = (1.5, 0.5)$

Soln: Take $F_1 = x_1^2 + x_1 x_2 + x_2^2 - 7$
$F_2 = x_1^3 + x_2^3 - 9$
and $F = (F_1, F_2)$

$$\therefore J_F(x^{(k)}) = \begin{bmatrix} \frac{\partial F_1(x^{(k)})}{\partial x_1} & \frac{\partial F_1(x^{(k)})}{\partial x_2} \\ \frac{\partial F_2(x^{(k)})}{\partial x_1} & \frac{\partial F_2(x^{(k)})}{\partial x_2} \end{bmatrix}$$

Solve: $$\begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = -\left(J_F(x^{(k)})\right)^{-1} \begin{bmatrix} F_1(x^{(k)}) \\ F_2(x^{(k)}) \end{bmatrix}$$

OR,

$$\begin{bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \end{bmatrix} + \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \end{bmatrix} - \left(J_F(x^{(k)})\right)^{-1} \begin{bmatrix} F_1(x^{(k)}) \\ F_2(x^{(k)}) \end{bmatrix}$$

OR (in short) Solve:

$$x^{(k+1)} = x^{(k)} - \left(J_F(x^{(k)})\right)^{-1} F(x^{(k)})$$

where $k = 0, 1, 2, \cdots$

□

Lec. Notes-8

Roots for a function of complex variable

— The root of an equation $f(z)=0$, z is a complex variable, can be obtained by using the method (Newton's) for the system of non-linear equations:

$$f(z)=0 \quad , \quad z=x+iy$$

$$\Rightarrow f(z)=u(x,y)+iv(x,y)=0 \qquad \text{——} (1)$$

OR

$$\left.\begin{aligned} u(x,y)&=0\\ v(x,y)&=0 \end{aligned}\right\} \qquad \text{——} (2)$$

Thus, the problem of finding a complex root of (1) reduces to solving a system of two non-linear equations (2).

Example:

$$f(z)=z^3+1=0$$

with initial guess $(x_0,y_0)=(0.25,0.25)$

$$f(x+iy)=u(x,y)+iv(x,y)=(x^3-3xy^2+1)+i(3x^2y-y^3)=0$$

$$\therefore \quad u(x,y)=x^3-3xy^2+1=0$$

$$v(x,y)=3x^2y-y^3=0$$

Jacobian matrix $J=\begin{bmatrix} u_x & u_y \\ v_x & v_y \end{bmatrix}=\begin{bmatrix} 3x^2-3y^2 & -6xy \\ 6xy & 3x^2-3y^2 \end{bmatrix}$

and $|J|=9(x^2+y^2)^2$

$$\therefore \quad J^{-1} = \frac{1}{|J|}\begin{bmatrix} 3(x^2-y^2) & 6xy \\ -6xy & 3(x^2-y^2) \end{bmatrix}$$

Newton's method:

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \end{bmatrix} - \frac{1}{|J_k|}\begin{bmatrix} 3(x_k^2-y_k^2) & 6x_ky_k \\ -6x_ky_k & 3(x_k^2-y_k^2) \end{bmatrix}\begin{bmatrix} x_k^3-3x_ky_k^2+1 \\ 3x_k^2y_k-y_k^3 \end{bmatrix}$$

with $\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} 0.25 \\ 0.25 \end{bmatrix}$ $\quad k = 0, 1, 2, \cdots$

Start interation to get: $X^{(1)} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$, $X^{(2)} = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$, $\cdots$.

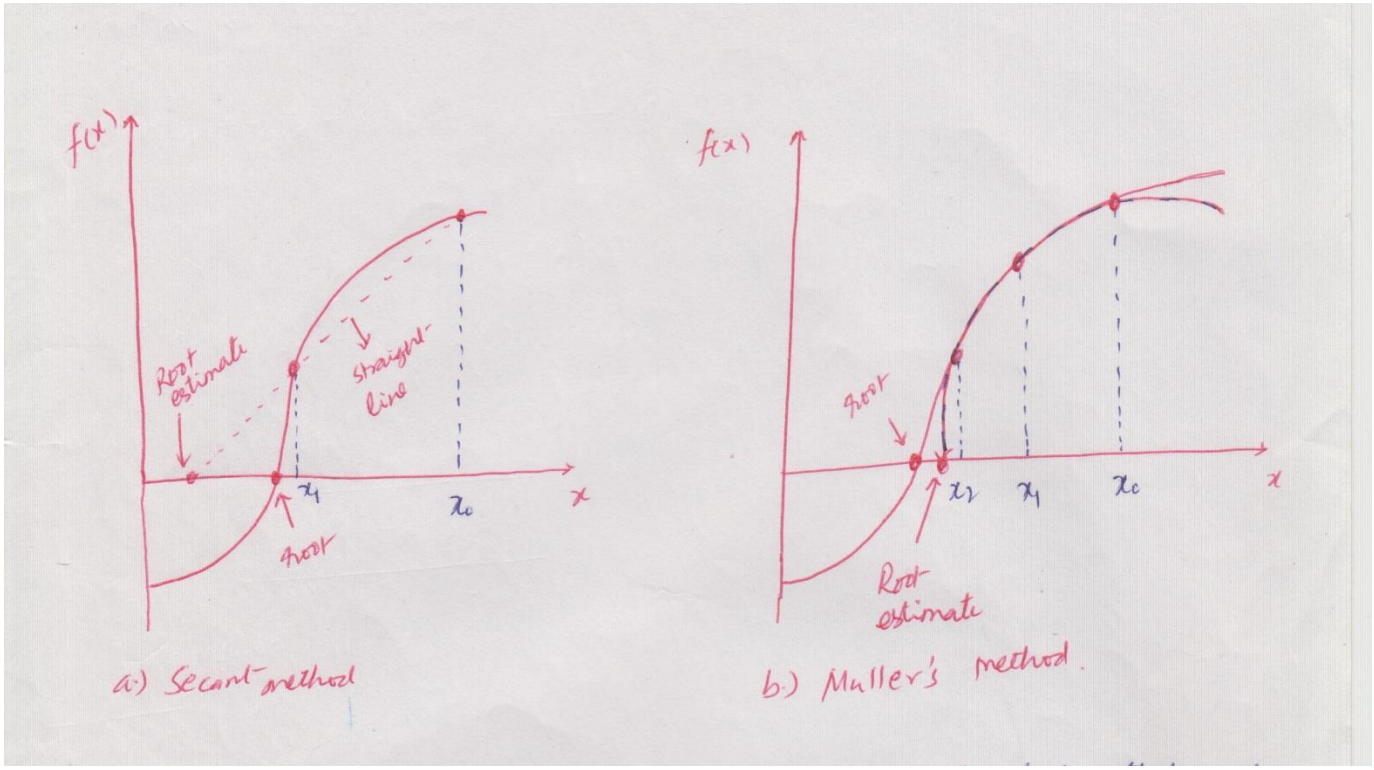
□

To find the real and complex roots of polynomial

- Müller's method ✓
- Bairstow method

Müller's method:

- As secant method finds a root estimate by projecting a line to the x-axis through two function values. Müller's method follow a similar approach, but projets a parabola through three points.
- The method consists of desriving the coefficients of the parabola passing through three points.


f(x)
Root estimate
straight-line
x1
x0
x
root
a) Secant-method
f(x)
root
x2
x1
x0
x
Root estimate
b) Muller's method.

These coefficients can then be substituted into the quadratic formula to obtain the point where the parabola intercepts the x-axis — that is, the root estimate.

$$f_2(x) = a(x-x_2)^2 + b(x-x_2) + c \quad \text{——(3)}$$

We want this parabola to intersect the three points:

$(x_0, f(x_0))$, $(x_1, f(x_1))$ and $(x_2, f(x_2))$

Coefficient of (3) can be obtained as:

$$\left.\begin{aligned} f(x_0) &= a(x_0-x_2)^2 + b(x_0-x_2) + c \\ f(x_1) &= a(x_1-x_2)^2 + b(x_1-x_2) + c \\ f(x_2) &= a(x_2-x_2)^2 + b(x_2-x_2) + c \end{aligned}\right\} \quad \text{——(4)}$$

$$c = f(x_2)$$

and,

$$\left.\begin{aligned} f(x_0) - f(x_2) &= a(x_0-x_2)^2 + b(x_0-x_2) \\ f(x_1) - f(x_2) &= a(x_1-x_2)^2 + b(x_1-x_2) \end{aligned}\right\} \quad \text{——(5)}$$

How to get a and b?

Notations: $h_0 = x_1 - x_0$, $h_1 = x_2 - x_1$

$$\delta_0 = \frac{f(x_1)-f(x_0)}{x_1-x_0}, \quad \delta_1 = \frac{f(x_2)-f(x_1)}{x_2-x_1}$$

∴ From (5):

$$(h_0+h_1)b-(h_0+h_1)^2a = h_0\delta_0+h_1\delta_1$$

$$h_1 b - h_1^2 a = h_1\delta_1$$

Solve for a and b:

$$a=\frac{\delta_1-\delta_0}{h_1+h_0}\,,\quad b=ah_1+\delta_1 \qquad \text{——— (6)}$$

and $c=f(x_2)$

— To find the root, we apply the quadratic formula to eqn. (3), however, because of potential round-off error, we use the alternative formulation to yield:

$$x_3-x_2=\frac{-2c}{b\pm\sqrt{b^2-4ac}} \quad \text{OR} \quad x_3=x_2+\left[\frac{-2c}{b\pm\sqrt{b^2-4ac}}\right] \qquad \text{——— (7)}$$

— Note that the use of quadratic formula means that both real and complex roots can be located. This is the major benefit of the ~~formula~~ method.

- Note that (7) yields two roots corresponding to $\pm$ term in the denominator. In Müller's method, the sign is chosen to agree with the sign of b. This choice will result in the largest denominator, and hence will give the root estimate that is closest to x_2.

- Once x_3 is determined, the process is repeated. This brings up the issue of which point is discarded. Two general strategies are typically used:

1.) If only real roots are being located, we choose the two original points that are nearest the new root estimate x_3.

2.) If both real and complex roots are being evaluated, a sequential approach is employed. That is, just like the secant method x_1, x_2 and x_3 take the place of x_0, x_1 and x_2.

Example:

Perform three iterations using Müller's method to find the smallest +ve root of the equation:

$$f(x) = x^3 - 5x + 1 = 0$$

Sol^n: Note that the smallest +ve root of $f(x)$ lies in $(0,1)$.

Take: $x_0 = 0$, $x_1 = 0.5$, $x_2 = 1.0$ (initial guesses)

$\Rightarrow f(x_0) = 1.0, \quad f(x_1) = -1.375, \quad f(x_2) = -3.0$

$\therefore \quad c = f(x_2) = -3.0, \quad a = \dfrac{\delta_1 - \delta_0}{h_1 + h_0} = 1.5$

$b = ah_1 + \delta_1 = -2.5$

Note that since $\underline{b < 0}$, we get:

$$x_3 = x_2 - \frac{2c}{b - \sqrt{b^2 - 4ac}} = 0.191857$$

For new iterations, choose: $x_0 = 0.5, \; x_1 = 1.0, \; x_2 = 0.1918$

Then

$$c = 0.0477, \quad a = 1.6991, \quad b = -5.1385$$

again, as $\underline{b < 0}$, we get

$$x_4 \Rightarrow \text{as} \quad x_3 = x_2 - \frac{2c}{b - \sqrt{b^2 - 4ac}} = 0.2011$$

again, choose: $x_0 = 1.0, \; x_1 = 0.1918, \; x_2 = 0.2011$

⋮

□

Interpolation

— Consider the problem of approximating a given function by a class of simpler functions (polynomials).

— Main uses of interpolation:

a.) In reconstructing the function $f(x)$ when it is not given explicitly, and only the values and/or its certain order derivatives are given at a set of points.

b.) To replace the $f(x)$ by a interpolating polynomial $p(x)$ so that many common operations such as determination of roots, differentiation and integration etc. which are intended for $f(x)$ may be performed using $p(x)$.

Definition: A polynomial $p(x)$ is called an interpolating polynomial if the values of $p(x)$ and/or its certain order derivatives coincide with those of $f(x)$ and/or its same order derivatives at one or more points.

— In general, if there are $(n+1)$ distinct points:

$$a \leq x_0 \leq x_1 \leq \cdots \leq x_n \leq b$$

Then the problem of interpolation is to obtain $p(x)$ satisfying

the condition:

i) $p(x_i) = f(x_i), \quad i = 0, 1, 2, \cdots, n$ ——— (1)

OR

ii) $p(x_i) = f(x_i)$
$p'(x_i) = f'(x_i), \quad i = 0, 1, \cdots, n$ } ——— (2)

— Condition in Eq. (1) give rise to Lagrange interpolation, and (2) gives Hermite interpolation.

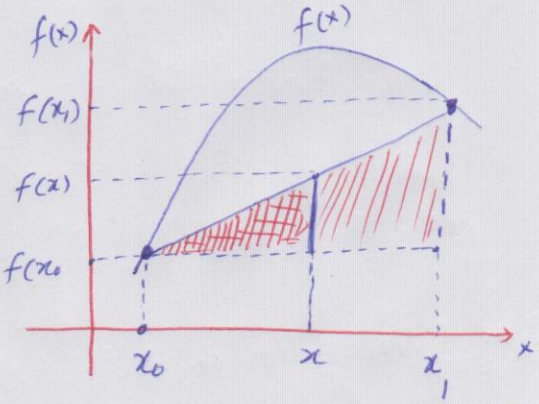
Note: The derivative condition in Eq. (2) may be replaced with more general derivative condition involving higher-order derivatives.

Next, we discuss the interpolations of various degrees.

— Linear interpolation (Lagrange):

— Using property of similar triangles:

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



OR

$$f_1(x) = \left(\frac{x-x_1}{x_0-x_1}\right) f(x_0) + \left(\frac{x-x_0}{x_1-x_0}\right) f(x_1)$$

$$= l_0(x) f(x_0) + l_1(x) f(x_1) \quad \text{——— (3)}$$

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

— functions $l_0(x)$ and $l_1(x)$ are called Lagrange fundamental polynomials. It can be verified that:

$$l_0(x) + l_1(x) = 1$$

$$\begin{cases} l_0(x_0) = 1, & l_0(x_1) = 0 \\ l_1(x_0) = 0, & l_1(x_1) = 1 \end{cases}$$

OR

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

δ_{ij} ↓ Kronecker delta

— Also, Eqn. (3) is a weighted average of $f(x_0)$ and $f(x_1)$. Here, weights ($l_i(x)$) are inversely related to the distance from end points to the unknown point; the closer point has more influence than the farther point. Thus the weights $\left(\frac{x-x_1}{x_0-x_1}\right)$ and $\left(\frac{x-x_0}{x_1-x_0}\right)$ which are normalized distances between the unknown point and each end point.

Truncation error (bounds):

In linear interpolation (Lagrange) case, polynomial $f_1(x)$ coincides with the function $f(x)$ at x_0 and x_1, and it deviates at all other points in interval (x_0, x_1) (as shown in Fig. 1). This deviation will introduce truncation error and denoted as:

$$E_1 = f(x) - f_1(x) \quad \text{—— (4)}$$

Expression for $E_1(f, x)$:

— Note that at $x = x_0$ or $x = x_1 \Rightarrow E_1(f, x) = 0$

$\therefore$ If $x \in (x_0, x_1)$, then define a function $g(t)$ as:

$$g(t) = f(t) - f_1(t) - \left(f(x) - f_1(x)\right) \frac{(t - x_0)(t - x_1)}{(x - x_0)(x - x_1)} \quad \text{—— (5)}$$

Check that: $g(t) = 0$ if $t = x_0$, $t = x_1$, or $t = x$

$\Rightarrow g(t)$ satisfies Rolle's theorem.

Applying Rolle's theorem on the intervals (x_0, t) and (t, x_1) separately, we get:

$$g'(\xi_1) = 0, \quad x_0 < \xi_1 < t \quad \text{and} \quad g'(\xi_2) = 0, \quad t < \xi_2 < x_1$$

— $\Rightarrow g'(t)$ also satisfies the conditions of Rolle's theorem.

∴ Applying Rolle's theorem for $g'(t)$ on interval (ξ_1, ξ_2) we obtain $g''(\xi) = 0$, $\xi_1 < \xi < \xi_2$ OR $x_0 < \xi < x_1$

From (5):

$$g''(t) = f''(t) - \frac{2(f(x) - f_1(x))}{(x-x_0)(x-x_1)} \quad \text{—— (6)}$$

∴ $g''(\xi) = 0$, from (6), we have

$$f(x) = f_1(x) + \frac{1}{2}(x-x_0)(x-x_1)f''(\xi)$$

$$\Rightarrow \boxed{E_1(f,x) = f(x) - f_1(x) = \frac{1}{2}(x-x_0)(x-x_1)f''(\xi)}$$

— If we can determine a bound for $f''(x)$ in $[x_0, x_1]$, that is,

$$|f''(x)| \le M_2, \text{ for all } x \in [x_0, x_1]$$

then $$|E_1(f,x)| = \left|\frac{1}{2}(x-x_0)(x-x_1)f''(\xi)\right|$$

OR $$\boxed{|E_1(f,x)| \le \frac{1}{2} \max_{x_0 \le x \le x_1} |(x-x_0)(x-x_1)|\, M_2} \quad \text{—— (7)}$$

Lagrange Interpolating polynomials (in general):

— In general, Lagrange interpolating polynomials can be represented as:

$$f_n(x) = \sum_{i=0}^{n} l_i(x) f(x_i) \quad \text{— (8)}$$

where $$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

For linear ($n=1$) case:

$$f_1(x) = \left(\frac{x - x_1}{x_0 - x_1}\right) f(x_0) + \left(\frac{x - x_0}{x_1 - x_0}\right) f(x_1) \quad \text{— (*)}$$

and the second order version ($n=2$) is:

$$f_n(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1) + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} f(x_2)$$

— (**)

Also, note that

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

and

$$\sum_{i=0}^{n} l_i(x) = 1$$

□

Interpolation (contd.)

Example: Given that $f(0)=1$, $f(1)=3$, $f(3)=55$ find the unique polynomial of degree 2 (or less) which fits the given data:

x	$f(x)$
0	1
1	3
3	55

∴ We have $x_0=0$, $x_1=1$, $x_2=3$

and $f(x_0)=1$, $f(x_1)=3$, $f(x_2)=55$

∴ Lagrang fundamental polynomials are:

$$l_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} = \frac{(x-1)(x-3)}{(-1)(-3)} = \frac{1}{3}(x^2-4x+3)$$

$$l_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} = \frac{1}{2}(3x-x^2)$$

$$l_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} = \frac{1}{6}(x^2-x)$$

Hence, the Lagrange quadratic interpolating polynomial $f_2(x)$ is:

$$f_2(x) = \sum_{i=0}^{2} l_i(x) f(x_i) = 8x^2-6x+1$$

□

Exercise: Consider the following values of the function $f(x) = \sin x + \cos x$ as:

x	$f(x)$
$\pi/18$	1.1585
$\pi/9$	1.2817
$\pi/6$	1.3660

Construct the quadratic interpolating polynomial $f_2(x)$ that fits the data. Hence, find $f(\pi/2)$. compare with the exact value.

Error in quadratic interpolation (Lagrange):

$$E_2(f,x) = f(x) - f_2(x)$$

To get the truncation error bound, take

$$g(t) = f(t) - f_2(t) - \left(f(x) - f_2(t)\right) \frac{(t-x_0)(t-x_1)(t-x_2)}{(x-x_0)(x-x_1)(x-x_2)}$$

— Apply Rolle's theorem repeatedly for $g(t)$, $g'(t)$ and $g''(t)$.

$\Rightarrow g'''(\xi_1) = 0$, $\quad x_0 < \xi_1 < x_2$

$\Rightarrow$ $$\boxed{E_2(f,x) = \frac{1}{3!}(x-x_0)(x-x_1)(x-x_2) f'''(\xi_1)} \quad \text{——} (1)$$

— Now if we can find +ve real M_3 such that

$$M_3 = \max_{x_0 \le x \le x_2} |f'''(x)|$$

then error bound is:

$$|E_2| = |f(x) - f_2(x)| \le \frac{1}{6} M_3 \left(\max_{x_0 \le x \le x_2} |(x-x_0)(x-x_1)(x-x_2)| \right)$$

□

— Similarly, in general, for higher order interpolating polynomial, say of degree n:

$$E_n(f,x) = \frac{1}{(n+1)!}(x-x_0)(x-x_1)\cdots(x-x_n) f^{(n+1)}(\xi)$$

with $x_0 < \xi < x_n$

——— (3)

and

$$|E_n| \le \frac{1}{(n+1)!} M_n \left(\max_{x_0 \le x \le x_n} |(x-x_0)(x-x_1)\cdots(x-x_n)| \right)$$

Where

$$M_n = \max_{x_0 \le x \le x_n} |f^{(n+1)}(x)|$$

□

— Let's find the truncation error bound for Example 1:

$$|E_2| \le \frac{1}{6} M_2 \left(\max_{0 \le x \le 3} |(x-0)(x-1)(x-3)| \right)$$

$$= \frac{1}{6}(2.1126) M_2$$

□

Find max. of $|x(x-1)(x-3)|$.
Check that it occurs at
$x = 2.1126$

— There are two alternatives that are well suited for computer implementation

i) Lagrange form

ii) Newton' form (Newton's divided difference form)

— Let's derive the quadratic interpolating polynomial $f_2(x)$ in Newton's form:

If there are three data points are ~~avail~~ available, we can use a second-order polynomial (or parabo) given as:

$$f_2(x) = a_0 + a_1 (x - x_0) + a_2 (x - x_0)(x - x_1) \quad \text{— (4)}$$

at $x = x_0$: $a_0 = f(x_0)$ — (5)

Using a_0 in (4), at $x = x_1$:

$$a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \quad \text{— (6)}$$

Finally, using ⑤ and ⑥ in ④ at $x = x_2$ gives:

$$a_2 = \frac{\dfrac{f(x_2) - f(x_1)}{x_2 - x_1} - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0} \quad \text{——⑦}$$

Values of coefficients obtained in Eqs. ⑤, ⑥, ⑦ can be substituted in ④ to get $f_2(x)$.

– General form of Newton interpolating polynomials:

– To fit an n^{th}-order polynomial to $(n+1)$ data points, the n^{th}-order polynomial can be written as:

$$f_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}) \quad \text{——⑧}$$

Here also data points can be used to evaluate the coefficients $a_0, a_1, \cdots, a_n$, as:

$$\left.\begin{aligned} a_0 &= f(x_0) \\ a_1 &= f[x_1, x_0] \\ a_2 &= f[x_2, x_1, x_0] \\ &\vdots \\ a_n &= f[x_n, x_{n-1}, \cdots, x_1, x_0] \end{aligned}\right\} \quad \text{——⑨}$$

Where, [ ] denotes the finite divided differences.
For example, the first finite divided difference is:

$$f[x_i, x_j] = \frac{f(x_i) - f(x_j)}{x_i - x_j}$$

The second divided difference, which is the difference of two first divided differences, is expressed as:

$$f[x_i, x_j, x_k] = \frac{f[x_i, x_j] - f[x_j, x_k]}{x_i - x_k}$$

Similarly, the n^{th} finite divided difference is:

$$f[x_n, x_{n-1}, \cdots, x_1, x_0] = \frac{f[x_n, x_{n-1}, \cdots x_1] - f[x_{n-1}, x_{n-2}, \cdots, x_0]}{x_n - x_0}$$

i	x_i	$f(x_i)$	First	Second	Third
0	x_0	$f(x_0)$	$f[x_1, x_0]$		
1	x_1	$f(x_1)$	$f[x_2, x_1]$	$f[x_2, x_1, x_0]$	
2	x_2	$f(x_2)$	$f[x_3, x_2]$	$f[x_3, x_2, x_1]$	$f[x_3, x_2, x_1, x_0]$
3	x_3	$f(x_3)$			

Table 2: Recursive nature of finite-divided difference.

∴ Using (9) in terms of divided difference:

$$f_n(x) = f(x_0) + (x-x_0) f[x_1, x_0] + (x-x_0)(x-x_1) f[x_2, x_1, x_0] + \cdots + (x-x_0)(x-x_1)\cdots(x-x_{n-1}) f[x_n, x_{n-1}, \cdots, x_1, x_0] \quad \text{—— (10)}$$

Example 2:

x_i	$f(x_i)$
2·5	14
3·2	15
2·0	8

Use Newton's divided difference to calculate $f_2(x)$. Also find $f_2(2{\cdot}8)$.

i	x_i	$f(x_i)$	$f[x_{i+1}, x_i]$	$f[x_{i+2}, x_{i+1}, x_i]$
0	2·5	14	1·42857	
1	3·2	15		−8·80952
2	2·0	8	5·83333	

∴ Second order interpolation can be implemented as:

$$f_2(x) = f(x_0) + (x-x_0) f[x_1, x_0] + (x-x_0)(x-x_1) f[x_2, x_1, x_0]$$

at $x = 2{\cdot}8$:

$$f_2(2{\cdot}8) = 15.48571$$

□

Errors of Newton's Interpolating Polynomials

— As you have already seen that n^{th}-order interpolating polynomial (in Newton' form) is:

$$f_n(x) = f(x_0) + (x-x_0)f[x_1, x_0] + (x-x_0)(x-x_1)f[x_2, x_1, x_0] + \cdots + (x-x_0)(x-x_1)\cdots(x-x_{n-1})f[x_n, x_{n-1}, \cdots, x_0] \quad \text{——①}$$

Newton's divided difference interpolating polynomial.

— Note that Eq.① is similar to the Taylor series expansion, and the terms are finite divide differences.

— Similar to Taylor series expansion, here also the truncation part (error) could be expressed as:

$$R_n = f[x, x_n, x_{n-1}, \cdots, x_1, x_0](x-x_0)(x-x_1)\cdots(x-x_n) \quad \text{——②}$$

where $f[x, x_n, x_{n-1}, \cdots, x_0]$ is the $(n+1)^{th}$ finite divided difference.

