# Resumen de Métodos y Fórmulas – Análisis Numérico I

# Errores (Guía 1)

Error relativo:

$$R_a = \frac{\Delta a}{\hat{a}}$$

Error de truncamiento:

$$\Delta Y = |Y_{N+1} - Y_N|$$

Fórmula general de propagación de errores de entrada:

$$\Delta y = \sum_{i=1}^{n} \left| \left[ \frac{\partial F}{\partial x_i} \right]_P \right| \Delta x_i$$

Fórmula para gráfica de procesos (con errores relativos):

$$R_y = \sum_{j=1}^{n} |f_j| * R_j + \sum_{i=1}^{m} |g_i| * u$$

$$R_y = C_P * R + F_u * u$$

Número de condición del problema:

$$C_P = \sum_{j=1}^{n} |f_j|$$

Número de condición del Algoritmo:

$$C_A \equiv \frac{F_u}{C_P}$$

## Perturbaciones Experimentales

Análisis de propagación del error relativo:

$$Perturbación\ relativa\ de\ la\ variable\ x_j \rightarrow\ \rho_j \equiv \left| \frac{\varepsilon_j}{x_j} \right|$$

$$Variación\ relativa\ del\ resultado\ por\ la\ perturbación\ de\ x_j \rightarrow\ \omega_j \equiv \left| \frac{Y_j - Y}{Y} \right|$$

Suponiendo despreciable el error de redondeo tenemos que: $|f_j| \approx \frac{\omega_j}{\rho_j}$

Análisis de propagación del error de redondeo:

$$\text{Variación relativa del resultado por el cambio de precisión} \rightarrow \omega_{2t} \equiv \left| \frac{Y_{2t} - Y}{Y_{2t}} \right|$$

$$F_u \approx \frac{\omega_{2t}}{u}\ (u: unidad\ de\ máquina)$$

# Ecuaciones NO lineales (Guía 3.a)

El problema tipo a resolver es:

$$Hallar\ x\ /\ F(x) = 0$$

Orden de convergencia:

$$P = \frac{\ln\left[\frac{e_{k+1}}{e_k}\right]}{\ln\left[\frac{e_k}{e_{k-1}}\right]}$$

Constante asintótica del error:

$$\frac{e_{k+1}}{{e_k}^P} = \mathrm{C} = \frac{e_k}{{e_{k-1}}^P}$$

Métodos de arranque:

- Bisección ($k_0 = 1$)

$$m_{k+1} = \frac{b_{k+1} + a_{k+1}}{2} \quad ; \quad \Delta m_{k+1} = \frac{b_{k+1} - a_{k+1}}{2} = \frac{b_0 - a_0}{2^{k+1}} \quad ; \quad r_{k+1} = \frac{\Delta m_{k+1}}{m_{k+1}}$$

- Regula-Falsi

$$m_{k+1} = a_k - (b_k - a_k)\frac{F(a_k)}{F(b_k) - F(a_k)}$$

$$\Delta m_{k+1} = \frac{b_{k+1} - a_{k+1}}{2} = |m_{k+1} - m_k| \quad ; \quad r_{k+1} = \frac{\Delta m_{k+1}}{m_{k+1}} = \frac{|m_{k+1} - m_k|}{m_{k+1}}$$

Métodos de punto fijo:

- Planteo común

$$g(x_k) = x_{k+1} = x_k - F(x_k)$$

- Newton-Raphson

$$g_{NR}(x_k) = x_k - \frac{F(x_k)}{F'(x_k)}$$

- Secante

$$g_{Sec}(x_k) = x_k - (x_k - x_{k-1})\frac{F(x_k)}{F(x_k) - F(x_{k-1})}$$

Para todos los casos, la cota del error absoluto y relativo se calcula como:

$$\Delta x_{k+1} = |x_{k+1} - x_k| \quad ; \quad r_{k+1} = \frac{\Delta x_{k+1}}{x_{k+1}} = \frac{|x_{k+1} - x_k|}{x_{k+1}}$$

Existen dos condiciones suficientes pero no necesarias que garantizan la convergencia del método de punto fijo en un intervalo $[a, b]$.

1) **Condición de existencia**: garantiza que existe al menos una raíz dentro del intervalo.

$$Debe\ existir\ una\ función\ g(x)\ continua\ en\ [a,b]\ /\ g_{(x)}\epsilon[a,b]\ \forall\ x\epsilon[a,b]$$

2) **Condición de unicidad**: garantiza que dicha raíz es única.

$$Debe\ \exists g'_{(x)}\ \in (a,b)\ y\ \exists\ 0 < K < 1\ /\ |g'_{(x)}\ | \leq K < 1\ \ \forall\ x\epsilon[a,b]$$

Condiciones extras de convergencia para Newton - Raphson

Además de cumplir con las condiciones comunes para cualquier método de punto fijo, debe cumplir que:

1) $\exists\ F'(x) \neq 0\ \forall\ x \in (a,b)\ tal\ que\ \exists\ g_{NR}(x)$
2) $\exists\ F''(x)\ \forall\ x \in (a,b)\ tal\ que\ \exists\ {g_{NR}}'(x)$. Pues:

$${g_{NR}}'(x) = 1 - \frac{F'_{(x)} * F'_{(x)} - F_{(x)} * F''_{(x)}}{{F'_{(x)}}^2} = \frac{{F'_{(x)}}^2 - {F'_{(x)}}^2 + F_{(x)} * F''_{(x)}}{{F'_{(x)}}^2}$$

$${g_{NR}}'(x) = \frac{F_{(x)} * F''_{(x)}}{{F'_{(x)}}^2}$$

3) $|{g_{NR}}'(x)| = \left|\frac{F_{(x)} * F''_{(x)}}{{F'_{(x)}}^2}\right| < 1\ \forall\ x \in (a,b).$

# Aproximaciones (Guías 4 y 5)

## Ajuste discreto por cuadrados mínimos

Ecuación Normal:

$$\sum_{j=0}^{n} C_j \langle \varphi_{k(x_i)}, \varphi_{j(x_i)} \rangle = \langle \varphi_{k(x_i)}, f_{(x_i)} \rangle \quad con \quad \begin{cases} k = 0, \cdots, n \\ i = 0, \cdots, m \end{cases}$$

En forma matricial: $\varphi_k$ y $\varphi_j$ son vectores y $\langle \varphi_k, \varphi_j \rangle$ indica el producto interno entre ellos.

$$A_{jk} * c = b_k$$

$$\begin{bmatrix} < \varphi_0, \varphi_0 > & < \varphi_0, \varphi_1 > & \cdots & < \varphi_0, \varphi_n > \\ < \varphi_1, \varphi_0 > & < \varphi_1, \varphi_1 > & \cdots & < \varphi_1, \varphi_n > \\ \vdots & \vdots & \ddots & \vdots \\ < \varphi_n, \varphi_0 > & < \varphi_n, \varphi_1 > & \cdots & < \varphi_n, \varphi_n > \end{bmatrix} * \begin{bmatrix} C_0 \\ C_1 \\ \vdots \\ C_n \end{bmatrix} = \begin{bmatrix} < f, \varphi_0 > \\ < f, \varphi_1 > \\ \vdots \\ < f, \varphi_n > \end{bmatrix}$$

Error cuadrático global absoluto:

$$\|e\| = \sqrt{\sum_{i=0}^{m} {e_i}^2} = \sqrt{\sum_{i=0}^{m} \left[{f^*}_{(x_i)} - f_{(x_i)}\right]^2}$$

