Contents:

1. Solution of nonlinear equations.
2. Interpolation, approximation
3. Numerical differentiation, integration
4. Solution of ODE
5. Solution of PDE

SOLUTIONS OF NONLINEAR EQUATIONS:

Linear iteration,

⋮

algorithm, procedure, iterative algorithm.

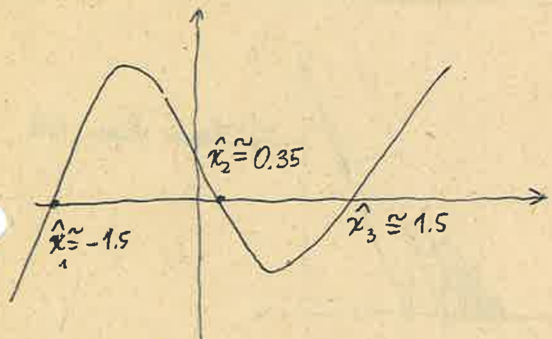
26021980

This expression evaluates the error e_{i+1} in terms of error e_i. For a fast convergence, we require that error → 0 sharply; when e_i becomes fairly small, e_i^2, e_i^3 ... etc become much small and ignored, then the most significant error term is the one with lowest order.

If 1st term $g'(\hat{x})$ is nonzero, it is a linear iterative.
" 2nd term $g''(\hat{x})$ " " , but $g'(x)$ is zero then it is quadratically convergent iteration.

Example:

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



$x = g(x)$ find $g(x)$; Trial and error process:

$$x = \underbrace{\frac{1}{3}(x^3+1)}_{g(x)}$$

$$g'(x) = x^2 \longrightarrow |g'(x)| < 1$$
$$|x^2| < 1 \rightarrow -1 \leq x \leq 1$$

$x_0 = 0.5$

$x_1 = \frac{1}{3}(0.5^3 + 1) = 0.375$

$x_2 = \frac{1}{3}(0.052873+1) = 0{,}35095 \cong \hat{x}$

a parameter method for determining g function

$f(x) = 0$

$f(\hat{x}) = 0$ or $\hat{x} = g(\hat{x})$

$k\hat{x} = kg(\hat{x})$

or further $\hat{x} = (1-k)\hat{x} + kg(\hat{x})$ or iteratively

$$x_{i+1} = \underbrace{(1-k)x_i + kg(x_i)}_{G(x_i)}$$

$|G'(x_i)| < 1$ required. We may satisfy this requirement by adjusting k. properly.

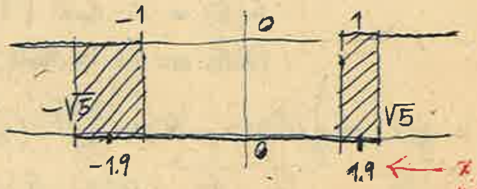
Choose $k = -\frac{1}{2}$

$$G'(x) = (1-k) + kg'(x)$$
$$= (1-k) + kx^2$$
$$G'(x) = (1+\tfrac{1}{2}) - \tfrac{1}{2}x^2 = \underbrace{\tfrac{3}{2} - \tfrac{1}{2}x^2}_{G(x)}$$

$$\left|\tfrac{3}{2} - \tfrac{1}{2}x^2\right| < 1 \longrightarrow \left.\begin{array}{l} \tfrac{3}{2} - \tfrac{1}{2}x^2 < 1 \\ -\tfrac{3}{2} + \tfrac{1}{2}x^2 < 1 \end{array}\right\}$$ These must be satisfied simultaneously

$3 - x^2 - 2 \leq 0 \longrightarrow x \leq \pm 1$

$-3 + x^2 - 2 \leq 0 \longrightarrow x = \pm\sqrt{5}$



← x initially chosen within this range

you perform the iterations yourself.

How computers take square root by software?

AITKEN'S DELTA-SQUARE PROCESS:

$x_{i+1} = (1-k)x_i + kg(x_i) \longleftarrow k = \frac{1}{2}$ $\quad x = g(x) = \frac{a}{x}$

if $x = \sqrt{a} = ?$ $\quad g'(x) = -\frac{a}{x^2}$

$f(x) = x - \sqrt{a} = 0$

$G(x) = \frac{1}{2}\left(1 - \frac{a}{x^2}\right)$ note that as $x \to \hat{x} = \sqrt{a}$
$G'(x) \to 0$

Example: $a = 4$

$x = \sqrt{4} = ?$

$x_0 = 1$

$x_{i+1} = \frac{1}{2}x_i + \frac{1}{2}a/x_i$

$x_1 = \frac{1}{2}(1+4) = 2.5$

$x_2 = \frac{1}{2}\left(2.5 + \frac{4}{2.5}\right) = 2.05$

$x_3 = \frac{1}{2}\left(2.05 + \frac{4}{2.05}\right) = 2.00$

NEWTON RAPHSON METHOD :

This method is the most popular quadratically convergent root finding method. At any value of x_i, the error is

$$x_{i+1} - x_i = \Delta x_i$$

We may write, $\underbrace{f(x_i + \Delta x)}_{x_{i+1}} \rightarrow 0$

Now expand $f(x_i + \Delta x_i)$ around x_i by Taylor series

$$f(x_i+\Delta x_i) = f(x_i) + f'(x_i)\Delta x_i + \frac{f''(x_i)}{2}\Delta x_i^2 \longrightarrow 0$$

as x_i approaches to $\hat{x}_i$ Δx becomes small, its square becomes much smaller, neglected.

yielding : $\Delta x_i = -\dfrac{f(x_i)}{f'(x_i)}$

02031981

$$\underbrace{\boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}}}_{g(x_i)} \longleftarrow \text{Newton Raphson eqn.}$$

$$g'(x) = \frac{f(x)f''(x)}{[f'(x)]^2}$$

note that as we converge to the root $\hat{x}$
$g'(x) \longrightarrow 0$ since
$f(\hat{x}) = 0$ and $f'(\hat{x}) \neq 0$
(roots are not multiple)

$$x_{i+1} = g(x_i) = \underbrace{g(\hat{x})}_{\hat{x}} + \underbrace{g'(\hat{x})}_{0}\underbrace{(x_i-\hat{x})}_{e_i} + \frac{g''(\hat{x})}{2}\overbrace{(x_i-\hat{x})^2}^{e_i^2} + \Theta(x_i-\hat{x})^3$$

Thus convergence is quadratic ie $e_{i+1} = g''(\hat{x})\dfrac{e_i^2}{2} + \cdots$

Now, if we have multiple root :

Assume that we have a double root at $\hat{x}$. Then $g'(x)$:

$$g'(x) = \lim_{x\to\hat{x}} \frac{\frac{d}{dx}[f(x)f''(x)]}{\frac{d}{dx}\{[f'(x)]^2\}} = \frac{1}{2} + \frac{f(\hat{x})f'''(\hat{x})}{2f'(\hat{x})f''(\hat{x})}$$

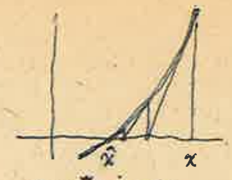
Apply L'hospital's rule one more ;

you'll obtain , $g'(\hat{x}) = \frac{1}{2} = \text{constant} = k$

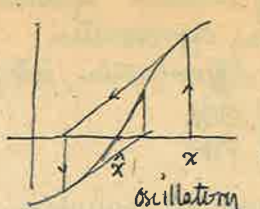
Disadvantages of NR (Newton-Raphson)

1. You must know (guess) the initial solution point x_0. If x_0 is not close to the root, then NR may diverge, oscillate, or converge to a non-desired root.
2. NR requires evaluation of the derivative or Jacobien of f(x) at each solution point; which may not be possible or easy.

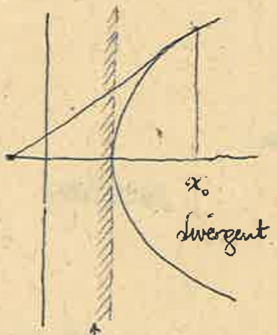
Possible convergence patterns of NR



monotonically convergence

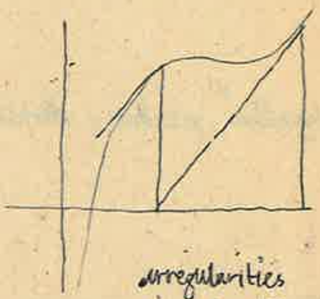


oscillatory convergence

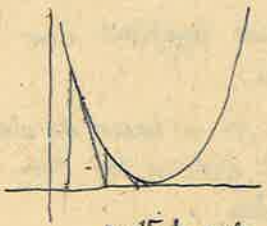


divergent

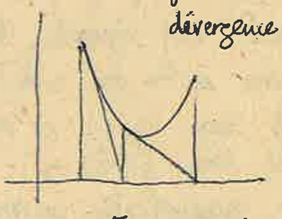
non convergent region



irregularities in a function may delay convergence or cause divergence.

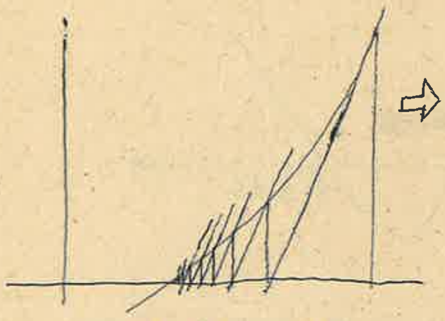


multiple root (linearly conv)



roots are complex (divergent)

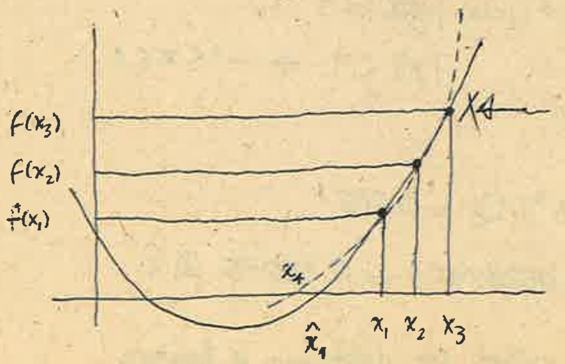
FIXED TANGENT METHOD (parallel tangents meth.



⇨ slower than NR.

Convergence of this method is slower than NR., but since the derivative is not evaluated at each point the overall computational work may be less than NR.

QUADRATIC SPLINE METHOD (Muller)



a parabola : $a_0 + a_1x + a_2x^2$

Muller's Spline :

Determines the real and complex roots of any function $f(x)$. Let $\{x_{i-2}, x_{i-1}, x_i\}$ be three distinct initial guesses about the root. Since the above set defines a unique polynomial,

$$p(x) = a_0 + a_1 x + a_2 x^2$$

This polynomial has the roots,

$$x = \frac{2a_0}{-a_1 \pm \sqrt{a_1^2 - 4a_0 a_2}}$$

hint: define $y = \frac{1}{x}$ solve for y then set $x = \frac{1}{y}$

choose the sign so that the denominator is largest in magnitude.

The above x value is the next approximation for the root $\hat{x}$. Continue iteration with new set;

$$\{x_{i-1}, x_i, x = x_{i+1}\}$$

algorithm suitable for programming

S1: Let $\{x_{i-2}, x_{i-1}, x_i\}$ be the guesses of $\hat{x}$. If no idea is available about the value of $\hat{x}$, then use $x_0 = 1$, $x_1 = 0$, $x_2 = 1$

S2: Compute sequentially the followings;

$h = x_i - x_{i-1}$ ($i = 2$ initially)

$$\chi_i = \frac{h}{x_{i-1} - x_{i-2}}$$

Define $\delta_i = 1 + \lambda_i$

S3: Compute $g_i = f(x_{i-2})\lambda_i^2 - f(x_{i-1})\delta_i^2 + f(x_i)(\lambda_i + \delta_i)$

S4: Compute $\lambda_{i+1} = \dfrac{-2f(x_i)\delta_i}{g_i \pm [g_i^2 - 4f(x_i)\delta_i\lambda_i(f(x_{i-2})\lambda_i - f(x_{i-1})\delta_i + f(x_i))]^{1/2}}$

choose this sign which makes denom. largest in magnitude

S5: $x_{i+1} = x_i + h\lambda_{i+1}$

S6: Compute $f(x_{i+1})$

S7: Repeat S1 to S6 until $|f(x_{i+1})| \leq \epsilon$ where ϵ is an arbit. error tolerance

Deflation (= extraction)

Once a root $\hat{x}_1$ or complex pair $\hat{x}_1^*, x_1$ is found, the remaining roots may be found by extracting-out the found root from the function.

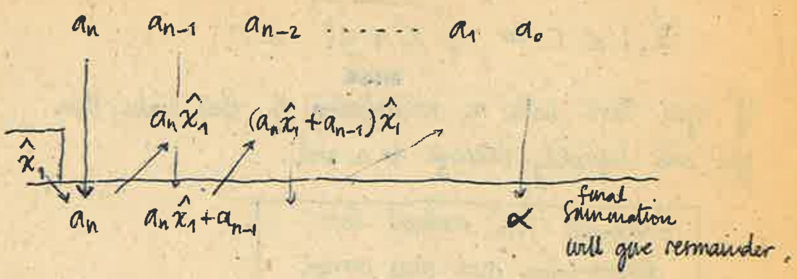
1) Synthetic division;

Let $f(x) = \sum_{i=0}^{n} a_i x^i$ be a polynomial of degree n, $a_n \neq 0$

Suppose that we have found a root $\hat{x}_1$.

Then $P(x) = \underbrace{(x - \hat{x}_1)}_{\text{divider}}\underbrace{q(x)}_{\text{dividend}} + \alpha$ where α is constant remainder here

If $\hat{x}_1$ is a root then α must be zero.



Now, what we will do when the found roots are complex (conjugate)

2) Bairstow's synthetic division (quadratic div)

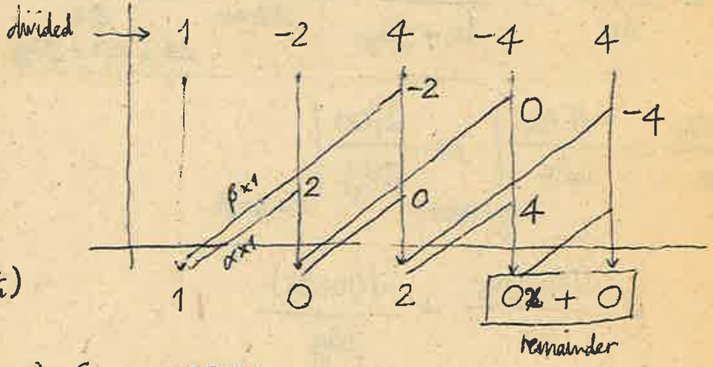
Example We have, $P(x) = x^4 - 2x^3 + 4x^2 - 4x + 4$

assume that we have found;

$(x - \hat{x}_1)(x - \hat{x}_1^*) = x^2 - \alpha x - \beta$ (divider)

Then $\underset{\text{divided}}{P(x)} = \underbrace{(x^2 - \alpha x - \beta)}_{g(x)}\underset{\text{dividend}}{q(x)} + \underbrace{(x + \gamma)}_{r(x)}$

assume that: $\alpha = 2$, $\beta = -2$ divider



3) General deflation:

If you have a general function $f(x)$ and found a root $\hat{x}_1$, then you may define a new function $q(x)$

$$q(x) = \frac{f(x)}{(x - \hat{x}_1)}$$

It may be shown that this $q(x)$ does no longer involve the found root.

Theorem: (A method of estimating x_0, so that convergence is guaranteed.)

Let $f(x)$ be a polynomial of order n;

$f(x) = P(x) = a_n x^n + a_{n-1}x^{n-1} \cdots + a_0$

a condition: $a_n = 1$

otherwise divide the polynomial by a_n. Then $\exists$ a circle, with radius:

$r = 1 + \max|a_i|$ $i =$ ~~0, ..., n-1~~ 1, ..., n i.e if the roots are real, then $|\hat{x}_i| \le r$

if your roots are complex,

$$\hat{z}_i = \hat{x}_i + j\hat{y}_i$$

$$|\hat{z}_i| \le r \quad \text{or} \quad \sqrt{x_i^2 + y_i^2} \le r$$

If you start with x_0 at ~~outside~~ INSIDE of this circle, then you will definitely converge to a root.

$$\boxed{\text{Extended NR method for determining real and comp. roots of any function :}}$$

$z = x + jy$

$f(z) = 0$ is the function to be solved for $\hat{z}$.

$$f(z) = u + jv = 0 \qquad \begin{matrix} u = 0 \\ v = 0 \end{matrix}$$

$$z_{i+1} = z_i - \frac{f(z_i)}{f'(z_i)} \qquad i = 0 \ldots$$

↑ must satisfy Cauchy - Riemann.

Definition of $f'(z)$:

$$f'(z) = \frac{df(z)}{dz} = \frac{df(z)}{dx + jdy} \cong \lim_{\Delta z \to 0} \frac{f(z+\Delta z) - f(z)}{\Delta z}$$

$\Delta z = \Delta x + j\Delta y$

$$\frac{df(z)}{dz} = \frac{df(z)}{dx}\Bigg|_{y=\text{const}} = \frac{df(z)}{jdy}\Bigg|_{x=\text{Const.}}$$

$$= \frac{d(u+jv)}{dx} = \frac{d(u+jv)}{jdy}$$

or $u_x + jv_x = \frac{1}{j}(u_y + jv_y)$ where subscript denotes partial w.r.t. x or y

Thus $\boxed{\begin{matrix} u_x = v_y \\ u_y = -v_x \end{matrix}}$ CR conditions

If CR conditions hold then $\frac{df(z)}{dz} = \underbrace{u_x + jv_x}_{\text{choose one of them}} = \underbrace{v_y - ju_y}$

$$\text{Then : } x_{i+1} + jy_{i+1} = x_i + jy_i - \frac{u + jv}{u_x + jv_x}$$

$$= x_i + jy_i - \frac{(u+jv)(u_x - jv_x)}{|u_x + jv_x|^2}$$

or decomposing into its components you will obtain two separate real equations.

$$x_{i+1} = x_i - \frac{uu_x + vv_x}{u_x^2 + v_x^2}\Bigg|_{x_i} \qquad (1)$$

$$y_{i+1} = y_i - \frac{vu_x - uv_x}{u_x^2 + v_x^2}\Bigg|_{x_i} \qquad (2)$$

Example : Find the roots of $f(z) = z^2 + 1 = 0$

check for CR conditions :

$$f(z) = (x+jy)^2 + 1 = \underbrace{(x^2 - y^2 + 1)}_{u} + j\underbrace{(2xy)}_{v}$$

$$\left.\begin{matrix} \frac{\partial u}{\partial x} = 2x \ , \ \frac{\partial v}{\partial y} = 2x \\ \frac{\partial u}{\partial y} = -2y \ , \ \frac{\partial v}{\partial x} = 2y \end{matrix}\right\} \text{CR holds.}$$

z_0 may be found by the theorem start, for example :

$$z_0 = x_0 + jy_0 = 0.5 + j0.5$$

$$x_{i+1} = x_i - \frac{(x_i^2 - y_i^2 + 1)2x_i + 2x_iy_i 2y_i}{(2x_i)^2 + (2y_i)^2}$$

$$y_{i+1} = y_i - \frac{2x_iy_i 2x_i - (x_i^2 - y_i^2 + 1)2y_i}{(2x_i)^2 + (2y_i)^2}$$

use successive substitution instead of simultaneous substitution !

$$\begin{tikzcd} x_0 & y_0 \\ x_1 \arrow[u] & y_1 \arrow[u] \end{tikzcd} \Bigg\} \text{simultaneous}$$

$$\begin{tikzcd} x_0 & y_0 \\ x_1 \arrow[u] & y_1 \arrow[u] \end{tikzcd}$$

→ without waiting the calculation of y_0 in terms of old x_0 insert new value x_1

090381

Convergence conditions for complex iterations ;

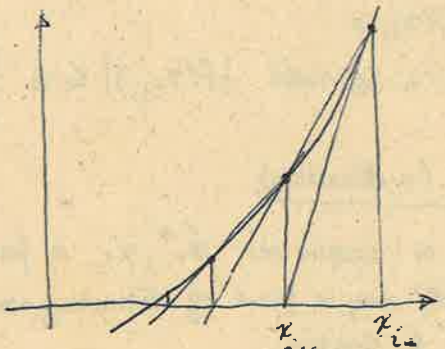
we may test either :

$$|f(z)| \le \epsilon \quad \text{or} \quad |\Delta z| \le \epsilon$$

$$|(\Delta x)^2 + (\Delta y)^2| \le \epsilon$$

where $\Delta z = z_{i+1} - z_i$

Multi - step iteration algorithms :



The NR equation was :

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

If we approximate the derivative by its definition ;

$$\lim_{x_{i+1} \to x_i} f'(x_i) = \frac{f(x_i) - f(x_{i+1})}{x_i - x_{i+1}}$$

then NR becomes:

minus

$$x_{i+1} = x_i - \frac{f(x_i)\left[x_i - x_{i+1}\right]}{f(x_i) - f(x_{i+1})}$$

SYSTEM OF NONLINEAR EQUATIONS

A system of nonlinear eq'ns may be shown as,

$$\underline{F}(\underline{x}) = \underline{0}$$

where $\underline{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ $\quad \underline{F} = \begin{bmatrix} f_1(x_1 \cdots x_n) \\ \vdots \\ f_n(x_1 \cdots x_n) \end{bmatrix}$

Methods:

1. Simple iteration:

Transform $F(x) = 0$ to the form $X = G(x)$ and iterate:

$$x_{i+1} = G(x_i) \qquad i = 0 \ldots n \quad \text{where } G = \begin{bmatrix} g_1(x_1 \cdots x_n) \\ \vdots \\ \vdots \end{bmatrix}$$

if converges, the above iteration converges linearly.