— As finite divided difference contains the unknow $f(x)$, therefore it can not be solved for the error.

— However, if an additional data point $f(x_{n+1})$ is available, then

$$R_n \cong f[x_{n+1}, x_n, \cdots, x_0](x-x_0)(x-x_1)\cdots(x-x_n) \quad \text{——③}$$

□

Example 1: (Quadratic interpolation)

$x_0 = 1 \quad f(x_0) = 0$

$x_1 = 4 \quad f(x_1) = 1.386294$

$x_2 = 6 \quad f(x_2) = 1.791759$

Check that:

$$f_2(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1)$$

Where

$$a_0 = f(x_0) = 0$$

$$a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = f[x_1, x_0] = 0.4620981$$

$$a_2 = \frac{\left(\frac{f(x_2) - f(x_1)}{x_2 - x_1}\right) - \left(\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right)}{x_2 - x_0} = f[x_2, x_1, x_0] = -0.05187$$

$$\therefore f_2(x) = 0 + 0.4620981(x-1) - 0.0518731(x-1)(x-4)$$

Hence:
$f_2(2) = 0.5658444$

Example 2: Using Eq. (3) estimate the error for the quadratic interpolating polynomial obtained in Example-1.

— If ~~value~~ function $f(x)$ is not known to us, then we can estimate the error using (3) as (provided we know the additional value at x_3):

$$R_2 = f[x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2)$$

OR

$$R_2 = 0.007865(x-1)(x-4)(x-6)$$

Note that at $x=2$:

$$R_2 = 0.0629242$$

□

— Now let's do an example to get the idea about the significance of the position and sequence of the given data.

— This is discussed in Example 3:

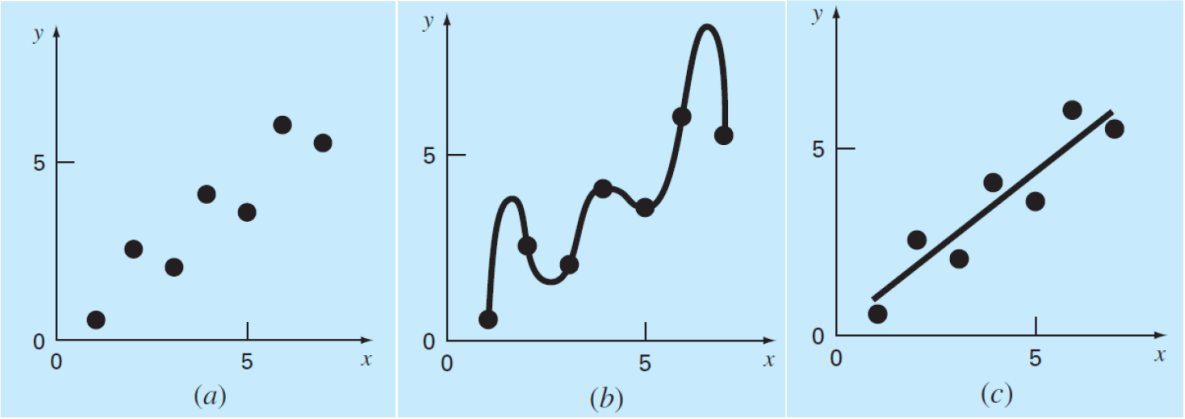


Fig. 1. Diffrence between interpolation and regression: (a) data points; (b) polynomial fit (iterpolation); (c) using least-squares (regression) fit.

Remaks:

Figure 1. shows that when data is highly scatterd, polynomial interpolation is inappropriate. A more appropraite technique for such cases is to derive an approximating function that fits the shape or general trend of the data (without matching its values at given data points)

Example-1: See the following data set for $f(x) = \ln x$ to evaluate $f(x)$ at $x = 2$

x	f(x) = ln x
1	0
4	1.3862944
6	1.7917595
5	1.6094379
3	1.0986123
1.5	0.4054641
2.5	0.9162907
3.5	1.2527630

Order	F(x)	Error
0	0.000000	0.462098
1	0.462098	0.103746
2	0.565844	0.062924
3	0.628769	0.046953
4	0.675722	0.021792
5	0.697514	-0.003616
6	0.693898	-0.000459
7	0.693439	

Fig. 2 Tabulated data (left side) and different orders interpolation. Note that true value is $\ln 2 = 0.6931472$

- Note from Fig. 2 that error reduces as the order increases.

- It can be concluded that the fifth order version yields a good estimate and that higher-order terms do not significantly enhance the prediction.

Inverse interpolation:

- Interpolation: finding some value $f(x)$ for some x that is between given data points.

- Inverse interpolation: Find the x for which $f(x)$ is a certain value.

For example: $f(x) = 1/x$ (Example 3.)

x	1	2	3	4	5	6	7
$f(x)$	1	0.5	0.3333	0.25	0.2	0.1667	0.1429

$f(x)$	0.1429	0.1667	0.2	0.25	0.3333	0.5	1
x	7	6	5	4	3	2	1

- Now suppose you are asked to determine the value of x that corresponds to $f(x) = 0.3$. Such a problem is known as inverse interpolation.

— Unfortunately, when you reverse the variables, then in general the values along the new abscissa will not be evenly spaced.

— Such non-uniform spacing on the abscissa often leads to oscillations in the resulting interpolating polynomial. This can occur even for lower order polynomials.

— Therefore, rather than finding an interpolation of x as a function of $f(x)$, it may be useful to find an equation for $f(x)$ as a function of x using interpolation and then solve the corresponding root problem:

— That is interpolation problem reduces to a root problem.

For the problem mentioned in Example 3:

— Fit a quadratic polynomial to the three points:

$(2, 0.5)$, $(3, 0.3333)$ and $(4, 0.25)$

$$\Rightarrow f_2(x) = 1.08333 - 0.375x + 0.041667x^2$$

Now the answer to the inverse interpolation of finding the x corresponding to $f(x) = 0.3$ would therefore involve determining the root of

$$0.3 = 1.08333 - 0.375x + 0.041667x^2$$

Using quadratic formula (for roots):

$$x = \frac{0.375 \pm \sqrt{(-0.375)^2 - 4(0.041667)\,0.78333}}{2(0.041667)} = \begin{matrix} 5.704158 \\ 3.295842 \end{matrix}$$

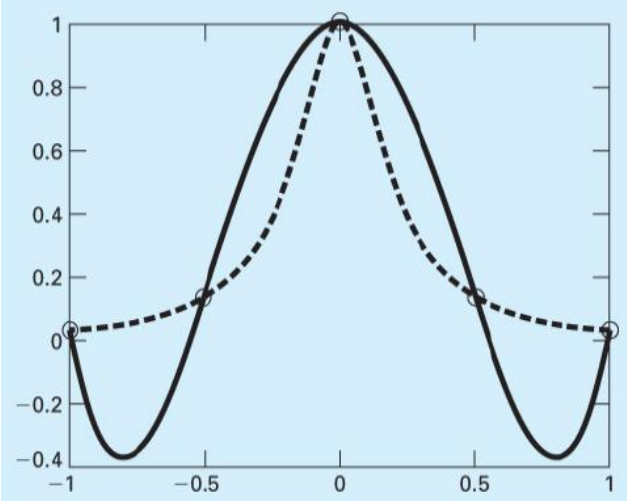
Thus, the second root, 3.296, is a good approximation of the true value of 3.3333. □

Oscillations (Interpolation)

Higher-order polynomials can not only lead to round-off errors due to ill-conditioning, but can also introduce oscillations to an interpolation problem. Consider the following function known as Runge's function:

$$f(x)=\frac{1}{1+25x^2}$$

In the figures below, the dashed line represents Runge's function, the circles represent data points of the function, and the solid line represents the results of a polynomial interpolation:



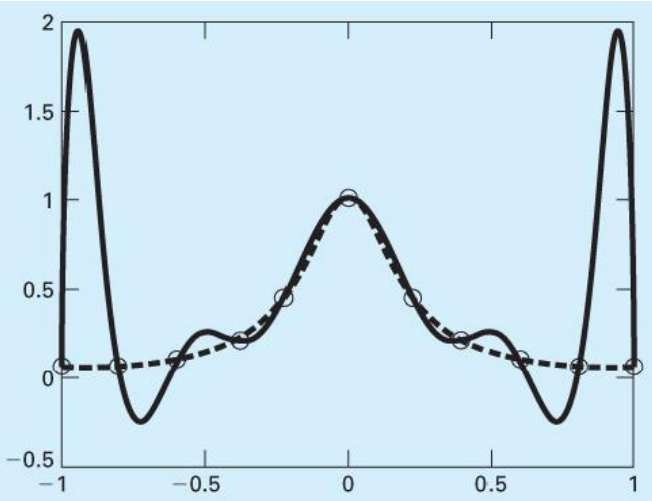


Figure 1:

a. Interepolation using fourth degree polynomial with 5 data points.

b. Interpolation using tenth degree polynomial with 11 data points

Spline Interpolation:

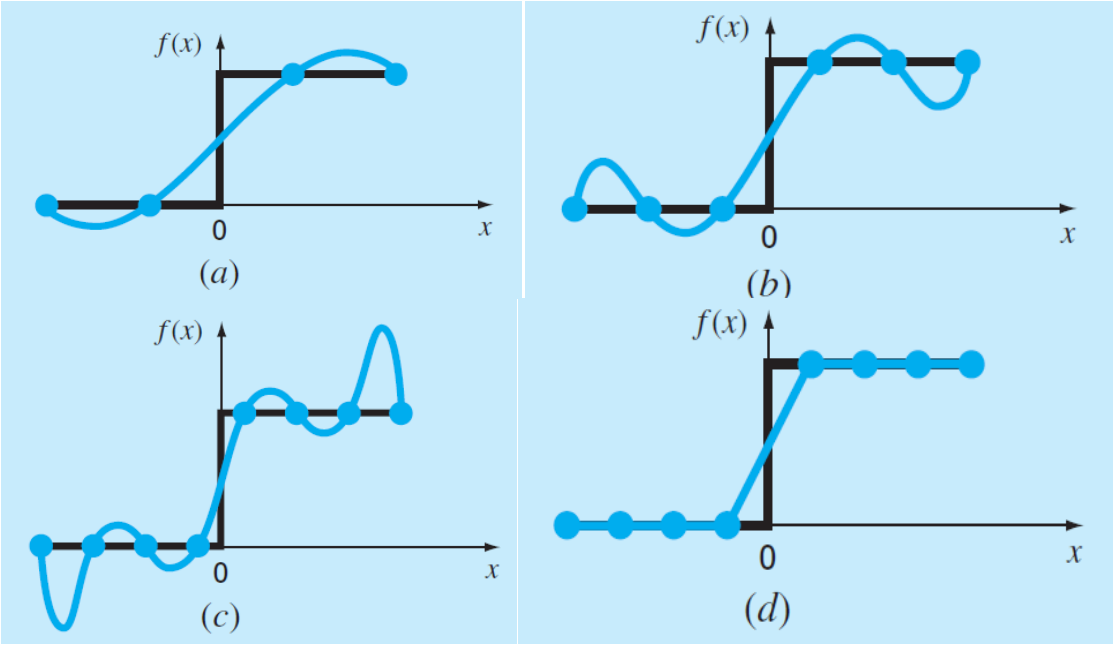


Figure 2: Rpresentation of a situation where splines are superior to higher-order interpolating polynomials. The function to be fit undergoes an abrupt increase at $x = 0$. Parts (a) through (c) indicate that the abrupt change induces oscillations in interpolating polynomials. In contrast, a linear spline (d) provides a much more acceptable approximation.

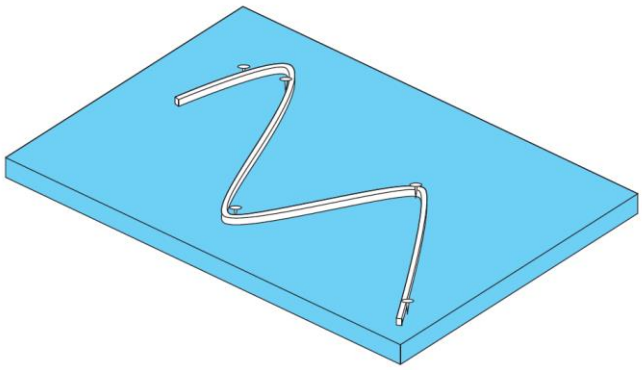


Figure 3: The concept of spline originated from the drafting technique of using a thin flexible strip (spline) to draw smooth curves through a set of points.

Spline Interpolation

— An alternative approach — to apply lower-order polynomials to subset of data points. Such connecting polynomials are called Spline functions.

Linear splines

— The simplest connection between two (data) points is a straight line

— The first-order splines for a collection of ordered data points can be defined as a set of linear functions:

$$\left.\begin{aligned} f(x) &= f(x_0) + m_0(x - x_0), && x_0 \le x \le x_1 \\ f(x) &= f(x_1) + m_1(x - x_1), && x_1 \le x \le x_2 \\ &\vdots \\ f(x) &= f(x_{n-1}) + m_{n-1}(x - x_{n-1}), && x_{n-1} \le x \le x_n \end{aligned}\right\} \quad \text{——— (1)}$$

where m_i $(i = 0, 1, \cdots, n-1)$ is the slope of straight line joining the points:

$$m_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \qquad \text{——— (2)}$$

— These equations can be used to evaluate the function at any point between x_0 and x_n by first locating the interval within which the point lies.

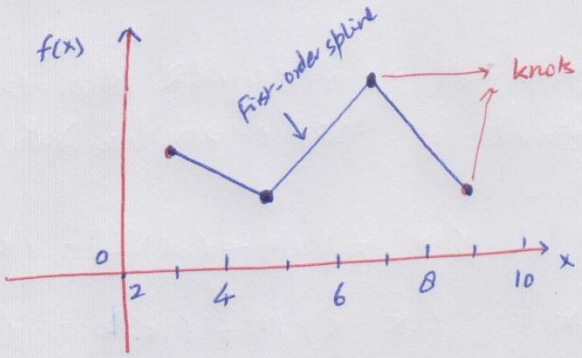
Example:

x	$f(x)$
3.0	2.5
4.5	1.0
7.0	2.5
9.0	0.5

Fit the data using first-order splines. Evaluate the function at $x=5$.

Here, 5 lies between $x=4.5$ and $x=7.0$

Therefore, $$m = \frac{2.5-1}{7-4.5} = 0.60$$



Knots: data points where two splines meet

$$f(x) = f(4.5) + 0.60(x-4.5)$$

at $x=5$, $f(5) = 1.3$ □

— Quadratic splines:

— To ensure that n^{th} derivatives are continuous at the knots, a spline of atleast $(n+1)$ order must be used.

— Third-orde polynomials (cubic splines) ensures continuous first and second derivatives — Frequently used for engineering applications.

— Quadratic splines — having continuous first derivatives at the knots.

— Derive a second-order polynomial for each interval between data points. Therefore, in general:

$$f(x_i) = a_i x^2 + b_i x + c_i \qquad (3)$$

For $(n+1)$ data points, there are n intervals and $3n$ unknowns.

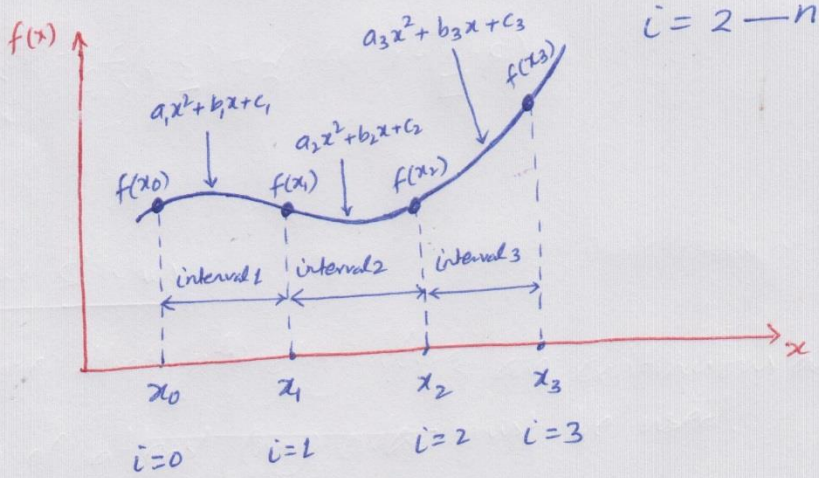
— To evaluate $3n$ unknowns, $3n$ conditions are required:

a.) The function values of adjacent polynomials must be equal at the interior knots:

$$a_{i-1} x_{i-1}^2 + b_{i-1} x_{i-1} + c_{i-1} = f(x_{i-1}) \qquad (4)$$

$$a_i x_{i-1}^2 + b_i x_{i-1} + c_i = f(x_{i-1}) \qquad (5)$$

$i = 2 - n.$



— Eqs. (4) and (5) each provide $n-1$ conditions for a total of $2n-2$ conditions.

b.) The first and last functions must pass through the end points:

$$a_1 x_0^2 + b_1 x_0 + c_1 = f(x_0) \quad \text{—— (6)}$$

$$a_n x_n^2 + b_n x_n + c_n = f(x_n) \quad \text{—— (7)}$$

—— 2 conditions

Total $2n$ conditions.

c.) The first derivatives at the interior knots must be equal.

— The first derivative of (3) is

$$f'(x) = 2ax + b$$

∴ the conditions are:

$$2a_{i-1} x_{i-1} + b_{i-1} = 2a_i x_{i-1} + b_i \quad \text{—— (8)}$$

$i = 2 - n$

Eq. (8) provides another $n-1$ conditions for a total of

$2n + n - 1 = 3n - 1$ conditions.

d.) Assume that the second derivative is zero at the first point

$$\Rightarrow a_1 = 0 \quad \text{—— (9)}$$

That is the first two points will be connected by a straight line.

□

Example 2: Fit quadratic splines for the data given in Example-1. Also use the result to estimate the value at $x = 5$.

— We are given with 4 data points and 3 intervals.

$\therefore 3n = 9$ unknowns must be determined.

Eqs. (4) and (5) yields 4 conditions:

$$20.25\,a_1 + 4.5\,b_1 + c_1 = 1.0$$
$$20.25\,a_2 + 4.5\,b_2 + c_2 = 1.0$$
$$49\,a_2 + 7\,b_2 + c_2 = 2.5$$
$$49\,a_3 + 7\,b_3 + c_3 = 2.5$$

Eqs. (6) and (7) gives:

$$9\,a_1 + 3\,b_1 + c_1 = 2.5$$
$$81\,a_3 + 9\,b_3 + c_3 = 2.5$$

Continuity of derivatives creates: (Eq. (8)):

$$9\,a_1 + b_1 = 9\,a_2 + b_2$$
$$14\,a_2 + b_2 = 14\,a_3 + b_3$$

Finally, Eq. (9) gives: $a_1 = 0$. So, we have to solve (only) eight simultaneous equations.

— In matrix form:

$$\begin{bmatrix} 4.5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 20.25 & 4.5 & 1 & 0 & 0 & 0 \\ 0 & 0 & 49 & 7 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 49 & 7 & 1 \\ 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 81 & 9 & 1 \\ 1 & 0 & -9 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 14 & 1 & 0 & -14 & -1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \\ a_3 \\ b_3 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2.5 \\ 2.5 \\ 2.5 \\ 0.5 \\ 0 \\ 0 \end{bmatrix}$$

Solving this:

$a_1 = 0$ $\quad b_1 = -1$ $\quad c_1 = 5.5$

$a_2 = 0.64$ $\quad b_2 = -6.76$ $\quad c_2 = 18.46$

$a_3 = -1.6$ $\quad b_3 = 24.6$ $\quad c_3 = -91.3$

Substituting these ~~too~~ into original quadratic equations to develop the following relationships for each interval:

$$f_1(x) = -x + 5.5, \quad 3.0 \le x \le 4.5$$

$$f_2(x) = 0.64x^2 - 6.76x + 10.46, \quad 4.5 \le x \le 7.0$$

$$f_3(x) = -1.6x^2 + 24.6x - 91.3, \quad 7.0 \le x \le 9.0$$

∴ Use f_2 to get the estimated value at $x = 5$, which is:

$$f_2(5) = 0.66.$$

□

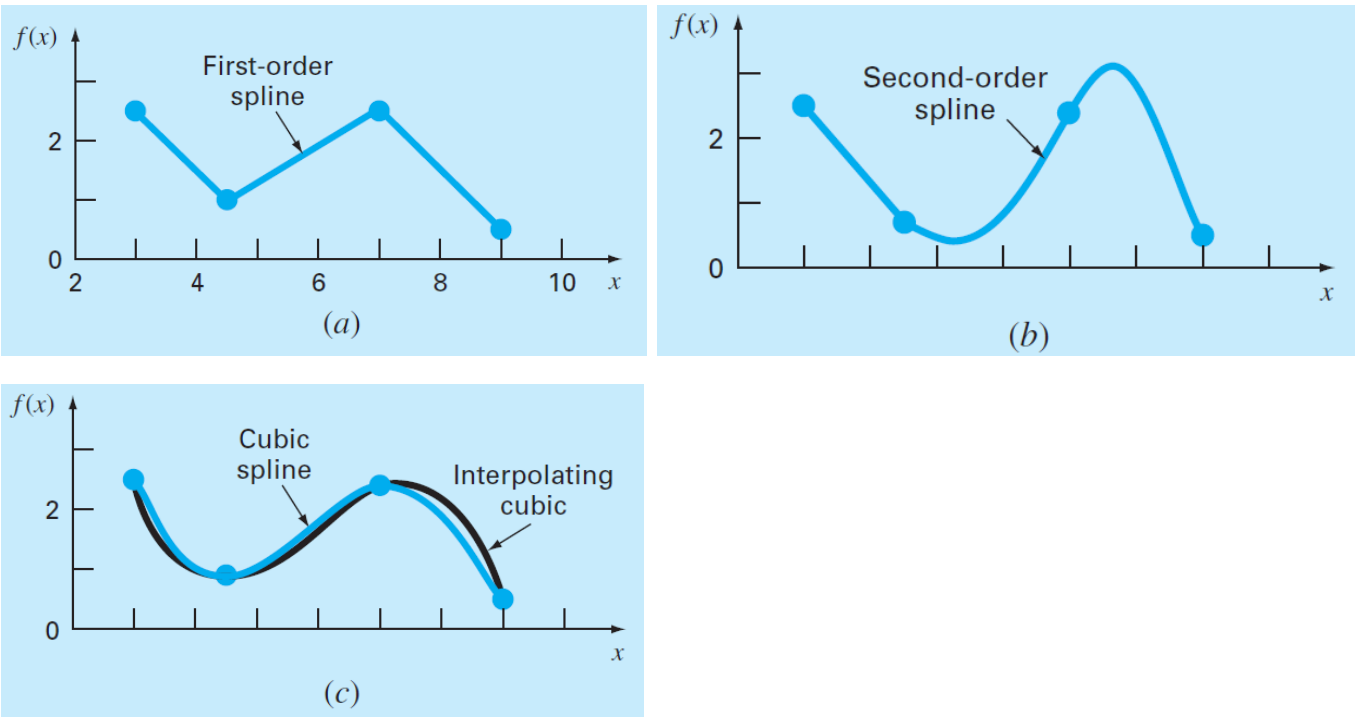


Figure 4: Spline fits of a set of four points (as discussed in Examples 1-3): (a) linear splines; (b) quadratic spline; (c) cubic spline (also with cubic interpolating polunomial)

Cubic splines:

— Desrive a third-order polynomial for each interval between knots as:

$$f_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \quad \text{——} (10)$$

∴ for $(n+1)$ data points with n intervals, there are $4n$ unknowns $\Rightarrow$ $4n$ conditions are required.

1. The function values must be equal at the interior knots ($2n-2$ conditions)

2. The first and last function must pass through the end points (2 conditions)

3. The first derivatives at the interior knots must be equal ($n-1$ conditions)

4. The second derivatives at the interior knots must be equal ($n-1$ conditions)

5. The second derivative at the end knots are zero (2 conditions)

Example 3: Fit cubic splines to the same data used in Examples 1 and 2. Estimate value at $x=5$.

Derivation of Cubic Splines

— As each pair of knots is connected by a cubic polynomial
⇒ second derivative within each interval is a straight line.
∴ second derivative can be represented by a first-order (linear) Lagrange interpolating polynomial as:

$$f_i''(x) = f_i''(x_{i-1})\left(\frac{x - x_i}{x_{i-1} - x_i}\right) + f''(x_i)\left(\frac{x - x_{i-1}}{x_i - x_{i-1}}\right) \quad \text{— (11)}$$

Where $f_i''(x)$ — value of the second-order derivative at any x within the i^{th} interval.

— Eq. (11) is a straight line connecting the second derivative at the first knot $f''(x_{i-1})$ with the second derivative at the second knot $f''(x_i)$.