Error cuadrático global relativo:

$$\|r\| = \frac{\|e\|}{\|f\|} \quad con\ \|f\| = \sqrt{\sum_{i=0}^{m} {f_{(x_i)}}^2}$$

## Ajuste continuo por cuadrados mínimos

Ecuaciones Normales:

$$\sum_{k=0}^{n} a_k \int_a^b \left[x^k \cdot x^j\right] dx = \int_a^b \left[f_{(x)} \cdot x^j\right] dx$$

En forma matricial:

$$\bar{\bar{H}} * \bar{a} = \bar{r}$$

Los elementos del vector de coeficientes $\bar{a}$ son $a_k \; con \; k = 0, \cdots, n$ . Los coeficientes de la Matriz de Hilbert $\left(\bar{\bar{H}}\right)$ se calculan como:

$$H_{j+1,k+1} = \frac{b^{j+k+1} - a^{j+k+1}}{j+k+1}$$

Los elementos del vector de términos independientes $\bar{r}$ son $r_{j+1}$ y se calculan como:

$$r_{j+1} = \int_a^b \left[f_{(x)} * x^j\right] dx$$

Los problemas se pueden resolver obteniendo la matriz de Hilbert o bien se puede resolver analíticamente por medio de la definición del error:

$$E = \int_a^b e^2 = \int_a^b \left[f_{(x)} - P_{n(x)}\right]^2 dx$$

## Interpolación discreta

Polinomio de Lagrange:

$$p_L(x) = \sum_{k=0}^{n} L_{nk}(x) \cdot f(x_k) \quad ; \quad L_{nk}(x) = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{x - x_i}{x_k - x_i} \rightarrow \; L_{nk}(x) = \begin{cases} 0 \; si \; i \neq k \\ 1 \; si \; i = k \end{cases}$$

Polinomio de Newton:

$$p_N(x) = C_0 + \sum_{j=1}^{m} C_j (x - x_0)(x - x_1) \cdots (x - x_{j-1})$$

$$Esquema \; de \; cálculo: \quad \begin{array}{cc|ccc} x_i & f_{(x_i)} & & & \\ x_0 & f_{(x_0)} & f_{[x_0,x_1]} & & \\ x_1 & f_{(x_1)} & f_{[x_1,x_2]} & f_{[x_0,x_1,x_2]} & \\ x_2 & f_{(x_2)} & f_{[x_2,x_3]} & f_{[x_1,x_2,x_3]} & f_{[x_0,x_1,x_2,x_3]} \\ x_3 & f_{(x_3)} & f_{[x_3,x_4]} & f_{[x_2,x_3,x_4]} & f_{[x_1,x_2,x_3,x_4]} \\ x_4 & f_{(x_4)} & & & \end{array} \; f_{[x_0,x_1,x_2,x_3,x_4]}$$

Polinomio de Hermite (Método Directo por Lagrange):

$$H_{2m+1}(x) = \sum_{j=0}^{m} f(x_j) * H_{mj}(x) + \sum_{j=0}^{m} f'(x_j) * \widehat{H}_{mj}(x)$$

Donde:

- $H_{mj}(x) \equiv \left[1 - 2 * \left(x - x_j\right) * {L'}_{mj}(x_j)\right] * {L^2}_{mj}(x)$
- $\widehat{H}_{mj}(x) \equiv \left(x - x_j\right) * {L^2}_{mj}(x)$
- $L_{mj}(x)$: $Polinomio\ de\ Lagrange$
- ${L'}_{mj}(x)$: $Derivada\ del\ Polinomio\ de\ Lagrange$

Polinomio de Hermite (Método generalizado por diferencias divididas de Newton): agrega las derivadas en el esquema de Newton.

$$
\begin{array}{cc|ccccccc}
x_i & f_{(x_i)} & & & & & & & \\
x_0 & f_{(x_0)} & f_{[x_0,x_0]} = f'_{(x_0)} & & & & & & \\
x_0 & f_{(x_0)} & & f_{[x_0,x_1]} & & & & & \\
x_1 & f_{(x_1)} & f_{[x_1,x_1]} = f'_{(x_1)} & & f_{[x_0,x_0,x_1]} & & & & \\
x_1 & f_{(x_1)} & & f_{[x_1,x_2]} & & f_{[x_0,x_0,x_1,x_1]} & & & \\
x_2 & f_{(x_2)} & f_{[x_2,x_2]} = f'_{(x_2)} & & f_{[x_0,x_1,x_1]} & & f_{[x_0,x_0,x_1,x_1,x_2]} & & \\
x_2 & f_{(x_2)} & & f_{[x_2,x_3]} & & f_{[x_0,x_1,x_2,x_2]} & & f_{[x_0,x_0,x_1,x_1,x_2,x_2]} & \\
x_3 & f_{(x_3)} & f_{[x_3,x_3]} = f'_{(x_3)} & & f_{[x_1,x_1,x_2]} & & f_{[x_0,x_1,x_1,x_2,x_2]} & & f_{[x_0,x_0,x_1,x_1,x_2,x_2,x_3]} \\
x_3 & f_{(x_3)} & & f_{[x_3,x_4]} & & f_{[x_1,x_2,x_2,x_3]} & & f_{[x_0,x_1,x_1,x_2,x_2,x_3]} & \\
x_4 & f_{(x_4)} & f_{[x_4,x_4]} = f'_{(x_4)} & & f_{[x_2,x_2,x_3]} & & f_{[x_1,x_1,x_2,x_2,x_3]} & & f_{[x_0,x_0,x_1,x_1,x_2,x_2,x_3,x_3]} \\
x_4 & f_{(x_4)} & & & & & & & \\
\end{array}
$$

$$
\begin{array}{cc|ccc}
f_{[x_0,x_0,x_1,x_1,x_2,x_2,x_3]} & & & & \\
f_{[x_0,x_1,x_1,x_2,x_2,x_3,x_3]} & f_{[x_0,x_0,x_1,x_1,x_2,x_2,x_3,x_3]} & & & \\
f_{[x_1,x_1,x_2,x_2,x_3,x_3,x_4]} & f_{[x_0,x_1,x_1,x_2,x_2,x_3,x_3,x_4]} & f_{[x_0,x_0,x_1,x_1,x_2,x_2,x_3,x_3,x_4]} & & \\
f_{[x_1,x_2,x_2,x_3,x_3,x_4,x_4]} & f_{[x_1,x_1,x_2,x_2,x_3,x_3,x_4,x_4]} & f_{[x_0,x_1,x_1,x_2,x_2,x_3,x_3,x_4,x_4]} & f_{[x_0,x_0,x_1,x_1,x_2,x_2,x_3,x_3,x_4x_4]} &
\end{array}
$$

Polinomio de Tchebychef: **para eliminar el fenómeno de Runge.**

1) Por medio de la siguiente transformación lineal se convierten los valores $i \in [a,b]\ con\ i = 1, \dots, m$ a valores $t_i \in [-1,1]$. El $i = 0$ no se utiliza ya que $t_0 = t_1$ por lo que, de usarse, se estaría duplicando un dato.

$$t_i = \cos\left(\frac{2i-1}{n} * \frac{\pi}{2}\right)$$

Siendo $n$ la cantidad de puntos dato, es decir: $n = m + 1$

2) Luego, a través de la expresión a continuación, se obtienen las abscisas de Tchebychef que serán los nuevos puntos $\tilde{x}_i$ dato, pertenecientes al intervalo $[a,b]$ para poder calcular el polinomio interpolador.

$$\tilde{x}_i = a + \frac{b-a}{2}(t_i + 1)$$

3) Se evalúan los nuevos puntos $\tilde{x}_i$ en la función $f_{(x)}$ y se obtienen las imágenes $f_{(\tilde{x}_i)}$.