Convergence conditions:

- A sufficient (but not necessary) convergence condition is as follows;

1. $g_1, g_2, \cdots, g_n$ and their first partials are continuous and bounded in a closed region R in the neighborhood of the root $\hat{X}$.

2- $$\left. \| J(x) \| \right|_{x \cong \hat{x}} = k \leq 1$$

$$\begin{vmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{vmatrix} = \begin{vmatrix} g_1(x_1 \cdots x_n) \\ \\ \\ g_n(x_1 \cdots x_n) \end{vmatrix}$$

where $J(x)$ is known as the Jacobian of $G(x)$ defined as follows:

$$J_{ij}(x) = \frac{\partial g_i(x)}{\partial x_j} \qquad i, j = 1, \ldots n$$

and $\|\cdot\|$ is known as matrix norm, and is defined as

$$\|A\| = \max_i \sum_{j=1}^{n} |a_{ij}| = \max_j \sum_{i=1}^{n} |a_{ij}|$$

known as maximal row norm or column norm respectively.

Simple iteration is also known as Jacobian iteration. (simultaneous disp.)

2. Gauss Seidel iteration: (Successive displacement)

$$x_1 = g_1(x_1 \ldots x_n)$$
$$\vdots$$
$$x_n = g_n(x_1 \ldots x_n)$$

Example:

Apply GS iteration to solve the following sys. of equations:

$$f_1 \longrightarrow x_1^2 - x_2^2 - x_1 = 0$$

$$f_2 \longrightarrow x_1 - 4x_2 = 0$$

g functions:

$$x_1 = x_1^2 - x_2^2 \longleftarrow g_1$$

$$x_2 = \frac{x_1 - x_2}{3}$$

$$G(x) = \begin{bmatrix} x_1^2 - x_2^2 \\ (x_1 - x_2)/3 \end{bmatrix} \qquad x^0 = \begin{bmatrix} 0.25 \\ -0.25 \end{bmatrix}$$

$$x_1^1 = g_1(x_1^0, x_2^0) = (x_1^0)^2 - (x_2^0)^2 = 0$$

$$x_2^1 = g_2(x_1^1, x_2^0) = \frac{0 + 0.25}{3} \qquad x^1 = \begin{bmatrix} 0 \\ 0.0833 \end{bmatrix}$$

Similarly:

$x^2 \;\ldots\; x^3$

120381

NR algorithm for nonlinear sys. of equations:

$$F(x) = 0$$

$$x^{p+1} = x^p - \left[J(x^p)\right]^{-1} F(x^p)$$

$$\Delta x^{p+1} = -\left[J(x^p)\right]^{-1} F(x^p) \qquad p: \text{ iteration index } 0, 1 \ldots$$

where $J(x)$ is the Jacobian of F w.r.t X, defined as

$$J_{ij}(x) = \left. \frac{\partial f_i}{\partial x_j} \right|_{x = x^p} \qquad i, j = 1 \ldots n$$

This inversion (inversion of $J(x)$) is performed by usual techniques.

Conditions for convergence:

1. Let $f_1 \ldots f_n$ and all their cross-partials be continuous and bounded.

2. This region should contain $\hat{x}$.
   i.e $\hat{x} \in R$
3. $J(x^P)$ must be ~~nonsighed~~ singular at all points ~~of~~ x^P
4. x_0 should be sufficiently close to root.

<u>Example</u> : 2 equations, 2 unknowns.

$$\underbrace{\begin{bmatrix} f_1(x_1,x_2) = x_1^2 - x_2^2 - x_1 \\ f_2(x_1,x_2) = x_1 - 4x_2 \end{bmatrix}}_{F(x)} \begin{matrix} = 0 \\ = 0 \end{matrix} \qquad x^0 = \begin{bmatrix} 0.25 \\ -0.25 \end{bmatrix}$$

$$\Delta x^P = -J(x^P)^{-1} F(x^P)$$

$$J(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 - 1 & -2x_2 \\ 1 & -4 \end{bmatrix}\Bigg|_{x = x^0}$$

$$J(x^0) = \begin{bmatrix} -0.5 & 0.5 \\ 1 & -4 \end{bmatrix}$$

$$[J(x^0)]^{-1} = \begin{bmatrix} -2.6667 & -0.333 \\ -0.66667 & -0.333 \end{bmatrix}$$

mismatch :

$$F(x^0) = \begin{bmatrix} 0.25^2 - 0.25 - (-0.25)^2 \\ 0.25 - 4(-0.25) \end{bmatrix} = \begin{bmatrix} -0.25 \\ 1.25 \end{bmatrix}$$

Now ;

$$\Delta x^{P+1} = -J(x^P)^{-1} F(x^P)$$

$$= -\begin{bmatrix} -2.66 & -0.3 \\ -0.66 & -0.2 \end{bmatrix} \begin{bmatrix} -0.25 \\ 1.25 \end{bmatrix} = \begin{bmatrix} -0.25 \\ 0.25 \end{bmatrix}$$

$$x^1 = x^0 + \Delta x = \begin{bmatrix} 0.25 \\ -0.25 \end{bmatrix} + \begin{bmatrix} -0.25 \\ 0.25 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

<u>Steepest descent method</u> :

$$F(x) = 0 \qquad x = \hat{?}$$

$$F = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Define a new function (known as <u>cost</u> function) $L(x)$,

$$L(x) = (f_1(x))^2 + \dots + (f_n(x))^2$$

at the solution point this function must be minimized.

The gradient of cost function wrt x will be,

$$\frac{\partial L(x)}{\partial x} = \nabla L(x)$$

The gradient direction points the angle in which $L(x)$ increases most rapidly.

choose $-\nabla L(x)$ direction.

The normed direction vector $(U(x))$, is obtained by normalizing ∇L with its norm.

$$u(x) = \frac{-\nabla L(x)}{\|\nabla L(x)\|} \qquad \text{where } \|x\| = \sqrt{x_1^2 + \dots x_n^2}$$

α is a step length. (scaler)

$$x^{P+1} = x^P + \alpha u$$

To determine α there are several ways ~~are~~ possible. One possible, (but not best nor computation efficient) way :

→ Set α to a large value. So that

$$L(x^{P+1}) > L(x^P)$$

which means α is too large for that step length.

Then divide α by two and retry.

Then $x^{P+1} = x^P + \alpha u$.

This method usually is used for highly-nonlinear problems.

SOLUTION OF LINEAR SYSTEM OF EQUATIONS

<u>Definition</u>: A linear system of equations is defined as

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1$$
$$\vdots$$
$$a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n$$

in matrix form:

$$Ax = B$$

<u>Solvability condition</u> :

The linear system of equations $Ax = B$ has a solution or <u>consistent</u> iff : $\text{rank}(A) = \text{rank}(A|B)$ otherwise if $\text{rank}\,A < \text{rank}(A|B)$ then sys. is said to be inconsistent.

if $\text{rank}(A) = \text{rank}(A|B) = n$ then the solution is unique.

Rank of a matrix A is equal to its size n, iff $\det A \neq 0$

If the above condition is satisfied, the A is nonsingular.

<u>Definition</u> : A system of equations

$AX = B$

is called homogeneous if $B = 0$,

If $Ax = B$ is homogenous and have a unique solution then this solution is trivial, i.e $x = 0$ if rank $A < n$, then there are other solutions $\Rightarrow x \neq 0$

<u>Theorem</u> : For the linear system of equations

$Ax = B$

if rank A = rank $(A|B) = K < n$

then, n-k of unknowns x_i are not unique (or arbitrary) The remaining k unknowns are dependent upon these n-k unknowns.

<u>Example</u> :

1. has a unique solution:

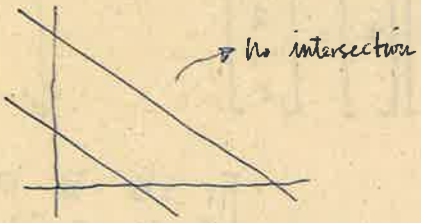
$$x_1 + 2x_2 = 4 \qquad \text{rank} A = \text{rank}(A|B) = n = 2$$
$$x_1 - x_2 = 2$$

2. inconsistent case:

$$x_1 + 2x_2 = 4$$
$$2x_1 + 4x_2 = 5$$

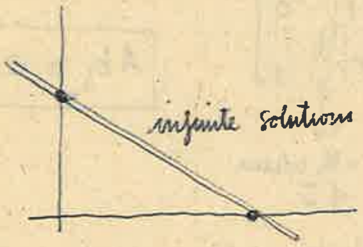
$$\begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \end{bmatrix} \qquad \text{rank} A = 1$$

rank $(A|B) = 2$

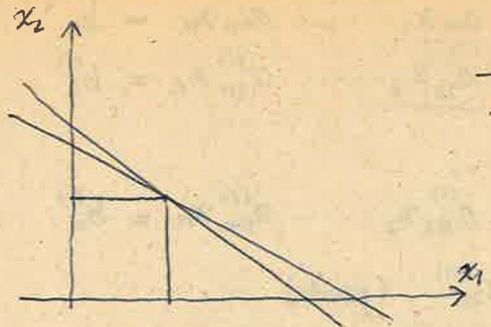


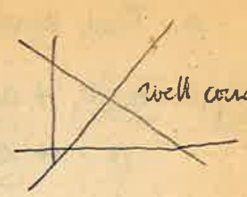
③ Consistent, but ∞ solutions:
(n-k unknowns are arbitrary)
(Rank deficient)

$$x_1 + 2x_2 = 4 \qquad \text{rank} A = \text{rank}(A|B) = 1$$
$$2x_1 + 4x_2 = 8$$



<u>Numerically ill conditioning</u> :





IMPLICIT REAL*8 (A-H, Ø-Z)

<u>Example</u> :

$$\left.\begin{aligned} 2x_1 + 6x_2 &= 8 \\ 2x_1 + 6.00001x_2 &= 8.00001 \end{aligned}\right\} \quad \begin{aligned} &\text{Solution:} \\ &x_1 = 1 \\ &x_2 = 1 \end{aligned}$$

while;

$$\left.\begin{aligned} 2x_1 + 6x_2 &= 8 \\ 2x_1 + 5.99999x_2 &= 8.00002 \end{aligned}\right\} \quad \begin{aligned} x_1 &= 10 \\ x_2 &= -2 \end{aligned}$$

<u>Solution of linear system of equations</u>

1. Direct solution. ⟶ are used in problems with n<100 usually., <u>dense</u> coeff. matrix (percent of non-zero entries greater than 60)
   - accurate
2. Iterative solution
   ⟶ large n; sparse matrices (15% <)
   - accuracy depends on ∈

<u>Types of errors in numerical errors</u> :

① Round-off errors.
② Truncation errors
③ Iteration errors. ε

<u>Direct Methods</u> :

– <u>Gaussian Elimination</u>

Let the system of linear equations be:

$$\begin{matrix} a_{11}x_1 \cdots & a_{1n}x_n & = b_1 \\ \vdots & & \vdots \\ a_{n1}x_1 \cdots & a_{nn}x_n & = b_n \end{matrix}$$

algorithm:

1. Choose the first row, take a_{11}
2. ~~Multiply~~ Add $\rightarrow \left(\frac{-a_{21}}{a_{11}}\right)$ times first row to second row.
3. 〃 $\rightarrow \left(\frac{-a_{31}}{a_{11}}\right)$ 〃 〃 〃 to 3rd row.

4. Then equations become:

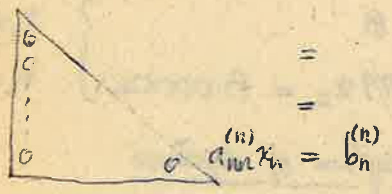
$$\begin{aligned} \text{called pivot} \leftarrow \boxed{a_{11}} x_1 + a_{12} x_2 - - - \quad & a_{1n} x_n = b_1 \\ 0 \quad + \boxed{a_{22}^{(1)}} \boxed{x_2} \cdots\cdots \quad & a_{2n}^{(1)} x_n = b_2^{(1)} \\ \vdots \qquad\qquad \downarrow \qquad & \\ 0 \quad + a_{n2}^{(1)} x_2 - - - - \quad & a_{nn}^{(1)} x_n = b_n^{(1)} \end{aligned}$$

now consider $a_{22}^{(1)}$ (pivot)

add $-\dfrac{a_{32}^{(1)}}{a_{22}^{(1)}}$ times second row to 3rd row

add $-\dfrac{a_{42}}{a_{22}}$ " " " " 4th "

$\downarrow$



Step 2: back substitution

$$\begin{bmatrix} a_{11} & - - - & a_{1n} \\ & a_{22}^{(1)} - - - & \\ \boxed{0 \longleftrightarrow 0} & & a_{nn}^{(n)} \end{bmatrix} \begin{bmatrix} X \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ \vdots \\ b_n^{(n)} \end{bmatrix}$$

$$a_{nn}^{(n)} x_n = b_n^{(n)}$$

$$x_n = \frac{b_n^{(n)}}{a_{nn}^{(n)}}$$

$$a_{n-1,n-1} x_{n-1} + a_{n-1,n} x_n = b_{n-1}$$

$\downarrow$

<u>algorithm</u>:

$$x_n = b_n^{(n)} / a_{nn}^{(n)}$$

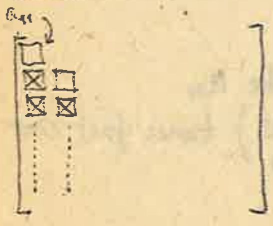
$$x_{n-1} = \left(b_{n-1}^{(n-1)} - a_{n-1}^{(n-1)} x_n \right) / a_{n-1,n-1}^{n-1}$$

$$\vdots$$

$$x_i = \left(b_i^{(i)} - \sum_{k=i+1}^{n} a_{ik}^{(i)} x_k \right) / a_{ii}^{(i)} \quad i = n-1, \cdots 1$$

160381

<u>CROUT ELIMINATION</u>:



This method is same as Gaussian elimination (GE) but each row is processed until its final form is obtained.

<u>Example</u>:

$$\begin{bmatrix} 2 & -1 & -1 & 0 \\ -2 & 2 & 1 & 2 \\ 0 & 2 & 4 & 3 \\ 2 & 3 & 11 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \\ -1 \\ -14 \end{bmatrix}$$

<u>GE</u>

Step I

$$\left[\begin{array}{cccc|c} 2 & -1 & -1 & 0 & 3 \\ 0 & 1 & 0 & 2 & 2 \\ 0 & 2 & 4 & 3 & -1 \\ 0 & -2 & 12 & -5 & -17 \end{array}\right]$$

Step 2

$$\begin{matrix} 2 & -1 & -1 & 0 & 3 \\ 0 & 1 & 0 & 2 & 2 \\ 0 & \boxed{0} & 4 & -1 & -5 \\ 0 & \boxed{0} & 12 & -1 & -13 \end{matrix}$$

Step 3:

$$\begin{matrix} 2 & -1 & -1 & 0 & 3 \\ 0 & 1 & 0 & 2 & 2 \\ 0 & 0 & 4 & -1 & -5 \\ 0 & 0 & \boxed{0} & 2 & 2 \end{matrix}$$

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

<u>CE</u>

$$\begin{matrix} 2 & -1 & -1 & 0 & 3 & \leftarrow \text{1st row.} \\ 1 & -\frac{1}{2} & -\frac{1}{2} & 0 & \frac{3}{2} & \leftarrow \text{normalized 1st row.} \end{matrix}$$

Step 1:

$$\begin{matrix} -2 & 2 & 1 & 2 & -1 & \leftarrow \text{initial form of 2nd row.} \\ 0 & 1 & 0 & 2 & 2 & \leftarrow \text{final form of 2nd row} \end{matrix}$$

Step 2

$$\begin{matrix} 0 & 2 & 4 & 3 & -1 & \leftarrow \text{initial form of 3rd row} \\ 0 & 0 & 4 & -1 & -5 & \leftarrow \text{final form of 3rd row} \\ 0 & 0 & 1 & -\frac{1}{4} & -\frac{5}{4} & \nwarrow \text{normalized form of 3rd row.} \end{matrix}$$

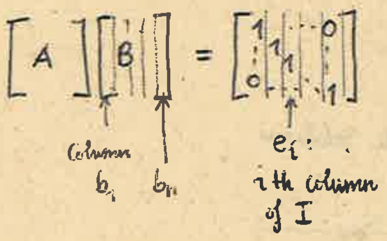
$$\begin{matrix} 2 & -3 & 11 & -5 & -14 & \leftarrow \text{initial 4th row} \\ 0 & -2 & 12 & -5 & -17 & \\ 0 & 0 & 12 & -1 & -13 & \\ 0 & 0 & 0 & 2 & 2 & \leftarrow \text{final 4th row.} \\ 0 & 0 & 0 & 1 & 1 & \nwarrow \text{normalized 4th row.} \end{matrix}$$

$\downarrow$

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

<u>inverse of a Matrix by GE</u>

$$\overset{n \times n}{A} \overset{n \times n}{B} = I$$



$$\boxed{A b_i = e_i \quad i = 1, \ldots, n}$$

<u>an alternate method:</u> (Gauss – Jordan Method)

$$\left[\begin{array}{ccc|c} a_{11} & - - - & a_{1n} & \\ \vdots & & \vdots & I \\ a_{n1} & - - - & a_{nn} & \end{array}\right]$$

Perform GE on the above augmented matrix and obtain :

Suppose that we have an n dimensional problem. Evaluate by yourself and see that the # of arithmetic operations performed for each type of method are as follows:

$$GE: \begin{cases} \left(\frac{n^2}{2} - \frac{n}{2}\right) \text{ divisions} \\ \left(\frac{n^3}{3} - \frac{n}{3}\right) \text{ multiplication + addition} \end{cases}$$

Back substitution: n divisions $+ \frac{n(n-1)}{2}$ mult + addition

Total number of operations:

$$\frac{n^3}{3} + \frac{n^2}{2} - \frac{5n}{6} \text{ (mult + add)} + \frac{n^2}{2} + \frac{n}{2} \text{ (division)}$$

for Gauss Jordan method:

$n + n(n-1) = n^2$ divisions

$n(n^2-1)/2 \cong \frac{n^3}{2}$ (mult + add)

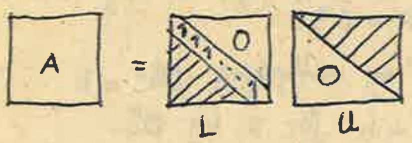
For large size of n, GJ requires 50% more operations than GE.

TRIANGULAR FACTORIZATION:

$$[A][x] = [b]$$

Theorem: Any real square ? matrix A can be decomposed into two factor matrices,

A = LU where L is lower triangular
U is an upper triangular matrix.



$Ax = b \longrightarrow (LU)x = b$

if you define $Ux = b'$
then $Lb' = b$
or $b' = L^{-1}b$ ← Forward substitution
then:
$x = U^{-1}b'$ ← back substitution

Theorem: Any real matrix A can be reduced to an ◁ form by Elementary Row Operations (ERO) and these ERO's may be represented by a factor matrix M which multiply A from the left. Thus M matrix is obtained by performing the same ERO's on an identity matrix.

Notes:

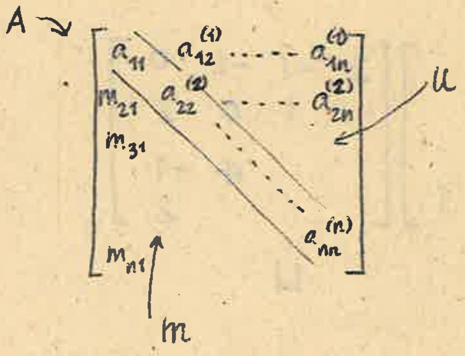
1. Matrix M is a lower triangular matrix with unity diagonals.
2. After triangulating lower part of A becomes and remains zero.
3. U and M are constant for a constant A. Thus they are evaluated once and used for all b's.
4. U and M are not stored into different arrays. The locations used for the original matrix are used to store L and U.
5. The unity diagonals of M are not stored by 'remembered' in programming.
5. The equation then may be solved for an arbitrary number of b vectors by operating

$Mb = b'$ (f.s) → forward substitution
$Ux = b'$ (b.s) → back "

When an element below the diagonal is eliminated its value becomes and remains zero, this location may be used to store the multiplier that was used to eliminate this element.