— Integrate Eq. (11) twice to yield an expression for $f_i(x)$. Note that the expression will have <u>two</u> constants of integration and to evaluate them use:

$$f(x) = f(x_{i-1}) \quad \text{at} \quad x = x_{i-1}$$

$$\text{and} \quad f(x) = f(x_i) \quad \text{at} \quad x = x_i$$

$$\Rightarrow f_i(x) = \frac{f_i''(x_{i-1})}{6(x_i - x_{i-1})}(x_i - x)^3 + \frac{f_i''(x_i)}{6(x_i - x_{i-1})}(x - x_{i-1})^3 +$$

PTO.

$$\left[\frac{f(x_{i-1})}{(x_i - x_{i-1})} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6}\right](x_i - x)$$

$$+\left[\frac{f(x_i)}{(x_i - x_{i-1})} - \frac{f''(x_i)(x_i - x_{i-1})}{6}\right](x - x_{i-1}) \qquad \text{——— (12)}$$

The second derivatives appearing in Eq. (12) can be evaluated by using the condition that the first derivatives at the knots must be continuous:

$$f_i'(x_i) = f_{i+1}'(x_i) \qquad \text{——— (13)}$$

Now differentiate Eq. (12) and use condition (13) for both the $(i-1)^{th}$ and i^{th} intervals:

$$\Rightarrow (x_i - x_{i-1}) f''(x_{i-1}) + 2(x_{i+1} - x_{i-1}) f''(x_i) + (x_{i+1} - x_i) f''(x_{i+1})$$

$$= \frac{6}{(x_{i+1} - x_i)}\left[f(x_{i+1}) - f(x_i)\right] + \frac{6}{(x_i - x_{i-1})}\left[f(x_{i-1}) - f(x_i)\right] \qquad \text{——— (14)}$$

- Note that if Eq. (14) is written for all interior knots $(n-1)$, then $(n-1)$ simultaneous equations result with $(n+1)$ unknown second derivatives.
- Since, for (natural) cubic splines, the second derivative at the end knots are zero.

∴ Problem reduces to $(n-1)$ equations in $(n-1)$ unknowns. □

In summary for cubic splines we have the following:

– Cubic equation for each interval:

$$f_i(x) = \frac{f_i''(x_{i-1})}{6(x_i - x_{i-1})}(x_i - x)^3 + \frac{f_i''(x_i)}{6(x_i - x_{i-1})}(x - x_{i-1})^3 + \left[\frac{f(x_{i-1})}{(x_i - x_{i-1})} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6}\right](x_i - x) + \left[\frac{f(x_i)}{(x_i - x_{i-1})} - \frac{f''(x_i)(x_i - x_{i-1})}{6}\right](x - x_{i-1}) \quad \text{——— (A)}$$

– Two unknowns – $f''(x_{i-1})$ and $f''(x_i)$ – at the end of each interval – are evaluated using:

$$(x_i - x_{i-1})f''(x_{i-1}) + 2(x_{i+1} - x_{i-1})f''(x_i) + (x_{i+1} - x_i)f''(x_{i+1}) = \frac{6}{(x_{i+1} - x_i)}[f(x_{i+1}) - f(x_i)] + \frac{6}{(x_i - x_{i-1})}[f(x_{i-1}) - f(x_i)] \quad \text{——— (B)}$$

Remark: Equation (B) generates $(n-1)$ equations in $(n-1)$ unknowns for interior knots (as second derivatives at the end knots are zero)

Example 3: Fit cubic spline to the same data (as in Example 1). Estimate the value at $x = 5$.

x	$f(x)$
3·0	2·5
4·5	1·0
7·0	2·5
9·0	0·5

First Step: Employ (B) to get the set of simultaneous equations to determine the second derivatives at the knots:

For the first interior knot:

$$x_0 = 3 \qquad f(x_0) = 2{\cdot}5$$
$$x_1 = 4{\cdot}5 \qquad f(x_1) = 1{\cdot}0$$
$$x_2 = 7 \qquad f(x_2) = 2{\cdot}5$$

Using (B):

$$(4{\cdot}5 - 3)f''(3) + 2(7-3)f''(4{\cdot}5) + (7 - 4{\cdot}5)f''(7)$$
$$= \frac{6}{(7-4{\cdot}5)}(2{\cdot}5-1) + \frac{6}{(4{\cdot}5-3)}(2{\cdot}5-1)$$

Because of the second derivative is zero at the end knot:

$$\Rightarrow f''(3) = 0$$

$$8f''(4{\cdot}5) + 2{\cdot}5f''(7) = 9{\cdot}6 \qquad \text{—— (i)}$$

Similarly, (B) can be applied to the second interior point (knot) (which is x_2)

$$\Rightarrow \quad 2{\cdot}5f''(4{\cdot}5) + 9f''(7) = -9{\cdot}6 \qquad \text{—— (ii)}$$

Solving (i) and (ii):

$$\left.\begin{aligned} f''(4.5) &= 1.67909 \\ f''(7) &= -1.53308 \end{aligned}\right]$$

These can be substituted in (A):

$$f_1(x) = \frac{1.67909}{6(4.5-3)}(x-3)^3 + \frac{2.5}{(4.5-3)}(4.5-x) + \left[\frac{1}{(4.5-3)} - \frac{1.67909(4.5-3)}{6}\right](x-3)$$

OR

$$f_1(x) = 0.186566(x-3)^3 + 1.666667(4.5-x) + 0.246894(x-3)$$

— which is the cubic spline for the first interval.

Similarly, we can get equations for second and third intervals:

$$f_2(x) = 0.111939(7-x)^3 - 0.102205(x-4.5)^3 - 0.29962(7-x)$$

and

$$f_3(x) = -0.127757(9-x)^3 + 1.761027(9-x) + 0.25(x-7)$$

∴ the value at $x=5$ (which falls within the second interval) is:

$$f_2(5) = 1.102886$$

□

Numerical Integration

— Find

$$I = \int_a^b f(x)\,dx \cong \int_a^b f_n(x)\,dx \qquad \text{——— (1)}$$

where $f_n(x)$ — a polynomial approximation of $f(x)$

$$f_n(x) = a_0 + a_1 x + \cdots + a_n x^n$$

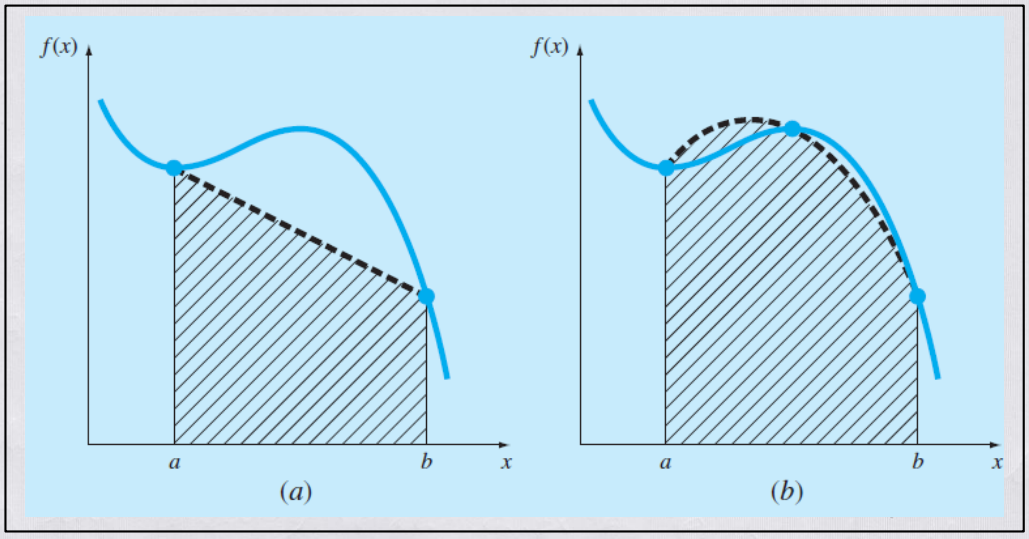


Fig. 1: a.) a first-order polynomial (a straight line) is used as an approximation of $f(x)$.

b.) a parabola is used for the approximation of given function $f(x)$.

Newton-Cotes integration formulas:

- Based on the strategy of replacing a complicated function or tabulated data with an approximating function.

- i) Closed form Newton-Cotes formulas.
  ii) Open form Newton-Cotes formulas.

- The closed form are those where the data points at the begining and end of the limits of integration are known.

- The open forms have integration limits that extend beyond the range of the data. (Generally, not preferred for definite integrals).

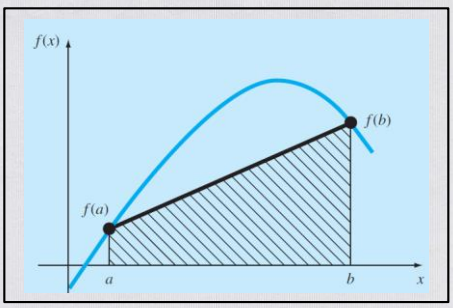
Closed form Newton-Cotes formulas:

A) Trapezoidal Rule:

This corresponds to the case where the polynomial in Eq. (1) is of first-order (linear).

That is:

$$I = \int_a^b f(x)\,dx \cong \int_a^b f_1(x)\,dx \qquad \text{——} (2)$$



– Straight line joining (two) data points:

$$f_1(x) = f(a) + \frac{f(b)-f(a)}{(b-a)}(x-a) \quad \text{——} (3)$$

– Area under this straight line is an estimate of the $\int_a^b f(x)\,dx$.

$$\Rightarrow I \cong \int_a^b \left[f(a) + \frac{f(b)-f(a)}{(b-a)}(x-a) \right] dx$$

OR

$$I \cong (b-a)\left[\frac{f(a)+f(b)}{2}\right] \quad \text{——} (4)$$

Remark:

1. Geometrically, from fig. 2:

$$\left.\begin{aligned} I &\cong \text{width} * \text{average height} \\ &= (b-a) * \left(\frac{f(b)+f(a)}{2}\right) \end{aligned}\right\} \text{——} (5)$$

2. All the Newton-Cotes formulas can be expressed in the general format as given in Eq. (5). In fact they differ only with respect to the formulation of average height.

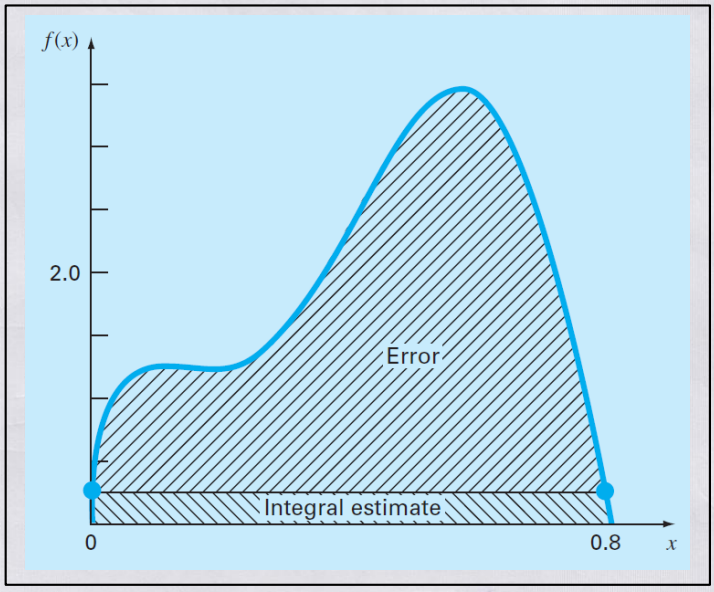
Error of the Trapezoidal Rule:

- Since we are using linear approximation for the function, therefore, the error will be substantial.

- For example, choose a function:

$$f(x) = 0.2 + 25x - 200x^2 + 675x^3 - 900x^4 + 400x^5$$

with $a = 0$ and $b = 0.8$



- An estimate for the local truncation error for a single application of the trapezoidal rule is:

$$E_t = -\frac{1}{12} f''(\xi)(b-a)^3 \quad \text{——} (6)$$

where ξ lies between a and b.

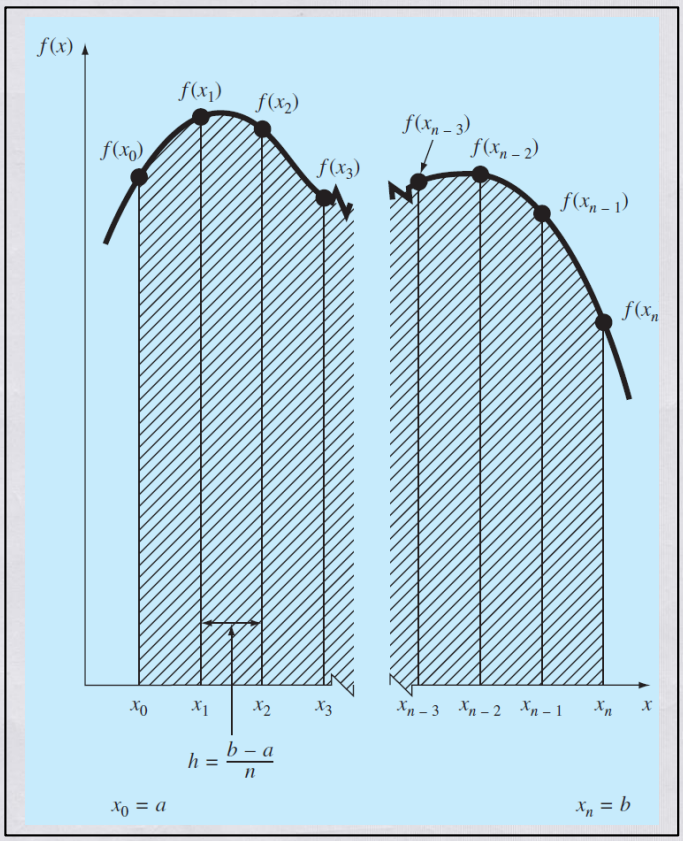
- Equation (6) indicates that trapezoidal rule is exact if $f(x)$ is linear.

The multiple application of Trapezoidal Rule

— To improve the accuracy of the trapezoidal rule — divide the integration interval from a to b into a number of segments and apply method to each segment:

— If there are (n+1) equally-spaced base point: $x_0, x_1, \cdots, x_n$ then there are n-segments of equal width:

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



— The total integral:

$$I = \int_{x_0}^{x_1} f(x)\,dx + \int_{x_1}^{x_2} f(x)\,dx + \cdots + \int_{x_{n-1}}^{x_n} f(x)\,dx$$

— Using trapezoidal rule in each segment:

$$I \cong h\left(\frac{f(x_0)+f(x_1)}{2}\right) + h\left(\frac{f(x_1)+f(x_2)}{2}\right) + \cdots + h\left(\frac{f(x_{n-1})+f(x_n)}{2}\right)$$

OR $$= \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right] \qquad \text{——— (7)}$$

$$= (b-a) * \left(\frac{f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)}{2n}\right)$$

(b−a): width; the fraction: average height

— Error expression for the multiple application of trapezoidal rule:

$$E_t = -\frac{(b-a)^3}{12n^3}\sum_{i=1}^{n} f''(\xi_i) \qquad \text{——— (8)}$$

Where ξ_i is located in the segment i. Equation (8) can be simplified by taking the mean value of the second-derivative as:

$$\bar{f}'' \cong \frac{\sum_{i=1}^{n} f''(\xi_i)}{n} \Rightarrow \sum_{i=1}^{n} f''(\xi_i) \cong n\bar{f}''$$

$$\therefore \boxed{E_a = -\frac{(b-a)^3}{12n^2}\bar{f}''} \qquad \text{——— (9)}$$

— To improve the efficiency of the methods — Employ higher order polynomial approximation of $f(x)$.

Simpson's Rule

— Aside from applying the trapezoidal rule with finer segmentation, another way to obtain a more accurate estimate of an integral is to use higher-order polynomial to connect the points:

A.) Simpson's 1/3 Rule

a second-order interpolating polynomial is substituted in Eq. (1):

$$I = \int_a^b f(x)\,dx \cong \int_a^b f_2(x)\,dx$$

Let $a = x_0$, $b = x_2$ (as three points: x_0, x_1 & x_2 are required for a second-order fit)

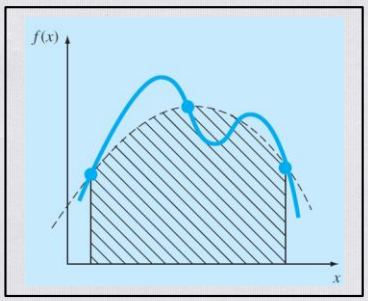
$$I \cong \int_{x_0}^{x_2} \left[\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1) \right.$$

$$\left. + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} f(x_2) \right] dx$$

After integration:

$$I \cong \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right] \quad \text{——— (10)}$$

where $h = \frac{b-a}{2}$.



- Eq. (10) can also be represented as:

$$I \cong (b-a)\left(\frac{f(x_0) + 4f(x_1) + f(x_2)}{6}\right)$$

width — average height

Note:

Here $a = x_0$, $b = x_2$ and x_1 is the mid-point, which is, $\left(\frac{b+a}{2}\right)$.

- Single application of Simpson's 1/3 rule has a truncation error:

$$E_t = -\frac{1}{90} h^5 f^{(4)}(\xi)$$

where ξ lies in the interval a to b. As, $h = \frac{b-a}{2}$

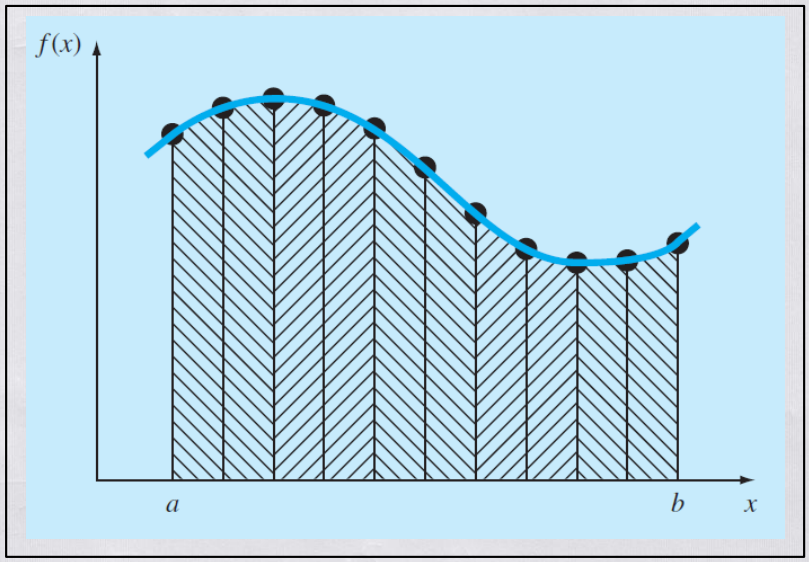
$$\Rightarrow \boxed{E_t = -\frac{(b-a)^5}{2880} f^{(4)}(\xi)} \quad \text{——— (11)}$$

Multiple application of Simpson's 1/3 rule :

— Similar to trapezoidal rule, Simpson's rule can be improved by dividing interval into a number of segments of equal width :

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

— Note that the method can be employed only if the number of segments is even.



— The total integral : $I = \int_{x_0}^{x_2} f(x)\,dx + \int_{x_2}^{x_4} f(x)\,dx + \cdots + \int_{x_{n-2}}^{x_n} f(x)\,dx$

Using Simpson's 1/3 rule for the individual integral

$$I \cong 2h\left(\frac{f(x_0) + 4f(x_1) + f(x_2)}{6}\right) + \cdots + 2h\left(\frac{f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)}{6}\right)$$

OR

$$I \cong (b-a)\left(f(x_0) + 4\sum_{i=1,3,5}^{n-1} f(x_i) + 2\sum_{j=2,4,6}^{n-2} f(x_j) + f(x_n) \right)$$

width — average height

- Truncation error:

$$\boxed{E_a = -\frac{(b-a)^5}{180\,n^4}\,\bar{f}^{(4)}}$$

□

Simpson's 3/8 Rule

— As Simpson's 1/3 rule is limited to situations where are an even number of segments (and an odd number of points).

— Therefore, Simpson's 1/3 rule is — an odd-segment even-point formula.

— An odd-segment-even-point formula — Simpson's 3/8 rule — is used in conjuction with the 1/3 rule to permit evaluation of both even and odd number of segments.

— $$I = \int_a^b f(x)\,dx \cong \int_a^b f_3(x)\,dx$$

$f_3(x)$ → Third Lagrange interpolating polynomial (which require four data points)

$x_0 \quad x_1 \quad x_2 \quad x_3$

$x_0 = a$, $x_3 = b$

$$I \cong \int_{a=x_0}^{b=x_3} \left[\frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} f(x_0) + \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} f(x_1) \right.$$

$$\left. + \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} f(x_2) + \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} f(x_3) \right] dx$$

Int. w.r.t - x

$$I \cong \frac{3h}{8}\left[f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)\right] \quad \text{——— (1)}$$

where $h = \frac{(b-a)}{3}$.

OR

$$I \cong (b-a)\left(\frac{f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)}{8}\right)$$

width — average height

— Simpson 3/8 rule has an error of

$$E_t = -\frac{3}{80}h^5 f^{(4)}(\xi), \quad a < \xi < b$$

OR

$$E_t = -\frac{(b-a)^5}{6480}f^{(4)}(\xi) \quad \text{——— (2)}$$

Simpson's 3/8 Rule

Problem Statement.

(a) Use Simpson's 3/8 rule to integrate

$$f(x) = 0.2 + 25x - 200x^2 + 675x^3 - 900x^4 + 400x^5$$

from $a = 0$ to $b = 0.8$.

(b) Use it in conjunction with Simpson's 1/3 rule to integrate the same function for five segments.

Solution.

(a) A single application of Simpson's 3/8 rule requires four equally spaced points:

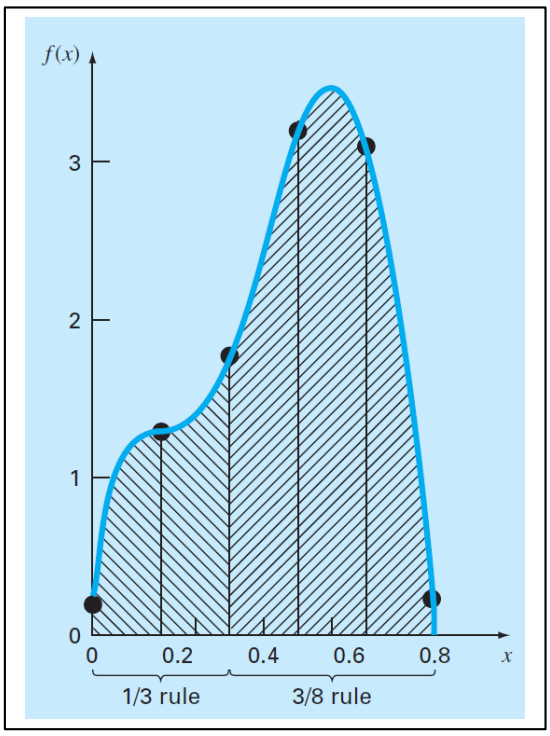
$$f(0) = 0.2 \qquad f(0.2667) = 1.432724$$
$$f(0.5333) = 3.487177 \qquad f(0.8) = 0.232$$

Using Eq. (1):

$$I \cong 0.8\frac{0.2 + 3(1.432724 + 3.487177) + 0.232}{8} = 1.519170$$

$$E_t = 1.640533 - 1.519170 = 0.1213630 \qquad \varepsilon_t = 7.4\%$$

$$E_a = -\frac{(0.8)^5}{6480}(-2400) = 0.1213630$$



(**b**) The data needed for a five-segment application ($h = 0.16$) is

$$f(0) = 0.2 \qquad f(0.16) = 1.296919$$
$$f(0.32) = 1.743393 \qquad f(0.48) = 3.186015$$
$$f(0.64) = 3.181929 \qquad f(0.80) = 0.232$$

The integral for the first two segments is obtained using Simpson's 1/3 rule:

$$I \cong 0.32\frac{0.2 + 4(1.296919) + 1.743393}{6} = 0.3803237$$

For the last three segments, the 3/8 rule can be used to obtain

$$I \cong 0.48\frac{1.743393 + 3(3.186015 + 3.181929) + 0.232}{8} = 1.264754$$

The total integral is computed by summing the two results:

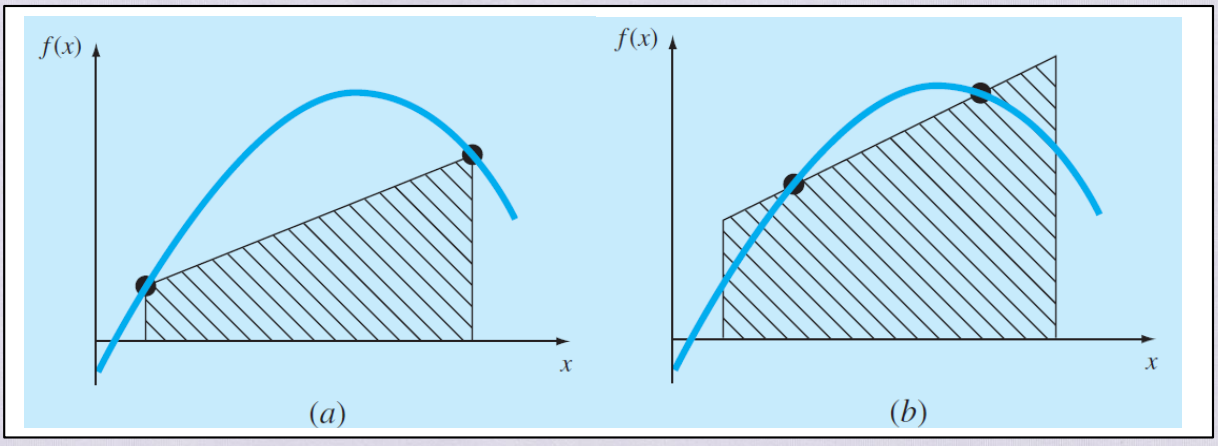
$$I = 0.3803237 + 1.264753 = 1.645077$$
$$E_t = 1.640533 - 1.645077 = -0.00454383 \qquad \varepsilon_t = -0.28\%$$

Gauss-Quadrature

— So far we have studied numerical integration or quadrature formulas known as the Newton-Cotes formulas. In these formulas, the integral estimate was based on evenly spaced function values. Therefore, the location of the base points used in these equations was fixed.

— Now, suppose that the constraint of fixed base points was removed and we were free to evaluate the area under a straight line joining any two points on the curve.

— That is can we improve Trapezoidal rule ?



- Gauss-quadrature is the class of technique to implement such a strategy.

- We will discuss about Gauss-Legendre formulas.