4) Finalmente se emplea alguno de los métodos conocidos (Lagrange, Newton o Hermite) para generar un nuevo polinomio interpolante a partir de los $\tilde{x}_i$, el cual habrá aumentado su precisión respecto al enseñado en el gráfico anterior.

Cabe aclarar que el polinomio será de grado $m - 1$, ya que al contar con $m$ datos $\tilde{x}_i$ el orden de este se reduce en uno, dado que un nodo se emplea para el término independiente.

# Sistemas de Ecuaciones Lineales (Guía 2)

Norma de una matriz:

$$\left.\begin{array}{l} Norma\ infinito \rightarrow \left\|\bar{\bar{A}}\right\|_{\infty} = \max\limits_{1\le i\le n}\sum\limits_{j=1}^{n}|a_{ij}| \\ Norma\ uno \rightarrow \left\|\bar{\bar{A}}\right\|_{1} = \max\limits_{1\le j\le n}\sum\limits_{i=1}^{n}|a_{ij}| \end{array}\right\} siendo\ \bar{\bar{A}} \in R^{nxn}$$

## Eliminación de Gauss:

Las operaciones básicas con las que se cuentan son:

- $\gamma E_i \rightarrow E_i\ con\ \gamma \neq 0$ → Multiplicar una fila por una constante
- $E_i + \gamma E_j \rightarrow E_i$ → Sumar a una fila otra multiplicada por una constante
- $E_i \leftrightarrow E_j$ → Intercambiar dos filas de la matriz.

El algoritmo del método es:

- Siempre que $a_{ii} \neq 0$, se aplica la transformación lineal: $(E: ecuación)$

$$E_j - \frac{a_{ji}}{a_{ii}}E_i \rightarrow E_j \quad con \quad \begin{cases} j = k, k+1, \cdots, n \\ k = i+1 \end{cases}$$

Donde $m_{ij} = \frac{a_{ij}}{a_{ii}}$ se denomina **multiplicador.** Luego, para obtener las incógnitas del problema se realiza una **sustitución hacia atrás.**

$$x_i = \frac{{a_{i,n+1}}^{(i)}}{{a_{ii}}^{(i)}} - \frac{1}{{a_{ii}}^{(i)}}\sum_{j=i+1}^{n}{a_{ij}}^{(i)}x_j\ \ con\ i = n-1, n-2, \cdots, 1$$

Matriz aumentada:

$$[\bar{\bar{A}}, \bar{b}]^{(n)} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & a_{1,n+1} \\ 0 & {a_{22}}^{(2)} & \cdots & {a_{2n}}^{(2)} & {a_{2,n+1}}^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & {a_{nn}}^{(n)} & {a_{n,n+1}}^{(n)} \end{bmatrix}$$

Técnicas de pivoteo

- **Pivoteo parcial:** se busca usar como pivote $(a_{ii})$ el menor entero de la columna $i$. Por lo que se permutan las filas de la matriz aumentada.
- **Pivoteo total o completo:** se busca usar como pivote $(a_{ii})$ el menor entero entre la fila $i$ y la columna $i$. Por lo que deberán permutarse dos columnas o bien dos filas. En el primer caso corresponde permutar las filas del vector incógnita.

Propagación de errores

Propagación de **errores de entrada** ($K_{(A)}$: número de condición de A):

$$\frac{\left\|\overline{\delta x}\right\|}{\|\bar{x}\|} \le K_{(A)}\left(\frac{\left\|\overline{\delta b}\right\|}{\|\bar{b}\|} + \frac{\left\|\overline{\overline{\delta A}}\right\|}{\left\|\bar{\bar{A}}\right\|}\right) \quad con\ K_{(A)} = \left\|\bar{\bar{A}}\right\|\left\|\bar{\bar{A}}^{-1}\right\|$$

- Si $K_{(A)}$ es pequeño, léase aproximado a orden 1 → $\bar{\bar{A}}$ se encuentra bien condicionada.
- Si $K_{(A)}$ es grande, léase mucho mayor a orden 1 → $\bar{\bar{A}}$ se encuentra mal condicionada.

IMPORTANTE: $K_{(A)}$ no puede ser menor que 1, por definición.

Propagación de los **errores de redondeo,** debido al redondeo al resolver se obtiene una solución $\tilde{\boldsymbol{x}}$ que NO satisface el sistema original, aparece entonces el residuo $\bar{\boldsymbol{R}}$:

$$\bar{R} = \bar{\bar{A}}\bar{x} - \bar{\bar{A}}\tilde{x} = \bar{b} - \tilde{b}$$

$$\bar{x} - \tilde{x} = \bar{\bar{A}}^{-1}\bar{R}$$

Entonces, acotando y usando que $\|\bar{b}\| \leq \|\bar{\bar{A}}\|\,\|\bar{x}\|$:

$$\frac{\|\bar{x} - \tilde{x}\|}{\|\bar{x}\|} \leq K_{(A)} \frac{\|\bar{R}\|}{\|\bar{b}\|}$$

Observaciones:

- Para obtener $\tilde{b}$ se utiliza doble precisión para minimizar el error de redondeo.
- Además, se realizan muchas menos operaciones para calcular $\bar{\bar{A}}\tilde{x}$ que para calcular $\bar{\bar{A}}$.

## Descomposición LU

Consiste en descomponer la matriz $\bar{\bar{A}}$ como el producto entre dos matrices $\bar{\bar{L}}$ (lower) y $\bar{\bar{U}}$ (upper):

$$\bar{\bar{A}} = \bar{\bar{L}} * \bar{\bar{U}}$$

$$\bar{\bar{L}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21}^{(1)} & 1 & 0 & 0 \\ m_{31}^{(1)} & m_{32}^{(2)} & 1 & 0 \\ m_{41}^{(1)} & m_{42}^{(2)} & m_{43}^{(3)} & 1 \end{bmatrix} \; ; \; \bar{\bar{U}} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & a_{14}^{(1)} \\ 0 & a_{22}{}^{(2)} & a_{23}{}^{(2)} & a_{24}{}^{(2)} \\ 0 & 0 & a_{33}{}^{(3)} & a_{34}{}^{(3)} \\ 0 & 0 & 0 & a_{44}{}^{(4)} \end{bmatrix}$$

Se resuelven entonces dos SEL:

$$\bar{\bar{A}}\,\bar{x} = \bar{b} \rightarrow \bar{\bar{L}}\,\bar{\bar{U}}\,\bar{x} = \bar{b} \quad \Longrightarrow \quad \bar{\bar{U}}\bar{x} = \bar{y} \quad \Longrightarrow \quad \bar{\bar{L}}\,\bar{y} = \bar{b}$$

Existen tres formas de armar las matrices para la descomposición LU:

1) Método de Doolittle → $l_{kk} = 1$
2) Método de Croot → $u_{kk} = 1$
3) Método de Chlesky → $l_{kk} = u_{kk}$

Nosotros utilizamos el Método de Doolittle porque guardamos los multiplicadores durante el proceso.