Where multipliers are defined as;

$$m_{ij} = -\frac{a_{ij}}{a_{jj}} \quad (i > j) \qquad m_{ii} = 1.0$$



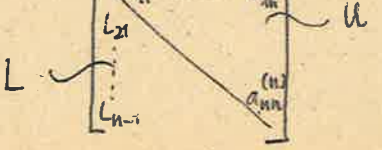
$M^{-1} = L$

This method stores M matrix but we need L actually. For the purpose of FS.

An alternate method:

We may store L instead of M directly as follows:

S1: Triangulate the A matrix by ERO's, fill the zeroed entries in the ▷ part by − of the multipliers to obtain



S2 : The equation is now solved as

$$LUx = b$$
$$Ux = b'$$
$$Lb' = b \ (FS) \rightarrow b' \rightarrow b_1' = b_1$$
$$UX = b' \ (BS) \rightarrow X \qquad b_2' = b_2 + L_{21} b_1'$$
$$\vdots$$
$$b_n' = b_n + \sum_{k=1}^{n-1} L_{ik} b_k'$$

Example :

$$+1 \left\{ \begin{bmatrix} 2 & -1 & -1 & 0 \\ -2 & 2 & 1 & 2 \\ 0 & 2 & 1 & 3 \\ 2 & -3 & 11 & -5 \end{bmatrix} \right. \begin{bmatrix} \\ x \\ \\ \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \\ -1 \\ -14 \end{bmatrix}$$

S1 :

$$\begin{array}{rrrrr} & 2 & -1 & -1 & 0 \\ -\left(\frac{a_{21}}{a_{11}}\right) \rightarrow & \boxed{-1/0} & 1 & 0 & 2 \\ -\left(\frac{a_{31}}{a_{11}}\right) \rightarrow & \boxed{0/0} & 2 & 4 & 3 \\ -\left(\frac{a_{41}}{a_{11}}\right) \rightarrow & \boxed{1/0} & -2 & 12 & -5 \end{array}$$

S2 :

$$\begin{array}{rrrr} 2 & -1 & -1 & 0 \\ -1/0 & 1 & 0 & 2 \\ 0/0 & \boxed{2/0} & 4 & -1 \\ 1/0 & \boxed{-2/0} & 12 & -1 \end{array}$$

S3

$$\begin{array}{rrrr} 2 & -1 & -1 & 0 \\ \boxed{-1/0} & 1 & 0 & 2 \\ \boxed{0/0} & \boxed{2/0} & 4 & -1 \\ \boxed{1/0} & \boxed{-2/0} & \boxed{3/0} & 2 \end{array}$$

$$= \underset{L}{\begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ 0 & 2 & 1 & \\ 1 & -2 & 3 & 1 \end{bmatrix}} \underset{U}{\begin{bmatrix} 2 & -1 & -1 & 0 \\ & 1 & 0 & 2 \\ & & 4 & -1 \\ & & & 2 \end{bmatrix}}$$

$$b = \begin{bmatrix} 3 \\ -1 \\ -1 \\ -14 \end{bmatrix}$$

FS : Solve $Lb' = b$ for b'

$$\begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ 0 & 2 & 1 & \\ 1 & -2 & 3 & 1 \end{bmatrix} \begin{bmatrix} \\ b' \\ \\ \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \\ -1 \\ -14 \end{bmatrix}$$

$$b_1' = 3$$
$$b_2' = -1 + 3\times 1 = 2$$
$$b_3' = -5$$
$$b_4' = 2$$

BS : Solve $Ux = b'$ for x

$$\begin{bmatrix} 2 & -1 & -1 & 0 \\ & 1 & 0 & 2 \\ & & 4 & -1 \\ & & & 2 \end{bmatrix} \begin{bmatrix} \\ x \\ \\ \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ -5 \\ 2 \end{bmatrix} \rightarrow \begin{array}{l} x_4 = 1 \\ x_3 = \frac{(-5+1\times 1}{4} \\ \quad = -1 \\ x_2 = -(2-2\times 1 \\ \quad = 0 \\ x_1 = 1 \end{array}$$

Note : See that each solution for x (ie FS+BS) requires only $\frac{n^2}{2} + \frac{n^2}{2} = n^2$ operations after factorizing A.

For symmetric matrices : $(a_{ij} = a_{ji})$

$$L = U^T$$
$$U^T U = LL^T = A$$

Symmetrical triangular factorization :

If A is symmetrical ;

then $L = U^T$

$L^T = U$

$$A = LU = U^T U$$

Then there is no need to store Δ L matrix since it is transpose of U.

Steps of solution for symmetrical equations :

S1 : Triangulate A into LDU but do not store L

S2 : Solve $U'^T b' = b$ for b' by FS.
where $U' = D^{1/2} U$

S3 : Obtain b'' by multiplying $Db'' = b$

S4 : Solve $U'x = b''$ for x by BS.

COMPLEX - LINEAR SYS. OF EQUATIONS :

Suppose that we have a complex syst. of equations :

$$CZ = W$$

where

C is a complex coefficient matrix.

Z is the complex unknown vector.

W " " known RHS vector.

since C is complex :

$$C = A + jB$$
$$Z = X + jY$$
$$W = U + jV$$

Thus ;

$$(A+jB)(X+jY) = U + jV$$

or

$$AX - BY = U$$
$$BX + AY = V$$

or :

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} x \\ \hline y \end{bmatrix} = \begin{bmatrix} U \\ V \end{bmatrix}$$

which may be solved by by the GE, GJ, TF methods.

Example:

$$\begin{bmatrix} 2+j3 & -1-j1 \\ -1+j0 & 4+j2 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 3+j4 \\ -3+j4 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 & -3 & 1 \\ -1 & 4 & 0 & -2 \\ 3 & -1 & 2 & 1 \\ 0 & 2 & -4 & 4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 3 \\ -3 \\ 4 \\ 4 \end{bmatrix}$$

230381

ITERATIVE SOL'N OF SYSTEM OF EQUATIONS:

$$Ax = b$$

$$n\begin{bmatrix} A \end{bmatrix}_{n}\begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} b \end{bmatrix}$$

$$Ax - b = 0$$

$$F(x) = 0$$

$$x^{i+1} = G(x^i) \qquad i = 0,1,2 \cdots$$

Iterative methods compute successive approximations of the solution x for a given A, b starting from an initial x_0

Convergence (if achieved) is asymptotic.

Thus the iterative procedure has to be terminated when a pre-specified degree of accuracy is satisfied.

The general iterative form is

$$x^{i+1} = G^i(x^i) \qquad i = 0,1, \cdots$$

$\uparrow$ if G function is changed during iteration; it is said, that is "non-stationary" iterative algorithm.

We will consider only stationary iterative algorithms:

$$x^{i+1} = G(x^i)$$

iterative methods are used particularly when;

i) The system size is large.

and/or ii) A is sparse. (you may store only non-zero terms)

Their convergence properties are very uncertain except for particular problems.

Method:

Let the given system be:

$$Ax = b \begin{cases} a_{11}x_1 + a_{12}x_2 \cdots\cdots & a_{1n}x_n = b_1 \\ \vdots & \\ a_{n1}x_1 + \cdots & a_{nn}x_n = b_n \end{cases}$$

we may arrange the above system as follows,

$$x_1 = \frac{1}{a_{11}}\left[b_1 - a_{12}x_2 - \cdots\cdots - a_{1n}x_n\right]$$

$$\vdots$$

$$x_n = \frac{1}{a_{nn}}\left[b_n - a_{n1}x_1 - \cdots\cdots \right]$$

or recursively:

$$x_j = \frac{1}{a_{jj}}\left[b_j - \sum_{\substack{k=1 \\ k\neq j}}^{n} a_{jk}x_k\right] \qquad j = 1,2 \cdots n \cdot (*)$$

we assume that diagonal coefficients $a_{jj} \neq 0$ otherwise we may switch the order of equations so that $a_{jj} \neq 0$,

Note that:

It is sufficient to store only the non-zero terms a_{jk} of A for iteration.

METHOD 1 (Jacobien iteration)

~~In this method the RHS (right hand side) of~~

In this method, first we assume an initial x^0 and Subtitute this x^0 to the RHS (right hand side) of $(*)$ and obtain x^1,

$$x_j^{i+1} = \frac{1}{a_{jj}}\left[b_j - \sum_{\substack{k=1 \\ k\neq j}}^{n} a_{jk}x_k^i\right] \qquad \begin{matrix} j = 1 \cdots n \\ i = 0,1, \cdots \end{matrix}$$

In this method the RHS contains solution vector x^i at iteration i.

The vector $x^{i+1} \neq x^i$ are stored in different vectors during the iteration, and at the end of each iteration cycle x^i is replaced by x^{i+1}.

$$Ax = b$$

we may decompose A as follows,

$$A = \underset{\substack{\downarrow \\ \text{lower} \\ \text{triangular} \\ \text{matrix}}}{L} + \underset{\substack{\searrow \\ \text{diagonal}}}{D} + U \rightarrow \text{upper triangular}$$

or

$$L = \begin{bmatrix} 0 & & & 0 \\ a_{21} & \ddots & & \\ \vdots & & \ddots & \\ a_{n1} & & a_{n,n-1} & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} a_{11} & & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & \ddots & & \vdots \\ & & \ddots & \\ 0 & & & 0 \end{bmatrix}$$

Original form :

$$(L+D+U)x = b$$

recursive form becomes :

$$x^{i+1} = \underbrace{D^{-1}b}_{d} - \underbrace{D^{-1}(L+U)}_{H}x^{i} \qquad i = 0,1\ldots$$

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

$d = D^{-1}b$

Then $\underline{\underline{x^{i+1} = Hx^{i} + d}}$ $\quad i = 0,1\ldots$

GAUSS-SEIDEL METHOD (Method 2)

$$(j=1\ldots n)\quad x_j^{i+1} = \frac{1}{a_{jj}}\left[b_j - \sum_{k=1}^{j-1} a_{jk}x_k^{i+1} - \sum_{k=j+1}^{n} a_{jk}x_k^{i}\right]$$

whenever calculated new value of $x^{i}(x^{i+1})$ is inserted immediately. (Similar to substitution method)

Gauss-Seidel always converges when Jacobian method converges, or it may converge when J diverges.

It's conv. rate is almost always faster than J.

$$x^{i+1} = D^{-1}\left[b - Lx^{i+1} - Ux^{i}\right]$$

or which may be solved for x^{i+1}, yielding

$$\boxed{x^{i+1} = (I + D^{-1}L)^{-1}(D^{-1}b - D^{-1}Ux^{i})}$$

or further

$$x^{i+1} = \underbrace{-(D+L)^{-1}U}_{H'}x^{i} + \underbrace{(D+L)^{-1}b}_{d'}$$

then

$$\underline{\underline{x^{i+1} = H'x^{i} + d'}} \qquad i = 0,1\ldots$$

260381

Convergence condition of GS Method:

$$x^{i+1} = H'x^{i} + d' \qquad i = 0,1,2,\ldots \quad (*)$$

let us assume that the error in i^{th} iteration is ;

$$e^{i} = x^{i} - \hat{x}$$

Then the initial error :

$$e^{0} = x^{0} - \hat{x} \quad \text{where } \hat{x} \text{ is exact solution.}$$

$$\longrightarrow x^{i} = e^{i} + \hat{x}$$

Then substitute this, x^{i} to $(*)$

$$\hat{x} + e^{i} = H'(\hat{x} + e^{i \overset{\text{minus}}{+} 1}) + d.$$

since $\hat{x}$ is a solution it will satisfy $(*)$ exactly.

Thus cancelling ;

$$e^{i} = H'e^{i-1} \qquad i = 0,1\ldots$$

$$e^{i-1} = H'e^{i-2}$$

$$\vdots$$

$$e^{1} = H'e^{0}$$

Hence,

$$\boxed{e^{i} = (H')^{i}e^{0}}$$

If we want this iteration to converge, then we must have the norm of the error vector approach to zero progressively with the iterations.

$$\lim_{i\to\infty} \|H^{i}\| = 0 \qquad\qquad \lim_{i\to\infty}(H^{i}) \longrightarrow \underline{0}$$

where $\|\cdot\|$ is defined in general sense.

Thus H should approach to zero as it is powered.

Theorem :

<u>A square matrix H^{p}</u> ← integer exponent. <u>approaches to zero matrix as p increases when the magnitude of its eigenvalues are less than 1.0.</u>

Eigenvalues of a matrix are the roots of the polynomial (known as the characteristic polynomial) which is defined as :

$$|SI - H| = 0$$

Property :

i) For every square matrix H with distinct eigenvalues, there is a nonsingular matrix Q known as the modal matrix satisfying

$$Q^{-1}HQ = \Lambda$$

Where Λ is a diagonal matrix with eigenvalues on its diagonals.

ii) Furthermore, any function of H matrix satisfy the following,

$$f(H) = Qf(\Lambda)Q^{-1}$$

Proof of theorem ;

$$f(H) = H^{p} \quad \text{hence} \quad H^{p} = Q\underset{\uparrow}{\Lambda^{p}}Q^{-1} \qquad \square$$

$$\begin{bmatrix} \lambda_1^{p} & & 0 \\ & \lambda_2^{p} & \\ 0 & & \ddots\ \lambda_n^{p} \end{bmatrix}$$

Conclusion : GS or JI will converge if max. eigenvalue in magnitude of H is less than 1.0.

Theorem : <u>If λ_m is the maximum in magnitude eigenvalue of H, then</u>

$$\underline{|\lambda_m| \leq \|H\|}$$

where ‖·‖ is defined either

i) $\|H\| = \max\limits_{1\le j\le n} \sum_{i=1}^{n} |h_{ij}|$ column norm.

ii) $\|H\| = \max\limits_{1\le \underset{i}{\cancel{j}}\le n} \sum_{j=1}^{n} |h_{ij}|$ row norm

proof : if λ is an eigenvalue then $\exists$ a vector V known as the eigenvector satisfying

$$HV = \lambda V$$

or

$$\begin{bmatrix} h_{11} & \cdots & h_{1n} \\ \vdots & & \vdots \\ h_{n1} & \cdots & h_{nn} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_n \end{bmatrix} = \lambda \begin{bmatrix} V_1 \\ \vdots \\ V_n \end{bmatrix}$$

or

$$\sum_{j=1}^{n} h_{ij} V_j = \lambda V_i \qquad i = 1,2 \ldots n$$

$$\left| \sum_{j=1}^{n} h_{ij} V_j \right| = \left| \lambda V_i \right|$$

$$\sum_{j=1}^{n} |h_{ij}||V_j| \geqslant |\lambda||V_i| \qquad i = 1,2 \ldots n$$

or explicitly,

$i=1 \quad |h_{11}||V_1| + |h_{21}||V_2| + \cdots + |h_{1n}||V_n| \geqslant |\lambda||V_1|$

$i=n \quad |h_{n1}||V_1| + \ldots + |h_{nn}||V_n| \geqslant |\lambda||V_n|$

adding the corresponding term columnwise,

$$\underbrace{\sum_{k=1}^{n} |h_{k1}||V_1|}_{\text{norm of the first column}} + \ldots + \underbrace{\sum_{k=1}^{n} |h_{kn}||V_n|}_{\text{norm of the last column}} \geqslant |\lambda| \sum_{k=1}^{n} |V_k|$$

$$\max_{1\le j\le n} \sum_{k=1}^{n} |h_{kj}| \underbrace{\left[|V_1| + \cdots + |V_n| \right]}_{\cancel{\sum_{k=1}^{n} |V_k|}} \geqslant |\lambda| \cancel{\sum_{k=1}^{n} |V_k|}$$

sufficient but not necessary → $\max\limits_{1\le j\le n} \sum_{k=1}^{n} |h_{kj}| \geqslant |\lambda|$ or

$\geqslant |\lambda_{max}|$

Theorem : If the coefficient matrix A in the system

$$Ax = b$$

is diagonally dominant then the algorithm will allways converges.

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

Possible convergence patterns:

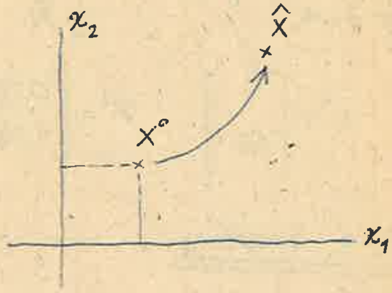
$Ax = b$

Assume that A : 2×2

i.e.

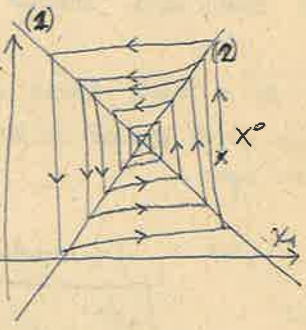
$a_{11}x_1 + a_{12}x_2 = b_1$ (1)

$a_{12}x_1 + a_{22}x_2 = b_2$ (2)

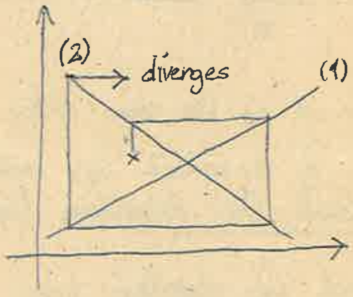


$x_1 = \frac{1}{a_{11}}\left[b_1 - a_{12}x_2\right]$

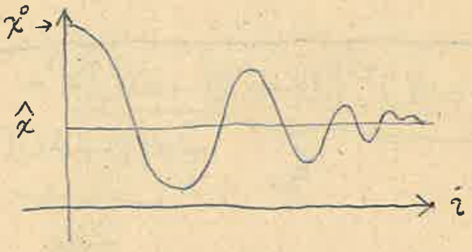
$x_2 = \frac{1}{a_{22}}\left[b_2 - a_{21}x_1\right]$



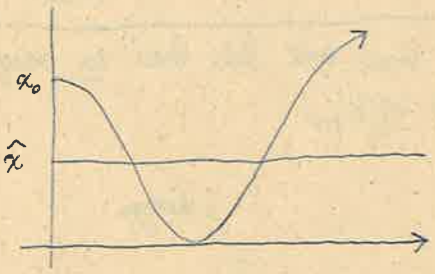
Cobweb, oscillatory convergent



decrease of error with the iterations :

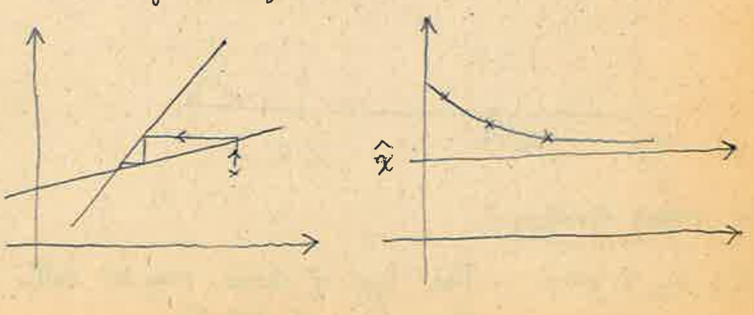


oscillatory convergent.



divergent.

monotonically convergent :



Example :

$$\begin{aligned} 2x_1 + x_2 &= 2 \\ x_1 - 2x_2 &= -2 \end{aligned} \longrightarrow \begin{aligned} \hat{x}_1 &= 0.4 \\ \hat{x}_2 &= 1.2 \end{aligned}$$

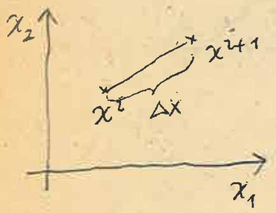
$$\left.\begin{aligned} x_1 &= \frac{-1}{2}x_2 + 1 \\ x_2 &= \frac{1}{2}x_1 + 1 \end{aligned}\right\} \rightarrow H = \begin{bmatrix} 0 & -1/2 \\ 1/2 & 0 \end{bmatrix}, d = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$x^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad x^1 = \begin{bmatrix} 1 \\ 1.5 \end{bmatrix} \quad x^2 = \begin{bmatrix} 0.25 \\ 1.25 \end{bmatrix} \quad x^3 = \begin{bmatrix} 0.403 \\ 1.201 \end{bmatrix}$$

Acceleration of iterations :

GS method is never used without acceleration in practice.

- Acceleration is a small linear extrapolation of the solution point applied to every variable at each iteration cycle.