- **Method of Undetermined Coefficients:**
  - To derive the Trapezoidal rule $\left(I \cong (b-a)\left(\frac{f(b)+f(a)}{2}\right) \right)$ ——— (i)

Express Eq. (i) as:

$$I \cong C_0 f(a) + C_1 f(b) \quad \text{——— (ii)} \qquad C_i\text{'s constant}$$

- As trapezoidal rule yields exact result if the integrand is a constant or straight line.

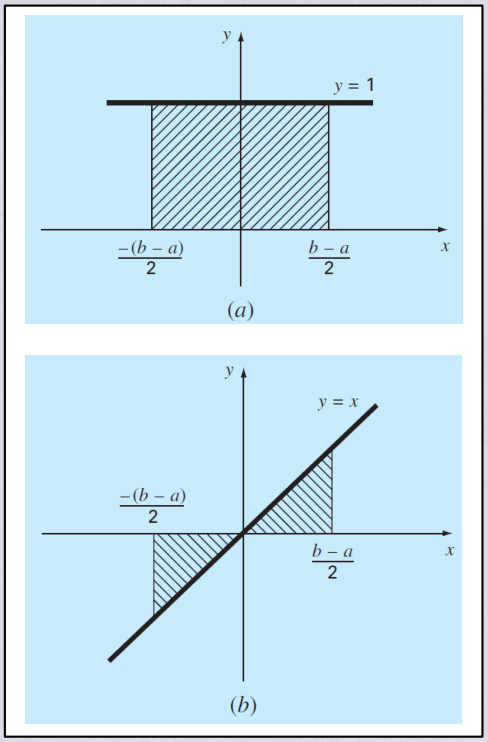
- Simple equations that represent these cases are: $y = 1$ and $y = x$.

$$\therefore \quad C_0 + C_1 = \int_{-\left(\frac{b-a}{2}\right)}^{\left(\frac{b+a}{2}\right)} 1 \, dx$$

and

$$-C_0\left(\frac{b-a}{2}\right) + C_1\left(\frac{b-a}{2}\right) = \int_{-\left(\frac{b-a}{2}\right)}^{\left(\frac{b+a}{2}\right)} x \, dx$$

$$\Rightarrow \left.\begin{aligned} C_0 + C_1 &= (b-a) \\ -C_0\left(\frac{b-a}{2}\right) + C_1\left(\frac{b-a}{2}\right) &= 0 \end{aligned}\right\}$$



$$\Rightarrow c_0 = c_1 = \frac{b-a}{2}$$

Substitute in (ii):

$$I \cong \left(\frac{b-a}{2}\right) f(a) + \left(\frac{b-a}{2}\right) f(b)$$

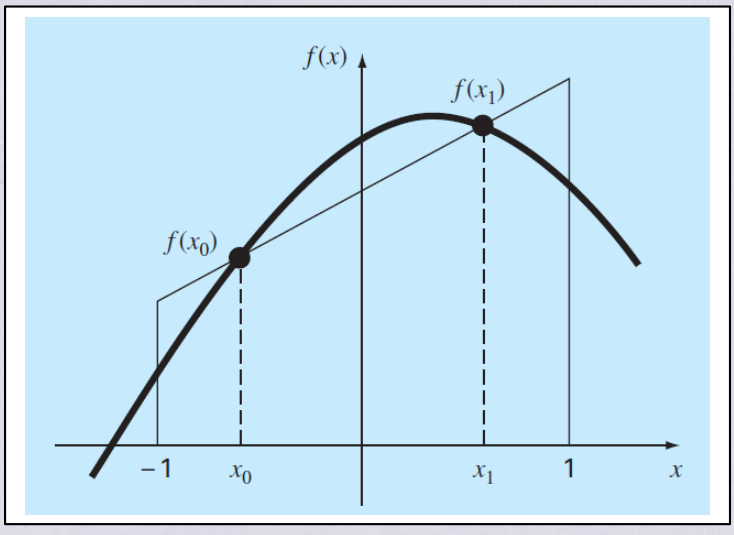
$$= (b-a)\left(\frac{f(b)+f(a)}{2}\right)$$

Derivation of two-point Gauss-Legendre formula:

— (Similar to the above derivation of trapezoidal rule) The object of Gauss-quadrature is to determine the coefficient of an equation of the form:

$$I \cong c_0 f(x_0) + c_1 f(x_1) \quad \text{——— (3)}$$

Here, unlike the trapezoidal rule that used fixed end points a and b, the function argument x_0 and x_1 are not fixed at the end points, but are unknowns.



— Here, we have total of four unknowns.

⇒ we require four conditions to determine them exactly.

— Here we assume that the integrals of a constant, linear, parabolic and cubic functions are exact.

$y = 1$, $y = x$, $y = x^2$, $y = x^3$

$$\therefore \quad C_0 f(x_0) + C_1 f(x_1) = \int_{-1}^{1} 1\,dx = 2 \qquad C_0 + C_1 = 2$$

$$C_0 f(x_0) + C_1 f(x_1) = \int_{-1}^{1} x\,dx = 0 \qquad C_0 x_0 + C_1 x_1 = 0$$

$$C_0 f(x_0) + C_1 f(x_1) = \int_{-1}^{1} x^2\,dx = 2/3 \qquad C_0 x_0^2 + C_1 x^2 = 2/3$$

$$C_0 f(x_0) + C_1 f(x_1) = \int_{-1}^{1} x^3\,dx = 0 \qquad C_0 x_0^3 + C_1 x_1^3 = 0$$

On solving these four equations:

$$C_0 = C_1 = 1$$
$$x_0 = -\tfrac{1}{\sqrt{3}}, \quad x_1 = \tfrac{1}{\sqrt{3}}$$

} Substitute in (3)

$$\boxed{I \cong f\left(-\tfrac{1}{\sqrt{3}}\right) + f\left(\tfrac{1}{\sqrt{3}}\right)}$$

— Gauss-Legendre two-point formula

□

Gauss-quadrature

- Note that for the Gauss-Legendre quadrature rule integration limits in eqns. are from -1 to 1.
- A simple change of variables can be used to translate other limits of integration into this form.
- Let x_d be the new variable and x old variable. x_d and x are related as:

$$x = a_0 + a_1 x_d \quad \text{——} (1)$$

Such that $x = a$ then $x_d = -1 \Rightarrow a = a_0 + a_1(-1)$ —— (2)

And $x = b$ then $x_d = 1 \Rightarrow b = a_0 + a_1(1)$ —— (3)

From (2) and (3): $a_0 = \frac{b+a}{2}$ and $a_1 = \frac{b-a}{2}$

∴ From (1) $x = \frac{(b+a) + (b-a)x_d}{2}$

$dx = \left(\frac{b-a}{2}\right) dx_d$

→ x and dx can be substituted before applying Gauss-Legendre rule $\int_a^b f(x)\,dx$

Two-Point Gauss-Legendre Formula

Problem Statement. Use G-L rule to evaluate the integral of

$$f(x) = 0.2 + 25x - 200x^2 + 675x^3 - 900x^4 + 400x^5$$

between the limits $x = 0$ to 0.8. Recall that this was the same problem that we solved in Chap. 21 using a variety of Newton-Cotes formulations. The exact value of the integral is 1.640533.

Solution. Before integrating the function, we must perform a change of variable so that the limits are from -1 to $+1$. To do this, we substitute $a = 0$ and $b = 0.8$ into Eq. (1) to yield

$$x = 0.4 + 0.4x_d$$

The derivative of this relationship is

$$dx = 0.4\,dx_d$$

Both of these can be substituted into the original equation to yield

$$\int_0^{0.8} (0.2 + 25x - 200x^2 + 675x^3 - 900x^4 + 400x^5)\,dx$$

$$= \int_{-1}^{1} [0.2 + 25(0.4 + 0.4x_d) - 200(0.4 + 0.4x_d)^2 + 675(0.4 + 0.4x_d)^3 - 900(0.4 + 0.4x_d)^4 + 400(0.4 + 0.4x_d)^5]0.4\,dx_d$$

Therefore, the right-hand side is in the form that is suitable for evaluation using Gauss quadrature. The transformed function can be evaluated at $-1/\sqrt{3}$ to be equal to 0.516741

and at $1/\sqrt{3}$ to be equal to 1.305837. Therefore, the integral according to G-L rule is

$$I \cong 0.516741 + 1.305837 = 1.822578$$

Error in Gauss-quadrature :

- The error in the Gauss-Legendre rule is :

$$E_t = \frac{2^{2n+3}[(n+1)!]^4}{(2n+3)[(2n+2)!]^3} f^{(2n+2)}(\xi_1) \quad \text{——} (4)$$

n = the number of points minus one.

and $f^{(2n+2)}(\xi_1)$ = the $(2n+2)^{th}$ derivative of the function after the change of variable with ξ_1 located somewhere between -1 to 1.

— Eq: (4) indicates the superiority of Gauss quadrature to Newton-Cotes formula.

— From (4), for two-point Gauss-Legendre rule:

$$\boxed{E_t = \frac{1}{135} f^{(4)}(\xi_1), \quad -1 < \xi_1 < 1} \quad \text{——} (5)$$

— Note: For many functions confronted in engineering practice, Gauss-quadrature provides an efficient means for evaluating integrals.

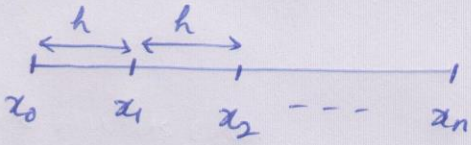
□

Numerical Differentiation

- Base on Taylor's series expansion.

$$f(x+h) = f(x) + h f'(x) + \frac{h^2}{2!} f''(x) + \cdots$$

- Suppose we have equi-distant points:



$$x_{i+1} = x_i + h$$

$$i = 0, \cdots, n-1.$$

$$\Rightarrow f(x_{i+1}) = f(x_i + h) = f(x_i) + h f'(x_i) + \frac{h^2}{2!} f''(x_i) + \cdots$$

OR

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} - \frac{h}{2} f''(x_i) + O(h^2) \quad \text{— (A)}$$

OR

$$\boxed{f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} + O(h)} \quad \text{— (6)}$$

— First Forward difference formula and h is the step-size.

— Similarly

$$f(x_{i-1}) = f(x_i - h) = f(x_i) - h f'(x_i) + \frac{h^2}{2!} f''(x_i) + \cdots$$

— (B)

OR

$$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{h} + O(h) \quad \text{——} (7)$$

— First Backward difference formula.

— Again, subtracting (A) from (B):

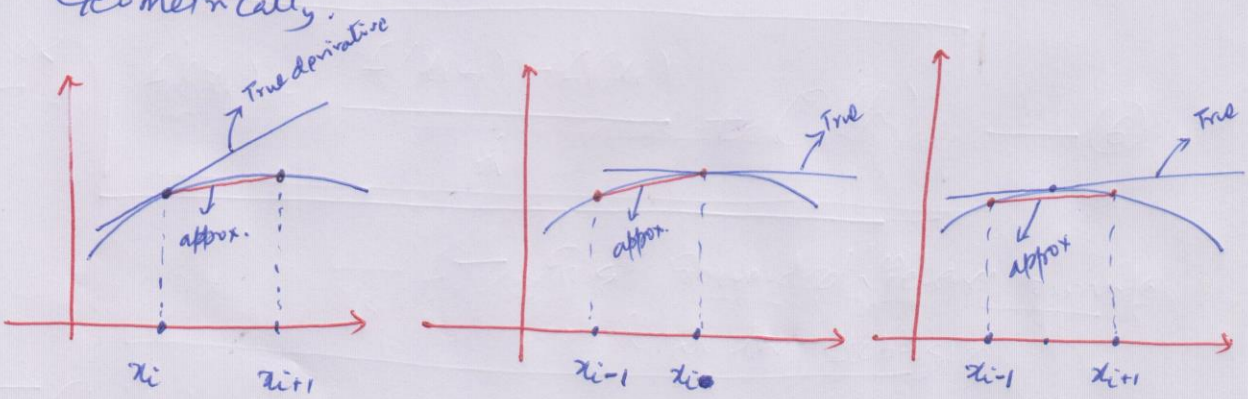
$$f(x_{i+1}) - f(x_{i-1}) = 2hf'(x_i) + \frac{2h^3}{3!} f'''(x_i) + \cdots$$

OR

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h} + O(h^2) \quad \text{——} (8)$$

— Centered difference approximation to the first derivative.

— Geometrically:



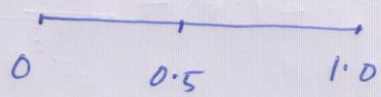
Forward Backward Centered

□

Example: $f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$

Estimate first derivative of $f(x)$ at $x = 0.5$ with step-size $h = 0.5$ using Forward, Backward & Centred formulas:

— For $h = 0.5$:



$x_{i-1} = 0 \Rightarrow f(x_{i-1}) = 1.2$

$x_i = 0.5 \Rightarrow f(x_i) = 0.925$

$x_{i+1} = 1.0 \Rightarrow f(x_{i+1}) = 0.2$

Forward:

$$f'(0.5) \cong \frac{f(x_{i+1}) - f(x_i)}{h} = \frac{0.2 - 0.925}{0.5} = -1.45$$

Backward:

$$f'(0.5) \cong \frac{f(x_i) - f(x_{i-1})}{h} = -0.55$$

Centered:

$$f'(0.5) \cong \frac{f(x_{i+1}) - f(x_{i-1})}{2h} = -1.0$$

— Exact value $f'(0.5) = -0.9125$

□

Curve Fitting

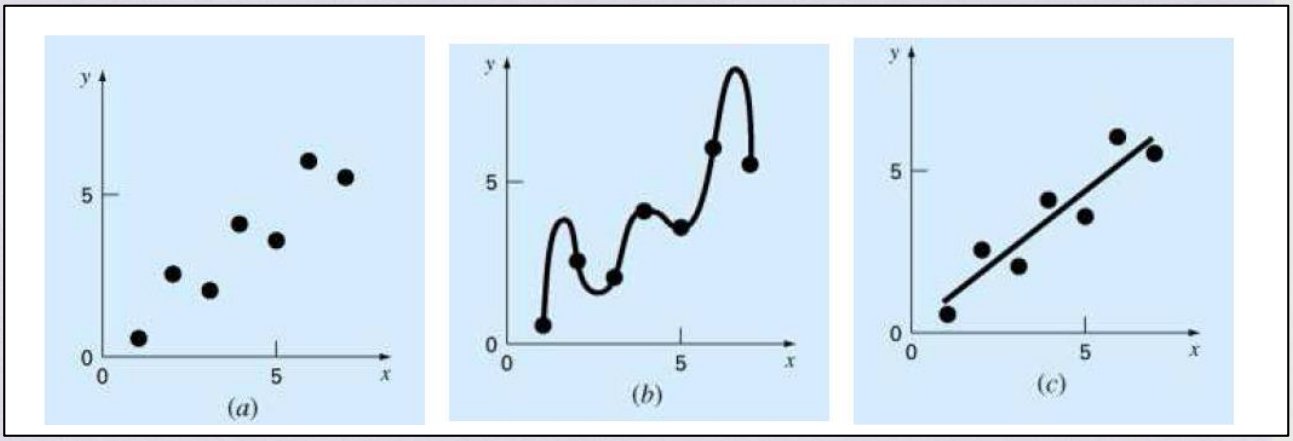
- Two types of curve fitting:

A.) Least square regression:

- For a given data set, derive a single curve that represent the general trend of the data.
- Used when the given data exhibit a significant degree of error or noise.

B.) Interpolation:

- For a given data set, fit a curve or a series of curves that pass directly through each of the data points.
- Used when data are very precise.



– Linear Regression:

– Fitting a straight line to a set of paired observations: $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$.

Mathematical expression for the straight line (model)

$$y = a_0 + a_1 x \quad , \quad a_0 - \text{intercept}, \; a_1 - \text{slope}$$

Define:

$$e_i = y_i|_{\text{measured}} - y_i|_{\text{model}} = y_i - (a_0 + a_1 x_i)$$

Criterion for best fit:

$$\min_{a_0, a_1} S_r = \min_{a_0, a_1} \sum_{i=1}^{n} e_i^2 = \min_{a_0, a_1} \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2$$

Find a_0 and a_1?

$$\frac{\partial S_r}{\partial a_0} = -2 \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i) = 0 \quad \text{——} \; (1)$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum_{i=1}^{n} [(y_i - a_0 - a_1 x_i) x_i] = 0 \quad \text{——} \; (2)$$

From Eq. (1):

$$\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a_0 - \sum_{i=1}^{n} a_1 x_i = 0$$

OR

$$n a_0 + \sum_{i=1}^{n} x_i a_1 = \sum_{i=1}^{n} y_i \quad \text{—— (3)}$$

From Eq. (2):

$$\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} a_0 x_i - \sum_{i=1}^{n} a_1 x_i^2 = 0$$

OR

$$\sum_{i=1}^{n} x_i a_0 + \sum_{i=1}^{n} x_i^2 a_1 = \sum_{i=1}^{n} x_i y_i \quad \text{—— (4)}$$

- Eqns. (3) and (4) are called <u>normal</u> equations.

From Eq. (3):

$$a_0 = \frac{1}{n} \sum_{i=1}^{n} y_i - \frac{1}{n} \sum_{i=1}^{n} x_i a_1$$

$$= \bar{y} - \bar{x} a_1$$

Where, $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$, $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$ — means of x & y.

From Eq. (4):

$$\sum_{i=1}^{n} x_i \left(\frac{1}{n} \sum_{i=1}^{n} y_i - \frac{1}{n} \sum_{i=1}^{n} x_i a_1 \right) + \sum_{i=1}^{n} x_i^2 a_1 = \sum_{i=1}^{n} x_i y_i$$

OR

$$a_1 = \frac{\sum_{i=1}^{n} x_i y_i - \frac{1}{n} \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \frac{1}{n} \left(\sum_{i=1}^{n} x_i \right)^2}$$

OR

$$a_1 = \frac{n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i \right)^2}$$

Definitions:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2$$

Standard error of the estimate:

$$S_{y/x} = \sqrt{\frac{S_r}{n-2}}$$

– spread around the regression line.

Standard deviation of data points:

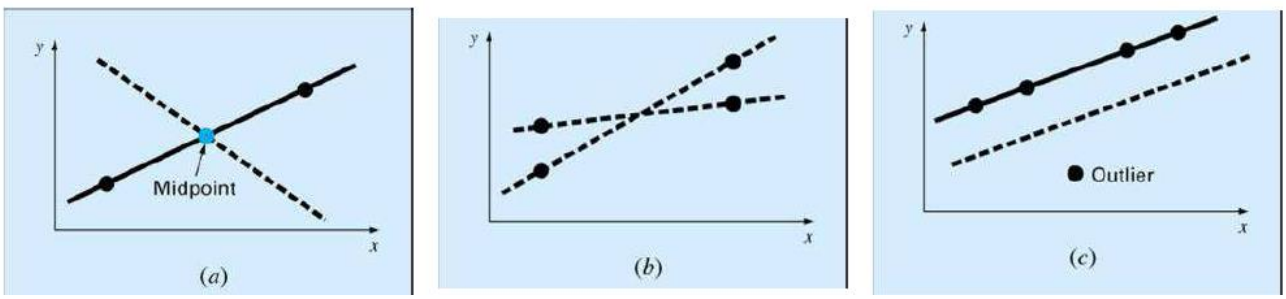
$$S_y = \sqrt{\frac{S_t}{n-1}} = \sqrt{\frac{\sum_{i=1}^{n} (y_i - y_i)^2}{n-1}}$$

where, $S_t = \sum_{i=1}^{n} (y_i - \bar{y})^2$, that is, spread around the mean value $\bar{y}$.

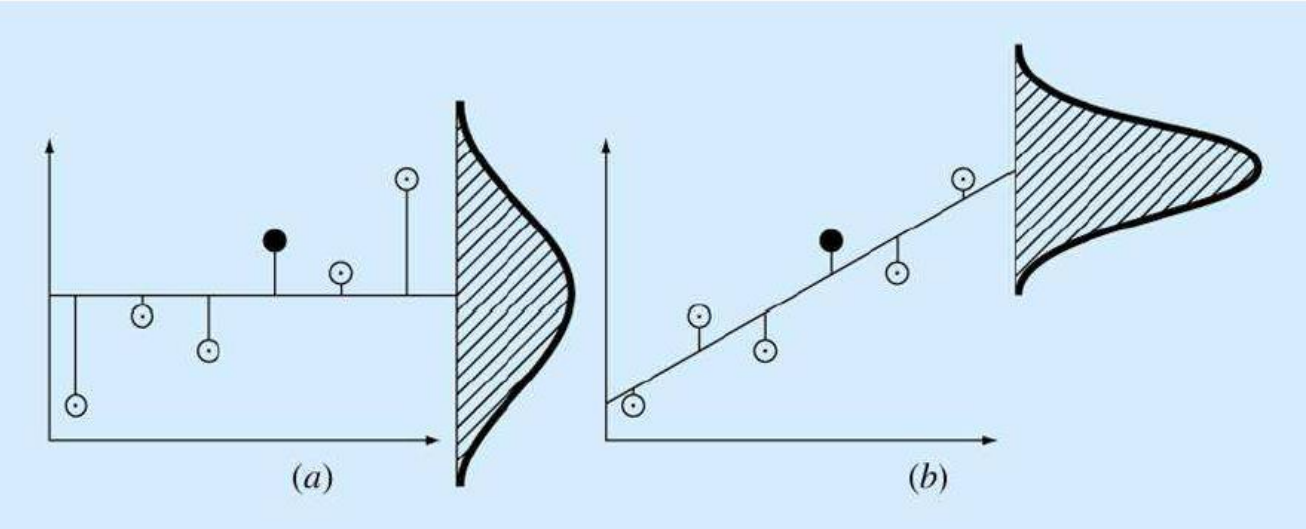
Correlation coefficient:

$$r = \sqrt{\frac{S_t - S_r}{S_t}}$$

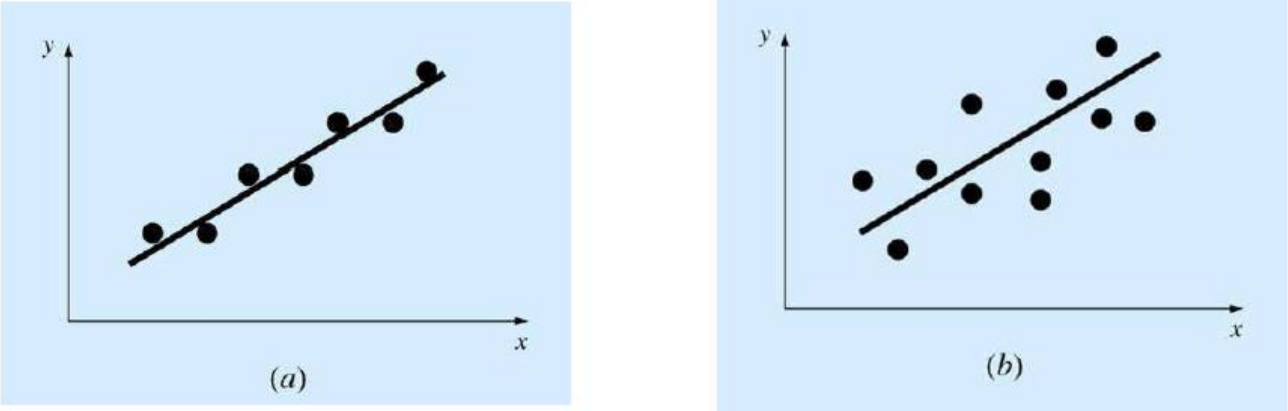
— Improvement or error reduction due to describing the data in terms of a straight line rather than as an average value.



Examples of some other criteria for best fit that are inadequate for regression: (a) min Σ e_i , (b) min Σ $|e_i|$ and (c) min max e_i.



(a) Spread of data around mean of the dependent variable, (b) spread of the data around best-fit line.



(a) Illustration of linear regression with (a) small and (b) large residual errors.

Example:

x	1	2	3	4	5	6	7
y	0.5	2.5	2.0	4.0	3.5	6.0	5.5

$$\sum x_i = 1+2+\cdots+7 = 28$$

$$\sum y_i = 0.5+2.5+\cdots+5.5 = 24$$

$$\sum x_i^2 = 1^2+2^2+\cdots+7^2 = 140$$

$$\sum x_i y_i = 1\times 0.5 + 2\times 2.5 + \cdots + 7\times 5.5 = 119.5$$

$$a_1 = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \qquad (\text{Here, } n=7)$$

$$= 0.8393$$

$$a_0 = \bar{y} - \bar{x}a_1 = \frac{1}{n}\sum_{i=1}^{n} y_i - \frac{a_1}{n}\sum_{i=1}^{n} x_i = 0.07143$$

Model:

$$\boxed{y = 0.07143 + 0.8393x}$$

$$S_r = \sum_{i=1}^{n} e_i^2, \qquad e_i = y_i - (a_0 + a_1 x_i)$$

$$e_1 = 0.5 - 0.07143 - 0.8393 \times 1 = -0.410$$

$$e_2 = 2.5 - 0.07143 - 0.8393 \times 2 = 0.750$$

$$e_3 = 2.0 - 0.07143 - 0.8393 \times 3 = -0.589$$

$$\vdots$$

$$e_7 = 5.5 - 0.07143 - 0.8393 \times 7 = -0.446$$

$$\Rightarrow S_r = (-0.410)^2 + 0.750^2 + (-0.589)^2 + \cdots + (-0.446)^2$$

$$= 2.9911$$

and

$$S_t = \sum_{i=1}^{n} (y_i - \bar{y})^2 = 22.714$$

Standard deviation of the data points:

$$S_y = \sqrt{\frac{S_t}{n-1}} = \sqrt{\frac{22.714}{6}} = 1.946$$

Standard error of the estimate:

$$S_{y/x} = \sqrt{\frac{S_r}{n-2}} = \sqrt{\frac{2.9911}{5}} = 0.774$$

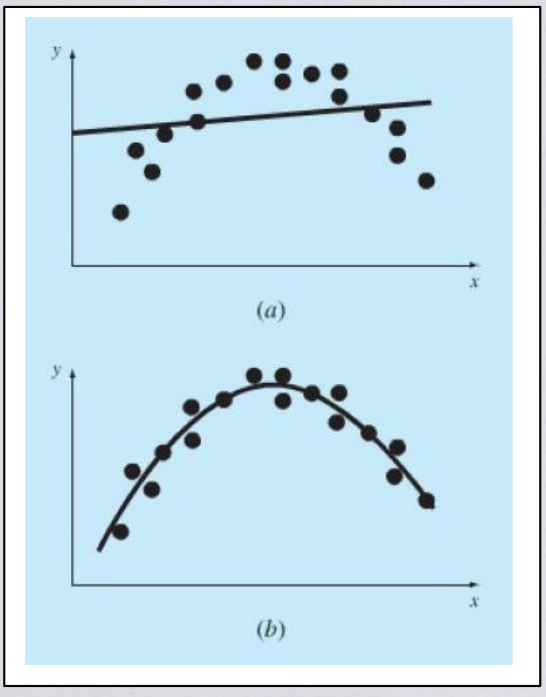
$$r^2 = \frac{S_t - S_r}{S_t} = 0.868$$

$S_{y/x} < S_y$, $S_r < S_t$. Correlation coefficient $r = \sqrt{\frac{S_t - S_r}{S_t}} = 0.932$ □

Lecture-10 Numerical Methods

- Polynomial Regression :

- Developed a strategy to derive the equation of straight line using the least-squares criterion.
- Some practical data, although exhibiting a marked pattern (as shown in the fig. below), is poorly represented by a straight line. For such cases, a curve would be better suited to fit the data.
- The alternative is to fit polynomials to the data using polynomial regression. And the least-squares approach can be extended to fit the data to a higher-order polynomial.