## Refinamiento iterativo

Residuo:

$$\bar{R}^{(1)} = \bar{b} - \tilde{b} = \bar{b} - \bar{\bar{A}}\tilde{x}^{(1)}$$

Método de refinamiento iterativo:

$$\bar{R}^{(k)} = \bar{b} - \bar{\bar{A}}\tilde{x}^{(k)} \quad \Longrightarrow \quad \bar{\bar{A}}\,\overline{\delta x}^{(k)} = \bar{R}^{(k)} \quad \Longrightarrow \quad \tilde{x}^{(k+1)} = \tilde{x}^{(k)} + \overline{\delta x}^{(k)} \quad \Longrightarrow \quad repito$$

Fórmulas útiles:

$$q = t - p \quad ; \quad p = \log\left(K_{(A)}\right) \quad ; \quad K_{(A)} \approx \frac{\left\|\overline{\delta x}^{(1)}\right\|}{\left\|\bar{x}^{(1)}\right\|} * 10^t$$

- La $q$ nos da una idea de cuantos dígitos vamos a ganar por cada paso iterativo realizado, por lo tanto, observando el resultado de esta fórmula podremos saber si tiene sentido o no utilizar el procedimiento de refinamiento iterativo y cuantas veces lo deberemos aplicar para una deseada precisión en el resultado final.
- El valor de $p$ se define al inicio y queda fijo para el resto del problema.
- La última expresión es un atajo para calcular $K_{(A)}$ y esta planteada con norma infinito y solo se debe plantear para la primera aproximación del refinamiento.

## Métodos Iterativos

Resuelven:

$$\bar{x}^{(k)} = \bar{\bar{T}} * \bar{x}^{(k-1)} + \bar{c}$$

## Método de Jacobi

$$T_{ij} = -\frac{a_{ij}}{a_{ii}}\left(1 - \delta_{ij}\right) \quad c_i = \frac{b_i}{a_{ii}} \quad con\ i, j = 1, \cdots, n$$

Forma matricial:

$$\bar{\bar{T}}_J = \bar{\bar{D}}^{-1}\left[\bar{\bar{L}} + \bar{\bar{U}}\right], \qquad y, \qquad \bar{c}_J = \bar{\bar{D}}^{-1}\bar{b}$$

Con:

$$\bar{\bar{A}} = \bar{\bar{D}} - \bar{\bar{L}} - \bar{\bar{U}}$$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix} - \begin{bmatrix} 0 & 0 & \cdots & 0 \\ -a_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & 0 \end{bmatrix} - \begin{bmatrix} 0 & -a_{12} & \cdots & -a_{1n} \\ 0 & 0 & \cdots & -a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

Criterios de convergencia (validos para los otros métodos)

Para lograr la convergencia para toda semilla $\bar{x}^{(0)}$ es **condición necesaria y suficiente** que el radio espectral de la matriz $\bar{\bar{T}}_J$ sea menor a 1, es decir:

$$\rho\left(\bar{\bar{T}}_J\right) = \max_{1 \le i \le n} |\mu_i| < 1$$

Donde $\rho\left(\bar{\bar{T}}_J\right)$ representa al **radio espectral** de la matriz $\bar{\bar{T}}_J$ y $\mu_i$ representa a los autovalores de dicha matriz.

Además:

$$Si\ \left\|\bar{\bar{T}}_J\right\| < 1\ el\ método\ converge\ \forall \bar{x}^{(0)}$$

Como consecuencia de ello se tiene la siguiente **propiedad:** si $\bar{\bar{A}}$ es estrictamente diagonal dominante, el método converge $\forall \bar{x}^{(0)}$.

Criterio de corte

$$\frac{\left\|\bar{x}^{(k)} - \bar{x}^{(k-1)}\right\|}{\left\|\bar{x}^{(k)}\right\|} < Tolerancia$$

Error de truncamiento

Si se cumple $\|\bar{\bar{T}}_J\| < 1$, la **cota para el error de truncamiento** viene dada por cualquiera de las siguientes expresiones:

$$\begin{cases} \|\bar{x} - \bar{x}^{(k)}\| \leq \dfrac{\|\bar{\bar{T}}_J\|^k}{1 - \|\bar{\bar{T}}_J\|} \|\bar{x}^{(1)} - \bar{x}^{(0)}\| \\ \|\bar{x} - \bar{x}^{(k)}\| \leq \dfrac{\|\bar{\bar{T}}_J\|}{1 - \|\bar{\bar{T}}_J\|} \|\bar{x}^{(k)} - \bar{x}^{(k-1)}\| \end{cases}$$

Si $\|\bar{\bar{T}}_J\| < \frac{1}{2}$, la cota para dicho error puede tomarse directamente como $\|\bar{x}^{(k)} - \bar{x}^{(k-1)}\|$. Se utiliza la norma infinito.

Propagación de errores de entrada

$$\|\hat{x}^{(k)} - \bar{x}\| \leq \frac{\|\overline{\overline{\delta T}}\|}{1 - \|\bar{\bar{T}}\|} \|\bar{x}^{(k-1)}\| + \frac{\|\overline{\delta c}\|}{1 - \|\bar{\bar{T}}\|}$$

Donde $\hat{x}^{(k)}$ es la mejor aproximación obtenida, es decir, la originada al finalizar el proceso en la iteración $k$. $\overline{\overline{\delta T}}$ y $\overline{\delta c}$ representan a los errores absolutos de entrada transformados.

## Método de Gauss-Seidel

Es Jacobi pero utilizando los valores conseguidos en el mismo paso.

$$x_i{}^{(k)} = -\frac{1}{a_{ii}} \sum_{j=1}^{i-1} a_{ij} x_j{}^{(k)} - \frac{1}{a_{ii}} \sum_{j=i+1}^{n} a_{ij} x_j{}^{(k-1)} + \frac{b_i}{a_{ii}}, \qquad con\ i = 1, \cdots, n\ y\ k \geq 1$$

$$\begin{aligned} x_1^{(k)} &= \tfrac{1}{10}x_2^{(k-1)} - \tfrac{1}{5}x_3^{(k-1)} + \tfrac{3}{5}, \\ x_2^{(k)} &= \tfrac{1}{11}x_1^{(k)} + \tfrac{1}{11}x_3^{(k-1)} - \tfrac{3}{11}x_4^{(k-1)} + \tfrac{25}{11}, \\ x_3^{(k)} &= -\tfrac{1}{5}x_1^{(k)} + \tfrac{1}{10}x_2^{(k)} + \tfrac{1}{10}x_4^{(k-1)} - \tfrac{11}{10}, \\ x_4^{(k)} &= -\tfrac{3}{8}x_2^{(k)} + \tfrac{1}{8}x_3^{(k)} + \tfrac{15}{8}. \end{aligned}$$

## Método de SOR (Succesive Over-Relaxation)

$$x_i{}^{(k)} = x_i{}^{(k-1)} + \boldsymbol{\omega} \frac{r_i{}^{(k)}}{a_{ii}}$$

Siendo $\omega$ el factor de peso o parámetro de relajación. Con:

$$r_i{}^{(k)} = \begin{cases} b_i - \sum_{j=1}^{i-1} a_{ij} x_j{}^{(k)} - \sum_{j=i}^{n} a_{ij} x_j{}^{(k-1)} \rightarrow Para\ Gauss - Seidel \\ b_i - \sum_{j=1}^{n} a_{ij} x_j{}^{(k-1)} \rightarrow Para\ Jacobi \end{cases}$$

La programación del método resulta directa a partir del Método de Gauss-Seidel:

$$x_i^{(k)} = (1-\omega)x_i^{(k-1)} + \omega\left[x_i^{(k)}\right]_{GS}$$

O en forma matricial:

$$\bar{\bar{T}}_\omega = \left[\bar{\bar{D}} - \omega\bar{\bar{L}}\right]^{-1}\left[(1-\omega)\bar{\bar{D}} + \omega\bar{\bar{U}}\right] \quad ; \quad \bar{c}_\omega = \omega\left[\bar{\bar{D}} - \omega\bar{\bar{L}}\right]^{-1}\bar{b}$$

Donde:

- Si $0 < \omega < 1$ se trata de sub-relajaciones.
- Si $1 \leq \omega < 2$ se trata de sobre relajaciones.
- Si $\omega = 1$ se trata del método de Gauss-Seidel.