1 ≤ acceleration factor ≤ 2

$$\boxed{x_j^{i+1} = x_j^i + \alpha(\underbrace{x_j^{i+1} - x_j^i}_{\Delta x_j^{i+1}})}$$

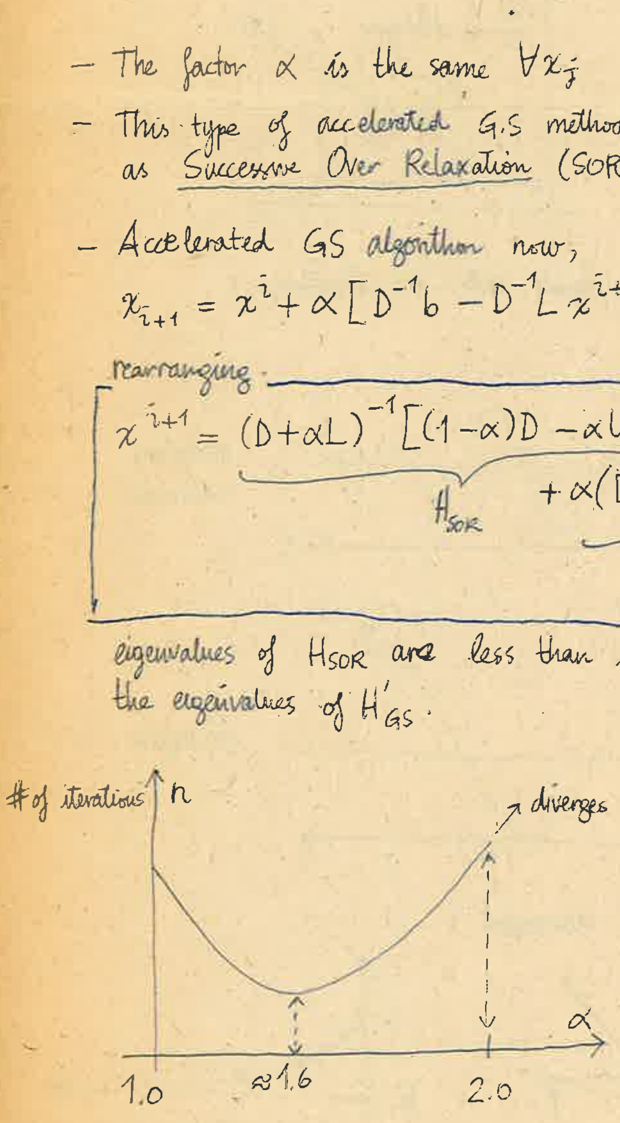
- The factor α is the same $\forall x_j$
- This type of accelerated G.S method is known as Successive Over Relaxation (SOR) method.
- Accelerated GS algorithm now,

$$x_{i+1} = x^i + \alpha[D^{-1}b - D^{-1}Lx^{i+1} - D^{-1}Ux^i - x^i]$$

rearranging

$$\boxed{x^{i+1} = \underbrace{(D+\alpha L)^{-1}[(1-\alpha)D - \alpha U]}_{H_{SOR}}x^i + \underbrace{\alpha(D+\alpha L)^{-1}b}_{d_{SOR}}}$$

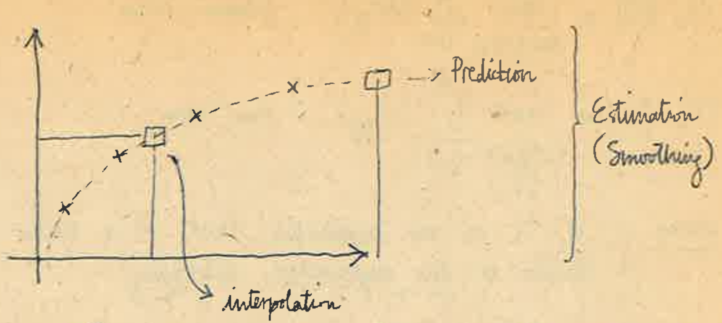
eigenvalues of H_{SOR} are less than in magnitude the eigenvalues of H'_{GS}.



For complex iterations

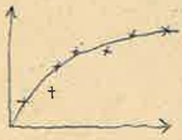
$\alpha = \alpha_R + j\alpha_I$ This type of choice may be better than using a real α.

NUMERICAL APPROXIMATION

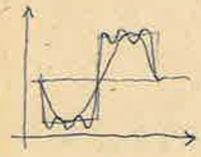


Definition : Numerical approximation deals with the problem of approximating non-arithmetic (experimental) quantities by analytical quantities and specifying the possible error bound associated with the approximation.
Two important criteria in the approximation are ;
1. Degree of accuracy
2. The amount of computation to achieve that accuracy.

Types of approximation :



least-squares approximation :
Determination of a curve which passes through the given sets of points in a least-squares error sense.



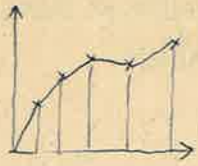
Fourier approximation
Representation of a "complicated" function in terms of relatively simpler functions (sin, cos etc)

$\left.\begin{matrix} \sin nx \\ \cos nx \end{matrix}\right\}_{n=0}^{n_{max}}$ basis functions

Taylor approximation

" " (Taylor here)
↓
polynomials

Linear approximation



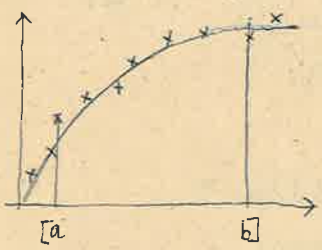
There, we assume that the function is linear between two adjacent points.
(Trapezoidal Rule)

The approximating function may then be defined in terms of the above basis functions as follows:

$$f(x) = \sum_{i=0}^{n} a_i g_i(x) \qquad g_i(x) = \begin{cases} \{\sin ix, \cos ix\}_{i=0}^{n} & \text{(Fourier)} \\ \{x^i\}_{i=0}^{n} & \text{(Taylor)} \end{cases}$$

↓ coefficients of linear combination

① Least squares approximation ;



$$P(x) = \sum_{j=0}^{n} a_j x^j$$

Let the approximated function be f(x)
" " approximating " " P(x)
Let the set of data points be $\left\{\begin{matrix} x_i \\ f(x_i) \end{matrix}\right\}_{i=1}^{m}$

Then the error will be

$$\begin{aligned} e(x_i) &= f(x_i) - P(x_i) \\ &= f(x_i) - \sum_{j=0}^{n} a_j x_i^j \end{aligned}$$

We want to minimize the sum of the squares of the errors

$$S = \sum_{i=1}^{m}\left[f(x_i) - \sum_{j=0}^{n} a_j x_i^j\right]^2$$

The problem is to minimize S w.r.t a_j coefficients.

Which is solved by differentiating S partially w.r.t each a_j and then equating to zero.

$$\frac{\partial S}{\partial a_j} = 0 \quad j = 0, \dots, n$$

$$S = \sum_{i=1}^{m} \left[a_0 + a_1 x_i + a_2 x_i^2 \dots\dots + a_n x_i^n - y_i\right]^2$$

$$\frac{\partial S}{\partial a_j} = 0$$

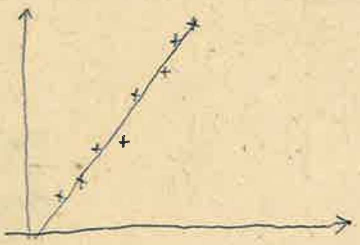
$$= \cancel{2} \sum_{i=1}^{m} \left[a_0 + a_1 x_i + \dots\dots + a_n x_i^n - y_i\right] \times x_i^{J} = 0$$

$$j = 0, 1 \dots, n$$

$$a_0, a_1, \dots, a_n$$

We have n+1 equations and unknowns

the least squares line :



Let the line be $P(x) = a_0 + a_1 x$

Chose the line so that

$$S = \sum_{j=1}^{m} (a_0 + a_1 x_j - y_j)^2 \text{ is a min.}$$

$$\frac{\partial S}{\partial a_0} = 2 \sum_{j=1}^{m} (a_0 + a_1 x_j - y_j) = 0$$

$$\frac{\partial S}{\partial a_1} = 2 \sum_{j=1}^{m} (a_0 + a_1 x_j - y_j)\, x_j = 0$$

020481

yielding ;

$$\sum_{j=1}^{m} y_j = m a_0 + a_1 \sum_{j=1}^{m} x_j$$

$$\sum_{j=1}^{m} y_j x_j = a_0 \sum_{j=1}^{m} x_j + a_1 \sum_{j=1}^{m} x_j^2$$

$$\begin{bmatrix} m & \sum_{j=1}^{m} x_j \\ \sum_{j=1}^{m} x_j & \sum_{j=1}^{m} x_j^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{m} y_j \\ \sum_{j=1}^{m} y_j x_j \end{bmatrix}$$

or, we can write ;

$$\underbrace{\begin{bmatrix} 1 & 1 & \dots\dots & 1 \\ x_1 & x_2 & \dots\dots & x_m \end{bmatrix}}_{H^T} \underbrace{\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_m \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} a_0 \\ a_1 \end{bmatrix}}_{X} = \underbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_m \end{bmatrix}}_{H^T} \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_{b}$$

the least squares parabola ;

$$\underbrace{\underbrace{\begin{bmatrix} 1 & 1 & \dots\dots & 1 \\ x_1 & x_2 & \dots\dots & x_m \\ x_1^2 & x_2^2 & \dots\dots & x_m^2 \end{bmatrix}}_{H^T} \underbrace{\begin{bmatrix} \; \\ \; \\ \; \end{bmatrix}}_{H}}_{A} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}}_{x} = \underbrace{\underbrace{\begin{bmatrix} \; & \; \\ \; & \; \end{bmatrix}}_{H^T} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_{b}$$

OR : n : # of points

$$\boxed{\begin{bmatrix} n & \sum x & \sum x^2 \\ \sum x & \sum x^2 & \sum x^3 \\ \sum x^2 & \sum_{j=1}^{m} x_j^3 & \sum x^4 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum y \\ \sum xy \\ \sum x^2 y \end{bmatrix}}$$

PARABOLA (degree = 2)

<u>Example</u> : Approximate fit (linear approximation, linear fitting curve fitting) (degree = 1)

Fit a least square line to the following data :

x	1	3	4	6	8	9	11	14
y	1	2	4	4	5	7	8	9

Solution:

$\sum y = 40$

$\sum x = 56$

$\sum x^2 = 524$

$\sum xy = 364$

$$\begin{bmatrix} 8 & 56 \\ 56 & 524 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 40 \\ 364 \end{bmatrix}$$

$\downarrow$

$a_0 = 0.545$

$a_1 = 0.636$

$P(x) = 0.545 + 0.636x$

<u>Lagrangian Interpolation</u> :

If the given data points are equally displaced on x axis, then they are called regular base points. Suppose that,

$\{x_i\}_{i=0}^{n}$ are n+1 <u>distinct</u> set of data points.

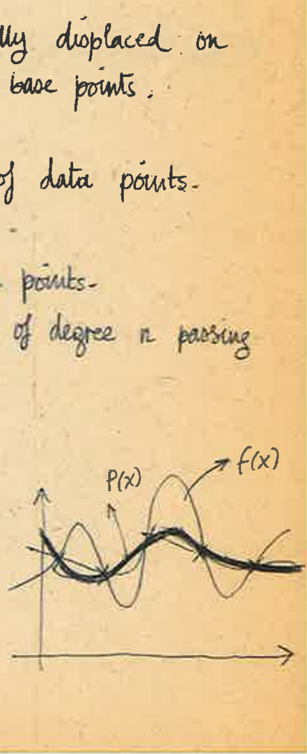
The data points are also called base points.

The problem is to find a polynomial of degree n passing through <u>all</u> these points.

$$P(x) = \sum_{j=0}^{n} a_j x^j$$

with the condition that,

$$P(x_i) = f(x_i) \quad i = 0, \dots, n$$



Problem is to determine the coefficients $\{a_i\}_{i=0}^{n}$

The easiest way is to write that,

$$\sum_{j=0}^{n} a_j x_i^j = f(x_i) \quad i=0,\dots,n$$

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}$$

if you choose distinct n+1 base points then this coefficient matrix can be proved to be always nonsingular.

This coefficient matrix is known as Van Der Monde matrix.

This method is although always applicable but, not used widely.

Lagrange IP :

We may express P(x) in the form

$$P(x) = L_0(x) f(x_0) + L_1(x) f(x_1) + \dots + L_n(x) f(x_n)$$

↑

Elementary Lagrange polynomials

$$P(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

Where each $L_k(x)$ $k=0,\dots,n$ is a polynomial of degree $\le n$ defined as ;

$$L_k(x) = \prod_{\substack{j=0 \\ j\ne k}}^{n} \frac{(x-x_j)}{(x_k-x_j)} = \frac{(x-x_0)(x-x_1)\cdots(x-x_n)}{(x_k-x_0)\cdots\cdots(x_k-x_n)}$$

there is no (x_k-x_k) term!

MT1
APRIL 30th

060481

we may write the 1st order div. diff. alternatively as ;

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

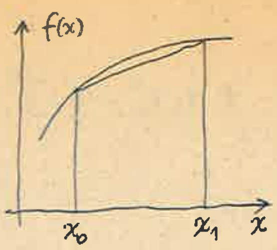
By induction it can be seen that, nth order div. diff. is :

$$f[x_n,x_{n-1},\dots,x_0] = \frac{f(x_n)}{(x_n-x_{n-1})(x_n-x_{n-2})\dots(x_n-x_0)} + \frac{f(x_{n-1})}{(x_{n-1}-x_n)(x_{n-1}-x_{n-2})\dots(x_{n-2}-x_0)} + \dots + \frac{f(x_0)}{(x_0-x_n)(x_0-x_{n-1})\dots(x_0-x_1)}$$

$$= \sum_{i=0}^{n} \frac{f(x_i)}{\prod_{j=0}^{n}(x_i-x_j)}$$

$(x_i - x_j)$ term is absent.

Application of div. differences to Interpolation theory



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

There is at least one point $\xi \in [x_0,x_1]$

$\ni \quad f[x_1,x_0] = f'(\xi)$

$f[x,x_0] \cong f[x_1,x_0]$ the interval is small

$$f(x) = \underbrace{\underbrace{f(x_0)}_{\text{constant}} + \underbrace{(x-x_0)}_{\text{poly}} \underbrace{f[x_1,x_0]}_{\text{slope}}}_{\substack{P(x) \\ \text{linear}}} + \underbrace{R(x)}_{\text{error}}$$

$$= P(x) + R(x)$$

Now, solve for R(x) ;

max. value of f'' in (x_0,x) occurs at ξ

$$R(x) = (x-x_0)(x-x_1) f[x,x_1,x_0] = (x-x_0)(x-x_1) f''(\xi)$$

$$\xi \in [x_0,x]$$

$f''(x)$ or $f[x,x_0,x_1]$

needs the evaluation of three distinct points x_0, x_1, x_2. However in linear interpolation we have only two points x_0, x; hence we can roughly estimate the error bound by using a dummy point ξ.

$$R(x) = (x-x_0) \underbrace{\frac{f(x)-f(x_0)}{(x-x_0)}}_{f[x,x_0]} - (x-x_0) f[x_1,x_0] =$$

$$(x-x_0) \underbrace{\overbrace{\left\{ \frac{f[x_1,x] - f[x_1,x_0]}{(x-x_1)} \right\}}^{\frac{f''(\xi)}{2} \quad \xi\in[x,x_0]}}_{\text{Second order div. diff.}} (x-x_1)$$

Now, error in the nth order interpolation with div. diff. is by induction

$$f(x) = P_n(x) + R_n(x)$$

where

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

$$+ \dots + (x-x_0)(x-x_1)(x-x_2)\dots (x-x_{n-1}) f[x_n,\dots,x_0]$$

$$= f[x_0] + \sum_{m=1}^{n} \left[\prod_{i=0}^{m-1} (x-x_i) f[x_n,\dots,x_0] \right]$$

(NEWTON's formula)

Similarly, the error polynomial :

$$R_n(x) = (x-x_0)(x-x_1)\dots(x-x_n)\underbrace{f[x, x_n, x_{n-1}, \dots, x_0]}_{\frac{f^{(n+1)}(\xi)}{(n+1)!}}$$

where $\xi \in [x, x_n, \dots, x_0]$

Example : Find an IP of degree ≤ 3 passing through

x	$f(x)$
1	−5
1	1
3	25
4	55

form the divided diff. table :

α_i		$f(x_i)$	$f[1]$	$f_2[,,]$	$f_3[,,,]$
$x_0=0$	$f(x_0) =$	-5	$f[x,x_0]=6$		
$x_1=1$	$f(x_1) =$	1	$f[x_2,x_1]=12$	$f[x_2,x_1,x_0]=2$	$f[x_3,x_2,x_1,x]$
$x_2=3$	$f(x_1) =$	25	$f[x_3,x_2]=30$	$f[x_3,x_2,x_1]=6$	$= 1 = \frac{6-2}{x_3-x_0} = \frac{4}{4}$
$x_3=4$	$f(x_3) =$	55			

We can use any one of these paths to write IP.

now, using NF :

$$\boxed{P(x) = f[x_0] + (x-x_0)f[x_1,x_0] + (x_0-x_0)(x-x_1)f[x_2,x_1,x_0] + (x-x_0)(x-x_1)(x-x_2)f[x_3,x_2,x_1,x_0]}$$

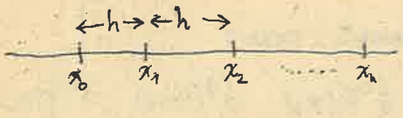
$$= -5 + (x-0)6 + (x-0)(x-1)2 + (x-0)(x-1)(x-3)1$$

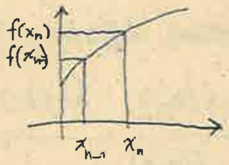
$$= x^3 - 2x^2 + 7x - 5$$

The path is immeterial, $P(x)$ is unigue

160481

POLYNOMIAL INTERPOLATION WITH REGULAR BASE POINTS





Suppose that the base points are regularly displaced, with step lengths h.

$$x_{i+1} - x_i = h$$

Difference Operators

- Forward difference operator :

$$\Delta^0 f(x) = f(x)$$

first order difference. $\Delta^1 f(x) = f(x+h) - f(x)$

$$\Delta^2 f(x) = \Delta^1 f(x+h) - \Delta^1(f(x)) = f(x+2h) - 2f(x+h) + f(x)$$

$$\vdots$$

$$\Delta^k f(x) = \Delta^{k-1} f(x+h) - \Delta^{k-1} f(x) \quad k = 1, 2 \dots$$

High order diff. operations may be expressed in terms of the function f(x) as follows (Show! by induction)

$$\Delta^j f(x) = \sum_{k=0}^{j} (-1)^{j-k} \frac{j!}{k!(j-k)!} f(x+kh) = \sum_{k=0}^{j} (-1)^{j-k} \binom{j}{k} \cdot f(x+kh)$$

Example :

Consider the polynomial :

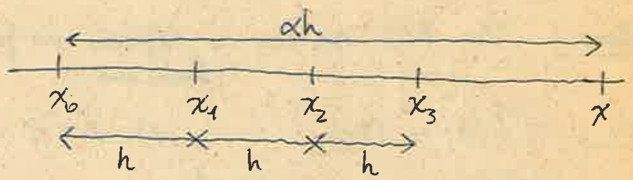
$P(x) = x^3 - 2x^2 + 7x - 5 = 0$ which is tabulated at 5 base points with $h = 1.0$

x_i	$f(x_i)$	$\Delta f(x_i)$	$\Delta^2 f(x_i)$	$\Delta^3 f(x_i)$	$\Delta^4 f(x_i)$
$x_0=0$	−5	6	2	6	0
$x_1=1$	1	8	8	6	
$x_2=2$	9	16	14		
$x_3=3$	25	30			
$x_4=4$	55				

h

Think about the proof of the property that, the m^{th} order difference operation of a polynomial with n^{th} order when $m>n$ is always zero.