– Polynomial Regression:

Given data: (x_i, y_i), $i = 1, 2, \cdots, n$

Fit a second-order polynomial:

$$y = a_0 + a_1 x + a_2 x^2 \quad \text{—— (model)}$$

$$e_i = y_i\Big|_{\text{measured}} - y_i\Big|_{\text{model}}$$

$$= y_i - (a_0 + a_1 x_i + a_2 x_i^2)$$

Criterion for best fit:

$$\min_{a_0, a_1, a_2} S_r = \min_{a_0, a_1, a_2} \sum_{i=1}^{n} e_i^2$$

$$= \min_{a_0, a_1, a_2} \sum_{i=1}^{n} (y_i - a_0 - a_1 x - a_2 x^2)^2$$

Find a_0, a_1 and a_2:

$$\frac{\partial S_r}{\partial a_0} = -2 \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i - a_2 x_i^2) = 0 \quad \text{—— ①}$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum_{i=1}^{n} [(y_i - a_0 - a_1 x_i - a_2 x_i^2) x_i] = 0 \quad \text{—— ②}$$

$$\frac{\partial S_r}{\partial a_2} = -2 \sum_{i=1}^{n} [(y_i - a_0 - a_1 x_i - a_2 x_i^2) x_i^2] = 0 \quad \text{—— ③}$$

OR

$$n a_0 + \sum_{i=1}^{n} x_i a_1 + \sum_{i=1}^{n} x_i^2 a_2 = \sum_{i=1}^{n} y_i \quad \text{——①}$$

$$\sum_{i=1}^{n} x_i a_0 + \sum_{i=1}^{n} x_i^2 a_1 + \sum_{i=1}^{n} x_i^3 a_2 = \sum_{i=1}^{n} x_i y_i \quad \text{——②}$$

$$\sum_{i=1}^{n} x_i^2 a_0 + \sum_{i=1}^{n} x_i^3 a_1 + \sum_{i=1}^{n} x_i^4 a_2 = \sum_{i=1}^{n} x_i^2 y_i \quad \text{——③}$$

Three (linear) equations in three unknowns: a_0, a_1 & a_2

$\therefore$ a_0, a_1 and a_2 can be calculated from ① – ③.

Note: In general, to fit p^{th} degree polynomial:

$$y = a_0 + a_1 x + a_2 x^2 + \cdots + a_p x^p$$

with least-squares regression approach, one need to solve a system of $(p+1)$ linear equations.

Standard error: $S_{y/x} = \sqrt{\dfrac{S_r}{n-(p+1)}}$

Multiple Linear Regression:

— Used when y (dependent variable) is a linear function of two or more independent variables.

Model: $y = a_0 + a_1 x_1 + a_2 x_2$

Given data: (x_{1i}, x_{2i}, y_i), $i = 1, 2, \cdots, n$.

$$e_i = y_i|_{measured} - y_i|_{model}$$

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_{1i} - a_2 x_{2i} \right)^2$$

Find a_0, a_1 and a_2 (which minimize S_r):

$$\frac{\partial S_r}{\partial a_0} = -2 \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_{1i} - a_2 x_{2i} \right) = 0 \quad \text{—— (A)}$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum_{i=1}^{n} \left[\left(y_i - a_0 - a_1 x_{1i} - a_2 x_{2i} \right) x_{1i} \right] = 0 \quad \text{—— (B)}$$

$$\frac{\partial S_r}{\partial a_2} = -2 \sum_{i=1}^{n} \left[\left(y_i - a_0 - a_1 x_{1i} - a_2 x_{2i} \right) x_{2i} \right] = 0 \quad \text{—— (C)}$$

OR

$$\begin{cases} n a_0 + \sum_{i=1}^{n} x_{1i} a_1 + \sum_{i=1}^{n} x_{2i} a_2 = \sum_{i=1}^{n} y_i \\ \sum x_{1i} a_0 + \sum x_{1i}^2 a_1 + \sum x_{1i} x_{2i} a_2 = \sum x_{1i} y_i \\ \sum x_{2i} a_0 + \sum x_{1i} x_{2i} a_1 + \sum x_{2i}^2 a_2 = \sum x_{2i} y_i \end{cases}$$

OR

$$\begin{bmatrix} n & \sum x_{1i} & \sum x_{2i} \\ \sum x_{1i} & \sum x_{1i}^2 & \sum x_{1i}x_{2i} \\ \sum x_{2i} & \sum x_{1i}x_{2i} & \sum x_{2i}^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \sum x_{1i}y_i \\ \sum x_{2i}y_i \end{bmatrix}$$

— In general for n-dimensional case:

Standard error: $S_{y/x} = \sqrt{\frac{S_r}{n-(m+1)}}$

—— * ——

— General Linear Least Squares:

Model:

$$y = a_0 Z_0 + a_1 Z_1 + a_2 Z_2 + \cdots + a_m Z_m$$

where: $Z_0, Z_1, \cdots, Z_m$ are $(m+1)$ different functions.

- Simple linear LSR: $Z_0 = 1$, $Z_1 = x$, $Z_i = 0$ for $i \geq 2$.
- Polynomial LSR: $Z_i = x^i$ $(Z_0 = 1, Z_1 = x, Z_2 = x^2 \cdots)$
- Multiple linear LSR: $Z_0 = 1$, $Z_i = x_i$ for $i \geq 1$.

Here, "linear" refers only to the model's dependence on its parameters — that is a's.

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i\big|_{measured} - y_i\big|_{model} \right)^2$$

$\therefore$ Given data: $\left(Z_{0i}, Z_{1i}, Z_{2i}, \cdots, Z_{mi}\}, y_i \right)$, $i = 1, 2, \cdots, n$

$$S_r = \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{m} a_j Z_{ji} \right)^2$$

Find a_j, $j = 0, 1, 2, \cdots, m$ to minimize S_r.

$$\therefore \quad \frac{\partial S_r}{\partial a_k} = -2 \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{m} a_j Z_{ji} \right) \cdot Z_{ki} = 0$$

OR

$$\sum_{i=1}^{n} y_i Z_{ki} = \sum_{i=1}^{n} \sum_{j=0}^{m} Z_{ki} Z_{ji} a_j \quad , \quad k = 0, 1, 2, \cdots, m$$

OR

$$\sum_{j=0}^{m} \sum_{i=1}^{n} Z_{ki} Z_{ji} a_j = \sum_{i=1}^{n} y_i Z_{ki}$$

OR

$$\boxed{Z^T Z \{A\} = Z^T \{Y\}}$$

Normal Equations.

$$Z = \begin{bmatrix} Z_{01} & Z_{11} & \cdots & Z_{m1} \\ Z_{02} & Z_{12} & \cdots & Z_{m2} \\ \vdots & & & \vdots \\ Z_{0n} & Z_{1n} & \cdots & Z_{mn} \end{bmatrix}$$

Where $\{A\}^T = (a_0, a_1, \cdots, a_m)$ and $\{Y\}^T = (y_1, y_2, \cdots, y_n)$

$\{A\}$: unknown coefficients

$\{Y\}$: observed values of the dep. variable

Numerical Methods

Example-1:

Fit $y = \alpha_2 x^{\beta_2}$ ——— (1) to the following data (using a logarithmic transformation of the data.

x	y	$\log x$	$\log y$
1	0.5	0	−0.301
2	1.7	0.301	0.226
3	3.4	0.477	0.534
4	5.7	0.602	0.753
5	8.4	0.699	0.922

Original data — log transformed data (from Eq. (1))

We can linearize power equation (power law) by taking log (base 10):

$$\log y = \beta_2 \log x + \log \alpha_2 \quad \text{——— (2)}$$

log transformed data is given in the above table.

A linear regression of the log transformed data yields:

$$\log y = 1.75 \log x - 0.300 \quad \text{(Check it!)}$$

∴ Intercept $\log \alpha_2 = -0.300$, and by taking antilog $\Rightarrow \boxed{\alpha_2 = 10^{-0.3}}$

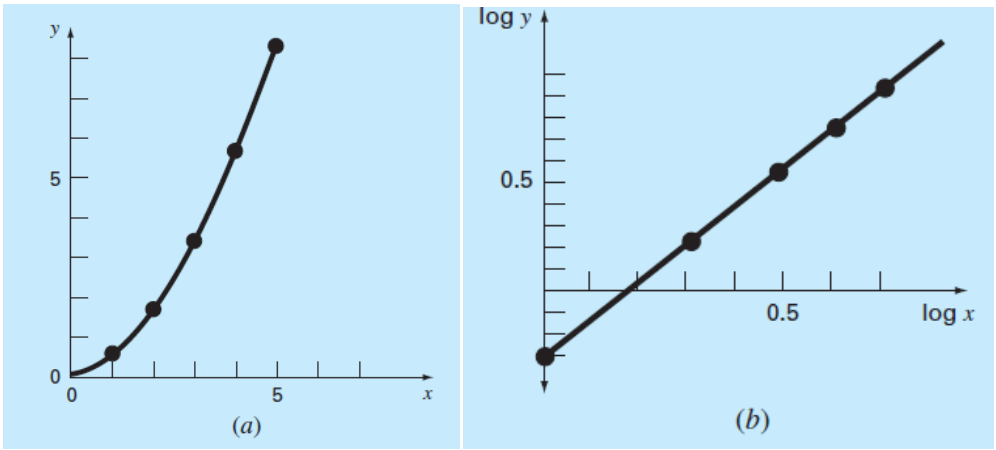
$= 0.5$

— slope $\beta_2 = 1.75$.

∴ The power equation (fit) is:

$$y = 0.5x^{1.75}$$

□



(*a*) Plot of untransformed data with the power equation that fits given data. (*b*) Plot of transformed data used to determine the coefficients of the power equation.

Example-2: (Polynomial regression) Fit a second-order poly. to the following data:

x_i	y_i	$(y_i - \bar{y})^2 = S_t$	$(y_i - a_0 - a_1 x_i - a_2 x_i^2) = S_r$
0	2.1		
1	7.7		
2	13.6		
3	27.2		
4	40.9		
5	61.1		

– Here, $m=2$, $\sum_{i=1}^{6} x_i = 15$, $\sum x_i^4 = 979$

$n=6$, $\sum y_i = 152.6$, $\sum x_i y_i = 585.6$

$\bar{x} = 2.5$, $\sum x_i^2 = 55$, $\sum x_i^2 y_i = 2488.8$

$\bar{y} = 25.433$, $\sum x_i^3 = 225$

∴ System of simultaneous linear equations:

$$\begin{bmatrix} 6 & 15 & 55 \\ 15 & 55 & 225 \\ 55 & 225 & 979 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 152.6 \\ 585.6 \\ 2488.8 \end{bmatrix}$$

Solving these (Gauss elimination): $a_0 = 2.47857$, $a_1 = 2.35929$

$a_2 = 1.86071$

∴ The least-squares quadratic equation:

$$\boxed{y = 2.47857 + 2.35929x + 1.86071x^2} \quad \text{—(2)}$$

$S_{y/x}$ (standard error of the estimate based on regression poly (2))

$$= \sqrt{\frac{S_r}{n-(m+1)}} = \sqrt{\frac{3.74657}{6-3}} = 1.12$$

← Coefficient of determination

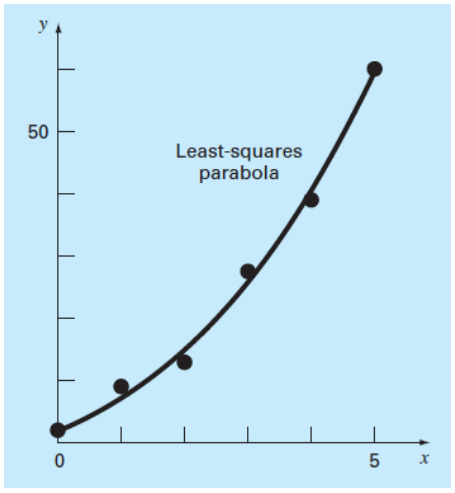
$$r^2 = \left(\frac{S_t - S_r}{S_t}\right) = \frac{2513.39 - 3.74657}{2513.39}$$

$$= 0.99851$$

and correlation coefficient: $r = \sqrt{0.99851} = 0.99925$

Note: That is 99.851 % of the original uncertainty has been explained by the model.

□



Fit of a second-order polynomial.

Example-3: (multilinear regression) Following data was calculated from the equation:

$$y = 5 + 4x_1 - 3x_2$$

x_1	x_2	y
0	0	5
2	1	10
2.5	2	9
1	3	0
4	6	3
7	2	27

Use multiple linear regression to fit this data:

Required computations to get the normal equations are given below:

	y	x_1	x_2	x_1^2	x_2^2	x_1x_2	x_1y	x_2y
	5	0	0	0	0	0	0	0
	10	2	1	4	1	2	20	10
	9	2.5	2	6.25	4	5	22.5	18
	0	1	3	1	9	3	0	0
	3	4	6	16	36	24	12	18
	27	7	2	49	4	14	189	54
Σ	54	16.5	14	76.25	54	48	243.5	100

Table: Computations required to develop the normal equations.

System of equations:

$$\begin{bmatrix} 6 & 16.5 & 14 \\ 16.5 & 76.25 & 40 \\ 14 & 40 & 54 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 54 \\ 243.5 \\ 100 \end{bmatrix}$$

$\Rightarrow a_0 = 5, \quad a_1 = 4, \quad a_2 = -3$

That is consistent with original data.

□

Nonlinear Regression:

How to fit nonlinear models to the data?

Here, models have a nonlinear dependence on their parameters:

For example: $$f(x) = a_0(1 - e^{-a_1 x}) + e \qquad \text{---- (i)}$$

∴ Nonlinear regression is based on determining the values of the parameters that minimize the sum of squares of residuals.

Being nonlinear case, sol^n procedure requires an iterative process.

Use Gauss-Newton method for minimizing the sum of the squares of the residuals between data and nonlinear eqtn.

— Gauss - Newton Method

— Taylor series expansion is used to linearize the original nonlinear equation.

— Then, least-squares technique can be used to obtain new estimate of the parameters that move in the direction of minimizing the residual.

Steps:

A.) Relationship between the nonlinear equation and the data can be expressed as:

$$y_i = \underbrace{f(x_i; a_0, a_1, \cdots, a_m)}_{\text{model}} + \underbrace{e_i}_{\text{error}}$$

(y_i: measured)

OR

$$y = f(x_i) + e_i \qquad \text{—— (ii)}$$

B.) Using Taylor series expansion: (For a two parameter case)

$$f(x_i)\Big|_{j+1} = f(x_i)\Big|_j + \frac{\partial f(x_i)|_j}{\partial a_0}\Delta a_0 + \frac{\partial f(x_i)|_j}{\partial a_1}\Delta a_1 \qquad \text{—— (iii)}$$

— $\Delta a_0 = a_0|_{j+1} - a_0|_j$

$\Delta a_1 = a_1|_{j+1} - a_1|_j$

$\begin{cases} j = \text{initial guess} \\ j+1 = \text{prediction} \end{cases}$

From (ii) and (iii):

$$y_i - f(x_i)|_j = \frac{\partial f(x_i)|_j}{\partial a_0} \Delta a_0 + \frac{\partial f(x_i)|_j}{\partial a_1} \Delta a_1 + e_i$$

OR, in matrix notations:

$$\{D\} = [Z_j]\{\Delta A\} + \{E\} \quad \text{——} \quad \text{(iv)}$$

Where

$$[Z_j] = \begin{bmatrix} \frac{\partial f_1}{\partial a_0} & \frac{\partial f_1}{\partial a_1} \\ \frac{\partial f_2}{\partial a_0} & \frac{\partial f_2}{\partial a_1} \\ \vdots & \vdots \\ \frac{\partial f_n}{\partial a_0} & \frac{\partial f_n}{\partial a_1} \end{bmatrix}$$

n = no. of data points

$\frac{\partial f_i}{\partial a_k}$ = partial derivative of the function wrt the k^{th} parameter evaluated at the i^{th} data point.

$$\{D\} = \begin{Bmatrix} y_1 - f(x_1) \\ y_2 - f(x_2 \\ \vdots \\ y_n - f(x_n) \end{Bmatrix}$$

↓ Diff. between measurement and function values.

$$\{\Delta A\} = \begin{Bmatrix} \Delta a_0 \\ \Delta a_1 \\ \vdots \\ \Delta a_n \end{Bmatrix}$$

changes in the parameter values.

(c.) Apply linear least-squares to Eq. (iv) in the following normal equations:

$$\left([Z_j]^T [Z_j]\right)\{\Delta A\} = [Z_j]^T \{D\} \quad \text{——} (v)$$

That is solve Eq. (v) for $\{\Delta A\}$, which can be used to get the improved values for the parameters, as:

$$a_0|_{j+1} = a_0|_j + \Delta a_0$$

and $$a_1|_{j+1} = a_1|_j + \Delta a_1$$

This procedure will be repeated until the sol^n converges, that is, until:

$$|\varepsilon_a|_k = \left| \frac{a_k|_{j+1} - a_k|_j}{a_k|_{j+1}} \right| 100\%$$

falls below acceptable stopping criterion □.

Numerical Methods

Example-1 (Nonlinear regression using Gauss-Newton Method)

Fit the function $f(x; a_0, a_1) = a_0(1 - e^{-a_1 x})$ to the data:

x	0.25	0.75	1.25	1.75	2.25
y	0.28	0.57	0.68	0.74	0.79

Use initial guesses of $a_0 = 1.0$ and $a_1 = 1.0$ for the parameters.

Soln:

$$f = a_0(1 - e^{-a_1 x})$$

$$\therefore \frac{\partial f}{\partial a_0} = 1 - e^{-a_1 x}$$

$$\frac{\partial f}{\partial a_1} = a_0 x e^{-a_1 x}$$

$$\therefore [Z_0] = \begin{bmatrix} \partial f_1/\partial a_0 & \partial f_1/\partial a_1 \\ \partial f_2/\partial a_0 & \partial f_2/\partial a_1 \\ \partial f_3/\partial a_0 & \partial f_3/\partial a_1 \\ \partial f_4/\partial a_0 & \partial f_4/\partial a_1 \\ \partial f_5/\partial a_0 & \partial f_5/\partial a_1 \end{bmatrix} = \begin{bmatrix} 0.2212 & 0.1947 \\ 0.5276 & 0.3543 \\ 0.7135 & 0.3581 \\ 0.8262 & 0.3041 \\ 0.8946 & 0.2371 \end{bmatrix}$$

$$[Z_0]^T[Z_0] = \begin{bmatrix} 2.3193 & 0.9489 \\ 0.9489 & 0.4404 \end{bmatrix}$$

Vector D – differences between the measurements and the model predictions.

$$\{D\} = \left\{ y_i - f(x_i) \right\} = \begin{Bmatrix} 0.0588 \\ 0.0424 \\ -0.0335 \\ -0.0862 \\ -0.1046 \end{Bmatrix}$$

$1 \le i \le 5$

$$\therefore [Z_0]^T\{D\} = \begin{bmatrix} -0.1533 \\ -0.0365 \end{bmatrix}$$

Vector ΔA – changes in the parameter values

$$\{\Delta A\} = \begin{Bmatrix} \Delta a_0 \\ \Delta a_1 \end{Bmatrix} = \begin{Bmatrix} -0.2714 \\ 0.5019 \end{Bmatrix}$$

∴ updated values of free parameters:

$$\begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix} + \begin{Bmatrix} -0.2714 \\ 0.5019 \end{Bmatrix} = \begin{Bmatrix} 0.7286 \\ 1.5019 \end{Bmatrix}$$

EG

——— (*)

- Error/residual:

$$e_i = \underbrace{f(x_i)}_{\text{measured}} - \underbrace{a_0(1-e^{-a_1 x})}_{\text{model}}$$

- Sum of squares of the residuals:

$$S_r = \sum_{i=1}^{5} e_i^2$$

$$S_r\Big|_{IG} = 0.0248$$

and $$S_r\Big|_{\circledast} = 0.0242$$

- Repeat the computations until stopping criterion is satisfied.

□

Ordinary Differential Equation:

Q. Given $y' = \frac{dy}{dx} = f(x, y)$, find $y(x)$.

Euler's Method:

- An iterative method:

  Given (x_i, y_i), $x_{i+1} = x_i + h$, $y_{i+1} = y_i + \phi h$

  where, ϕ is the estimated slope:

- In Euler's Method, first derivative is used to estimate the function slope, as:

$\phi = f(x_i, y_i)$ and $y_{i+1} = y_i + f(x_i, y_i) \cdot h$

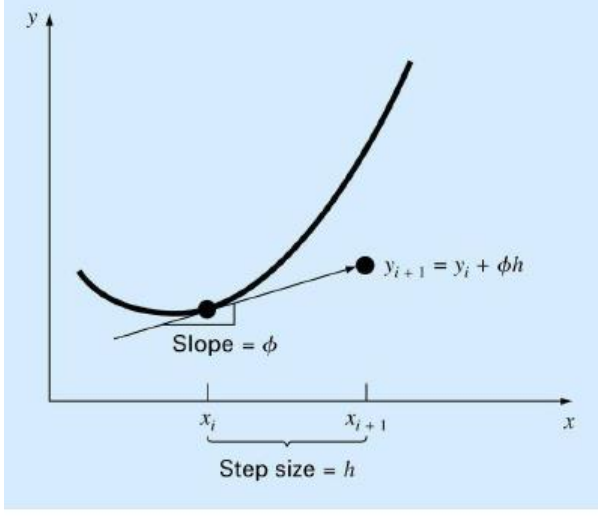


Figure 1: Illustration of iterative methods

- Use Taylor series expansion:

$$y_{i+1} = y_i + h y_i' + \frac{h^2}{2!} y_i'' + \cdots + \frac{h^n}{n!} y_i^{(n)} + R_n$$

with $h = x_{i+1} - x_i$ and $R_n = \frac{y^{(n+1)}(\alpha)}{(n+1)!} h^{n+1} = O(h^{n+1})$,

$$x_i < \alpha < x_{i+1}$$

$\because y' = f(x,y)$, we have $y_i' = f(x_i, y_i)$

$$y_i'' = f'(x_i, y_i)$$

$$\vdots$$

$$y_i^{(n)} = f^{(n-1)}(x_i, y_i)$$

$\therefore$

$$y_{i+1} = y_i + f(x_i, y_i)h + \frac{1}{2!} f'(x_i, y_i)h^2 + \cdots + \frac{1}{n!} f^{(n-1)}(x_i, y_i)h^n + O(h^{n+1})$$

Using Euler's method:

$$\boxed{y_{i+1} = y_i + f(x_i, y_i)h}$$

∴ The true local truncation error in using Euler's method:

$$E_t = \frac{1}{2} f'(x_i, y_i) h^2 + \cdots + \frac{1}{n!} f^{(n-1)}(x_i, y_i) h^n + O(h^{n+1})$$

Here, h is sufficiently small, higher powers of h can be neglected. Thus, approximated local truncation error is:

$$\boxed{E_a = \frac{1}{2} f'(x_i, y_i) h^2}$$

- E_a is proportional to h^2 and $f'(x_i, y_i)$
- Taylor series provides only the local truncation error.
- Global truncation error using Euler's method is proportional to the step-size, $O(h)$.
- Truncation error can be reduced by decreasing the step-size (h)
- Euler's method provides error free solution/prediction if the function $y(x)$ (that is soln. of diff. eqn) is linear.

□.

Example: Integrate the eq^n: $\frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5$

from $x = 0$ to $x = 1$

Using Euler's method with a step-size $h = 0.5$ and 0.25.