Nosotros siempre trabajamos con sobre relajaciones por lo que utilizaremos $\omega \in [1,2)$.

Observaciones:

- Si $\bar{\bar{A}}$ es definida positiva y se cumple $\omega \in (0,2)$ → SOR converge ∀ $\bar{x}^{(0)}$.
- Si $\bar{\bar{A}}$ es simétrica y definida positiva → SOR converge ∀ $\bar{x}^{(0)}$.

## Aplicaciones en Ingeniería

1) Si $\bar{\bar{A}}$ es definida positiva y tridiagonal, la máxima velocidad se obtiene con:

$$\omega_{optimo} = \frac{2}{1 + \sqrt{1 - \rho\left(\bar{\bar{T}}_J\right)^2}}$$

   Siendo $\bar{\bar{T}}_j = \bar{\bar{D}}^{-1}(\bar{\bar{L}} + \bar{\bar{U}})$ la matriz de Jacobi, y en este caso $\rho\left(\bar{\bar{T}}_\omega\right) = \omega - 1$.

2) Si $\bar{\bar{A}}$ no satisface la condición anterior y se tienen que resolver múltiples SEL definidos por la misma $\bar{\bar{A}}$ y distintos $\bar{b}$ es conveniente hallar $\omega$ en términos prácticos. Para ello se realiza un gráfico de
   - $k\ vs\ \omega$, donde para un valor constante del error absoluto se observa como varía el número de iteraciones $k$ que necesitas para alcanzar la precisión deseada respecto a distintos valores de $\omega$.
   - $error\ vs\ \omega$, donde para un número constante de iteraciones $k$ se observa como cambia el valor del error absoluto a medida que varía $\omega$.

En ambos casos, el $\omega_{optimo}$ estará dado por el mínimo de la curva graficada, por lo que, se guardará ese valor para utilizarlo en todos los cálculos y reducir el costo computacional.

# Sistemas de Ecuaciones NO Lineales (Guía 3b)

**Método de Punto Fijo:**

$$\bar{G}_{(\bar{x})} = \bar{x} - \bar{F}_{(\bar{x})}$$

Siendo $\bar{G} = (g_1, g_2, \cdots, g_n)$.

Contando con un sistema de 2x2 de la forma: $\begin{cases} x_1 - \frac{1}{3}\cos(x_2) = \frac{1}{3} \\ \frac{1}{20}e^{-x_1} + x_2 = -\frac{1}{2} \end{cases}$

$$\bar{G}_{(\bar{x})} = \bar{x} - \bar{F}_{(\bar{x})} = \begin{bmatrix} g_1(x_1, x_2) \\ g_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1 - x_1 + \frac{1}{3}\cos(x_2) + \frac{1}{3} \\ x_2 - \frac{1}{20}e^{-x_1} - x_2 - \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{1}{3}\cos(x_2) + \frac{1}{3} \\ -\frac{1}{20}e^{-x_1} - \frac{1}{2} \end{bmatrix}$$

Finalmente, mi sistema resulta:

$$\begin{cases} {x_1}^{k+1} = \frac{1}{3}\cos({x_2}^k) + \frac{1}{3} \\ {x_2}^{k+1} = -\frac{1}{20}e^{-{x_1}^k} - \frac{1}{2} \end{cases}$$

**Método de Gauss-Seidel:**

Recordamos que Gauss-Seidel plantea ir utilizando las variables calculadas dentro del mismo paso iterativo para obtener las demás variables restantes.

En consecuencia, el sistema antes presentado simplemente se modifica como:

$$\begin{cases} {x_1}^{k+1} = \frac{1}{3}\cos({x_2}^k) + \frac{1}{3} \\ {x_2}^{k+1} = -\frac{1}{20}e^{-{x_1}^{k+1}} - \frac{1}{2} \end{cases}$$

Observación: este método no siempre consigue acelerar la convergencia.

**Método de Newton:**

Recordando el caso para una dimensión del método Newton-Raphson:

$$g_{NR} = x - \frac{F(x)}{F'(x)} = x - \varphi(x)F(x), \qquad con\ \varphi(x) = \frac{1}{F'(x)}$$

Se puede generalizar el planteo para $n$ dimensiones como:

$$\bar{G}_{(\bar{x})} = \bar{x} - {J_{(\bar{x})}}^{-1}\bar{F}_{(\bar{x})}$$

Con $\varphi(x) = {J_{(\bar{x})}}^{-1}$ siendo $J_{(\bar{x})}$ la matriz jacobiana de la forma:

$$J_{ij}(x) = \frac{\partial F_i(x)}{\partial x_j}$$

La forma iterativa para el método resulta:

$$x^{k+1} = G(x^k) = x^k - {J_{(x^k)}}^{-1}F_{(x^k)}$$

Observaciones:

- Se obtiene una **convergencia cuadrática**
- El costo computacional es mayor ya que se requieren calcular las derivadas e invertir el Jacobiano.

Una forma más sencilla del método se puede obtener a partir de plantear el siguiente cambio de variables:

$$y^k = -{J_{(x^k)}}^{-1} F_{(x^k)} \rightarrow J_{(x^k)} y^k = -J_{(x^k)} {J_{(x^k)}}^{-1} F_{(x^k)} \rightarrow J_{(x^k)} y^k = -F_{(x^k)}$$

Siendo esta última expresión un sistema lineal ya que esta evaluado en $x^k$, es decir, son matrices y vectores con números. De manera que, se puede reemplazar en la ecuación para la forma iterativa del método y obtenemos:

$$x^{k+1} = x^k - {J_{(x^k)}}^{-1} F_{(x^k)} \rightarrow x^{k+1} = x^k + {J_{(x^k)}}^{-1} \left(-F_{(x^k)}\right) \rightarrow$$

$$x^{k+1} = x^k + {J_{(x^k)}}^{-1} J_{(x^k)} y^k \rightarrow$$

$$x^{k+1} = x^k + y^k$$

Es decir, el método de Newton consiste básicamente en resolver un Sistema de Ecuaciones Lineales. Para el ejemplo anterior correspondería:

$$\bar{F} = \begin{bmatrix} F_1(x_1, x_2) \\ F_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1 - \frac{1}{3}\cos(x_2) - \frac{1}{3} \\ \frac{1}{20} e^{-x_1} + x_2 + \frac{1}{2} \end{bmatrix} \rightarrow \begin{cases} {x_1}^{k+1} = \frac{1}{3}\cos\left({x_2}^k\right) + \frac{1}{3} \\ {x_2}^{k+1} = -\frac{1}{20} e^{-{x_1}^k} - \frac{1}{2} \end{cases}$$

$$J_{(x^k)} = \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{3} sen(x_2) \\ -\frac{1}{20} e^{-x_1} & 1 \end{bmatrix} \rightarrow Además, hay\ que\ invertirlo$$

**<u>Método Cuasi-Newton</u>**

Para evitar el cálculo del jacobiano del caso anterior, en casos reales de matrices de gran dimensión, se aplica la idea de la secante donde se aproxima la derivada como:

$$\frac{\partial F_i(x^k)}{\partial x_j} \approx \frac{F_i\left(x^k + h e_j\right) - F_i(x^k)}{h}$$

Con $e_j = (0,0,\cdots,1,\cdots,0,0)$, léase un incremento pequeño en la variables $x_j$.