Interpolation with forward differences :



if $x - x_0 = \alpha h$ (α is a positive integer)

$$x - x_1 = \alpha h - h = h(\alpha-1)$$

$$x - x_2 = h(\alpha-2)$$

$$\vdots$$

$$x - x_n = h(\alpha-n)$$

The Newton IP. with div. diff. was :

$$P(x) + R_n(x) = f[x_0] + (x-x_0)f[x_1,x_0] + (x-x_0)(x-x_1)f[x_2,x_1,x_0] + \dots + (x-x_0)(x-x_1)\dots(x-x_{n-1}) \cdot f[x_n, x_{n-1}, \dots x_0] + R_n(x)$$

Let's now modify the above NIP formula with forward difference operators:

$f[x_0] = f(x_0) \qquad x_1 = x_0 + h$

$$f[x_1,x_0] = \frac{f(x_1)-f(x_0)}{x_1-x_0} = \frac{f(x_0+h)-f(x_0)}{h} = \frac{\Delta f(x_0)}{h}$$

$$f[x_2,x_1,x_0] = \frac{f[x_2,x_1] - f[x_1,x_0]}{\underbrace{x_2-x_0}_{\to 2h}} = \frac{\frac{\Delta f(x_1)}{h} - \frac{\Delta f(x_0)}{h}}{2h}$$

$$= \frac{\Delta f(x_1) - \Delta f(x_0)}{2h^2} = \frac{\Delta^2 f(x_0)}{2h^2}$$

or, in general ;

$$f[x_n, x_{n-1}, \cdots x_0] = \frac{\Delta^n f(x_0)}{n!\, h^n}$$

note that
$x - x_0 = \alpha h$
$x = x_0 + \alpha h$

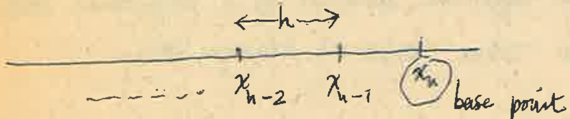
$$P(x) + R_n(x) = P(x_0 + \alpha h) + R_n(x_0 + \alpha h)$$
$$= f(x_0) + \alpha \cancel{h} \frac{\Delta f(x_0)}{\cancel{h}} + \alpha \cancel{h} \frac{(\alpha-1)\cancel{h}\,\Delta^2 f(x_0)}{2!\,\cancel{h^2}}$$
$$+ \dots + \frac{\alpha(\alpha-1)(\alpha-2)\dots(\alpha-n+1)}{n!}\Delta^n f(x_0)$$

The error, (or remainder) may be written as ;

$$R_n(x) = (x-x_0)(x-x_1)\dots(x-x_n)\frac{f^{(n+1)}(\xi)}{(n+1)!} \qquad \xi \in [x_0, x_n]$$

$$= \alpha h(\alpha-1)h \dots$$
$$= h^{n+1}\alpha(\alpha-1)(\alpha-2)\dots(\alpha-n)\frac{f^{(n+1)}(\xi)}{(n+1)!}$$

Backward dif. operator ;



Definition : $x = x_n + \alpha h \quad \alpha \leq 0$

Definition : The B.D. operator is shown and defined as follows ;

'Del' →

$$\nabla^0 f(x) = f(x) \qquad \alpha h \quad \alpha = -1$$
$$\nabla^1 f(x) = f(x) - f(x-h)$$
$$\nabla^2 f(x) = \nabla^1 f(x) - \nabla^1 f(x-h) = f(x) - 2f(x-h) + f(x-2h)$$

Similarly :

$$\nabla^i f(x) = \nabla^{i-1} f(x) - \nabla^{i-1} f(x-h) \qquad i = 1, \cdots 2$$

The backward difference table :

Same problem.

x_i	$f(x_i)$	$\nabla f(x_i)$	$\nabla^2 f(x_i)$	$\nabla^3 f(x_i)$	$\nabla^4 f(x_i)$
0	−5	6	2	6	0
1	1	8	8	6	
2	9	16	14		
3	25	30			
4	55				

F.D

B.D

Newton's formula in terms of B.D

$$x - x_n = \alpha h$$
$$x - x_{n-1} = (\alpha+1)h$$
$$x - x_{n-2} = (\alpha+2)h$$

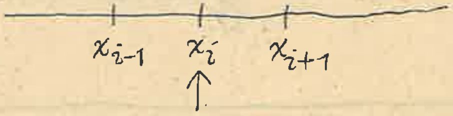
NBIF :

$$f(x_n + \alpha h) = f(x_n) + \alpha\nabla f(x_n) + \frac{\alpha(\alpha+1)}{2!}\nabla^2 f(x_n) + \frac{\alpha(\alpha+1)(\alpha+2)}{3!}\nabla^3 f(x_n) + \dots + \frac{\alpha(\alpha+1)(\alpha+2)\dots(\alpha+n-1)}{n!}\nabla^n f(x_n) + R_n(x_n + \alpha h)$$

Where ;

$$R_n(x_n + \alpha h) = h^{n+1}\alpha(\alpha+1)(\alpha+2)\dots(\alpha+n)\frac{f^{(n+1)}(\xi)}{(n+1)!}$$
$$\xi \in [x_n, x_0]$$

Central difference operations :



Definition : The central difference operations is shown and defined as follows.

$$\delta^0 f(x) = f(x)$$
$$\delta^1 f(x) = f(x + \tfrac{h}{2}) - f(x - \tfrac{h}{2})$$
$$\delta^2 f(x) = \delta^1 f(x + \tfrac{h}{2}) - \delta^1 f(x - \tfrac{h}{2})$$
$$= f(x+h) - 2f(x) + f(x-h)$$

$$\delta^k f(x) = \delta^{k-1} f(x + \tfrac{h}{2}) - \delta^{k-1} f(x - \tfrac{h}{2}) \qquad k = 1, 2 \dots$$

The central difference triangle now :

x_i	$f(x_i)$	$\delta f(x_i)$	$\delta^2 f(x_i)$	$\delta^3 f(x_i)$	$\delta^4 f(x_i)$
x_{-2}	$f(x_{-2})$				
x_{-1}	$f(x_{-1})$	$\delta f(x_{-1} - \frac{h}{2})$	$\delta^2 f(x_{-1})$	$\delta^3 f(x_0 - \frac{h}{2})$	
x_0	$f(x_0)$	$\delta f(x_0 - \frac{h}{2})$	$\delta^2 f(x_0)$		$\delta^4 f(x_0)$
x_1	$f(x_1)$	$\delta f(x_0 + \frac{h}{2})$	$\delta^2 f(x_1)$	$\delta^3 f(x_0 + \frac{h}{2})$	
x_2	$f(x_2)$	$\delta f(x_1 + \frac{h}{2})$			

h

unknown point $(x_{-1} - \frac{h}{2})$

Newton's Central Difference interpolation NCDIP

$$f(x_0 + \alpha h) = f(x_0) + \alpha\delta f(x_0 + \tfrac{h}{2}) + \frac{\alpha(\alpha-1)}{2!}\delta^2 f(x_0) + \alpha(\alpha-1)(\alpha+1)\delta^3 \frac{f(x_0 + \frac{h}{2})}{3!} + \dots + R_n(x_0 + \alpha h)$$

when $\alpha = \frac{x - x_0}{h} \longrightarrow x = x_0 + \alpha h$

NCDIP (for the upper path)

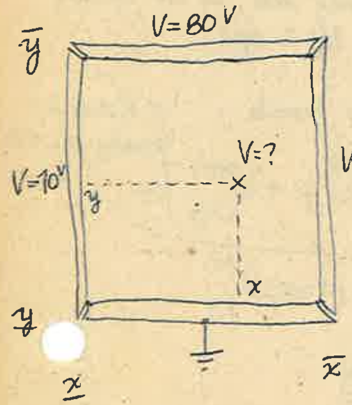
$$f(x_0+\alpha h) = f(x_0) + \alpha\,\delta f(x_0-\tfrac{h}{2}) + \frac{\alpha(\alpha+1)}{2!}\delta^2 f(x_0)$$

$$+ \alpha(\alpha-1)(\alpha+1)\frac{\delta^3 f(x_0-\frac{h}{2})}{3!} + \dots + R_n(x_0+\alpha h)$$

The Sterling formula :

is the average of above two formulae.

SOLUTION OF PDE



$\nabla^2 V = 0$ Laplace's equation

$V(x,y)$ $x \in [\underline{x}, \bar{x}]$
$y \in [\underline{y}, \bar{y}]$

$$\nabla^2 f(x) = \frac{d^2 f}{dx^2} = f(x+h) - 2f(x) + f(x-h)$$

$$\frac{\partial^2 V(x,y)}{\partial x^2} \cong V(x+h,y) - 2V(x,y) + V(x-h,y)$$

200481

The general form of PDE of two variables,

$$AU_{xx} + BU_{xy} + CU_{yy} + DU_x + EU_y + FU = 0$$

where, $U = U(x,y)$

$U_x = \dfrac{\partial U}{\partial x}$ $U_y = \dfrac{\partial U}{\partial y}$

$U_{xx} = \dfrac{\partial^2 U}{\partial x^2}$ $U_{yy} = \dfrac{\partial^2 U}{\partial y^2}$ $U_{xy} = \dfrac{\partial^2 U}{\partial x \partial y}$

A, B, C, D, E, F : constants

We also have initial and final boundary values.

If only, A and C are nonzero, then we obtain Laplace's equation.

$$AU_{xx} + BU_{yy} = -\sigma = \frac{\rho}{\varepsilon_0} \quad \text{Poisson's eq.}$$

$$= 0 \quad \text{Laplace's eqn.}$$

Possible conditions :

if $B^2 - 4AC > 0$ hyperbolic PDE
$B^2 - 4AC = 0$ Parabolic PDE
$B^2 - 4AC < 0$ Elliptic PDE

transformation of Partial derivatives to differences

(FD) Forward difference type transformation:

$$U_x = \frac{\partial U(x,y)}{\partial x} = \frac{U(x+h,y) - U(x,y)}{h}$$

(BD) Backward difference type representation

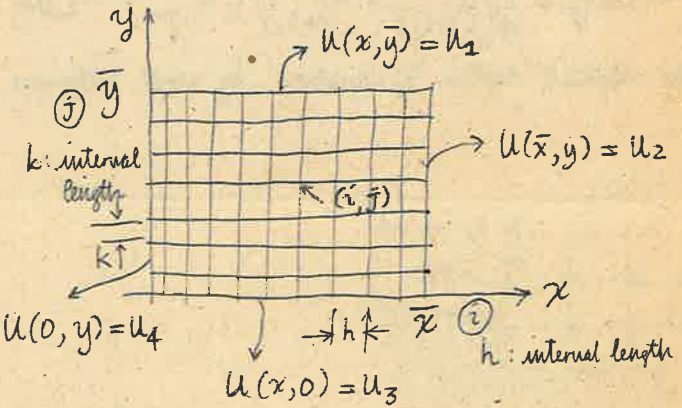
$$U_x = \frac{U(x,y) - U(x-h,y)}{h}$$

(CD) Central difference type

$$U_x = \frac{U(x+h,y) - U(x-h,y)}{2h}$$

Similarly for second partials :

$$U_{xx} = \frac{\partial^2 U(x,y)}{\partial x^2} = \frac{U(x+h,y) - 2U(x,y) + U(x-h,y)}{h^2}$$



Solve $\nabla^2 U$ in this region

$$AU_{xx} + BU_{yy} = 0$$

$$A\left[\frac{U(x+h,y) - 2U(x,y) + U(x-h,y)}{h^2}\right] + B\left[\frac{U(x,y+k) - 2U(x,y) + U(x,y-k)}{k^2}\right] = 0$$

Assume for simplicity $A = B = 1$ and multiply both sides by k^2

$$\frac{k^2}{h^2}\left[U_{i+1,j} - 2U_{ij} + U_{i-1,j}\right] + \left[U_{i,j-1} - 2U_{i,j} + U_{i,j+1}\right] = 0$$

$i = 0, 1, 2, \dots, n$
$j = 0, 1, 2, \dots, m$

$\bar{x} = nh$
$\bar{y} = mk$

We have $(n-1) \times (m-1)$ unknowns.

For simplicity again choose $k=h$

$$u_{i+1,j} + u_{i-1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j} = 0$$

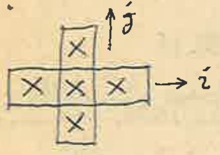
we may write $(n-1)\times(m-1)$ such equations for each interior points. hence # of equations = # of unknowns.

if you combine these equations in matrix form ;

$$\begin{bmatrix} 4 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 4 & -1 & \ddots & 0 & \vdots \\ 0 & -1 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} u_{1,2} \\ u_{1,3} \\ \vdots \\ u_{n-1,m-1} \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \end{bmatrix} \leftarrow \text{not true for every type!}$$

230481

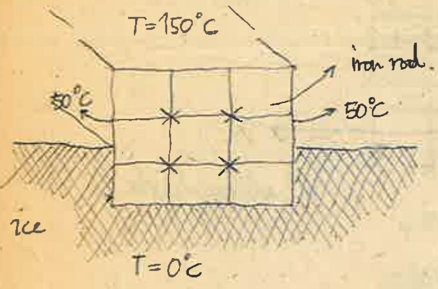
1. Write down the following rule for every interior node :



$$\longrightarrow u_{i,j} = \frac{1}{4}\left[u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}\right]$$

2. Solve the resultant system of equations by usual techniques.

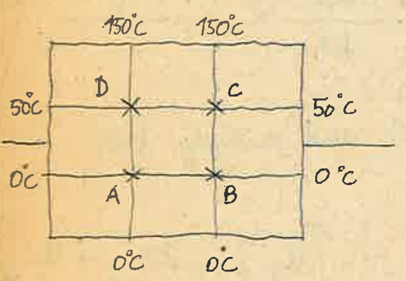
EX :



Determine the Temperature variation in the rod by using 4 interior nodes.

Laplace : $\nabla^2 T = 0$

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$



$$T_A = \frac{1}{4}\left[0 + 0 + T_D + T_B\right]$$

$$T_B = \frac{1}{4}\left[T_A + 0 + 0 + T_C\right]$$

$$T_C = \frac{1}{4}\left[150 + T_B + 50 + T_D\right]$$

$$T_D = \frac{1}{4}\left[50 + T_C + 150 + T_A\right]$$

We can solve the above system of eqn's by any suitable direct method.

Iterative methods (Relaxation methods) :

In iterative methods we write again the same equations for every interior node.

$$\longrightarrow u_{i,j} = \frac{1}{4}\left[u_{i,j+1} + u_{i,j-1} + u_{i+1,j} + u_{i-1,j}\right] \quad (*)$$

$i = 1, \ldots, m-1$
$j = 1, \ldots, m-1$ There are $(m-1)\times(n-1)$ interior nodes.

in iterative solution method, the presently available solution points are substituted to RHS of $(*)$ and a new estimate of a variable is obtained.

Thus the iterative form of $(*)$ may be written as ;

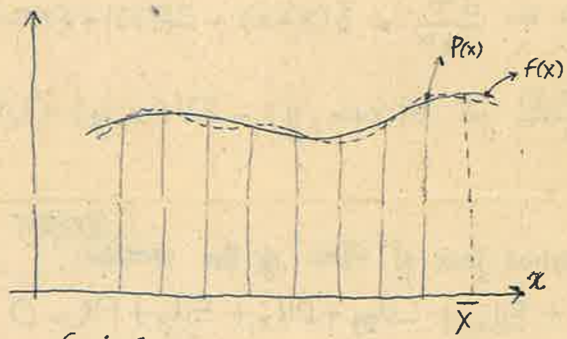
$$u_{i,j}^{(n+1)} = \frac{1}{4}\left[u_{i,j+1}^{(n)} + \underbrace{u_{i,j-1}^{(n+1)}} + u_{i+1,j}^{(n)} + \underbrace{u_{i-1,j}^{(n+1)}}\right]$$

(Assuming that we are sweeping from left to right) .

If we apply acceleration to the above formula : $1 \leq \alpha \leq 2$ usually $\alpha = 1.8$

$$u_{i,j}^{(n+1)} = \frac{\alpha}{4}\left[u_{i,j+1}^{(n)} + u_{i,j-1}^{(n+1)} + u_{i+1,j}^{(n)} + u_{i-1,j}^{(n+1)}\right] + (1-\alpha)\,u_{i,j}^{(n)}$$

NUMERICAL INTEGRATION



e.g : $\int \underbrace{e^{\sin x^2}\cos x}_{f(x)}\,dx$

x	f(x)
⋮	⋮

There are two types of numerical integration formulas :

1. Closed type numerical integration formula.
2. Open type " " " .

Closed integration formulas uses data about f(x) at both limits of integration, while open formulas do not require data about f(x) at both limits.

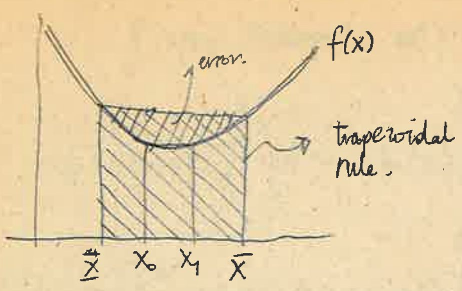
In this chapter, we will always assume regular base points.

i.e $x_{i+1} - x_i = h$ = step size of integration.

Consider Newton's forward formula : (NFF)

$$f(x_0+\alpha h) = f(x_0) + \alpha\Delta f(x_0) + \frac{\alpha(\alpha-1)}{2!}\Delta^2 f(x_0) + \frac{\alpha(\alpha-1)(\alpha-2)}{3!}\Delta^3 f(x_0) + \cdots + \cdots + \frac{\alpha(\alpha-1)\ldots(\alpha-n+1)}{n!}\Delta^n f(x_0) + R_n(x_0+\alpha h)$$

Newton-Cotes closed integration formula



Here we have 2 points only, thus we have a straight line passing thru these points.

NFF becomes:

$$f(x) = f(x_0+\alpha h) = f(x_0) + \alpha \Delta f(x_0) + \underbrace{R_1(x_0+\alpha h)}_{\text{error}}$$

where $R_1(x_0+\alpha h) = h^2 \alpha(\alpha-1) \frac{f^{(2)}(\xi)}{2!}$ $\quad \xi \in [x_0, x_1]$

Let's now integrate this f(x):

$$\int_{x_0}^{x_1} f(x)\,dx = \int_{x_0}^{x_1} P_1(x)\,dx + \int_{x_0}^{x_1} R_1(x)\,dx$$

Let's make a change of variable of integration:

$x = x_0 + \alpha h \qquad x = x_0 \rightarrow \alpha = 0$
$dx = h\,d\alpha \qquad x = x_1 \rightarrow \alpha = 1$

$$= h\int_0^1 P_1(x_0+\alpha h)\,d\alpha = h\int_0^1 \left[f(x_0) + \alpha \Delta f(x_0)\right] d\alpha$$

$$= h\left[\alpha f(x_0) + \frac{\alpha^2}{2}\Delta f(x_0)\right]_0^1$$

$$= h\left(f(x_0) + \frac{1}{2}\Delta f(x_0)\right)$$

$$= h\left[f(x_0) + \frac{f(\overset{x_1}{\widetilde{x_0+h}}) - f(x_0)}{2}\right]$$

$$= \frac{h}{2}\left[f(x_0) + f(x_1)\right] \text{ trapezoidal rule.}$$

The error in trapezoidal rule;

$$e = \int_{x_0}^{x_1} R_1(x)\,dx = h\int_0^1 R_1(x_0+\alpha h)\,d\alpha$$

$$= h^3 \int_0^1 \underbrace{\alpha(\alpha-1)}_{g(x)} \underbrace{\frac{f^{(2)}(\cancel{\alpha})}{2!}}_{f(x)}\,d\alpha \qquad \cancel{\xi \in [x_0, x_1]}$$

integral mean-value Theorem;

If two functions f(x) and g(x) are continuous over $[x_0, x_1]$ and g(x) is of the same sign over this interval,

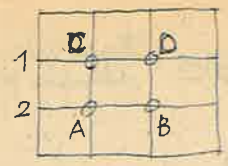
Then $\boxed{\int_{x_0}^{x_1} f(x) g(x)\,dx = f(\xi) \int_a^b g(x)\,dx \quad \text{when } \xi \in [x_0, x_1]}$

→ $\alpha(\alpha-1) < 0$ always

$$h^3 \int_0^1 \alpha(\alpha-1) \frac{f(2)(\cancel{\alpha})}{2!}\,d\alpha = h^3 \frac{f^{(2)}(\xi)}{2!} \overbrace{\int_0^1 \alpha(\alpha-1)\,d\alpha}^{-\frac{1}{6}}$$

$$e = \boxed{-\frac{h^3}{12} f^{(2)}(\xi)} \quad \xi \in [x_0, x_1]$$

Example about PDE



Solve using iterative method (GS)
(SØR)

$$A^{(n+1)} = \frac{1}{4}\left[B^{(n)} + C^{(n)} + 4\right] \qquad n = 0,1,2,\ldots$$

$$B^{(n+1)} = \frac{1}{4}\left[A^{(n+1)} + D^{(n)} + 2\right]$$

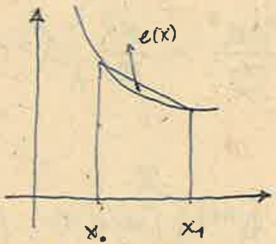
$$C^{(n+1)} = \frac{1}{4}\left[A^{(n+1)} + D^{(n)} + 2\right]$$

$$D^{(n+1)} = \frac{1}{4}\left[B^{(n+1)} + C^{(n+1)} + 4\right]$$

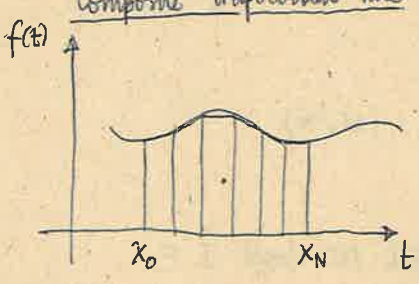
n	A	B	C	D
0	0	0	0	0
1	1	0.75	0.75	1.35
2	1.375	1.188	1.188	1.59
3	1.593	1.299	1.299	1.650
4	1.650	1.324	1.324	1.662
Exact Sol'n	1.667	1.333	1.333	1.667