The initial condition: at $x = 0$, $y = 1$

Euler's method with $h = 0.5$:

$x_1 = x_0 + h = 0.5$

$y_1 = y_0 + f(x_0, y_0)\,h = 1 + f(0,1) * 0.5 = 5.25$

$x_2 = x_1 + h = 1.0$

$y_2 = y_1 + f(x_1, y_1)\,h = 5.25 + f(0.5, 5.25) * 0.5 = 5.875$

(A)

Euler's method with $h = 0.25$:

$x_1 = x_0 + h = 0.25$

$y_1 = y_0 + f(x_0, y_0)\,h = 1 + f(0,1) * 0.25 = 3.1250$

$x_2 = x_1 + h = 0.5$

$y_2 = y_1 + f(x_1, y_1)\,h = 3.1250 + f(0.25, 3.1250) * 0.25 = 4.1797$

$x_3 = x_2 + h = 0.75$ | $x_4 = x_3 + h = 1.0$

$y_3 = 4.4922$ | $y_4 = 4.3438$

(B)

— Exact soln.

$$y = \int \left(-2x^3 + 12x^2 - 20x + 8.5\right) dx$$

$$= -\frac{1}{2}x^4 + 4x^3 - 10x^2 + 8.5x + C$$

Using IC: $x = 0,\ y = 1$

$$\Rightarrow y = -\frac{1}{2}x^4 + 4x^3 - 10x^2 + 8.5x + 1 \quad \text{——} \quad (C)$$

$$\therefore \boxed{y(1) = 3}$$

Percent relative error (with $h = 0.5$):

$$\varepsilon_t = \left| \frac{\text{true value} - \text{approximate value}}{\text{true value}} \right| * 100\ \%$$

$= 95.8\ \%$

Percent relative error (with $h = 0.25$)

$$\varepsilon_t = 44.79\ \%$$

$\Rightarrow$ Reducing step-size can reduce the approximation error.

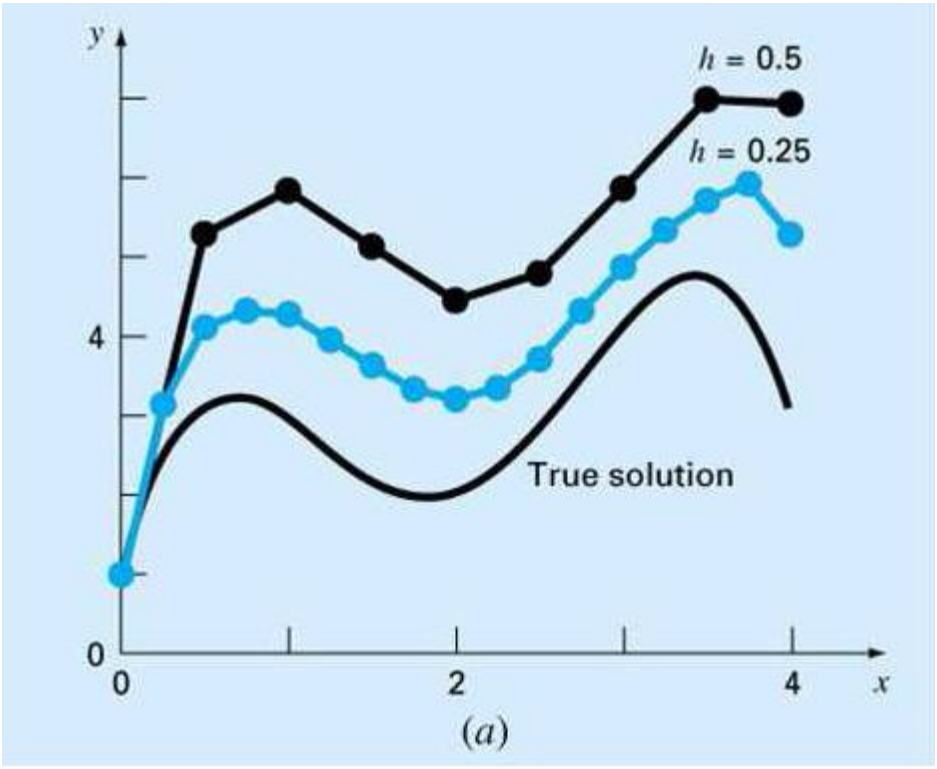


Figure 2: Euler's method for $\frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5$

Numerical Methods

— Higher-order Taylor Series Method:

Using second-order Taylor series:

$$y_{i+1} = y_i + y_i' h + \frac{y_i''}{2} h^2 = y_i + f(x_i, y_i) h + \frac{1}{2} f'(x_i, y_i) h^2$$

where $$f'(x,y) = \frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y} \frac{dy}{dx}$$ (chain-rule)

Local truncation error:

$$E_a = \frac{1}{3!} f''(x_i, y_i) h^3$$

where $$f''(x,y) = \frac{\partial f'}{\partial x} + \frac{\partial f'(x,y)}{\partial y} \frac{dy}{dx}$$

Note: $f'(x,y)$ and $f''(x,y)$ may be difficult to evaluate for complicated functions.

□

Runge - Kutta methods :

Runge - Kutta (RK) methods can achieve the accuracy of higher Taylor series BUT avoid evaluating the higher order derivatives.

The general form of RK method is

$$y_{i+1} = y_i + \phi(x_i, y_i, h)\,h$$

Where, $\phi(x_i, y_i, h)$ is called an increment function and is written in general form as:

$$\phi = a_1 k_1 + a_2 k_2 + \cdots + a_n k_n$$

Where,

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + p_1 h,\ y_i + q_{11} h k_1)$$

$$k_3 = f(x_i + p_2 h,\ y_i + q_{21} k_1 h + q_{22} k_2 h)$$

$$\vdots$$

$$k_n = f(x_i + p_{n-1} h,\ y_i\, q_{n-1,1} k_1 h + q_{n-2,2} k_2 h + \cdots + q_{n-1,n-1} k_{n-1} h)$$

— Various types of RK methods can be devised by choosing different number of terms in ϕ and different values of the parameters a's, p's and q's.

First-Order RK methods:

$n=1$, letting $a_1 = 1$, we have, $\phi(x_i, y_i, h) = a_1 k_1 = k_1$

$$\Rightarrow y_{i+1} = y_i + f(x_i, y_i)\,h$$ — Euler's Method

↓

first-order RK method.

Second-order RK methods:

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h$$

where

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + p_1 h,\ y_i + q_{11} k_1 h)$$

Q: How to find constants a_1, a_2, p_1 and q_{11}?

Use Taylor series:

$$y_{i+1} = y_i + y_i' h + \frac{1}{2} y_i'' h^2 + \underbrace{H.O.T}_{\text{(ignore them)}}$$

$$= y_i + f(x_i, y_i) h + \frac{1}{2} f'(x_i, y_i) h^2$$

$$= y_i + f(x_i, y_i) h + \frac{1}{2} \left[\frac{\partial f(x_i, y_i)}{\partial x} + \frac{\partial f(x_i, y_i)}{\partial y} y_i' \right] h^2 \quad \text{——— (1)}$$

Using 2nd - order RK method:

$$y_{i+1} = y_i + a_1 k_1 h + a_2 k_2 h$$

$$= y_i + a_1 f(x_i, y_i) h + a_2 k_2 h \quad \text{——— (2)}$$

Where,

$$k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h)$$

$$= f(x_i, y_i) + \frac{\partial f(x_i, y_i)}{\partial x} p_1 h + \frac{\partial f(x_i, y_i)}{\partial y} q_{11} k_1 h + H.O.T$$

$$= f(x_i, y_i) + \frac{\partial f(x_i, y_i)}{\partial x} p_1 h + \frac{\partial f(x_i, y_i)}{\partial y} q_{11} f(x_i, y_i) h \quad \text{——— (3)}$$

Substitute k_2 in ② by ③:

$$y_{i+1} = y_i + (a_1 + a_2) f(x_i, y_i) h + a_2 \frac{\partial f(x_i, y_i)}{\partial x} p_1 h^2 + a_2 \frac{\partial f(x_i, y_i)}{\partial y} q_{11} f(x_i, y_i) h^2 \quad \text{——— ④}$$

Comparing ④ and ①:

$$\left.\begin{aligned} a_1 + a_2 &= 1 \\ a_2 p_1 &= 1/2 \\ a_2 q_{11} &= 1/2 \end{aligned}\right\}$$ Three equations in Four unknowns. ⇒ infinite no. of sol's are possible.

A.) Heun Method:

$a_1 = 1/2$, $a_2 = 1/2$, and $p_1 = q_{11} = 1$

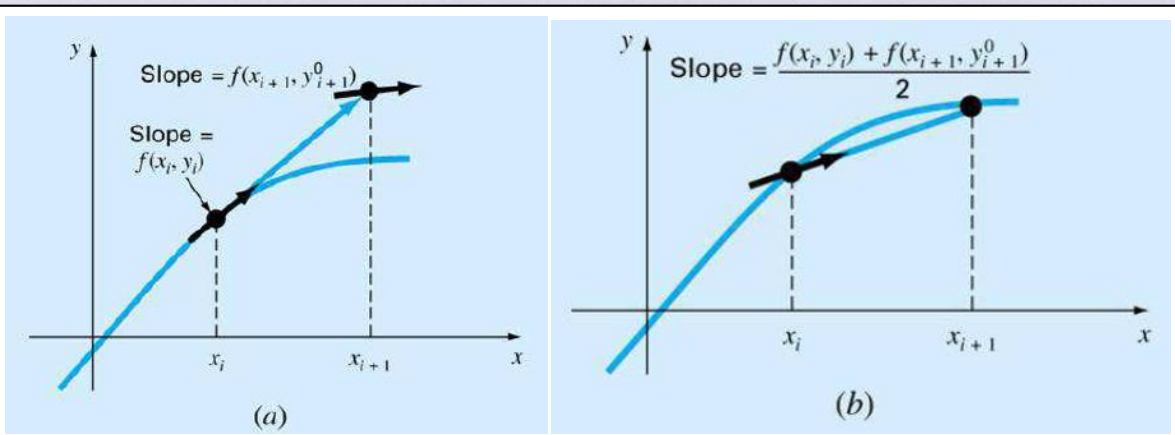
$$y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2)$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + h, y_i + k_1 h)$$

– Heun method can be represented in a predictor-corrector approach:

- Predictor: $y^0_{i+1} = y_i + f(x_i, y_i)h$
- Corrector: $y_{i+1} = y_i + \left(\frac{f(x_i, y_i) + f(x_{i+1}, y^0_{i+1})}{2}\right)h$



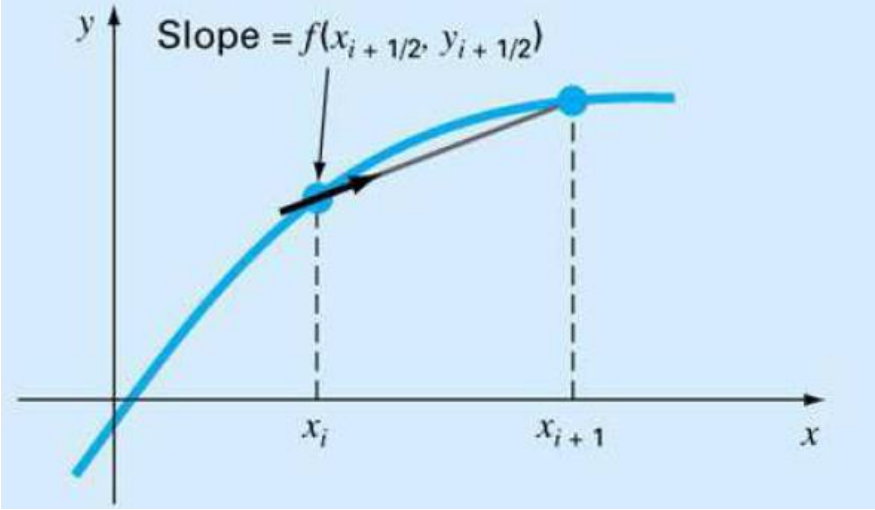
Heun method: (a) Predictor; (b) Corrector

B.) The Midpoint method $a_1 = 0$, $a_2 = 1$, and $p_1 = q_{11} = 1/2$

$$y_{i+1} = y_i + k_2 h$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{k_1 h}{2}\right)$$



Midpoint Method

Numerical Methods

— Fourth-order Runge-Kutta (RK) methods:

— Fourth-order RK methods have the following general form:

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2 + a_3 k_3 + a_4 k_4) h$$

Similar to second-order RK methods, there are infinite number of versions of 4th-order RK methods. The most commonly used form is:

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) h$$

Where,

$$k_1 = f(x_i, y_i)$$

$$k_2 = f\left(x_i + \frac{1}{2}h,\ y_i + \frac{1}{2}k_1 h\right)$$

$$k_3 = f\left(x_i + \frac{1}{2}h,\ y_i + \frac{1}{2}k_2 h\right)$$

$$k_4 = f(x_i + h,\ y_i + k_3 h)$$

↳ RK_4 method.

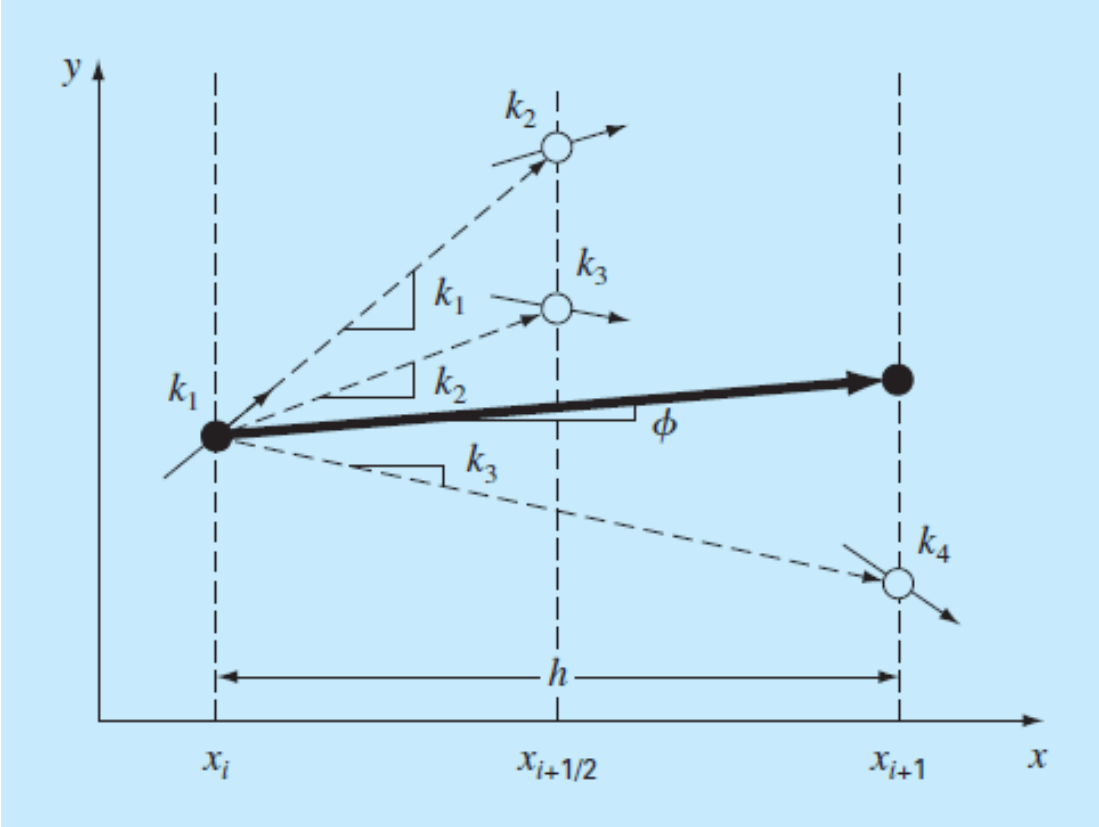


Fig. Slope estimates in the fourth-order RK method.

Example: $f(x,y) = -2x^3 + 12x^2 - 20x + 8.5$

Solve $\begin{cases} y' = f(x,y) \\ \text{with } y(0) = 1 \end{cases}$

Using RK_4 method with $h = 0.5$.

—

$i = 0, \quad x_0 = 0, \quad y_0 = 1$

$k_1 = f(x_0, y_0) = f(0,1) = 8.5$

$k_2 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{1}{2}k_1 h\right) = f(0.25, 3.125) = 4.21875$

$k_3 = f\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2 h\right) = f(0.25, 2.0547) = 4.21875$

$k_4 = f(x_0 + h, y_0 + k_3 h) = f(0.25, 3.10937) = 1.25$

$\Rightarrow x_1 = x_0 + h = 0.5$

$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) = 3.21875$

□

Also, $y|_{0.5} = 3.21875$ — Exat value.

This is because $y(x)$ is a 4^{th}-order polynomial
$\Rightarrow 4^{th}$-order RK methods give exact solution.

— In general: n^{th} – order RK methods

A.) Accurate to n^{th} – order polynomial.
B.) Equivalent to n^{th} – order Taylor series approx.
C.) Does not require to evaluate derivatives.

— System of ODEs:

For a system of simultaneous ODEs like:

$$\frac{dy_1}{dx} = f_1(x, y_1, y_2, \cdots, y_n)$$

$$\frac{dy_2}{dx} = f_2(x, y_1, y_2, \cdots, y_n)$$

$$\vdots$$

$$\frac{dy_n}{dx} = f_n(x, y_1, y_2, \cdots, y_n)$$

} sol. requires n initial conditions.

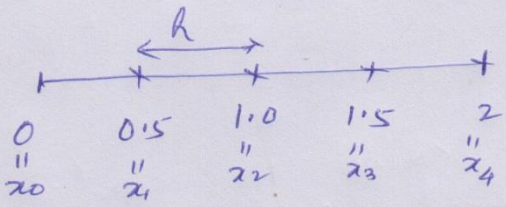
— RK methods discussed for a single ODE can be extended to system of ODEs.

Example 2: Use Euler's method to solve

$$\left.\begin{aligned} \frac{dy_1}{dx} &= -0.5\, y_1 \\ \frac{dy_2}{dx} &= 4 - 0.3\, y_2 - 0.1\, y_1 \end{aligned}\right\}$$

with $y_1(0) = 4$ and $y_2(0) = 6$.

Find solution at $x = 2$ with a step-size (h) $= 0.5$.



$$y_{i+1,1} = y_{i,1} + f_1(x_i, y_{i,1}, y_{i,2})$$

$$y_{i+1,2} = y_{i,2} + f_2(x_i, y_{i,1}, y_{i,2})$$

Here, $\left.\begin{aligned} f_1(x, y_1, y_2) &= -0.5\, y_1 \\ f_2(x, y_1, y_2) &= 4 - 0.3\, y_2 - 0.1\, y_1 \end{aligned}\right\}$

- $i = 0$, $x_1 = x_0 + h = 0.5$

$$\begin{cases} y_{2,1} = y_{0,1} + f_1(x_0, y_{0,1}, y_{0,2})h = 4 + f_1(0, 4, 6) = 3 \\ y_{2,2} = y_{0,2} + f_2(x_0, y_{0,1}, y_{0,2})h = 6 + f_2(0, 4, 6) = 6.9 \end{cases}$$

□

$i=1, \quad x_2 = x_1 + h = 1.0$

$$\begin{cases} y_{2,1} = y_{1,1} + f_1(x_1, y_{1,1}, y_{1,2})h = 3 + f_1(0.5, 3, 6.9) = 2.25 \\ y_{2,2} = y_{1,2} + f_2(x_1, y_{1,1}, y_{1,2})h = 6.9 + f_2(0.5, 3, 6.9) = 7.715 \end{cases}$$

$\vdots$

$y_{4,1} = \cdots$

$y_{4,2} = \cdots$

□

Q. Solve the prob. in Example 2 using RK_4 method.

$$\begin{cases} y_{i+1,1} = y_{i,1} + \frac{1}{6}(k_{1,1} + 2k_{2,1} + 2k_{3,1} + k_{4,1})h \\ y_{i+1,2} = y_{i,2} + \frac{1}{6}(k_{1,2} + 2k_{2,2} + 2k_{3,2} + k_{4,2})h \end{cases}$$

where,

$$\begin{aligned} k_{1,1} &= f_1(x_i, y_{i,1}, y_{i,2}) \\ k_{2,1} &= f_1(x_i + \frac{1}{2}h, y_{i,1} + \frac{1}{2}k_{1,1}h, y_{i,2} + \frac{1}{2}k_{1,2}h) \\ k_{3,1} &= f_1(x_i + \frac{1}{2}h, y_{i,1} + \frac{1}{2}k_{2,1}h, y_{i,2} + \frac{1}{2}k_{2,2}h) \\ k_{4,1} &= f_1(x_i + h, y_{i,1} + k_{3,1}h, y_{i,2} + k_{3,2}h) \end{aligned}$$

$$\begin{aligned} k_{1,2} &= f_2(x_i, y_{i,1}, y_{i,2}) \\ k_{2,2} &= f_2(x_i + \frac{1}{2}h, y_{i,1} + \frac{1}{2}k_{1,1}h, y_{i,2} + \frac{1}{2}k_{1,2}h) \\ k_{3,2} &= f_2(x_i + \frac{1}{2}h, y_{i,1} + \frac{1}{2}k_{2,1}h, y_{i,2} + \frac{1}{2}k_{2,2}h) \\ k_{4,2} &= f_2(x_i + h, y_{i,1} + k_{3,1}h, y_{i,2} + k_{3,2}h) \end{aligned}$$

Solution. First, we must solve for all the slopes at the beginning of the interval:

$$k_{1,1} = f_1(0, 4, 6) = -0.5(4) = -2$$
$$k_{1,2} = f_2(0, 4, 6) = 4 - 0.3(6) - 0.1(4) = 1.8$$

where $k_{i,j}$ is the ith value of k for the jth dependent variable. Next, we must calculate the first values of y_1 and y_2 at the midpoint:

$$y_1 + k_{1,1}\frac{h}{2} = 4 + (-2)\frac{0.5}{2} = 3.5$$

$$y_2 + k_{1,2}\frac{h}{2} = 6 + (1.8)\frac{0.5}{2} = 6.45$$

which can be used to compute the first set of midpoint slopes,

$$k_{2,1} = f_1(0.25, 3.5, 6.45) = -1.75$$
$$k_{2,2} = f_2(0.25, 3.5, 6.45) = 1.715$$

These are used to determine the second set of midpoint predictions,

$$y_1 + k_{2,1}\frac{h}{2} = 4 + (-1.75)\frac{0.5}{2} = 3.5625$$

$$y_2 + k_{2,2}\frac{h}{2} = 6 + (1.715)\frac{0.5}{2} = 6.42875$$

which can be used to compute the second set of midpoint slopes,

$$k_{3,1} = f_1(0.25, 3.5625, 6.42875) = -1.78125$$
$$k_{3,2} = f_2(0.25, 3.5625, 6.42875) = 1.715125$$

These are used to determine the predictions at the end of the interval

$$y_1 + k_{3,1}h = 4 + (-1.78125)(0.5) = 3.109375$$
$$y_2 + k_{3,2}h = 6 + (1.715125)(0.5) = 6.857563$$

which can be used to compute the endpoint slopes,

$$k_{4,1} = f_1(0.5, 3.109375, 6.857563) = -1.554688$$
$$k_{4,2} = f_2(0.5, 3.109375, 6.857563) = 1.631794$$

The values of k can then be used to compute [Eq. (25.40)]:

$$y_1(0.5) = 4 + \frac{1}{6}[-2 + 2(-1.75 - 1.78125) - 1.554688]0.5 = 3.115234$$

$$y_2(0.5) = 6 + \frac{1}{6}[1.8 + 2(1.715 + 1.715125) + 1.631794]0.5 = 6.857670$$

Proceeding in a like manner for the remaining steps yields

x	y_1	y_2
0	4	6
0.5	3.115234	6.857670
1.0	2.426171	7.632106
1.5	1.889523	8.326886
2.0	1.471577	8.946865

***

Numerical Methods

— Fourth-order Runge-Kutta (RK) methods:

— Fourth-order RK methods have the following general form:

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2 + a_3 k_3 + a_4 k_4) h$$

Similar to second-order RK methods, there are infinite number of versions of 4th-order RK methods. The most commonly used form is:

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) h$$

Where,

$$k_1 = f(x_i, y_i)$$

$$k_2 = f\left(x_i + \frac{1}{2}h,\ y_i + \frac{1}{2}k_1 h\right)$$

$$k_3 = f\left(x_i + \frac{1}{2}h,\ y_i + \frac{1}{2}k_2 h\right)$$

$$k_4 = f(x_i + h,\ y_i + k_3 h)$$

↳ RK_4 method.

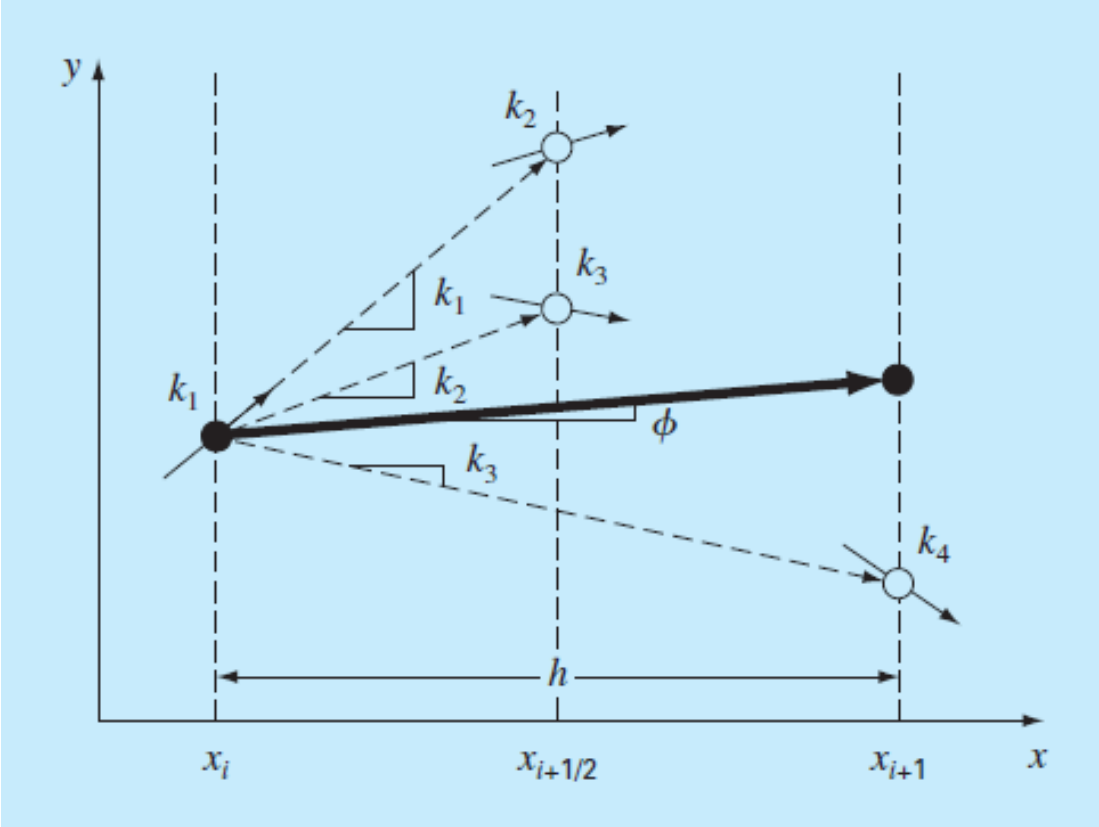


Fig. Slope estimates in the fourth-order RK method.