Generalmente la matriz jacobiana se calcula cada ciertos pasos con el fin de reducir el costo computacional un poco. Es decir, se utiliza una $J$ para cierta cantidad de iteraciones y luego se la vuelve a calcular.

# Integración (Guía 6)

La expresión general es:

$$I_i = h\left[f_m + \frac{1}{24}f_m''h^2 + \frac{1}{1920}f_m{}^{iv}h^4 + \cdots\right]$$

Es a la hora de truncar la expresión cuando surgen los distintos métodos de aproximación.

## Método del Rectángulo (R)

Corresponde a aproximar las sub-integrales locales al orden más bajo posible, o sea $n = 0$:

$$I \approx R = h\sum_{i=1}^{N} f_{\left(\frac{x_{i-1}+x_i}{2}\right)}$$

Error de truncamiento:

$$R - I = \sum_{i=1}^{N} R_i - \sum_{i=1}^{N} I_i = \sum_{i=1}^{N}(R_i - I_i) = -\frac{1}{24}h^2\left\{\sum_{i=1}^{N} f_m''h\right\} + O(h^4)$$

Resulta ser de **orden 2** siendo lo que está contenido entre llaves un valor constante.

## Método del Trapecio (T)

Dado que a veces no es posible evaluar la función $f$ en el punto medio del intervalo (por ejemplo, cuando se hace un muestreo), se dispone del método del trapecio para resolver dichos casos. Entonces, se hará uso de los puntos disponible como los puntos $x_i$. Su expresión es:

$$I \approx T = h\sum_{i=1}^{N}\left(\frac{f_{(x_{i-1})} + f_{(x_i)}}{2}\right)$$

Para minimizar el error se puede reducir el número de operaciones:

$$T = h\frac{f_0}{2} + h\sum_{i=1}^{N-1} f_i + h\frac{f_N}{2}$$

Error de truncamiento surge de la expresión:

$$T - I = \sum_{i=1}^{N} T_i - \sum_{i=1}^{N} I_i = \sum_{i=1}^{N}(T_i - I_i) = -\frac{1}{12}h^2\left\{\sum_{i=1}^{N} f_m''h\right\} + O(h^4)$$

Y sigue siendo de **orden 2**.

## Extrapolación de Richardson

Es necesario primero definir el concepto de **precisión.** La precisión de una fórmula de cuadratura está asociada con el polinomio de mayor grado que la fórmula puede integrar de manera exacta. Para las **reglas del Rectángulo y del Trapecio**, dado que utilizan rectas para aproximar polinomios poseen una **precisión de 1**. Es decir, solo pueden integrar de manera exacta polinomios de grado 1.

La Extrapolación de Richardson **sirve para lograr precisión 2 a partir de métodos de precisión 1.** Su expresión es:

$$N_j\left(\frac{h}{2}\right) = N_{j-1}\left(\frac{h}{2}\right) + \left(\frac{N_{j-1}\left(\frac{h}{2}\right) - N_{j-1}(h)}{2^{j-1} - 1}\right)$$

Donde $N$ representa a una regla numérica de integración (léase, trapecio, rectángulo u otra), $j$ representa el orden de la nueva fórmula obtenida siendo de orden $h^j$. Observación: dicha fórmula se obtiene de partir el intervalo $h$ en la mitad. Si se partiese de una forma diferente se conseguiría una fórmula distinta.

De forma más genérica la extrapolación de Richardson se puede expresar como:

$$I_{Rich} = I_{(h_1)} + \frac{I_{(h_1)} - I_{(h_2)}}{\left(\frac{h_2}{h_1}\right)^p - 1} + O(h^r)$$

Donde $h_2 > h_1$ son dos pasos que pueden ser o no iguales.

## Método de Simpson

Sabiendo que el método de Trapecios es de orden 2 en lo que respecta al error de truncamiento, se puede aplicar una extrapolación de Richardson y generar un método cuyo error de truncamiento sea de orden 3.

La **regla de Simpson 1/3 compuesta** para un intervalo global resulta:

$$S_{(h)} = \frac{h}{3}\left[f_{(a)} + 2\sum_{i=1}^{N-1} f_{2i} + 4\sum_{i=1}^{N} f_{2i-1} + f_{(b)}\right]$$

Esta ecuación aplica par N par con $x_i = a + ih$ con $i = 0, \dots, N$ y el paso resulta $h = \frac{b-a}{N}$.

De la misma forma, el error de truncamiento esta dado por:

$$S - I = -\frac{b-a}{180} f_{(\mu)}{}^{iv} h^4$$

## Método de Romberg

Este método aplica el uso consecutivo de la extrapolación de Richardson. La construcción de las aproximaciones se consigue por los siguientes pasos:

1) Utiliza la regla compuesta del trapecio para un intervalo global como aproximación preliminar.
2) Aplica la extrapolación de Richardson hasta conseguir el orden deseado.

## Cuadratura de Gauss

Permite obtener un mejor resultado con el mismo esfuerzo de cálculo. Para ello, se plantea una cuadratura con puntos y coeficientes de peso a definir:

$$I = \int_a^b f(x)dx \approx \sum_i c_i f_{(x_i)} = G$$

Con las siguientes condiciones:

- $x_1, x_2, \dots, x_n \in [a, b]$.
- $c_1, c_2, \dots, c_n$ deben ser tales que minimicen el error de truncamiento dado por $G - I$.

El criterio de selección de las $2n$ incógnitas $\{x_i, c_i\}$ es que integren en forma exacta el polinomio de mayor grafo que se puede construir con 2n parámetros.

$$p(x) \in P_{2n-1}(x)$$

Los coeficientes se pueden obtener de la fórmula de Lagrange:

$$c_i = \int_{-1}^{1} \left[ \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \right] dx$$

Los puntos de Gauss $(x_i)$ se obtienen como las raíces de los polinomios de Legendre:

$$P_2(x) = \tfrac{1}{2}(3x^2 - 1)$$
$$P_3(x) = \tfrac{1}{2}(5x^3 - 3x)$$
$$P_4(x) = \tfrac{1}{8}(35x^4 - 30x^2 + 3)$$
$$P_5(x) = \tfrac{1}{8}(63x^5 - 70x^3 + 15x)$$
$$P_6(x) = \tfrac{1}{16}(231x^6 - 315x^4 + 105x^2 - 5)$$
$$P_7(x) = \tfrac{1}{16}(429x^7 - 693x^5 + 315x^3 - 35x)$$