Trapezoidal rule (Continued)

$$\int_{x_0}^{x_1} f(x)\,dx = \frac{h}{2}\left[f(x_0) + f(x_1)\right] - \frac{h^3}{12} f^{(2)}(\xi) \quad \xi \in [x_0, x_1]$$



Composite trapezoidal rule



$$\int_A^B f(x)\,dx = \int_{x_0}^{x_1} f(x)\,dx + \ldots\ldots + \int_{x_{N-1}}^{x_N} f(x)\,dx$$

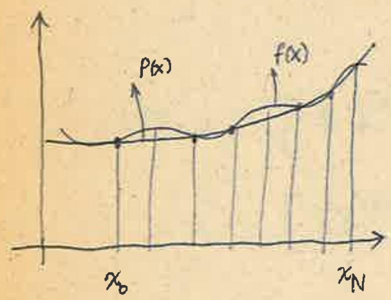
$$= \frac{h}{2}\left[f(x_0) + f(x_1)\right] + \ldots\ldots + \frac{h}{2}\left[f(x_{N-1}) + f(x_N)\right]$$

$$= h\left[\frac{1}{2} f(x_0) + f(x_1) + \ldots + f(x_{N-1}) + \frac{1}{2} f(x_N)\right]$$

The error is the sum of individual errors:

$$e_{TOT} = -\frac{(B-A)}{12} h^2 f^{(2)}(\xi) \quad \xi \in [A,B]$$

HIGHER ORDER INTEGRATION FORMULAS (Newton-Cotes closed integration formulas)



$$\int_a^b f(x)dx \cong \int_a^b P(x)dx$$

$x = x_0 + \alpha h$
$dx = h d\alpha$
$x = x_0 \Rightarrow \alpha = 0$
$x = b \Rightarrow$
$\alpha = \frac{b - x_0}{h} = \bar{\alpha}$

$$\int_a^b P(x)dx = h\int_0^{\bar{\alpha}} P_n(x_0+\alpha h)\,d\alpha$$

$$= h\int_0^{\bar{\alpha}} \Big[f(x_0) + \alpha\Delta f(x_0) + \frac{\alpha(\alpha-1)}{2!}\Delta^2 f(x_0) +$$

$$\frac{\alpha(\alpha-1)(\alpha-2)}{3!}\Delta^3 f(x_0) + \dots + \frac{\alpha(\alpha-1)\dots(\alpha-n+1)}{n!} \times \Delta^n f(x_0)\Big]d\alpha$$

$$= h\Big[\alpha f(x_0) + \frac{\alpha^2}{2}\Delta f(x_0) + \left(\frac{\alpha^3}{6} - \frac{\alpha^4}{4}\right)\Delta^2 f(x_0)$$

$$+ \left(\frac{\alpha^4}{24} - \frac{\alpha^3}{6} + \frac{\alpha^6}{6}\right)\Delta^3 f(x_0) +$$

$$\left(\frac{\alpha^5}{120} - \frac{\alpha^4}{16} + \frac{11\alpha^3}{72} - \frac{\alpha^2}{8}\right)\Delta^4 f(x_0)\Big] (*)$$

Similarly the error form:

$$e = h\int_0^{\bar{\alpha}} R_n(x_0+\alpha h)\,d\alpha = h^{n+2}\int_0^{\bar{\alpha}} \Big[\alpha(\alpha-1)\dots(\alpha-n)\frac{f^{(n+1)}(\xi)}{(n+1)!}\Big]d\alpha \quad (*,*)$$

$(*)$, $(*,*)$ are called NC closed I.F.

We can generate any integration formula with desired order using $(*)$, $(*,*)$

eg: if $n=1$; $\bar{\alpha}=1$ above formula reduces to trapezoidal rule.

SIMPSON'S RULE (3 pt. rule)

set $\bar{\alpha} = n = 2$ (two intermediate point)

$$\int_a^b f(x)dx = \int_{x_0}^{x_2} P_n(x)\,dx = h\int_0^2 P_h(x_0+\alpha h)\,d\alpha$$

$$= h\int_0^2 \Big[f(x_0) + \alpha\Delta f(x_0) + \frac{\alpha(\alpha-1)}{2!}\Delta^2 f(x_0) +$$

$$\frac{\alpha(\alpha-1)(\alpha-2)}{3!}\Delta^3 f(x_0) + \dots\Big]d\alpha$$

$$= h\Big[2f(x_0) + 2\Delta f(x_0) + \frac{1}{3}\Delta^2 f(x_0) +$$

$$\underset{\uparrow}{\underline{0\Delta^3 f(x_0)}} - \frac{1}{90}\Delta^4 f(x_0) + \dots\Big]$$

It may be shown that all even order integration rules are (Simpson (3), 3rd order) exact for polynomials with one degree above their degree.

<u>040581</u>

For the choice of 3 base pts. x_0, x_1, x_2 $f(x)$ may be approximated by a parabola,
$P_2(x) = P_2(x_0+\alpha h)$ ie $\boxed{n=2}$, retaining the first 3 Terms)

$$\int_{x_0}^{x_2} f(x)dx \cong h\Big[2f(x_0) + 2\{f(x_0+h) - f(x_0)\} +$$

$$\frac{1}{3}\{f(x_0+2h) - 2f(x_0+h) + f(x_0)\}\Big]$$

$$= \frac{h}{3}\Big[f(x_0) + 4f(x_1) + f(x_2)\Big]$$

is the most widely used integ. rule in numerical analysis.

<u>Error expression</u>:

<u>Because of the zero coefficient, the error term in Simpson's rule is not obtained when n=2, but when n=3</u> that is:

$$e = h\int_0^2 R_3(x_0+\alpha h)\,d\alpha = h^5\int_0^2 \alpha(\alpha-1)(\alpha-2)(\alpha-3)\frac{f^{(4)}(\xi)}{4!}d$$

$$\xi \in [x_0, x_2]$$

using again <u>integral mean value thm.</u>

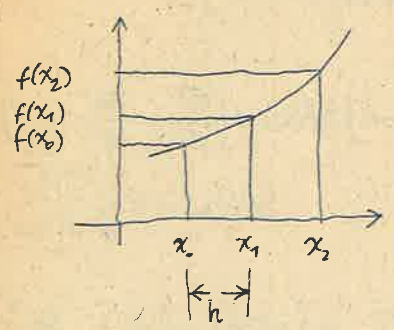
$$e = h^5\frac{f^{(4)}(\xi)}{4!}\int_0^2 \alpha(\alpha-1)(\alpha-2)(\alpha-3)\,d\alpha = -\frac{h^5 f^{(4)}(\bar{\xi})}{90}$$

$$\bar{\xi} \in [x_0, x_2]$$

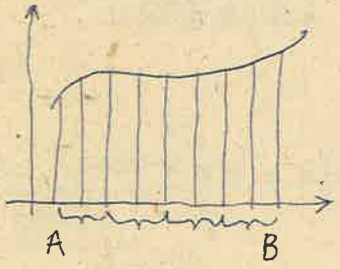
Thus in summary Simpson's rule may be given as ;

$$\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}\left[f(x_0)+4f(x_1)+f(x_2)\right] - \frac{h^5}{90}f^{(4)}(\xi)$$

$$\xi \in [x_0, x_2]$$



Successive application of Simpson R. ↓



We may extend S.R. to an integration ofer an [A,B] by dividing it into an even # 2N sub-intervals

So that :

$x_0 = A$ $x_{2n} = B$.
$x_1 = A+h$
$x_2 = A+2h$

If we sum up these sub-integrations over [A,B] we obtain :

$$\int_A^B f(x)dx = \frac{h}{3}\Big[f(x_0)+4f(x_1)+2f(x_2)+4f(x_3)\ldots$$

$$\ldots 4f(x_{2N-1})+f(x_{2N})\Big] - \frac{Nh^5}{90}f(\bar{\xi})$$

$$\underbrace{B-A = 2Nh}_{\frac{B-A}{180}h^4 f^{(4)}(\xi)}$$

Higher order integration rules :

These rules are not recommended and rarely used in practice due to the fact that coefficients in the formulas become large and so round-off errors become dominating.

$\bar{\alpha}=3$ $n=3$,

$$\int_{x_0}^{x_3} f(x)dx = \frac{3h}{8}\left[f(x_0)+3f(x_1)+3f(x_2)+f(x_3)\right] - \frac{3h^5}{80}f^{(4)}(\xi)$$

← it is not more accurate than simpson

$\bar{\alpha}=4$, $n=4$

$$\int_{x_0}^{x_4} f(x)dx = \frac{2h}{45}\Big[7f(x_0)+32f(x_1)+12f(x_2)+32f(x_3)+7f(x_4)\Big] - \frac{8h^7 f^{(6)}(\xi)}{945}$$

(SHOW!)

$\bar{\alpha}=5$, $n=5$

$$\int_{x_0}^{x_5} f(x)dx = \frac{5h}{288}\Big[19f(x_0)+75f(x_1)+50f(x_2)+ +75f(x_4)+19f(x_5)\Big] - \frac{279}{12096}h^7 f^{(6)}(\xi)$$

Example :

$P(x) = x^3-2x^2+7x-5$

$$\int_1^3 P(x)\,dx \ \underline{\text{exactly}} = 20.6667$$

x	$f(x)$
1	1
1.5	4.375
2	9
2.5	15.625
3	25

a) Trapezoidal rule :

$h=2$, $x_0=1$, $x_1=3$, $f(x_0)=1$, $f(x_1)=25$

$$\int \ = \frac{h}{2}\left[f(x_0)+f(x_1)\right] = 26$$

$$e = -\frac{8}{12}\underbrace{f^{(2)}(\xi)}_{6x-4} = -\frac{2}{3}(6\xi-4) \qquad \xi\in[1,3]$$

max $\in$ is at $x=3$

$$e_{max} = e\Big|_{\xi=3} = -\frac{28}{3}$$

Exact value : $26 \pm \frac{28}{3}$

$$20.66667 \in \left[26 \pm \frac{28}{3}\right]$$

b) Simpson's Rule:

$$\int_1^3 f(x)\,dx \qquad f(x) = x^3 - 2x^2 + 7x - 5$$

actual result : 20.6667

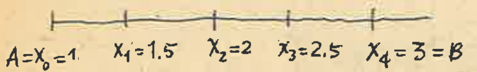
$x_0 = 1$, $x_1 = 2$, $x_2 = 3$, $h = 1$

$$\int_{x_0}^{x_2} f(x)\,dx = \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right] - \underbrace{\frac{h^5}{90} f^{(4)}(\xi)}_{\text{error term}}$$

error is zero, the result is exact.

$$= \frac{1}{3}\left[1 + 36 + 25\right] = 20.6667 \text{ exact.}$$

c) Composite trapezoidal rule.



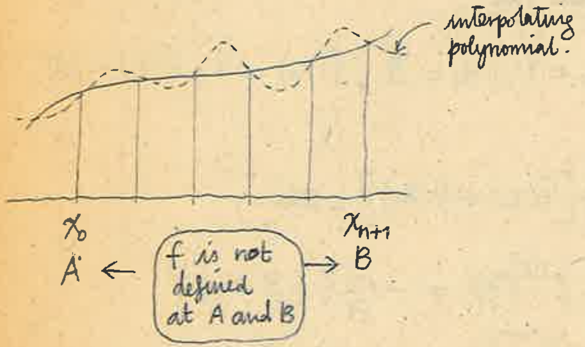
$h = 1/2$

$$\int_{x_0}^{x_4} f(x)\,dx = h\left[\frac{1}{2}f(x_0) + f(x_1) + f(x_2) + f(x_3) + \frac{1}{2}f(x_4)\right] - \frac{(B-A)}{12} h^2 f^{(2)}(\xi)$$

$$= \frac{1}{2}\left[\frac{1}{2} + 4.375 + 9 + 5.625 + 12.5\right] - \frac{2}{12}\left(\frac{1}{2}\right)^2 (6\xi - 4)'$$

$$= 21 - \frac{1}{24}(6\xi - 4) \qquad 1 \leq \xi \leq 3$$

Newton-Cotes Open type formulas :



We may derive integration formulas which use regular base points but do not have base points functional values at one or both of the integration limits. Here, for example we have an IP of N-2 degree

Let A be the lower and B be the upper integration limits.

$$\int_A^B f(x)\,dx \cong \int_A^B P_{N-2}(x)\,dx \quad \text{where } P_{N-2}(x) \text{ is the NFIP.}$$

Set $\alpha = \dfrac{x - x_0}{h}$, $\bar{\alpha} = \dfrac{B - x_0}{h}$, the

$$\cong h\int_0^{\bar{\alpha}} P_{N-2}(x_0 + \alpha h)\,d\alpha$$

$$P_{N-2}(x_0 + \alpha h) = f(x_1) + (\alpha-1)\Delta f(x_1) + \frac{(\alpha-1)(\alpha-2)}{2!}\Delta^2 f(x_1) + \frac{(\alpha-1)(\alpha-2)(\alpha-3)}{3!}\Delta^3 f(x_1) + \dots + \frac{(\alpha-1)\dots(\alpha-n+2)}{(n-2)!}\Delta^{n-2} f(x_1)$$

$$\int_A^B f(x)\,dx \cong h\int_0^{\bar{\alpha}}\left[f(x_1) + (\alpha-1)\Delta f(x_1) + \dots\dots\right]d\alpha$$

$$\cong h\left[\alpha f(x_1) + \left(\frac{\alpha^2}{2} - \alpha\right)\Delta f(x_1) + \left(\frac{\alpha^3}{6} - \frac{3\alpha^2}{4} + \alpha\right)\Delta^2 f(x_1) + \dots\dots\right]\Bigg|_0^{\bar{\alpha}}$$

$$\cong h\left[\bar{\alpha} f(x_1) + \left(\frac{\bar{\alpha}^2}{2} - \bar{\alpha}\right)\Delta f(x_1) + \left(\frac{\bar{\alpha}^3}{6} - \frac{3\bar{\alpha}^2}{4} + \bar{\alpha}\right)\Delta^2 f(x_1) + \dots\dots\right.$$

Error :

$$e = h\int_0^{\bar{\alpha}} R_{n-2}(x_0 + \alpha h)\,d\alpha = h^n \int_0^{\bar{\alpha}} \frac{(\alpha-1)(\alpha-2)\dots(\alpha-n+1)}{(n-1)!} f^{(n-1)}(\xi)\,d\alpha$$

$$\xi \in [x_0, B]$$

The above formulas are known as NC closed formulas. The upper limit B coincides with one of the base points then $\bar{\alpha}$ becomes an integer, and the integration is done throug m intervals each of width h.

i.e. $A = x_0$, $B = x_m$; $h = \dfrac{x_m - x_0}{m}$

Set $\bar{\alpha} = 2$ (i.e 2 intervals each of width h)

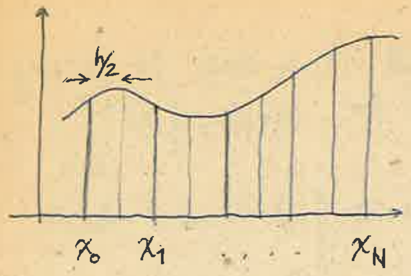
$$\int_{x_0}^{x_2} f(x)\,dx \cong \int_{x_0}^{x_2} P_{N-2}(x)\,dx = h\int_0^2 P_{N-2}(x_0 + \alpha h)\,d\alpha$$

$$= h\int_0^2 \left[f(x_1) + (\alpha-1)\Delta f(x_1) + \frac{(\alpha-1)(\alpha-2)}{2!}\Delta^2 f(x_1) \dots \dots\right]d\alpha$$

(Proceed the rest by yourself) (similar to closed formula)

$$\int_{x_0}^{x_2} f(x)\,dx = (x_2 - x_0) f(x_1) + \frac{(x_2 - x_0)^3}{24} f''(\xi) \quad \xi \in [x_0, x_2]$$

ROMBERG EXTRAPOLATION



Composite trapezoidal integration rule : (in closed form)

$$I = \frac{h}{2}\left[f_0 + f_N\right] + \sum_{i=1}^{N-1} h f_i$$

For composite trapezoidal rule, the error may be shown to be :

$$\underset{\text{exact}}{I} = \underset{\text{approx}}{T} + \underbrace{C_0h^2 + C_1h^4 + C_2h^6 \ldots\ldots \boxed{C_k h^{2(k+1)}}}_{\text{Error } = \Theta(h^2)}$$

interval

$$\text{exact} = I = \int_a^b f(x)\,dx \qquad h = \frac{b-a}{N}$$

$$\text{Thus, } I = \underbrace{T(h)}_{\substack{\text{approx.}\\ \text{when the}\\ \text{interval is } h}} + \underbrace{C_0h^2}_{\substack{\text{dominant}\\ \text{error}}} + \underbrace{\Theta(h^4)}_{\substack{\text{remaining}\\ \text{errors.}}} \ldots \quad ①$$

$$I = \underbrace{T(h/2)}_{\substack{\text{approx when the}\\ \text{interval is } h/2}} + C_0\frac{h^2}{4} + \Theta(h^4)$$

Multiply by 4 :

$$4 * I = 4T(h/2) + C_0h^2 + \Theta(h^4) \ldots \quad ②$$

Subtract ① from ② :

$$3I = 4T(h/2) + \Theta(h^4) - T(h)$$

$$\underset{\substack{\uparrow\\ \text{exact}}}{I} = \underbrace{\frac{4T(h/2) - T(h)}{3}}_{\text{approx.}} + \underbrace{\Theta(h^4)}_{C_1h^4 + \Theta(h^6)}$$

$T_0(h)$ > $T_1(h/2)$
$T_0(h/2)$

$T_0^{(0)}$ > $T_1^{(1)}$
$T_0^{(1)}$

Notation: $T_0^{(0)} = T(h)$

$T_0^{(1)} = T(h/2)$

$T_1^{(1)} = T_1(h/2)$

$$T_1^{(1)} = \frac{4T_0^{(1)} - T_0^{(0)}}{3}$$

$$I = T_1^{(1)} + C_1h^4 + \Theta(h^6)$$

using this notation :

$I = T_1^{(1)} + C_1h^4$ ① halving the interval once more.

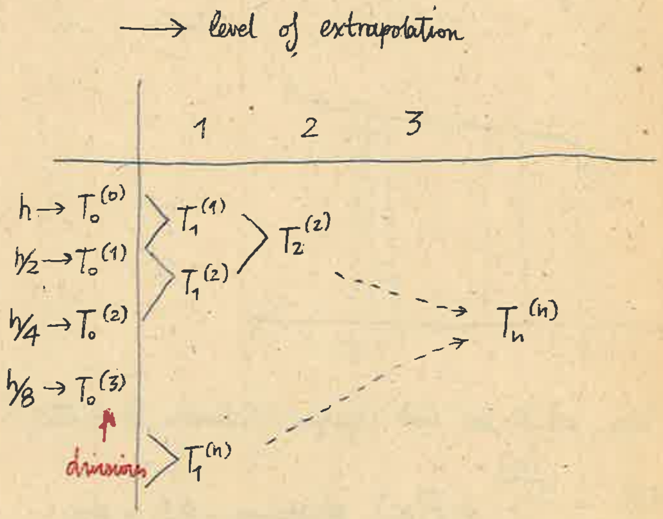
$I = T_1^{(2)} + C_1\left(\frac{h}{2}\right)^4 \ldots$ ② multiply both sides by 16, subtract ① from ②.

$$I = \underbrace{\frac{16T_1^{(2)} - T_1^{(1)}}{15}}_{T_2^{(2)}} + \Theta(h^6)$$

or in general (show!)

$$\boxed{T_{i+1}^{(n+1)} = \frac{4^{i+1}\,T_i^{(n+1)} - T_i^{(n)}}{4^{i+1} - 1}}$$

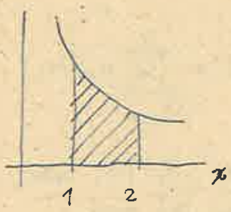
ROMBERG EXTRAPOLATION FORMULA



Example : Evaluate and extrapolate the following integral by ROMBERG.

$$I = \int_1^2 \frac{1}{x}\,dx \qquad \left.\begin{matrix} a=1\\ b=2 \end{matrix}\right\} h=1$$

apply trapez. rule :



$$h\,\frac{f(a) + f(b)}{2} = 0.75$$

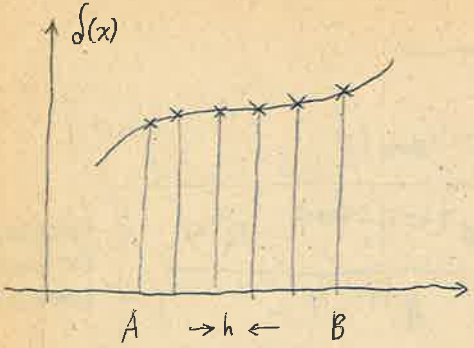
	0	1	2	3 ← i
h=1	0,75	0,69444	0,69317	0,693142
h=0.5	0,70833	0,69325	0,693143	
h=0.25	0,69702	0,69315	0,693153	
h=0.125	0,69402	0,69314		
h=0.0625	0.69339			

$$\frac{4 \times 0{,}70833 - 0{,}75}{3}$$

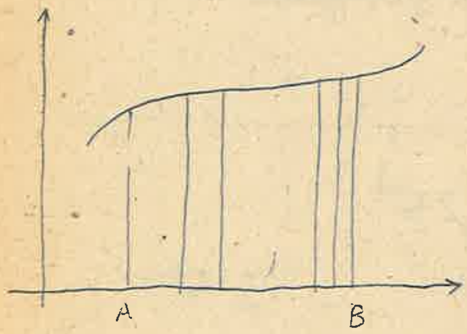
$$\frac{16 \times 0{,}69325 - 0{,}69444}{15}$$

$$\frac{64 \times 0{,}693143 - 0{,}6931}{63}$$

GAUSSIAN QUADRATURE



(ordinary method, regular base points)



(base points are not regular: Gaussian Quadrature)

$$\int_a^b f(x)\,dx = \sum_{i=0}^{n} a_i f(x_i) \quad \#\text{unknowns} = \#A_i + \#x_i$$

Greater accuracy with a certain number of points may be obtained if both the abscissa and the coefficients in the integration formula are unrestricted.