Example: $f(x,y) = -2x^3 + 12x^2 - 20x + 8.5$

Solve $\begin{cases} y' = f(x,y) \\ \text{with } y(0) = 1 \end{cases}$

Using RK_4 method with $h = 0.5$.

—

$i = 0, \quad x_0 = 0, \quad y_0 = 1$

$k_1 = f(x_0, y_0) = f(0,1) = 8.5$

$k_2 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{1}{2}k_1 h\right) = f(0.25, 3.125) = 4.21875$

$k_3 = f\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2 h\right) = f(0.25, 2.0547) = 4.21875$

$k_4 = f(x_0 + h, y_0 + k_3 h) = f(0.25, 3.10937) = 1.25$

$\Rightarrow \quad x_1 = x_0 + h = 0.5$

$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) = 3.21875$

□

Also, $y|_{0.5} = 3.21875$ — Exat value.

This is because $y(x)$ is a 4^{th}-order polynomial
$\Rightarrow 4^{th}$-order RK methods give exact solution.

— In general: n^{th} – order RK methods

A.) Accurate to n^{th} – order polynomial.
B.) Equivalent to n^{th} – order Taylor series approx.
C.) Does not require to evaluate derivatives.

— System of ODEs:

For a system of simultaneous ODEs like:

$$\frac{dy_1}{dx} = f_1(x, y_1, y_2, \cdots, y_n)$$

$$\frac{dy_2}{dx} = f_2(x, y_1, y_2, \cdots, y_n)$$

$$\vdots$$

$$\frac{dy_n}{dx} = f_n(x, y_1, y_2, \cdots, y_n)$$

} sol. requires n initial conditions.

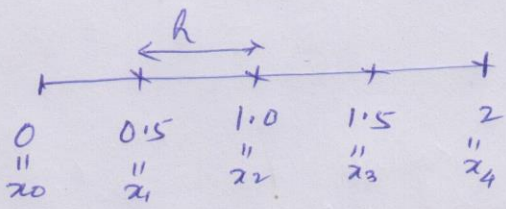
— RK methods discussed for a single ODE can be extended to system of ODEs.

Example 2: Use Euler's method to solve

$$\left.\begin{aligned} \frac{dy_1}{dx} &= -0.5\,y_1 \\ \frac{dy_2}{dx} &= 4 - 0.3\,y_2 - 0.1\,y_1 \end{aligned}\right\}$$

with $y_1(0) = 4$ and $y_2(0) = 6$.

Find solution at $x = 2$ with a step-size (h) $= 0.5$.



$$y_{i+1,1} = y_{i,1} + f_1(x_i, y_{i,1}, y_{i,2})$$

$$y_{i+1,2} = y_{i,2} + f_2(x_i, y_{i,1}, y_{i,2})$$

Here, $\left.\begin{aligned} f_1(x, y_1, y_2) &= -0.5\,y_1 \\ f_2(x, y_1, y_2) &= 4 - 0.3\,y_2 - 0.1\,y_1 \end{aligned}\right\}$

- $i = 0$, $x_1 = x_0 + h = 0.5$

$$\begin{cases} y_{2,1} = y_{0,1} + f_1(x_0, y_{0,1}, y_{0,2})h = 4 + f_1(0, 4, 6) = 3 \\ y_{2,2} = y_{0,2} + f_2(x_0, y_{0,1}, y_{0,2})h = 6 + f_2(0, 4, 6) = 6.9 \end{cases}$$

□

$i=1, \quad x_2 = x_1 + h = 1.0$

$$\begin{cases} y_{2,1} = y_{1,1} + f_1(x_1, y_{1,1}, y_{1,2})h = 3 + f_1(0.5, 3, 6.9) = 2.25 \\ y_{2,2} = y_{1,2} + f_2(x_1, y_{1,1}, y_{1,2})h = 6.9 + f_2(0.5, 3, 6.9) = 7.715 \end{cases}$$

$\vdots$

$y_{4,1} = \cdots$

$y_{4,2} = \cdots$

□

Q. Solve the prob. in Example 2 using RK_4 method.

$$\begin{cases} y_{i+1,1} = y_{i,1} + \frac{1}{6}(k_{1,1} + 2k_{2,1} + 2k_{3,1} + k_{4,1})h \\ y_{i+1,2} = y_{i,2} + \frac{1}{6}(k_{1,2} + 2k_{2,2} + 2k_{3,2} + k_{4,2})h \end{cases}$$

where,

$$\begin{cases} k_{1,1} = f_1(x_i, y_{i,1}, y_{i,2}) \\ k_{2,1} = f_1\left(x_i + \frac{h}{2}, y_{i,1} + \frac{k_{1,1}h}{2}, y_{i,2} + \frac{k_{1,2}h}{2}\right) \\ k_{3,1} = f_1\left(x_i + \frac{h}{2}, y_{i,1} + \frac{k_{2,1}h}{2}, y_{i,2} + \frac{k_{2,2}h}{2}\right) \\ k_{4,1} = f_1\left(x_i + \frac{h}{2}, y_{i,1} + \frac{k_{3,1}h}{2}, y_{i,2} + \frac{k_{3,2}h}{2}\right) \end{cases}$$

$$\begin{cases} k_{1,2} = f_2(x_i, y_{i,1}, y_{i,2}) \\ k_{2,2} = f_2\left(x_i + \frac{h}{2}, y_{i,1} + \frac{k_{1,1}h}{2}, y_{i,2} + \frac{k_{1,2}h}{2}\right) \\ k_{3,2} = f_2\left(x_i + \frac{h}{2}, y_{i,1} + \frac{k_{2,1}h}{2}, y_{i,2} + \frac{k_{2,2}h}{2}\right) \\ k_{4,2} = f_2\left(x_i + \frac{h}{2}, y_{i,1} + \frac{k_{3,1}h}{2}, y_{i,2} + \frac{k_{3,2}h}{2}\right) \end{cases}$$

Solution. First, we must solve for all the slopes at the beginning of the interval:

$$k_{1,1} = f_1(0, 4, 6) = -0.5(4) = -2$$
$$k_{1,2} = f_2(0, 4, 6) = 4 - 0.3(6) - 0.1(4) = 1.8$$

where $k_{i,j}$ is the ith value of k for the jth dependent variable. Next, we must calculate the first values of y_1 and y_2 at the midpoint:

$$y_1 + k_{1,1}\frac{h}{2} = 4 + (-2)\frac{0.5}{2} = 3.5$$

$$y_2 + k_{1,2}\frac{h}{2} = 6 + (1.8)\frac{0.5}{2} = 6.45$$

which can be used to compute the first set of midpoint slopes,

$$k_{2,1} = f_1(0.25, 3.5, 6.45) = -1.75$$
$$k_{2,2} = f_2(0.25, 3.5, 6.45) = 1.715$$

These are used to determine the second set of midpoint predictions,

$$y_1 + k_{2,1}\frac{h}{2} = 4 + (-1.75)\frac{0.5}{2} = 3.5625$$

$$y_2 + k_{2,2}\frac{h}{2} = 6 + (1.715)\frac{0.5}{2} = 6.42875$$

which can be used to compute the second set of midpoint slopes,

$$k_{3,1} = f_1(0.25, 3.5625, 6.42875) = -1.78125$$
$$k_{3,2} = f_2(0.25, 3.5625, 6.42875) = 1.715125$$

These are used to determine the predictions at the end of the interval

$$y_1 + k_{3,1}h = 4 + (-1.78125)(0.5) = 3.109375$$
$$y_2 + k_{3,2}h = 6 + (1.715125)(0.5) = 6.857563$$

which can be used to compute the endpoint slopes,

$$k_{4,1} = f_1(0.5, 3.109375, 6.857563) = -1.554688$$
$$k_{4,2} = f_2(0.5, 3.109375, 6.857563) = 1.631794$$

The values of k can then be used to compute [Eq. (25.40)]:

$$y_1(0.5) = 4 + \frac{1}{6}[-2 + 2(-1.75 - 1.78125) - 1.554688]0.5 = 3.115234$$

$$y_2(0.5) = 6 + \frac{1}{6}[1.8 + 2(1.715 + 1.715125) + 1.631794]0.5 = 6.857670$$

Proceeding in a like manner for the remaining steps yields

x	y_1	y_2
0	4	6
0.5	3.115234	6.857670
1.0	2.426171	7.632106
1.5	1.889523	8.326886
2.0	1.471577	8.946865

***

Numerical Methods

- All previous methods discussed so far to solve the IVP are one-step methods which utilize information at a single point x_i to predict a value of dep. variable y_{i+1} at a future point x_{i+1}. The multistep methods are based on the insight that, once the computation has begun, information from previous points can be used to estimate the function at a future point.

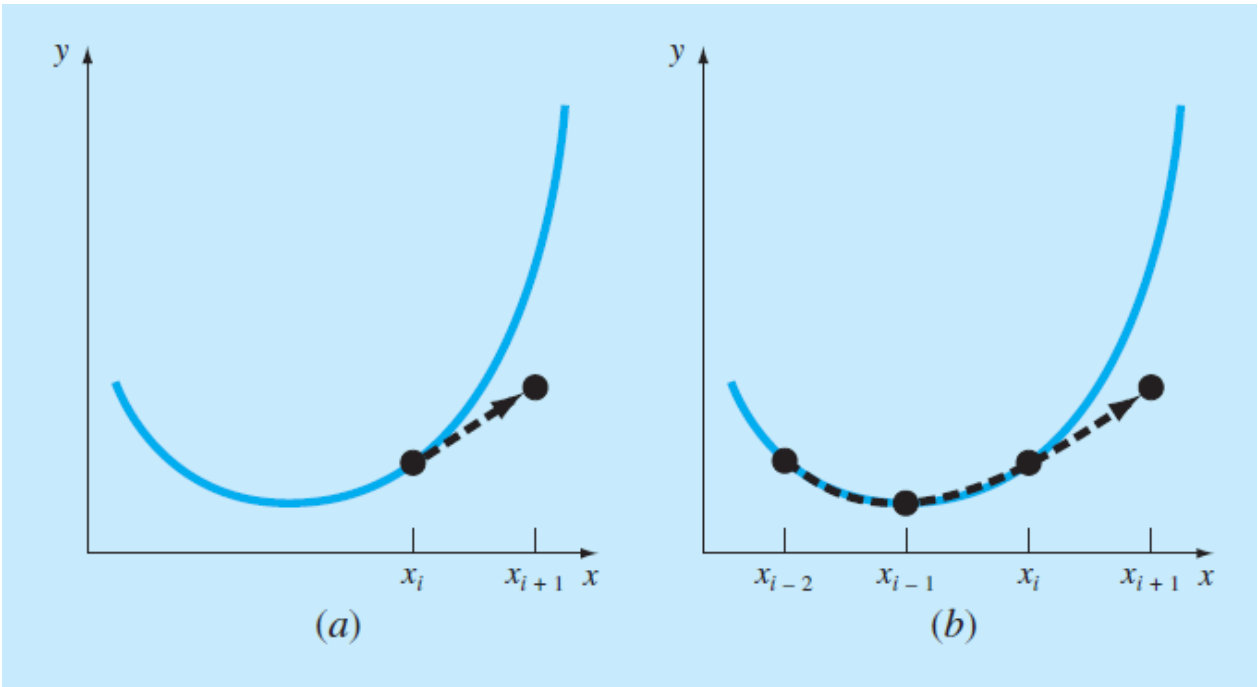


Fig. 1: Graphical depiction of the fundamental difference between (*a*) one-step and (*b*) multistep methods for solving ODEs.

The non-self-Starting Heun Method:

- Recall (from Lecture-21) that Heun approach uses Euler's method as a predictor:

$$y_{i+1}^0 = y_i + f(x_i, y_i)h \quad \text{——} (1)$$

And the trapezoidal rule as a corrector:

$$y_{i+1} = y_i + \left[\frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2}\right]h \quad \text{——} (2)$$

Note that here, predictor step has local truncation error of $O(h^2)$ and corrector step has local trunc. error of $O(h^3)$

$\Rightarrow$ Predictor is weak.

— We can improve Heun's method by devising a predictor that has a local truncation error of $O(h^3)$

New predictor:

$$y_{i+1}^0 = y_{i-1} + f(x_i, y_i)2h \quad \text{——} (3)$$

which have trunc. error of $O(h^3)$.

- Note that Eq. (3) is not self-starting, as it involves y_{i-1} (which would not be available). Due to this reason, Eqs. (2) and (3) are called as non-self-starting Heun method:

$$\text{Predictor: } y_{i+1}^{0} = y_{i-1}^{m} + f(x_i, y_i^m) \cdot 2h$$

$$\text{Corrector: } y_{i+1}^{j} = y_i^m + \left[\frac{f(x_i, y_i^m) + f(x_{i+1}, y_{i+1}^{j-1})}{2}\right] h$$

$$, \; j = 1, 2, \cdots, m$$

Where the corrector is applied iteratively from $j = 1$ to m to obtain refined solutions.

- The approximate percentage relative error is:

$$|\varepsilon_a| = \left| \frac{y_{i+1}^{j} - y_{i+1}^{j-1}}{y_{i+1}^{j}} \right| \times 100\%$$

- Termination criterion: $|\varepsilon_a| < \varepsilon_s$ (Tolerance)

□

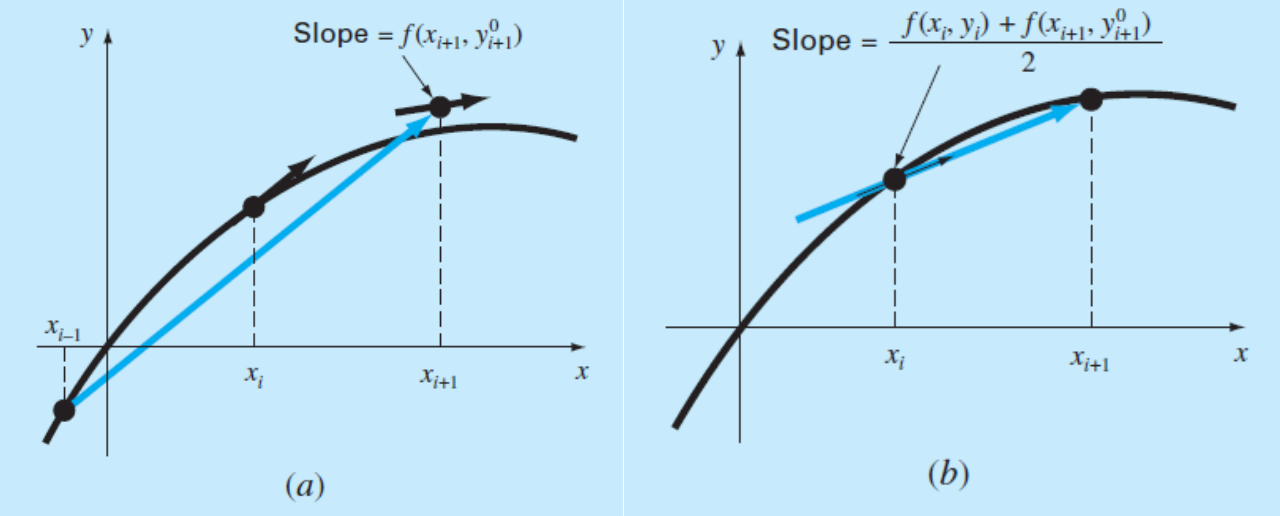


Fig. 2: Graphical depiction of the non-self-starting Heun method. (*a*) The midpoint method that is used as a predictor. (*b*) The trapezoidal rule that is employed as a corrector.

Example: Use the non-self-starting Heun method to integrate:

$$\left.\begin{aligned} y' &= 4e^{0.8x} - 0.5y \\ y(0) &= 2 \end{aligned}\right\}$$

with step-size $h = 1.0$. Additional information is required for the multistep method: $y = -0.3929953$ at $x = 1.0$.

Sol^:

$$x_{-1} = -1, \quad y_{-1} = -0.3929953, \quad x_0 = 0, \quad y_0 = 2$$

Step-I: $x_1 = x_0 + h$

Predictor step is used to extrapolate linearly from x_{-1} to x_1:

$$y_1^0 = y_{-1} + f(x_0, y_0)\,2h$$

$$= -0.3929953 + \left(4e^{0.8\times 0} - 0.5\times 2\right)\times 2\times 1$$

$$= 5.607005$$

The corrector is then used to compute the value:

When $j = 1$:

$$y_1^1 = y_0 + \left[\frac{f(x_0, y_0) + f(x_1, y_1^0)}{2}\right] h$$

$$= 6.549331$$

Approximate percentage relative error:

$$\varepsilon_a = \left|\frac{y_1^1 - y_1^0}{y_1^1}\right| \times 100\% = 14.39\%$$

- When $j = 2$:

$$y_1^2 = y_0 + \left[\frac{f(x_0, y_0) + f(x_1, y_1^1)}{2}\right] h$$

$$= 6.313749$$

Approximate percentage relative error:

$$\varepsilon_a = \left|\frac{y_1^2 - y_1^1}{y_1^2}\right| \times 100\% = 3.73\%$$

⋮

Step 2:

- Numerical Sol^{n} of Partial Differential Equations

- Three types of fundamental PDEs:

A) Elliptic: $\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$, $T = T(x, y)$

Laplace Equation —— (1)

B) Parabolic: $\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}$, $T = T(x, t)$

Heat Conduction Equation —— (2)

C) Hyperbolic: $\frac{\partial^2 U}{\partial t^2} = c^2 \frac{\partial^2 U}{\partial x^2}$, $U = U(x, t)$

—— (3)

Wave Equation

Elliptic PDEs: Numerical Sol^{n}

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \quad \text{(Laplace)}$$

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = f(x, y) \quad \text{(Poisson)}$$

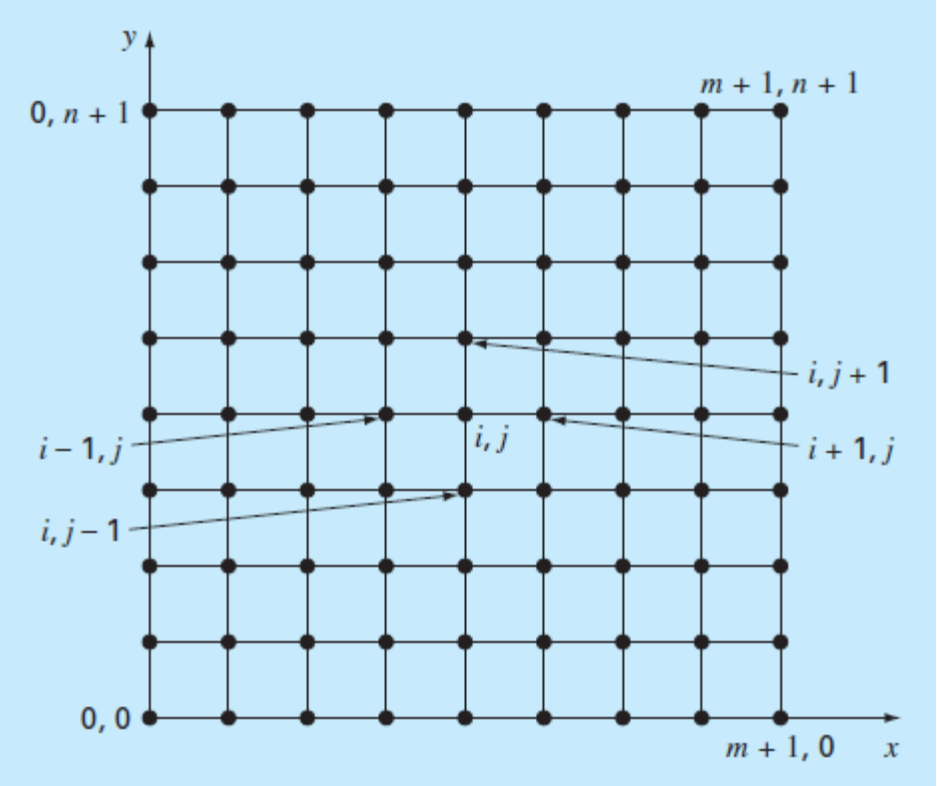


Fig. 1: A grid used for the finite-difference solution of elliptic PDEs in two independent variables- Laplace equation.

Consider 1D Laplace equation:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \qquad \text{——— (1)}$$

Use second-order centered difference formula to discretize eqn. (1)

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2} + O(\Delta x^2)$$

and $$\frac{\partial^2 T}{\partial y^2} = \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\Delta y^2} + O(\Delta y^2)$$

Substitute in Eq. (1):

$$\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\Delta y^2} = 0$$

If $\Delta x = \Delta y$

$$\Rightarrow \quad T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1} - 4T_{i,j} = 0 \qquad \text{——— (4)}$$

— Difference Equation

Boundary Conditions:

Boundary conditions along the domain boundaries must be specified to obtain a unique sol^n.

- If boundary condition is specified on the unknown (T)
  - Dirichlet Boundary Condition. ✓
- If boundary condition is specified on the derivative of the unknown (T)
  - Neumann Boundary Condition.

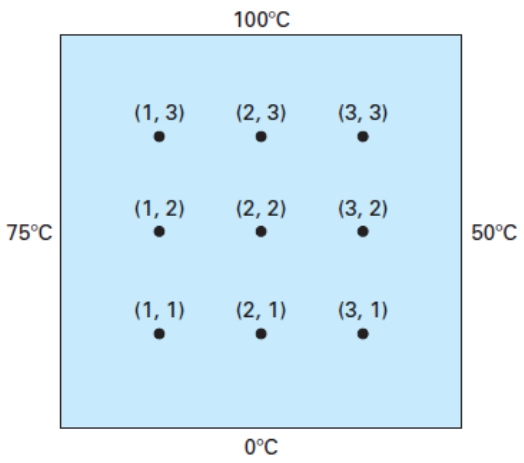


Fig. 2: A heated plate where boundary temperatures are held at constant levels- Dirichlet boundary condition.

$\therefore$ For node (1,1): (from (4)):

$$T_{2,1} + T_{0,1} + T_{1,2} + T_{1,0} - 4T_{1,1} = 0 \quad \text{(5)}$$

$\because T_{0,1} = 75$ and $T_{1,0} = 0 \Rightarrow -4T_{1,1} + T_{1,2} + T_{2,1} = -75$

- Similar equations can be developed for the other interior points.

- The resultant system of 9 simultaneous equations with 9 unknowns is given as:

$$
\begin{array}{rrrrrrrrrl}
4T_{11} & -T_{21} & & -T_{12} & & & & & & = 75 \\
-T_{11} & +4T_{21} & -T_{31} & & -T_{22} & & & & & = 0 \\
 & -T_{21} & +4T_{31} & & & -T_{32} & & & & = 50 \\
-T_{11} & & & +4T_{12} & -T_{22} & & -T_{13} & & & = 75 \\
 & -T_{21} & & -T_{12} & +4T_{22} & -T_{32} & & -T_{23} & & = 0 \\
 & & -T_{31} & & -T_{22} & +4T_{32} & & & -T_{33} & = 50 \\
 & & & -T_{12} & & & +4T_{13} & -T_{23} & & = 175 \\
 & & & & -T_{22} & & -T_{13} & +4T_{23} & -T_{33} & = 100 \\
 & & & & & -T_{32} & & -T_{23} & +4T_{33} & = 150
\end{array}
$$

OR

$$
\begin{bmatrix} \\ A \\ \\ \end{bmatrix}
\begin{bmatrix} T_{1,1} \\ T_{2,1} \\ T_{3,1} \\ T_{1,2} \\ T_{2,2} \\ T_{3,2} \\ T_{1,3} \\ T_{2,3} \\ T_{3,3} \end{bmatrix}
=
\begin{bmatrix} \\ b \\ \\ \end{bmatrix}
\qquad \text{—— (6)}
$$

- The Liebmann method (To solve system ⑥)

- based on Gauss-Seidel Method
- Here, Eq.(4) is expressed as:

$$T_{i,j} = \frac{T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}}{4} \quad \text{——⑦}$$

and ⑦ is solved iteratively for $j = 1$ to n
$i = 1$ to m

As, system ④ is diagonally dominant ⇒ convergence of iterative process.

- Sometimes Overrelaxation is employed to accelerate the rate of convergence by applying the following formula after each iteration:

$$\overline{T}_{i,j}^{new} = \lambda T_{i,j}^{new} + (1-\lambda) T_{i,j}^{old} \quad \text{——⑧}$$

where $T_{i,j}^{new}$ – present iteration value.

$T_{i,j}^{old}$ – previous iteration value.