De todas formas, como los polinomios NO dependen de los puntos se encuentran tabulados en tablas $(c_1 = w_1)$:

| $n$ | $w_i$ | $x_i$ | $n$ | $w_i$ | $x_i$ |
|---|---|---|---|---|---|
| 1 | 2.0 | 0.0 | 8 | 0.1012285363 | ±0.9602898565 |
| 2 | 1.0 | ±0.5773502692 | | 0.2223810345 | ±0.7966664774 |
| 3 | 0.5555555556 | ±0.7745966692 | | 0.3137066459 | ±0.5255324099 |
| | 0.8888888889 | 0.0 | | 0.3626837834 | ±0.1834346425 |
| 4 | 0.3478548451 | ±0.8611363116 | 9 | 0.0812743883 | ±0.9681602395 |
| | 0.6521451549 | ±0.3399810436 | | 0.1806481607 | ±0.8360311073 |
| 5 | 0.2369268851 | ±0.9061798459 | | 0.2606106964 | ±0.6133714327 |
| | 0.4786286705 | ±0.5384693101 | | 0.3123470770 | ±0.3242534234 |
| | 0.5688888889 | 0.0 | | 0.3302393550 | 0.0 |
| 6 | 0.1713244924 | ±0.9324695142 | 10 | 0.0666713443 | ±0.9739065285 |
| | 0.3607615730 | ±0.6612093865 | | 0.1494513492 | ±0.8650633667 |
| | 0.4679139346 | ±0.2386191861 | | 0.2190863625 | ±0.6794095683 |
| 7 | 0.1294849662 | ±0.9491079123 | | 0.2692667193 | ±0.4333953941 |
| | 0.2797053915 | ±0.7415311856 | | 0.2955242247 | ±0.1488743390 |
| | 0.3818300505 | ±0.4058451514 | | | |
| | 0.4179591837 | 0.0 | | | |

# Diferenciación (Guía 7)

## Fórmula de 2 puntos

Para $h$ suficientemente pequeño tenemos que:

$$f'_{(x)} \approx \frac{f_{(x_0+h)} - f_{(x_0)}}{h}$$

Y el error de truncamiento está acotado por $\frac{M}{2}h$ donde $M$ es tal que:

$$|f''(x)| \leq M \; \forall x \in [a,b]$$

Además, si $h > 0$ la derivada se calcula en adelanto y si $h < 0$ se está calculando en atraso.

## Formula de N+1 puntos

Para mejorar la precisión se utiliza la interpolación de Lagrange.

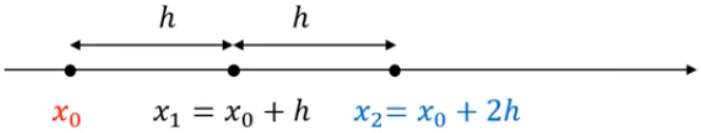
$$f_{(x)} = \sum_{k=0}^{N} f(x_k) L_{Nk}(x) + \frac{(x-x_0)(x-x_1)\dots(x-x_N)}{(N+1)!} f^{(N+1)}\left(\gamma_{(x)}\right)$$

Con $\gamma_{(x)} \in [a,b]$. Derivando y evaluando la expresión en $x = x_j$ para algún $0 \leq j \leq N$:

$$f'_{(x_j)} = \sum_{k=0}^{N} f(x_k) L'_{Nk}(x) + \frac{f^{(N+1)}\left(\gamma_{(x_j)}\right)}{(N+1)!} \prod_{\substack{K=0 \\ K \neq j}}^{N} (x_j - x_k)$$

Donde el **primer término** corresponde a la **fórmula de N+1 puntos** y el segundo determina el error de truncamiento.

Para tres nodos equidistantes tenemos:




$$f'_{(x_0)} = \frac{-f_{(x_2)} + 4f_{(x_1)} - 3f_{(x_0)}}{2h} + \frac{h^2}{3} f'''_{(\mu_0)} \quad en\ adelanto$$

$$f'_{(x_1)} \approx \frac{f_{(x_2)} - f_{(x_0)}}{2h} - \frac{h^2}{6} f'''_{(\mu_1)} \quad centrada$$

$$f'_{(x_2)} = \frac{3f_{(x_2)} - 4f_{(x_1)} + f_{(x_0)}}{2h} + \frac{h^2}{3} f'''_{(\mu_2)} \quad en\ atraso$$

Los segundos términos corresponden al error de truncamiento en cada caso.

## Método de los coeficientes indeterminados

Consiste en desarrollar $f_{(x)}$ por Serie de Taylor alrededor del punto deseado y solicitar con una fórmula de N+1 puntos ciertos requisitos.

Se busca la derivada primera con 3 nodos equidistantes de una función cualquiera. En forma genérica:

$$f'_n \approx af_{n-1} + bf_n + cf_{n+1}$$

Desarrollamos por Taylor en $f_{n-1}$ y $f_{n+1}$:

$$f_{n-1} = f_n - hf'_n + \frac{h^2}{2}f''_n - \frac{h^3}{6}f'''_n + O(h^4)$$

$$f_{n+1} = f_n + hf'_n + \frac{h^2}{2}f''_n + \frac{h^3}{6}f'''_n + O(h^4)$$

Reemplazando las expresiones en la forma genérica antes presentada:

$$f'_n + R_D = a\left[f_n - hf'_n + \frac{h^2}{2}f''_n - \frac{h^3}{6}f'''_n + O(h^4)\right] + bf_n + c\left[f_n + hf'_n + \frac{h^2}{2}f''_n + \frac{h^3}{6}f'''_n + O(h^4)\right]$$

Reordenando:

$$f'_n + R_D = (a+b+c)f_n + (-a+c)hf'_n + (a+c)\frac{h^2}{2}f''_n + (-a+c)\frac{h^3}{6}f'''_n + (a+c)O(h^4)$$

Para que ambos miembros sean iguales, deben cumplirse las siguientes condiciones:

$$\begin{cases} a+b+c=0 \\ (-a+c)h=1 \\ (a+c)\dfrac{h^2}{2}=0 \end{cases} \rightarrow \begin{cases} -\dfrac{1}{2h}+b+\dfrac{1}{2h}=0 \rightarrow b=0 \\ (c+c)h=1 \rightarrow c=\dfrac{1}{2h} \\ a=-c \end{cases} \rightarrow \begin{cases} a=-\dfrac{1}{2h} \\ b=0 \\ c=\dfrac{1}{2h} \end{cases}$$

Finalmente:

$$\begin{cases} f'_n \approx -\dfrac{1}{2h}f_{n-1} + \dfrac{1}{2h}f_{n+1} = \dfrac{f_{n+1}-f_{n-1}}{2h} \\ R_D = (-a+c)\dfrac{h^3}{6}f'''_n = \left(\dfrac{1}{2h}+\dfrac{1}{2h}\right)\dfrac{h^3}{6}f'''_n = \dfrac{h^2}{6}f'''_{(\mu)} \end{cases}$$

## Regla Práctica

$$OP = N - OD$$

Donde $N$ indica la cantidad de nodos o puntos disponibles, $OD$ es el orden de derivación léase qué derivada se desea calcular, y $OP$ indica el orden de precisión de la aproximación obtenida.