(We are free to move the x points to have a better accuracy)

$$\int_a^b f(x)dx = a_0 f(x_0) + \dots, + a_n f(x_n) = \sum_{i=0}^{n} a_i f(x_i)$$

The coefficients a_i and the abscissas x_i are now to be determined $(i=0,1\dots n)$. Hence we have $2(n+1)$ unknowns.

GQ: Gaussian integration is valid for the interval $[-1,1]$, if your integration is not performed in this interval we may convert it by the following transformation:

$$\boxed{x = \frac{1}{2}(b-a)t + \frac{1}{2}(b+a)} \qquad x=a \Rightarrow t=-1$$

$$\boxed{dx = \frac{1}{2}(b-a)dt} \qquad x=b \Rightarrow t=1$$

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{1} F(t)\,dt$$

Now we'll calculate

$$\int_{-1}^{1} F(t)dt = A_0 F(t_0) + \dots + A_n F(t_n) \qquad 2(n+1) \text{ unknowns}$$

Simplest case (2pt rule) $n=1 \Rightarrow \begin{matrix} A_0 & A_1 \\ t_0 & t_1 \end{matrix}$

$$\int_{-1}^{1} F(t)dt = A_0 F(t_0) + A_1 F(t_1) \qquad (1)$$

Determine A_0, A_1, t_0, t_1 so that (1) is exact for all polynomials of degree ≤ 3 i.e for $P_3(t) = \{t^k\}_{k=0}^{3}$

$$(2)\begin{cases} k=0\ ;\ P_3(t) = 1 \Rightarrow F(t) = 1 \Rightarrow \int_{-1}^{1} dt = 2 \Rightarrow 2 = A_0 + A_1 \\ k=1\ ;\ P_3(t) = t \Rightarrow F(t) = t \Rightarrow \int_{-1}^{1} t\,dt = 0 \Rightarrow 0 = A_0 t_0 + A_1 t_1 \\ k=2\ ;\ P_3(t) = t^2 \Rightarrow F(t) = t^2 \Rightarrow \int_{-1}^{1} t^2 dt = \frac{2}{3} \Rightarrow \frac{2}{3} = A_0 t_0^2 + A_1 t_1^2 \\ k=3\ ;\ P_3(t) = t^3 \Rightarrow F(t) = t^3 \Rightarrow \int_{-1}^{1} t^3 dt \Rightarrow 0 = A_0 t_0^3 + A_1 t_1^3 \end{cases}$$

4 equations, 4 unknowns. The solution to (2) is

$$A_0 = A_1 = 1 \ ,\ t_0 = -\frac{1}{\sqrt{3}} \cong -0.577,\ t_1 = \frac{1}{\sqrt{3}} \cong 0.577$$

Hence the integration formula becomes:

$$\boxed{\int_{-1}^{1} F(t)dt \cong F\left(-\frac{1}{\sqrt{3}}\right) + F\left(\frac{1}{\sqrt{3}}\right)}$$

: 2 point GQ rule exact for poly of degr. ≤ 3

error expression:

Truncation error in this rule for poly ≤ 3 is zero

For poly > 3: $\quad e = k F^{(4)}(\xi) \quad \xi \in [-1,1]$

where k is a coefficient to be calculated; to determine k:

Let $F(t) = t^4 \quad f^{(4)} = 24$

$e = k.(24)$

$$\int_{-1}^{1} t^4 = \frac{2}{5} \text{ exactly} \Rightarrow \frac{2}{5} = \left(-\frac{1}{\sqrt{3}}\right)^4 + \left(\frac{1}{\sqrt{3}}\right)^4 + 24k$$

$$k = \frac{1}{135}$$

then $\boxed{e = \frac{1}{135} F^{(4)}(\xi)} \quad \xi \in [-1,1]$

Example: Calculate $\int_0^4 x^3 dx$; use 2pt GQ rule.

(exact value = 64)

$$\int_a^b f(x)dx = 2\int_{-1}^{1} F(t)dt \qquad x = \frac{1}{2}(b-a)t + \frac{1}{2}(b+a)$$

$$x = 2t+2$$

$$F(t) = f(2t+2) = [2t+2]^3$$

$$2\int_{-1}^{1} F(t)dt = 2\left[F\left(-\frac{1}{\sqrt{3}}\right) + F\left(\frac{1}{\sqrt{3}}\right)\right]$$

$$= 2\left[\left\{F\left(\frac{-1}{\sqrt{3}}\right) + 2\right\}^3 + \left\{2\left(\frac{1}{\sqrt{3}}\right) + 2\right\}^3\right]$$

$$= 2(16+16) = 64 \ \underline{\underline{\text{exact}}}$$

Higher order GQ rules   Gaussian Quadrature

The same procedure may be used to derive a general n point GQ rule. F(t) is taken as a simple polynomial t^k (known as monomial) $k=0,1,\dots 2n+1$.

Observing that

$$\int_{-1}^{1} t^k dt = \begin{cases} 0 & \text{if } k \text{ is } \boxed{\text{even}} \\ \frac{2}{k+1} & \text{if } k \text{ is } \boxed{\text{odd}} \end{cases}$$

Hence we obtain the following sys. of nonlinear equations in terms of A's and t's:

$$\left.\begin{cases} k=0, & 2 = A_0 + A_1 + \dots\dots + A_n \\ k=1, & 0 = A_0 t_0 + A_1 t_1 + \dots + A_n t_n \\ k=2, & \frac{2}{3} = A_0 t_0^2 + \dots\dots + A_n t_n^2 \\ \vdots & \\ k=2n+1, & 0 = A_0 t_0^{2n+1} + \dots\dots + A_n t_n^{2n+1} \end{cases}\right\} \text{2n+2 nonlinear equations to 2n+2 unknowns.}$$

It can be shown that these equations possess a unique and real solution set for A_k, t_k $(k=0,1,\dots n)$

In practice these equations are never solved directly for A_k, t_k's.

Method of solution

Theorem: The set of t_k $(k=0,1,\dots n)$ which solves the above system of equations are the roots of the set of certain polynomials known as Legendre Polynomials. $P(t)=0$

The Legendre polynomials are defined as follows: (a recursive formula)

$$P_0(t) = 1$$
$$P_1(t) = t$$
$$\vdots$$
$$P_m(t) = \frac{1}{m}\left[(2m-1)tP_{m-1}(t) - (m-1)P_{m-2}(t)\right]$$
$$(m = 2,3,4\dots\dots)$$

Example:

$m=2 \Rightarrow P_2(t) = \frac{1}{2}\left[3t^2 - 1\right]$ roots: $\pm\sqrt{\frac{1}{3}}$

$m=3 \Rightarrow P_3(t) = \frac{1}{2}(5t^3 - 3t) \rightarrow$ roots: $t_0 = 0$, $t_{1,2} = \pm\sqrt{\frac{3}{5}}$

$m=4 \Rightarrow P_4(t) = \frac{1}{8}(35t^4 - 30t + 3)$

The solutions for t_k's are found now;
The coefficient of A_k $(k=0,1,\dots n)$ can be found by substituting t_k's to the system of equations and solving for A_k's.

Another way (simpler)

There is no need to solve the system of equations for A_k, but we may use the following formula:

$$A_i = \frac{2}{(1-t_i)^2\left[P'_{n+1}(t_i)\right]^2} \qquad (i=0,1\dots n)$$

no proof (think!)

Some properties of the Legendre Polynomials:

① $P_m(t)$ is a polynomial of degree m.
② The roots of $P_m(t)$ are all real and distinct, and in the interval $[-1,+1]$.
③ The roots are symmetrical around zero.
If m is odd, one root of P_m is always zero.

Order number of base points m.	Legendre polynomial	Roots t_k	Coefficient in GQ formula A_k
1	t	$t_0=0$	$A_0 = 2$
2	$\frac{1}{2}[3t^2-1]$	$t_0=-t_1=$ -0.57735027	$A_0=A_1= 1.0$
3	$\frac{1}{2}[5t^3-3t]$	$t_0=0$ $t_1=-t_2=$ 0.77459667	$A_1=0.8888889$ $A_2=A_0=$ 0.5555556
4	$\frac{1}{8}(35t^4-30t^2+3)$	$t_2=-t_1=$ 0.33998104 $t_3=-t_0=$ 0.86113631	$A_2=A_1=$ 0.65214515 $A_3=A_0=$ $0,34785485$

Note:

1. Coefficients A_k are all positive.
2. Coefficients corresponding to symmetrical t points are equal.

Example:

$$\int_0^3 x^2\cos x\,dx \quad \text{using GQ with 3 pts.}$$

$$x = \frac{b-a}{2}t + \frac{b+a}{2} = \frac{3}{2}(t+1) \qquad dx = \frac{3}{2}dt$$

$$I = \frac{3}{2}\int_{-1}^{1} \overbrace{\left[\frac{3}{2}(t+1)\right]^2 \cos\left[\frac{3}{2}(t+1)\right]}^{F(t)} dt \cong \frac{3}{2}\sum_{i=0}^{2} A_i F(t_i)$$

$$m=3\begin{cases} A_0 = \\ A_1 = \\ A_2 = \end{cases} \text{(look at the above table)}$$

$$m=3\begin{cases} t_0 = \\ t_1 = \\ t_2 = \end{cases} \left(\qquad '' \qquad\right) \qquad \begin{aligned} f(t_0) &= -0,32383 \\ f(t_1) &= 0,159158 \\ f(t_2) &= -3,99716 \end{aligned}$$

$$I = \frac{3}{2}\left[0,10784 \cdot 0,5555 + 0,159158 \dots\dots\dots\right]$$

$$= -4,936$$

The error in GQ rule :

$$e = \frac{2^{m+1}(m!)}{(2m+1)\left[(2m)!\right]^3} f^{(2m)}(\xi) \qquad \xi \in [-1,1]$$

Ref : Ralston / A first course in N.M.

Advantages of GQ

1. For the same # of points, GQ is twice accurate than ordinary formulas.
2.

Disadvantages of GQ

1. Abscissas are in general irrational numbers. (not a serious problem in computer application)
2. If $F(t)$ is given in tabular form only, (not in analytical) then the abscissa points may not coincide with the found t_k points. We can interpolate by sacrificing some accuracy.
3. If the integration is to be re-evaluated with more points, then the preceding points can't be re-used, since the location of t_k parameters are now changed.

NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS

The solution of an ODE is a function. A computer algorithm does not give an analytical solution, but produces its shape as a set of (x,y) pairs.

The general form of an ODE is :

$$\frac{d^n y}{dx^n} + \frac{d^{n-1}y}{dx^{n-1}} \cdots \cdots + \frac{dy}{dx} = f(x,y)$$

and the initial conditions :

$y(0) = y_0$

$y'(0) = y_1$

$\vdots$

Transformation Technique ;

$y_1 = y$

$y_2 = \frac{dy}{dx} = \frac{dy_1}{dx}$

$y_3 = \frac{d^2y}{dx} = \frac{d}{dx} y_2$

resulting in the following vector form :

$$\underbrace{\frac{d}{dx}\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} y_2 \\ y_3 \\ \vdots \\ f(x,y) - y_n - y_{n-1} - \ldots\ y_2 \end{bmatrix}}$$

$$\frac{d}{dx} Y = F(x,y)$$

$$\frac{d}{dx} Y = F(X,Y)$$

The general form of the ODE of 1st order, determine the solution $y(x)$ which satisfy,

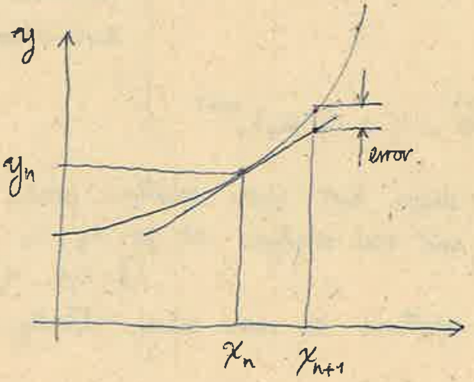
$$\frac{dy}{dx} = f(x,y)$$

with the condition $y(a) = b$

The LHS is the slope of the solution function $y(x)$ at point x.

The RHS is a function which may be evaluated easily at X and Y.

Hence a first order ODE is a rule which assigns a slope to the function y at each point x.



The error between the solution and the actual function increases. (named as propagation error)

EULER'S METHOD (not used in practice due to large propagation error)

Approximate the derivative on the RHS by its forward difference representation :

$$\frac{d}{dx} y \cong \frac{y(x+h) - y(x)}{h}$$

thus ;

$$\frac{y(x+h) - y(x)}{h} = f(x,y)$$

or ;

$$y(x+h) = y(x) + h f(x,y)$$

$$y(x_n+h) = y(x_n) + h f(x_n, y_n) \qquad n = 0,1,2 \ldots$$

or simply ;

$$\boxed{y_{n+1} = y_n + h f(x_n, y_n)}$$ EULER'S Formula

$y(0) = y_0$ ← initial cond. must be given.

The propagation error may be evaluated analytically. Expand $y(x_n+h)$ by Taylor :

$$y(x_n+h) = y(x_n) + h f(x_n,y_n) + \underbrace{\left\{ \frac{h^2}{2!} f'(x_n,y_n) + \ldots \right.}_{\text{Error terms.}}$$

$\theta(h^2)$ ← short representation

or ;

$$\text{error} = \frac{h^2}{2!} f'(x_n, y_n) + \frac{h^3}{3!} f''(x_n, y_n) + \ldots\ldots \quad \{\Theta(h^4)\}$$

Theorem : (from Conte) p. 361

The error in the Euler's formula at any point x_n where ;

$$x_n = x_0 + nh ,$$

is bounded as follows,

$$|e_n| \leq \frac{hY}{2L} \left[e^{(x_n - x_0)L} - 1 \right]$$

where Y and L are some constants, (bounds on the 1st ~~and 2nd~~ derivatives of $f(x,y)$ and $y(x)$ respectively)

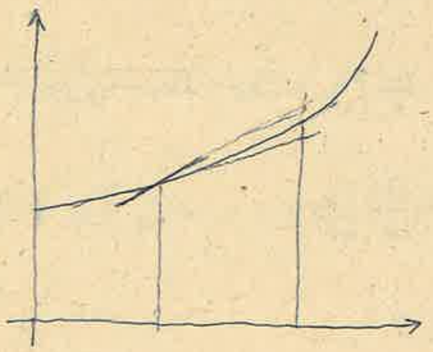
$$\frac{d}{dx} y(x) = f(x,y)$$

$$\left| \frac{\partial}{\partial y} f(x,y) \right| \leq L$$

$$\left| \frac{d^2 y(x)}{dx^2} \right| \leq Y$$

~~ⓧ there is an ambiguity!~~

HEUN'S METHOD



1. Obtain original slope at x_n, y_n.
   $\boxed{S_1 = f(x_n, y_n)}$
2. Obtain the next slope
   $S_2 = f(x_n + h, y_n + hS_1)$
3. Average the slopes
   $\bar{S} = \dfrac{S_1 + S_2}{2}$
4. Now, move from X_n again but now using $\bar{S}$

$$y(x_{n+1}) = y_n + \bar{S}h = y_n + \frac{h}{2}\left[f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n)) \right]$$

$n = 1, 2, \ldots$

HEUN'S formula.
(always better than EULER)

Example : Solve the following ODE, by EULER using step length of $h = 0.01$

$$\frac{dy}{dx} = y \quad y(0) = 1$$

Exact solution: $y = e^x$

$$y(0.01) = y_1 = y_0 + hf(x_0, y_0)$$

$$= 1 + 0.01 \times 1 = 1.01$$

$$y(0.02) = 1.01 + 0.01 \times 1.01 = 1.0201$$

$$y(0.03) = 1.0201 + 0.01 \times (1.0201) = 1.0303$$

$$y(0.04) = 1.0303 + 0.01 \times (1.0303) = 1.040606$$

(TRY HEUN!)

RUNGE-KUTTA METHODS

Gen. form:

$$y_{n+1} = y_n + h\left[\underset{0.5}{a} k_1 + \underset{0.5}{b} k_2\right]$$

Where $\begin{cases} k_1 = f(x_n, y_n) \\ k_2 = f(x_n + ph, y_n + qhk_1) \end{cases}$

2nd order RK formula :

choose $a = b = 1/2$

$p = q = 1$

$$y_{n+1} = y_n + \frac{h}{2}[k_1 + k_2]$$

and

$$k_1 = f(x_n, y_n)$$

$$k_2 = f(x_n + h, \underbrace{y_n(x_n + h)}_{y_n + hk_1})$$

4th order Runge-Kutta Method :

$$y_{n+1} = y_n + \frac{h}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right]$$

where $k_1 = f(x_n, y_n)$

$k_2 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right)$

$k_3 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right)$

$k_4 = f(x_n + h, y_n + hk_3)$.

Suggestion (for checking propagation error)

COLLATZ :

if $\dfrac{k_2 - k_3}{k_1 - k_2} >$ few $\overset{100,200,300}{\text{hundreds}}$ then it means that your "h" is too large. Reduce it.

Example : Solve the following ODE in the interval [1, 1.1] by using a 4th order RK method with step size h = 0.1.

$$\frac{d^2y}{dx^2} + x\frac{dy}{dx}e^{-2y} = 0$$

$y(1) = 0$ | (The actual solution is

$y'(1) = 1$ | $y(x) = \ln(x)$)

$$\left.\begin{aligned} y &= y_1 \\ \frac{dy}{dx} &= \frac{dy_1}{dx} = y_2 \\ \frac{d^2y}{dx^2} &= \frac{dy_2}{dx} \end{aligned}\right\}$$

Hence ;

$$\left[\begin{aligned} \frac{dy_1}{dx} &= y_2 \\ \frac{dy_2}{dx} + xy_1 e^{-2y_1} &= 0 \end{aligned}\right]$$

in vector form :

$$\underbrace{\frac{d}{dx}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} y_2 \\ -xy_2 e^{-2y_1} \end{bmatrix}}_{\frac{d}{dx}Y = F(x,Y)}$$

The initial conditions :

$$y(1) = y_1(1) = 0$$

$$y'(1) = \frac{d}{dx}y(x)\Big|_{x=1} = y_2(1) = 1$$

$$Y(1) = \begin{bmatrix} y_1(1) \\ y_2(1) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$y_1(n+1) = y_1(n) + \frac{h}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right]$$

$$y_2(n+1) = y_2(n) + \frac{h}{6}\left[c_1 + 2c_2 + 2c_3 + c_4\right]$$

$$k_1 = f_1(x_1, y_1(1); y_2(1)) = 1$$

$$c_1 = f_2(x_1, y_1(1), y_2(1)) = -1 \times 1e^{-2} = 1$$

$$k_2 = f_1(\underbrace{x_1 + \tfrac{h}{2}}_{\text{instead of } x_1}; \underbrace{y_1(1) + \tfrac{h}{2}k_1}_{\text{instead of } y_1}; y_2(1) + \tfrac{h}{2}c_1)$$

$$= f_1(1.05, 0.05, 0.95) = 0.95$$

$$c_2 = f_2(1.05, 0.05, 0.95) = -1.05 \times 0.95 \times e^{-2 \times 0.05}$$

$$\doteq -0.902575$$

$$k_3 = f_1(1.05, y_1(1) + \tfrac{h}{2}k_2, y_2(1) + \tfrac{h}{2}c_2)$$

$$= f_1(1.05, 0.0475, 0.95487) = 0.95487$$

$$c_3 = f_2(1.05, 0.0475, 0.95487) = -0.911769$$

$$k_4 = f_1(x_1 + h, y_1 + hk_3, y_2 + hc_3)$$

$$= f_1(1.1, 0.095487, 0.908825) = 0.908825$$

$$c_4 = -0.825912.$$

$$y_1(1) = y_1(1) + \frac{0.1}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right] = 0.0953094$$

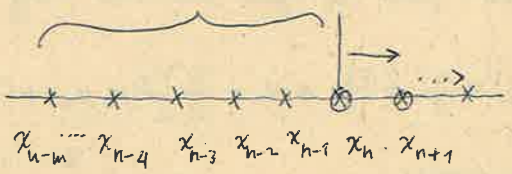
$$y_2(1) = y_2(1) + \frac{0.1}{6}\left[c_1 + \dots + c_4\right] = 0.9090906$$

The exact sol'n is :

$$\left.\begin{aligned} y_1(x) &= \ln x \\ y_2(x) &= \frac{dy_1}{dx} = \frac{1}{x} \end{aligned}\right\} \begin{aligned} \ln 1.1 &= 0.0953101 \\ \frac{1}{1.1} &= 0.90909 \end{aligned}$$

homework : Proceed one more step.