λ – relaxation parameter and $\lambda \in (1,2)$

↓ overrelaxation

— Stopping Criterion:

$$\left|(\varepsilon_a)_{i,j}\right| = \left| \frac{T_{i,j}^{new} - T_{i,j}^{old}}{T_{i,j}^{new}} \right| * 100\ \%$$

If $\left|(\varepsilon_a)_{i,j}\right| \leq \varepsilon_s$

↓

user defined tolerance

□

Example 1: Use Liebmann's method (Gauss-Seidel) to solve the temperature of the heated plate (given in Fig. 2). Use overrelaxation with a value of 1.5 for the weighting factor and iterate to $\varepsilon_s = 1\%$.

From Eq. (7), at $(i,j) = (1,1)$:

$$T_{1,1} = \frac{0 + 75 + 0 + 0}{4} = 18.75$$

Apply overrelaxation (given in Eq. (8)):

$$T_{1,1} = 1.5(18.75) + (1 - 1.5)0 = 28.125$$

Similarly, at $(i,j) = (2,1)$:

$$T_{2,1} = \frac{0 + 28.125 + 0 + 0}{4} = 7.03125$$

$$T_{2,1} = 1.5(7.03125) + (1 - 1.5)0 = 10.54688$$

For $(i,j) = (3,1)$:

$$T_{3,1} = \frac{50 + 10.54688 + 0 + 0}{4} = 15.13672$$

$$T_{31} = 1.5(15.13672) + (1 - 1.5)0 = 22.70508.$$

The computation is repeated for the other rows to give

$$T_{12} = 38.67188 \qquad T_{22} = 18.45703 \qquad T_{32} = 34.18579$$
$$T_{13} = 80.12696 \qquad T_{23} = 74.46900 \qquad T_{33} = 96.99554$$

Because all the $T_{i,j}$'s are initially zero, all ε_a's for the first iteration will be 100%.

For the second iteration the results are

$$T_{11} = 32.51953 \qquad T_{21} = 22.35718 \qquad T_{31} = 28.60108$$
$$T_{12} = 57.95288 \qquad T_{22} = 61.63333 \qquad T_{32} = 71.86833$$
$$T_{13} = 75.21973 \qquad T_{23} = 87.95872 \qquad T_{32} = 67.68736$$

The error for $T_{1,1}$ can be estimated as [Eq. (29.13)]

$$|(\varepsilon_a)_{1,1}| = \left|\frac{32.51953 - 28.12500}{32.51953}\right| 100\% = 13.5\%$$

Because this value is above the stopping criterion of 1%, the computation is continued. The ninth iteration gives the result

$$T_{11} = 43.00061 \qquad T_{21} = 33.29755 \qquad T_{31} = 33.88506$$
$$T_{12} = 63.21152 \qquad T_{22} = 56.11238 \qquad T_{32} = 52.33999$$
$$T_{13} = 78.58718 \qquad T_{23} = 76.06402 \qquad T_{33} = 69.71050$$

where the maximum error is 0.71%.

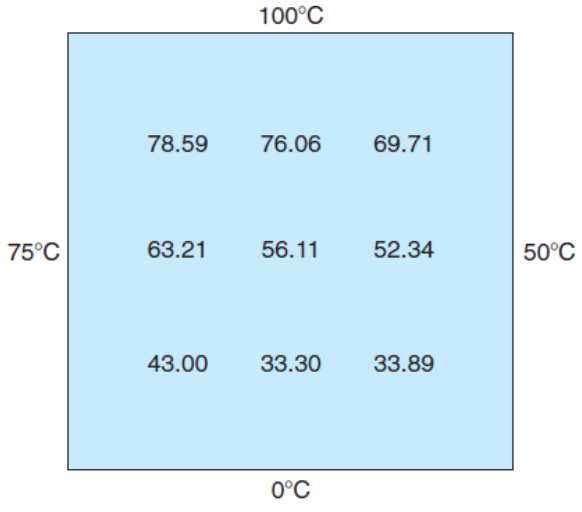


Fig. 3: Temperature distribution for a heated plate subject to fixed boundary conditions.

- Numerical Sol^{n} of Partial Differential Equations

- Three types of fundamental PDEs:

A) Elliptic: $\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$, $T = T(x, y)$

Laplace Equation —— (1)

B) Parabolic: $\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}$, $T = T(x, t)$

Heat Conduction Equation —— (2)

C) Hyperbolic: $\frac{\partial^2 U}{\partial t^2} = c^2 \frac{\partial^2 U}{\partial x^2}$, $U = U(x, t)$

—— (3)

Wave Equation

Elliptic PDEs: Numerical Sol^{n}

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \quad \text{(Laplace)}$$

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = f(x, y) \quad \text{(Poisson)}$$

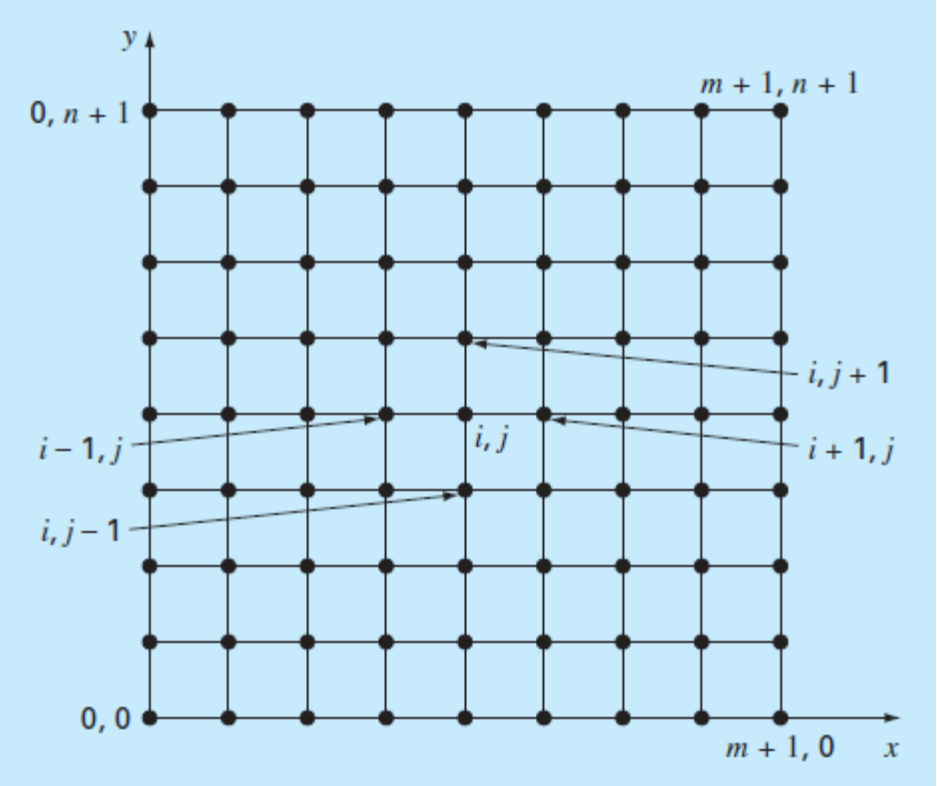


Fig. 1: A grid used for the finite-difference solution of elliptic PDEs in two independent variables- Laplace equation.

Consider 1D Laplace equation:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \qquad \text{——— (1)}$$

Use second-order centered difference formula to discretize eqn. (1)

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2} + O(\Delta x^2)$$

and $$\frac{\partial^2 T}{\partial y^2} = \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\Delta y^2} + O(\Delta y^2)$$

Substitute in Eq. (1):

$$\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\Delta y^2} = 0$$

If $\Delta x = \Delta y$

$$\Rightarrow T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1} - 4T_{i,j} = 0 \qquad \text{——— (4)}$$

— Difference Equation

Boundary Conditions:

Boundary conditions along the domain boundaries must be specified to obtain a unique sol^n.

- If boundary condition is specified on the unknown (T)
  - Dirichlet Boundary Condition. ✓
- If boundary condition is specified on the derivative of the unknown (T)
  - Neumann Boundary Condition.

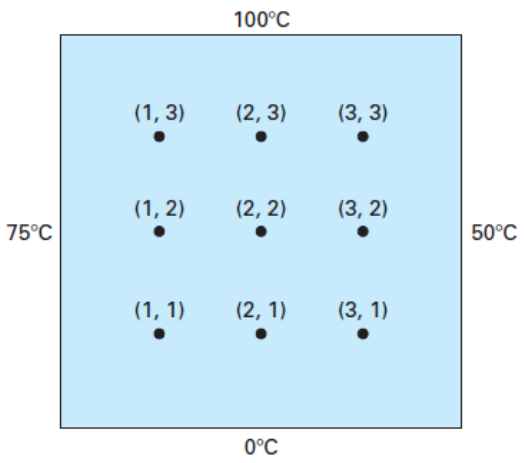


Fig. 2: A heated plate where boundary temperatures are held at constant levels- Dirichlet boundary condition.

$\therefore$ For node (1,1): (from (4)):

$$T_{2,1} + T_{0,1} + T_{1,2} + T_{1,0} - 4T_{1,1} = 0 \quad \text{(5)}$$

$\because T_{0,1} = 75$ and $T_{1,0} = 0 \Rightarrow -4T_{1,1} + T_{1,2} + T_{2,1} = -75$

- Similar equations can be developed for the other interior points.

- The resultant system of 9 simultaneous equations with 9 unknowns is given as:

$$
\begin{array}{rrrrrrrrrl}
4T_{11} & -T_{21} & & -T_{12} & & & & & & = 75 \\
-T_{11} & +4T_{21} & -T_{31} & & -T_{22} & & & & & = 0 \\
 & -T_{21} & +4T_{31} & & & -T_{32} & & & & = 50 \\
-T_{11} & & & +4T_{12} & -T_{22} & & -T_{13} & & & = 75 \\
 & -T_{21} & & -T_{12} & +4T_{22} & -T_{32} & & -T_{23} & & = 0 \\
 & & -T_{31} & & -T_{22} & +4T_{32} & & & -T_{33} & = 50 \\
 & & & -T_{12} & & & +4T_{13} & -T_{23} & & = 175 \\
 & & & & -T_{22} & & -T_{13} & +4T_{23} & -T_{33} & = 100 \\
 & & & & & -T_{32} & & -T_{23} & +4T_{33} & = 150
\end{array}
$$

OR

$$
\begin{bmatrix} \\ A \\ \\ \end{bmatrix}
\begin{bmatrix} T_{1,1} \\ T_{2,1} \\ T_{3,1} \\ T_{1,2} \\ T_{2,2} \\ T_{3,2} \\ T_{1,3} \\ T_{2,3} \\ T_{3,3} \end{bmatrix}
=
\begin{bmatrix} \\ b \\ \\ \end{bmatrix}
\qquad \text{—— (6)}
$$

- The Liebmann method (To solve system (6))

- based on Gauss-Seidel Method
- Here, Eq.(4) is expressed as:

$$T_{i,j} = \frac{T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1}}{4} \quad \text{——— (7)}$$

and (7) is solved iteratively for $j = 1$ to n
$i = 1$ to m

As, system (4) is diagonally dominant $\Rightarrow$ convergence of iterative process.

- Sometimes Overrelaxation is employed to accelerate the rate of convergence by applying the following formula after each iteration:

$$T_{i,j}^{new} = \lambda T_{i,j}^{new} + (1-\lambda) T_{i,j}^{old} \quad \text{——— (8)}$$

where $T_{i,j}^{new}$ – present iteration value.

$T_{i,j}^{old}$ – previous iteration value.

λ – relaxation parameter and $\lambda \in (0, 2)$

↓ overrelaxation

— Stopping Criterion:

$$\left|(\varepsilon_a)_{i,j}\right| = \left|\frac{T_{i,j}^{new} - T_{i,j}^{old}}{T_{i,j}^{new}}\right| * 100\ \%$$

If $\left|(\varepsilon_a)_{i,j}\right| \leq \varepsilon_s$

ε_s ↓ user defined tolerance

□

– Heat Conduction Equation:

$$\frac{\partial T}{\partial t} = k \frac{\partial^2 T}{\partial x^2} \qquad \text{k – thermal conductivity.}$$

——— (1)

– Explicit Numerical Methods:

$$\frac{\partial T}{\partial t} = k \frac{\partial^2 T}{\partial x^2}$$

Discretization:

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_i^{\ell} = \frac{T_{i+1}^{\ell} - 2T_i^{\ell} + T_{i-1}^{\ell}}{\Delta x^2} + \mathcal{O}(\Delta x^2)$$

——— (2)

$$\frac{\partial T}{\partial t} = \frac{T_i^{\ell+1} - T_i^{\ell}}{\Delta t} + \mathcal{O}(\Delta t)$$

——— (3)

Then from (1):

$$\frac{T_i^{\ell+1} - T_i^{\ell}}{\Delta t} = k\left(\frac{T_{i+1}^{\ell} - 2T_i^{\ell} + T_{i-1}^{\ell}}{\Delta x^2}\right)$$

OR,

$$\boxed{T_i^{\ell+1} = T_i^{\ell} + r\left(T_{i+1}^{\ell} - 2T_i^{\ell} + T_{i-1}^{\ell}\right)}$$

——— (4)

where $r = \frac{k\Delta t}{\Delta x^2}$

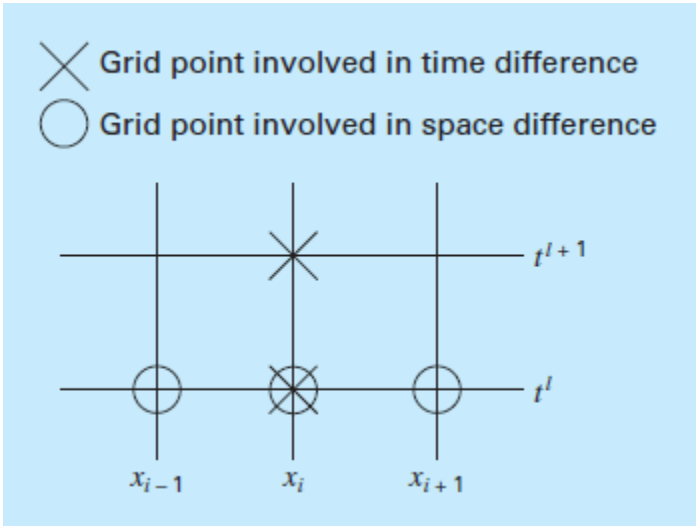
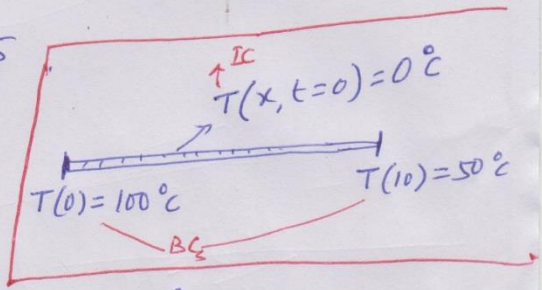


Fig. 1: Computational molecule for the explicit method

Example 1: Use explicit method to solve for the temperature distribution of a long, thin rod with a length 10 cm and the following values:

$k = 0.835\ cm^2/s$, $\Delta t = 0.1\ s$, $\Delta x = 2\ cm$

$$\Rightarrow d = \frac{k\Delta t}{\Delta x^2} = 0.020875$$



Use Eq. (4):

$$T_i^{l+1} = T_i^l + d\left(T_{i+1}^l - 2T_i^l + T_{i-1}^l\right)$$

$$T_1^1 = T_0^0 + d\left(T_2^0 - 2T_1^0 + T_0^0\right)$$

$$= 0 + 0.020875(0 - 2(0) + 100) = 2.0875$$

At the other interior points, $x = 4$, 6, and 8 cm, the results are

$$T_2^1 = 0 + 0.020875[0 - 2(0) + 0] = 0$$
$$T_3^1 = 0 + 0.020875[0 - 2(0) + 0] = 0$$
$$T_4^1 = 0 + 0.020875[50 - 2(0) + 0] = 1.0438$$

At $t = 0.2$ s, the values at the four interior nodes are computed as

$$T_1^2 = 2.0875 + 0.020875[0 - 2(2.0875) + 100] = 4.0878$$
$$T_2^2 = 0 + 0.020875[0 - 2(0) + 2.0875] = 0.043577$$
$$T_3^2 = 0 + 0.020875[1.0438 - 2(0) + 0] = 0.021788$$
$$T_4^2 = 1.0438 + 0.020875[50 - 2(1.0438) + 0] = 2.0439$$

- - -

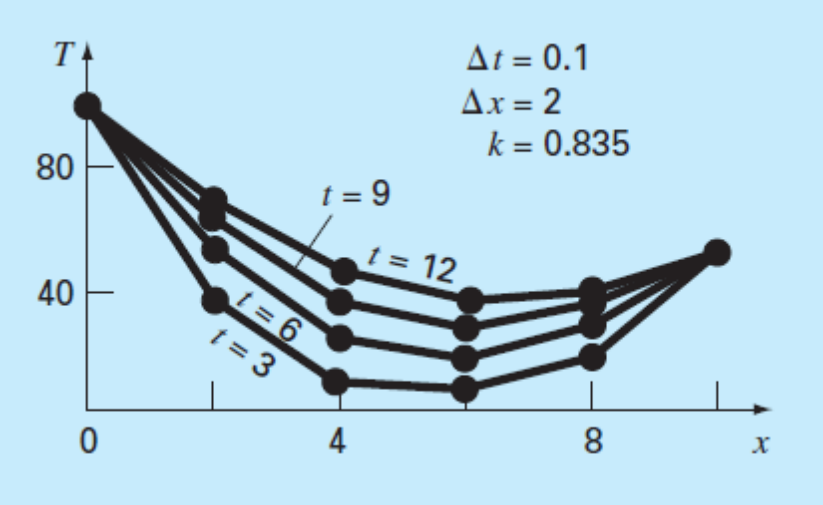


Fig. 2: Temperature distribution in a long, thin rod as computed with the explicit method.

Implicit Method :

$$\frac{\partial T}{\partial t} = k \frac{\partial^2 T}{\partial x^2} \qquad \text{——— (5)}$$

$$\left.\begin{aligned} \frac{\partial^2 T}{\partial x^2} &= \frac{T_{i+1}^{l+1} - 2T_i^{l+1} + T_{i-1}^{l+1}}{\Delta x^2} \\ \frac{\partial T}{\partial t} &= \frac{T_i^{l+1} - T_i^{l}}{\Delta t} \end{aligned}\right\} \qquad \text{——— (6)}$$

From (5) :

$$-\lambda T_{i-1}^{l+1} + (1+2\lambda) T_i^{l+1} - \lambda T_{i+1}^{l+1} = T_i^{l} \qquad \text{——— (7)}$$

where $\lambda = \frac{k \Delta t}{\Delta x^2}$

At left end of rod ($i=0$) :

$$T_0^{l+1} = \underbrace{f_0(t^{l+1})}$$

↓

describes how the boundary temp. changes with time

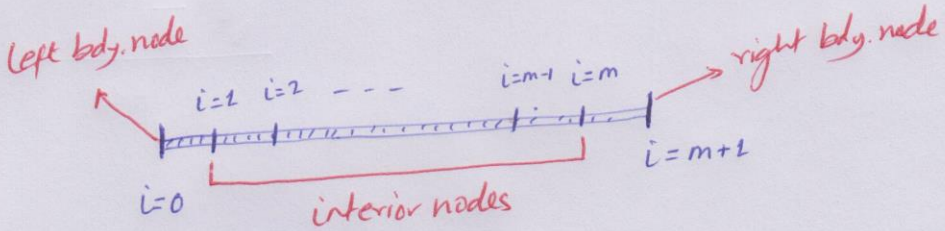
$\therefore$ $i=1$

$$(1+2\lambda) T_1^{l+1} - \lambda T_2^{l+1} = T_1^{l} + \lambda f_0(t^{l+1}) \qquad \text{——— (8)}$$

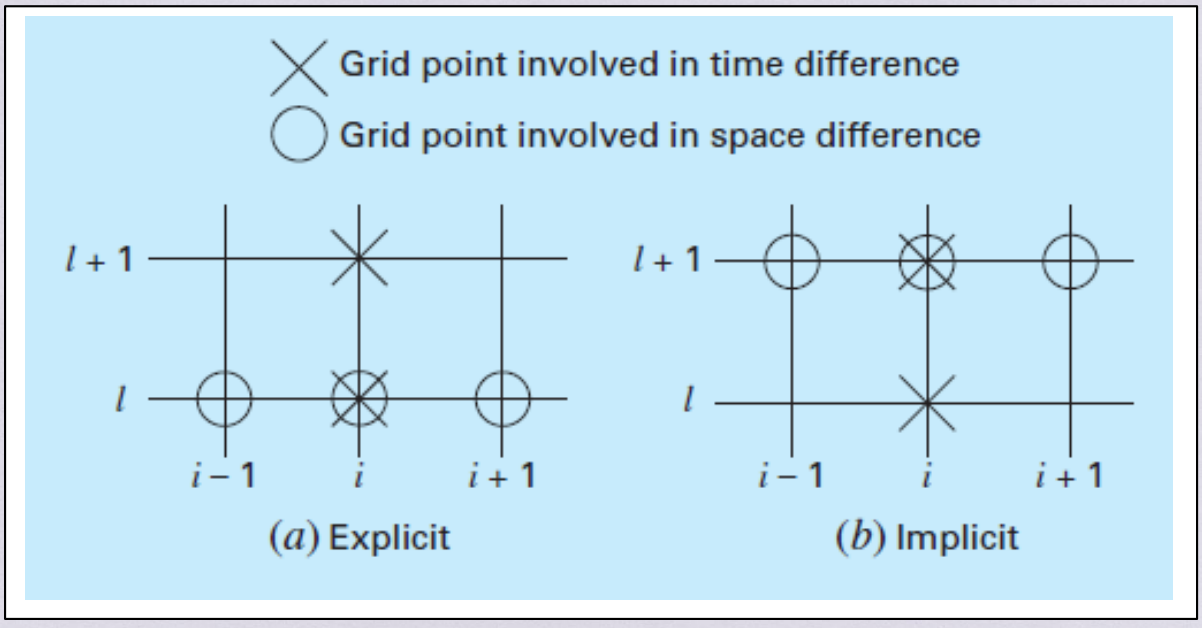
Similarly, at last interior node ($i=m$):

$$-\lambda T_{m-1}^{l+1} + (1+2\lambda) T_m^{l+1} = T_m^l + \lambda f_{m+1}(t^{l+1}) \qquad \text{——(9)}$$

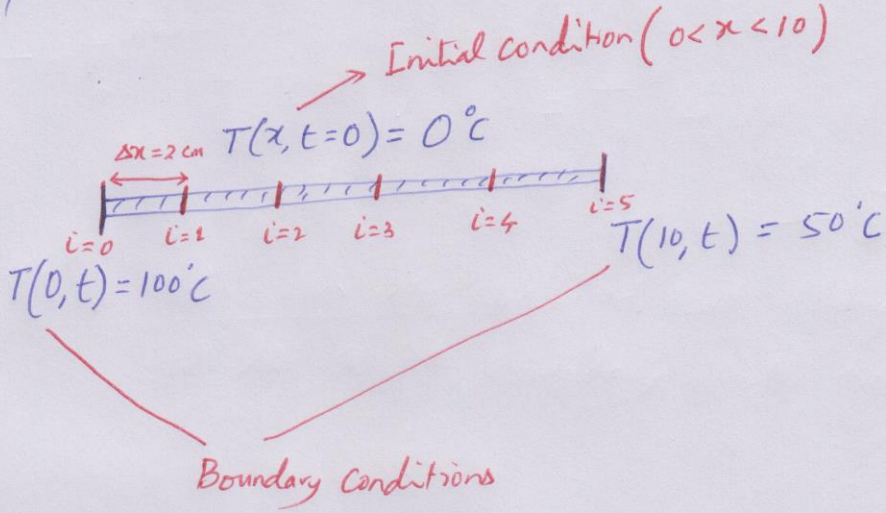
where $f_{m+1}(t^{l+1})$ describes the specified temp. change at the right boundary of the rod.



∴ Equations (7), (8) and (9) are for interior nodes i, $i=1,2,\cdots,m$,
That is m linear algebraic equations in m unknowns.



Example 2: Solve the problem given in Example 1 using implicit method.



$$\lambda = \frac{k\Delta t}{\Delta x^2} = 0.020875$$

Using Eq. (8) at $t = 0$: That is for Ist interior node ($i = 1$)

$$1.041\,75\, T_1^1 - 0.020875\, T_2^1 = 0 + 0.020875(100)$$

OR

$$1.041\,75\, T_1^1 - 0.020875\, T_2^1 = 2.0875$$

Similarly, using Eqns. (7) and (9), we have system of 4 equations in 4 unknowns.

$$\begin{bmatrix} 1.04175 & -0.020875 & & \\ -0.020875 & 1.04175 & -0.020875 & \\ & -0.020875 & 1.04175 & -0.020875 \\ & & -0.020875 & 1.04175 \end{bmatrix} \begin{Bmatrix} T_1^1 \\ T_2^1 \\ T_3^1 \\ T_4^1 \end{Bmatrix} = \begin{Bmatrix} 2.0875 \\ 0 \\ 0 \\ 1.04375 \end{Bmatrix}$$

which can be solved for the temperature at $t = 0.1$ s:

$$T_1^1 = 2.0047$$
$$T_2^1 = 0.0406$$
$$T_3^1 = 0.0209$$
$$T_4^1 = 1.0023$$

To solve for the temperatures at $t = 0.2$, the right-hand-side vector must be modified to account for the results of the first step, as in

$$\begin{Bmatrix} 4.09215 \\ 0.04059 \\ 0.02090 \\ 2.04069 \end{Bmatrix}$$

The simultaneous equations can then be solved for the temperatures at $t = 0.2$ s:

$$T_1^2 = 3.9305$$
$$T_2^2 = 0.1190$$
$$T_3^2 = 0.0618$$
$$T_4^2 = 1.9653$$