Excepción a la regla

Si la aproximación es centrada y la paridad (si es par o impar) de $N$ es distinta de la paridad del $OD$, entonces se gana 1 orden de precisión y la fórmula a utilizar es:

$$OP = N - OD + 1$$

Ejemplos:

$$\begin{cases} Primera\ derivada\ con\ 2\ nodos: N=2, OD=1 \rightarrow OP=2-1=1 \\ Primera\ derivada\ con\ 3\ nodos: N=3, OD=1 \rightarrow OP=3-1=2 \\ Segunda\ derivada\ con\ 3\ nodos: N=3, OD=2 \rightarrow OP=3-2+1=2 \end{cases}$$

# Problemas de Valores Iniciales (Guía 8)

## Métodos de PVI de paso simple

Euler explícito:

$$u^{n+1} = u^n + hf_{(t^n,u^n)} \quad con\, u^0 = \alpha$$

Euler implícito o inverso:

$$u^{n+1} = u^n + hf_{(t^{n+1},u^{n+1})}$$

Implícito Ponderado:

$$u^{n+1} = u^n + h\left[\beta f_{(t^{n+1},u^{n+1})} + (1-\beta) f_{(t^n,u^n)}\right]$$

Crank-Nicholson:

$$u^{n+1} = u^n + \frac{h}{2}\left[f_{(t^{n+1},u^{n+1})} + f_{(t^n,u^n)}\right]$$

Runge Kutta:

- Corrector-Predictor o del Punto Medio (RK-2)

$$\begin{cases} u^{n+\frac{1}{2}} = u^n + \frac{h}{2} f_{(t^n,u^n)} \\ u^{n+1} = u^n + hf_{\left(t^{n+\frac{1}{2}},u^{n+\frac{1}{2}}\right)} \end{cases}$$

- RK-4

$$\begin{cases} u_*^{n+\frac{1}{2}} = u^n + \frac{h}{2} f_{(t^n,u^n)} \\ u_{**}^{n+\frac{1}{2}} = u^n + \frac{h}{2} f_{\left(t^{n+\frac{1}{2}},u_*^{n+\frac{1}{2}}\right)} \\ u_*^{n+1} = u^n + hf_{\left(t^{n+\frac{1}{2}},u_{**}^{n+\frac{1}{2}}\right)} \end{cases}$$

$$u^{n+1} = u^n + \frac{h}{6}\left[f_{(t^n,u^n)} + 2u_*^{n+\frac{1}{2}} + 2u_{**}^{n+\frac{1}{2}} + f_{(t^n,u_*^{n+1})}\right]$$

Otra forma de expresarlo es:

$$\begin{cases} q_1 = hf_{(t^n,u^n)} \\ q_2 = hf_{\left(t^{n+\frac{1}{2}},\ u_n+\frac{1}{2}q_1\right)} \\ q_3 = hf_{\left(t^{n+\frac{1}{2}},\ u_n+\frac{1}{2}q_2\right)} \\ q_4 = hf_{(t^{n+1},\ u_n+q_3)} \end{cases}$$

$$u^{n+1} = u^n + \frac{1}{6}[q_1 + 2q_2 + 2q_3 + q_4]$$

## Métodos de PVI de paso múltiple

Para los métodos de paso múltiple tenemos la siguiente regla general:

$$u^{n+r} + \sum_{i=0}^{r-1} a_i u^{n+i} = h \sum_{i=0}^{r} b_i f_{(t^{n+i}, u^{n+i})}$$

Donde si $b_r = 0$ el método resulta explícito (predictor) y si $b_r \neq 0$ el método resulta implícito (corrector). Cada método queda definido por el conjunto $a_0, a_1, \cdots, a_{r-1}$ y $b_0, b_1, \cdots, b_{r-1}$. **Si $r = 1$ se obtienen los métodos de paso simple**.

Adams-Bashforth (Métodos explícitos de $m$ pasos):

$$AB\ de\ 2\ pasos: \quad u^{n+1} = u^n + \frac{h}{2}[3f^n - f^{n-1}] \quad O(h^2) = \frac{5}{12}h^2 y'''$$

$$AB\ de\ 3\ pasos: \quad u^{n+1} = u^n + \frac{h}{12}[23f^n - 16f^{n-1} + 5f^{n-2}] \quad O(h^3) = \frac{3}{18}h^3 y^{iv}$$

$$AB\ de\ 4\ pasos: \quad u^{n+1} = u^n + \frac{h}{24}[55f^n - 59f^{n-1} + 37f^{n-2} - 9f^{n-3}] \quad O(h^4) = \frac{251}{720}h^4 y^v$$

$$AB\ de\ 5\ pasos:$$

$$u^{n+1} = u^n + \frac{h}{720}[1901f^n - 2774f^{n-1} + 2616f^{n-2} - 1274f^{n-3} + 251f^{n-4}] \quad O(h^5) = \frac{95}{288}h^5 y^{vi}$$

Adams-Moulton (Métodos implícitos de $m$ pasos):

$$AM\ de\ 2\ pasos: \quad u^{n+1} = u^n + \frac{h}{12}[5f^{n+1} + 8f^n - 5f^{n-1}] \quad O(h^3) = \frac{-1}{24}h^3 y^{iv}$$

$$AM\ de\ 3\ pasos: \quad u^{n+1} = u^n + \frac{h}{24}[9f^{n+1} + 19f^n - 5f^{n-1} + f^{n-2}] \quad O(h^4) = \frac{-19}{172}h^4 y^v$$

$$AM-4: \quad u^{n+1} = u^n + \frac{h}{720}[251f^{n+1} + 646f^n - 264f^{n-1} + 106f^{n-2} - 19f^{n-3}] \quad O(h^5) = \frac{-3}{160}h^5 y^{vi}$$

## Métodos de PVI Conservativos

El método RK-2 no es apto para describir sistemas conservativos por lo que se cuenta con métodos que brindan mayor precisión.

Método de Nystrom:

Surge de discretizar la EDO a orden 2 según:

$$u'' \approx \frac{u^{n+1} - 2u^n + u^{n-1}}{h^2}$$

# Problemas de Valores de Contorno (Guía 9)

Problema diferencial tipo

Esta dado por la ecuación:

$$y'' = \frac{d^2y}{dx} = f_{(x,y,y')} \quad para\ a \leq x \leq b$$

Junto con las condiciones de contorno:

$$y_{(x=a)} = \alpha \quad y_{(x=b)} = \beta$$

De manera que la solución de la ecuación empleando las condiciones de contorno es $y_{(x)}$.

## Discretización

1) **Discretización del espacio:**

$$x \to x_i \quad con\ i = 0,1,\cdots,N$$

$$a \leq x_i \leq b \quad con\ x_0 = a\ , x_N = b$$

Siendo $\Delta x = x_{i+1} - x_i$ el paso espacial. Si el paso es constante tenemos que:

$$x_i = a + i\Delta x \quad con\ \Delta x = h = k = \frac{b-a}{N}$$

En general $\Delta x \neq cte$.

2) **Discretización de la ecuación diferencial ordinaria (EDO):**

$$u_i \approx y_{(x_i)}$$

La derivada segunda se calcula empleando la fórmula de dos puntos para la derivada segunda:

$$y'' \to \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} \quad (centrada\ de\ orden\ 2)$$

Para el término fuente (derivada primera) tenemos tres opciones:

$$y' \to \frac{u_i - u_{i-1}}{\Delta x} \quad (en\ atraso\ O(1))$$

$$y' \to \frac{u_{i+1} - u_i}{\Delta x} \quad (en\ adelanto\ O(1))$$

$$y' \to \frac{u_{i+1} - u_{i-1}}{2\Delta x} \quad (centrada\ O(2))$$

Es importante mantener el orden de la derivada elegido durante todo el cálculo.

3) **Discretización de las condiciones de contorno (CC):**

La forma más general de éstas es:

$$c_1 y'_{(a)} + c_2 y_{(a)} = \alpha \quad d_1 y'_{(b)} + d_2 y_{(b)} = \beta$$

De ellas se desprenden los siguientes casos particulares:

- Si $c_1 = d_1 = 0$ → CC tipo Dirichlet.

- Si $c_2 = 0 \; o \; d_2 = 0$ → CC tipo Neuman.
- Si las constantes son distintas de cero (caso genérico) → CC tipo Riemann.

Basta con tomar la EDO del problema, discretizar, despejar de la expresión los términos con $u_i$ de un lado y armar la matriz para la resolución del problema. Luego, se aplica un método de resolución de SEL para resolver.