Multi-Step Methods



Open type Methods :

Given

$$\frac{d}{dx}y(x) = f(x,y)$$

$$y(0) = y_0$$

Now, integrate the LHS & RHS from x_n to x_{n+1} ;

$$\int_{x_n}^{x_{n+1}} \cancel{\frac{d}{dx}} y(x) \cancel{dx} \Big|^{x_{n+1}} = \int_{x_n}^{x_{n+1}} f(x,y)\,dx$$

or

x_n

$$y_{n+1} = y(n) + \int_{x_n}^{x_{n+1}} f(x,y)\,dx$$

Interpolate $f(x,y)$ by Newton's backward interpolating polynomial of degree m.

$$P_m(x_n+\alpha h) = \sum_{k=0}^{m} (-1)^k \binom{-\alpha}{k} \nabla^k f_n \qquad (*)$$

$$\binom{a}{b} \triangleq \frac{a!}{b!(a-b)!}$$

$$\binom{-\alpha}{k} \triangleq \frac{(-\alpha)(-\alpha-1)(-\alpha-2)\ldots.}{k!(-\alpha-k)(-\alpha-k+1)\ldots.}$$

Joke : Show that $(*)$ is the same formula as the NBIP we have seen already in the interpolation chapter.

Thus :

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P_m(x+\alpha h)\,dx$$

$x = x_n + \alpha h$

$dx = h\,d\alpha$

$x = x_n \Rightarrow \alpha = 0$

$x = x_{n+1} \Rightarrow \alpha = 1$

$$= y_n + h\int_0^1$$

$$= y_n + h\sum_{k=0}^{m}\left[(-1)^k \int_0^1 \binom{-\alpha}{k} \nabla^k f_n\right] d\alpha$$

$$= y_n + h\left[\sum_{k=0}^{m} b_k \nabla^k f_n\right]$$

where $b_k = (-1)^k \int_0^1 \binom{-\alpha}{k} d\alpha \qquad k = 0,1\ldots m$

For example ;

$k=0$, $b_0 = 1$

$k=1$, $b_1 = \frac{1}{2}$

$k=2$, $b_2 = 5/12$

$k=3$, $b_3 = 3/8$

$k=4$, $b_4 = 251/720$

$k=5$, $b_5 = 95/288$

$$y_{n+1} = y_n + h\left[1 + \frac{1}{2}\nabla + \frac{5}{12}\nabla^2 + \frac{3}{8}\nabla^3 + \frac{251}{720}\nabla^4 + \frac{95}{288}\nabla^5\right] f_n$$

Implicit form of Adam-Bashford formula

Using : $\nabla f_n = f_n - f_{n-1}$.

$\nabla^2 f_n = f_n - 2f_{n-1} + f_{n+1}$

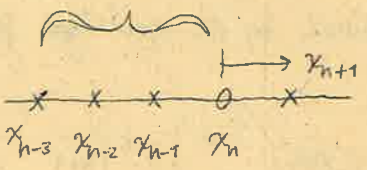
$\nabla^3 f_n = f_n - 3f_{n-1} + 3f_{n-2} - f_{n-3}$

Explicit form of AB formula :

$$y_{n+1} = y_n + \frac{h}{24}\left[55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}\right]$$

$f(x_n, y_n)$ $f(x_{n-1}, y_{n-1})$.

4 Step formula



The multistep formulas are not self-starting. Thus, we must generate those initial points by using a single-step formula, Such as Euler, Heun, Runge-Kutta.

$$\int_{x_{n-p}}^{x_{n+1}}$$

A number of multi-step formulas may be derived by integrating the RHS and LHS from x_{n-p} to x_{n+1} as follows ;

$$\int_{x_{n-p}}^{x_{n+1}} P_m(x+\alpha h)\,dx \overset{dx\to d\alpha}{=} h\int_{-p}^{1} P_m(x_n+\alpha h)\,d\alpha$$

$$\to b_k = (-1)^k \int_{-p}^{1} \binom{-\alpha}{k} d\alpha$$

Then , $y_{n+1} = y_{n-p} + h\sum_{k=0}^{m} b_k \nabla^k f_n$

Several formulas may be generated for different p points.

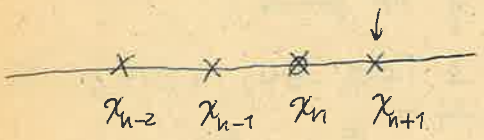
$p=1$; $y_{n+1} = y_{n-1} + h\left[2 + 0\nabla + \frac{1}{3}\nabla^2 + \frac{1}{3}\nabla^3 + \frac{29}{30}\nabla^4 + \ldots\right] f_n$

$p=3$; $y_{n+1} = y_{n-3} + h\left[4 - 4\nabla + \frac{8}{3}\nabla^2 + 0\nabla^3 + \frac{14}{45}\nabla^4 \ldots\right] f_n$

$\downarrow$

$$y_{n+1} = y_{n-3} + \frac{4h}{3}\left(2f_n - f_{n-1} + 2f_{n-2}\right)$$

Closed type formulas ;



$\boxed{\text{— prediction and correction methods.}}$

Closed type formulas require the point x_{n+1} y_{n+1} which may be obtained by an open type formula.

NBIP :

$$P_{m+1}(\alpha) = \sum_{k=0}^{m+1} (-1)^k \binom{1-\alpha}{k} \nabla^k f_{n+1} \quad \text{(think !)}$$

$$\cancel{\int_{x_{n-p}}^{x_{n+1}} \frac{dy}{dx}dx} \Big|_{x_{n-p}}^{x_{n+1}} = h\int_{-p}^{1} \sum_{k=0}^{m+1} (-1)^k \binom{1-\alpha}{k} \nabla^k f_{n+1}\, d\alpha$$

Or ;

$$y_{n+1} = y_{n-p} + h\sum_{k=0}^{m+1} b_k \nabla^k f_{n+1}$$

Where ,

$$b_k = (-1)^k \int_{-p}^{1} \binom{1-\alpha}{k} d\alpha \qquad k = 0, 1, 2 \ldots$$

$p=0$: $y_{n+1} = y_n + h\left[1 - \frac{1}{2}\nabla - \frac{1}{12}\nabla^2 - \frac{1}{24}\nabla^3 - \frac{19}{720}\nabla^4\right] f_{n+1}$

$p=1$; $y_{n+1} = y_{n-1} - h\left[2 - 2\nabla + \frac{1}{3}\nabla^2 - \frac{1}{90}\nabla^4\right] f_{n+1}$

~~interrupt priority~~ 4 No: 444 250581 1

Homework: Derive the explicit form of those formulas.

$$y_{n+1} = y_n + \frac{1}{3} h\left[4 - 8\nabla + \frac{20}{3}\nabla^2 - \frac{8}{3}\nabla^3 + \frac{14}{45}\nabla^4 - \dots\right] f_{n+1}$$

Some predictor – corrector methods ;

Milne's Method :

prediction : $y_{n+1}^{(p)} = y_{n-3} + h\left[4 - 4\nabla + \frac{8}{3}\nabla^2\right] f_n$

$= y_{n-3} + \frac{4h}{3}\left[2f_n - f_{n-1} + 2f_{n-2}\right]$ ← explicit form

correction ; $y_{n+1}^{(c)} = y_{n-1}^{(c)} + h\left[2 - 2\nabla + \frac{1}{3}\nabla^2\right] f_{n+1}$

$= y_{n-1}^{(c)} + \frac{h}{3}\left[f_{n+1}^{(p)} + 4f_n - f_{n-1}\right]$ ← explicit form

This method is unstable for large h's.

Adams - Moulton predictor - corrector Method

① Prediction : $y_{n+1}^{(p)} = y_n + \frac{h}{24}\left(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}\right)$

② intermediate step (evaluation) : $f_{n+1}^{(p)} = f\left(x_{n+1}, y_{n+1}^{(p)}\right)$

③ Correction $y_{n+1}^{(c)} = y_n + \frac{h}{24}\left[9f(x_{n+1}, y_{n+1}^{(p)}) + 19f_n - 5f_{n-1} + f_{n-2}\right]$

Iterate step 3 until $\left| y_{n+1}^{(c)^{i+1}} - y_{n+1}^{(c)^{i}} \right| \le \epsilon$

4th order Adams - Moulton Method :

prediction : $y_{n+1}^{(p)} = y_n + \frac{h}{12}\left[23f_n - 16f_{n-1} + 5f_{n-2}\right]$

Correction : $y_{n+1}^{(j+1)} = y_n + \frac{h}{24}\left[9f(x_{n+1}, y_{n+1}^{(j)}) + 19f_n - 5f_{n-1} + f_{n-2}\right]$

$j = 0, 1, 2 \dots$

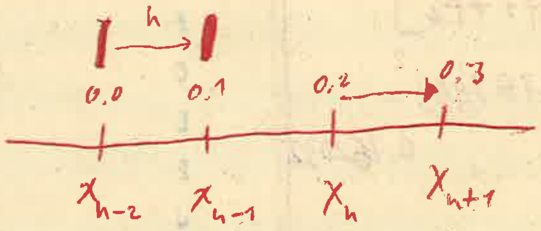
Example : Solve the following ODE by Adams-Moulton 4th order formula using 1 step size $h = 0.1$ in the interval $[0, 0.3]$ Obtain the initial points by Euler's method

Perform 1 iteration in the correction formula.

The exact soln $y(x) = \frac{x}{1+x^2}$

$$\frac{dy}{dx} = \frac{1}{1+x^2} - 2y^2 = f(x,y)$$

$y(0) = 0$



$x_0 \quad x_1 \quad x_2$

$y(x_0) = \quad \boxed{y(x_1)} \quad \boxed{y(x_2)}$

$\underbrace{f(x_0,y_0)}_{f_0} \quad \underbrace{f(x_1,y_1)}_{f_1} \quad \underbrace{f(x_2,y_2)}_{f_2}$

Solution : Calculation of initial points.

$y(0) = 0$

$f(0,0) = 1$

Euler : $y_1 = y_0 + hf(0,0) = 0 + 0.1\times 1 = 0.1$

$$f(0.1, 0.1) = \frac{1}{1+(0.1)^2} - 2(0.1)^2 = 0.9701.$$

Euler : $y_2 = y_1 + hf(0.1, 0.1) = 0.1 + 0.1\times 0.9701 = 0.19701$

$$f(x_2,y_2) = f(0.2, 0.19701) = \frac{1}{1+(0.2)^2} - 2\times(0.19701)^2 = 0.88391$$

Now using Adams Moulton formula : $(n=2)$

P : $y_{\underset{3}{n+1}} = y_3^{(0)} = y_2 + \frac{0.1}{12}\left[23f_2 - 16f_1 + 5f_0\right]$

$$= 0.19701 + \frac{0.1}{12}\left[23*0.88391 - 16*0.9701 + 5*1\right] = 0.278746$$

```
*   AV1176     2     GIRIS     6     6
    AV1176     2     GIRIS     6     5
    AV1176     2     GIRIS     6     9
    BF4145     5     GIRIS     5     4
    BF4145     5     GIRIS     5     2
    BF4145     5     GIRIS     5     3
    BF4145     5     GIRIS     5     7
    BF4145     5     GIRIS     5     9
******************************************************
                                                  KUYRUK
```

3

C ; $y_3^{(1)} = y_2 + \frac{0,1}{24}\left[9f(x_3, y_3^{(0)}) + 19f_2 - 5f_1 + f_0\right]$
Corrector

$$f(x_3, y_3^{(0)}) = \frac{1}{1+(0,3)^2} - 2\,(0,278746)^2 = 0,762032$$

Then ;

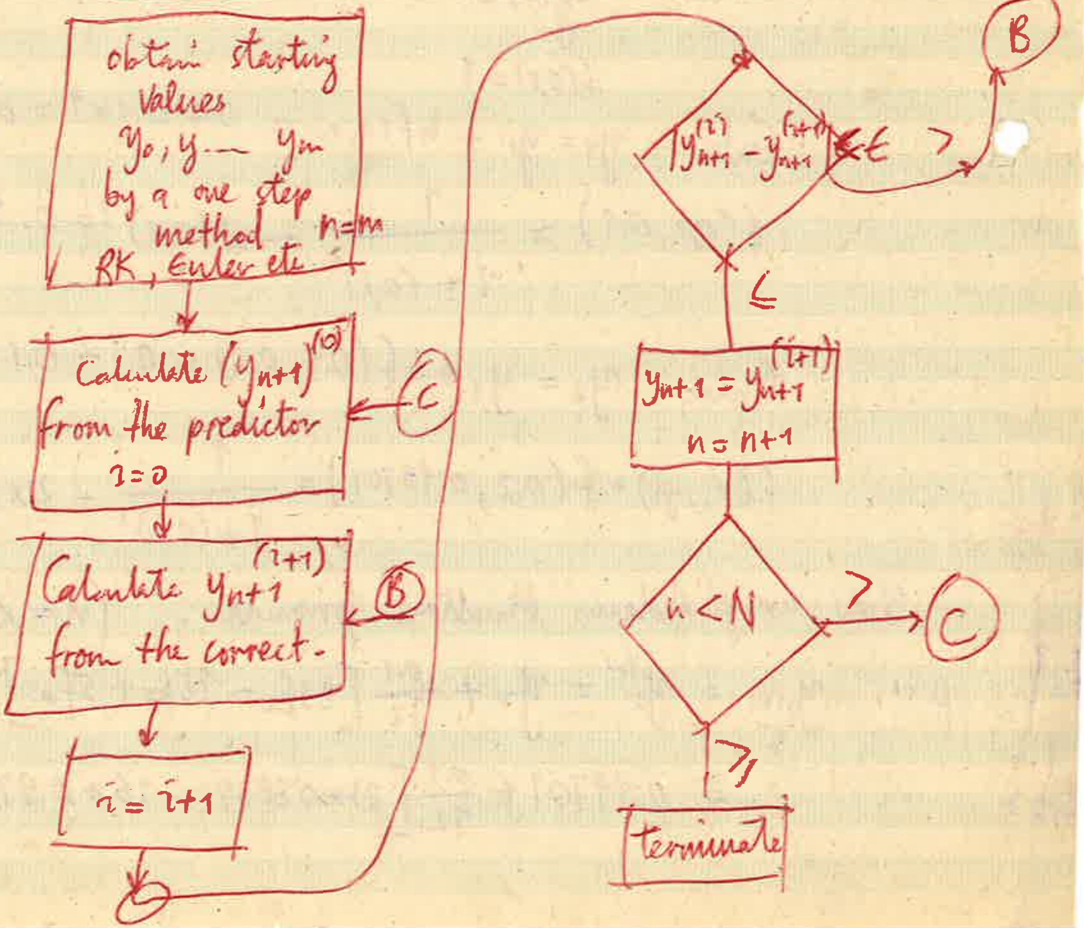
$$y_3^{(1)} = 0,2795186$$

The exact value at this pt is;

0,2752293

```
* GIRIS KUYRUKLARI *
KUYRUK ADI IS SAYISI
     A          31
     C          10
     D          30
     E          12
     J           5
     K           9
     F          12
     G          17
     V           5
     Z          20
     S          13
```

Prediction Correction Procedure



obtain starting values $y_0, y_1 \dots y_m$ by a one step method $n=m$ RK, Euler etc
Calculate $(y_{n+1})^{(0)}$ from the predictor $i=0$
C
Calculate $y_{n+1}^{(i+1)}$ from the correct.
B
$i = i+1$
$|y_{n+1}^{(i)} - y_{n+1}^{(i+1)}| < \epsilon$
$>$
B
$\leq$
$y_{n+1} = y_{n+1}^{(i+1)}$
$n = n+1$
$n < N$
$>$
C
$\geq$
terminate

B6900

TASK
JØB
QUEUE 0-99

an invalid character.
usually mult. punch 1,2,3.

```
$BEGIN JØB
  |
  |
$END JØB
```

Ex:

```
$BEGIN JØB  DERLEME/ØRNEG1 ;
 USER = ODTU/PARØLA;              optional
 CØMPILE  DERLENECEK/PRØGRAM  FØRTRAN  GØ ;
 FØRTRAN DATA
   {
   {
$DATA  FILE5                      card printer
   {
   {
$END JØB
```

Terminal usage: (via CANDE)

```
HELLØ  (XPIT)
```

→ Asks who are you?

```
ODTU/PARØLA
```

→ If it recognizes you:

```
MAKE  DERLENECEK/PRØGRAM  FØRTRAN
SEQ                                  sequence mode
10 ▷ FILE5 ----   data
  |
  |
?BRK
C  ← automatically compiles in FØRTRAN
     and outputs from printer
R  ← I/O at terminal.
  |
  |
BYE   ← closing.
```

Ex:

```
$BEGIN JØB .....   ( if we want to save our pgm. into our library)
TASK DERLE;
~~CØMPILE DERLENECEK/PRØGRAM [DERLE] FØRTRAN LIBRARY~~
CØMPILE  DERLENECEK/PRØGRAM  FØRTRAN [DERLE]  LIBRARY ;
```

```
FORTRAN DATA
{
{
$ IF DERLE IS COMPILED OK THEN
 BEGIN
 RUN DERLENECEK/PROGRAM ;
 DATA FILE5
        ↑
        card
        input (default)
```

if we want file5 in disk:

```
$ BEGIN JOB DERLEME/ORNEGGI;
 USER ODTU/PAROLA ;
 RUN DERLENECEK/PROGRAM;
 FILE FILES(KIND = DISK,
            TITLE = PROGRAMIN/
                    VERISI,
            MAXRECSIZE = 60,
            BLOCKSIZE = 180,
            UNITS = 1 ) ;
                    ↑
```

our m/c is word oriented so if we don't write UNITS = 1, MAXRESIZE = 60 × 6 byts assumed. so for BLOCKSIZE

```
$ END
$ END JOB
```

```
$ BEGIN JOB ...
 ¦
 ¦
WAIT ("BENIM BANDIMI TAKINIZ", OK); ←
RUN DERLENECEK/PROGRAM ;
FILE FILES ( KIND = TAPE ,
           ¦
           ¦
           ¦
           ) ;
```

for operator console

QUEUES : (limits for users)

```
$ BEGIN JOB DERLEME/ORNEGGI ;
  USER = ODTU/PAROLA;
  MAXIOTIME = 10;
  MAXPROCTIME = 5;
  MAXLINES = 4000;
```

Giving names to FILES :

```
  14
16/

BENIM/PROG/- - -

FIZIK/ ___
       ___
       ___ /___
```

14 level

```
REMOVE FIZIK/= ;     deletes 'fizik'
CHANGE FIZIK/= TO FIZIK01;    change name of FIZIK.
COPY FIZIK/TEST1 FROM FIZIKARC;
ADD FIZIK/= FROM FIZIKARC;
SECURITY FIZIK/TEST1 PUBLIC { I/O }  <-- (in general PRIVATE I/O; but
                            { I   }      if you want your file for
                            { O   }      public use insert this)
```

Derleyiciler:	additional:	Benzetim dilleri	terminal dilleri:
ALGOL	APL/700	GAMMA	NDL
PL/I	BASIC	SIMULA	MCS ← Message Control System
COBOL	SNOBOL	DYNAMO	CANDE (Command and edit)
FORTRAN	LISP	GPSS	
	POLGEN		
DM ALGOL			
DC ALGOL			
ESPOL			

B6900

FORTRAN IV H level.

Everything Same except DEFINE FILE for 6900

Running a program:

```
$BEGIN JOB <is adi>; USER = <kullanıcı kodu>;
 :
$ COMPILE <Program adı> <derleyici adı> GO; <derleyici adı> DATA
 [ Source program.
$ DATA <kütük adı>
 [ data
$ END JOB
```

Data Representation in 6900

At 6900   1 WORD = 52 bit.

bits: 0-47 (6 bytes) information
48-50 tag fields. ⟶ 000 (single precision)
010 (double precision)

51 parity bit