UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

INE-5369 – Análise Numérica II

Prof. Júlio Felipe Szeremeta

Versão 2.0
Florianópolis, Agosto de 2003

Sumário

Lista de Figuras

Lista de Algoritmos

Introdução

A elaboração desta apostila surgiu da necessidade observada por membros do PET Ciências da Computação da UFSC enquanto cursavam a disciplina de Análise Numérica II com o professor Júlio Felipe Szeremeta, na 3ª fase; pois até então não existia nenhum livro ou apostila que cobrisse toda matéria abordada em sala de aula.

Em conversa com professor tal idéia foi exposta e o mesmo foi de comum acordo; se propondo a elaborar uma apostila que cobrisse todo o conteúdo dado em sala de aula. Ao PET Ciências da Computação caberia a digitalização da mesma utilizando a linguagem de editoração de textos LaTeX(na época, Julho/2002, recém estudada por membros do PET).

Foi então, conciliando o novo aprendizado e a necessidade de um material de apoio à matéria, que a apostila foi desenvolvida; agora possibilitando um melhor acompanhamento e conseqüentemente aprendizado da matéria de Análise Numérica II.

Atualizada e revisada, já utilizada por duas turmas de graduação nos semestres 2002/2 e 2003/1, a mesma encontra-se em sua versão 2.0, terminada em Agosto 2003.

O grupo PET Ciências da Computação sente-se gratificado ao poder ajudar ao corpo discente e docente do Curso de Bacharelado em Ciências da Computação da UFSC.

Capítulo 1

Solução de Equações $f(x) = 0$

1.1 Fundamentos

Definio 1.1 *Uma equação a uma variável é toda expressão do tipo* $f(x) = 0$.

Definio 1.2 *Solução de* $f(x) = 0$ *ou raiz de* $y = f(x)$ *é todo* $\alpha \in \mathcal{C}$ / $f(\alpha) = 0$.

Assim, se $\alpha \in \mathbb{R} \Rightarrow$

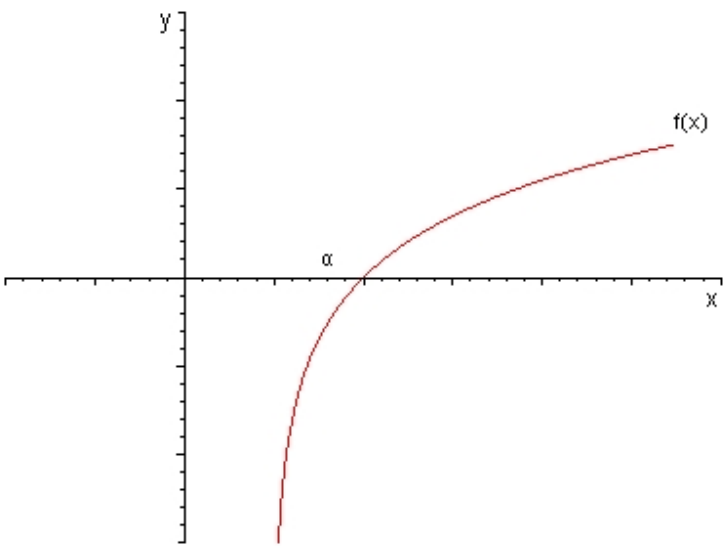


Figura 1.1: Significado geométrico

A solução de equações, primeiro problema da história de Matemática, hoje aparece com freqüência na maioria dos modelos matemáticos oriundos da aplicação do método científico nas mais variadas áreas. Historicamente, pode-se sintetizar a evolução deste problema como segue:

i) $ax + b = 0 \Rightarrow$Euclides (300aC) criou a solução via axiomas:

$ax + b = 0$

$\approx ax + b - b = -b$

$\approx ax = -b$

$\approx \alpha = -\ \frac{b}{a}$

ii) $ax^2 + bx + c = 0$

Pitágoras (500AC) já usava.

Sridhara (séc XI) criou o método divulgado por

Báscara (séc XII), que consiste em:

$ax^2 + bx + c = 0$

$\approx x^2 + \frac{b}{a}x = -\ \frac{c}{a}$

$\approx x^2 + \frac{b}{a}x + \frac{b^2}{4a^2} = -\ \frac{c}{a} + \frac{b^2}{4a^2}$

$\approx (x + \frac{b}{2a})^2 = \frac{b^2 - 4ac}{4a^2}$

$\Rightarrow \alpha = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

iii) $ax^3 + bx^2 + cx + d = 0 \Rightarrow$ (Longa história resolvida dia 10/02/1535)

iv) $e^{2x^3} = 5 \Rightarrow 2x^3 = \ln 5 \Rightarrow \alpha = \sqrt[3]{\frac{\ln 5}{2}}$;

v) $\sin x = 1 \Rightarrow \alpha = 2k\pi + \frac{\pi}{2}$, k $\in \mathbb{N}$

vi) $2\cos x = \ln x \Rightarrow \alpha =?; x - (\ln x)^x = 3 \Rightarrow \alpha =?$

vii) $e^x + 7 = 0 \Rightarrow \nexists \alpha \in \mathbb{R}$

1.1.1 Conclusões

1. Numa $f(x) = 0$ pode: $\exists! \alpha, \exists k\alpha, \exists \infty \alpha$ ou $\nexists \alpha$.
2. Solver $f(x) = 0$ por abstração pode ser: fácil, difícil ou impossível.
3. Necessitamos de outras metodologias, além da abstrativa. Aqui será abordada a *metodologia iterativa* (criada por Newton em 1670).

Definio 1.3 *Um método iterativo é uma técnica construtiva de solução de modelos matemáticos e que obedece a duas etapas em sua aplicação:*

a) isolamento da solução desejada (solução inicial).

b) refinamento da solução inicial até a precisão requerida.

A etapa do isolamento da solução é a mais difícil de aplicar por ser subjetiva (não existem métodos globais). Para uma equação $f(x) = 0$, o isolamento de suas raízes pode ser efetuado via:

a) esboço do gráfico da $y = f(x)$ (só para $\alpha \in \mathbb{R}$).

b) conhecimento teórico do problema modelado.

c) uso de propriedades algébricas de $y = f(x)$.

Ex:Prop.1: "Em $f(x) = 0$ se $y = f(x)$ for contínua em [a,b] e $f(a) * f(b) < 0 \Rightarrow \exists \alpha \in$ [a,b]".

d) agrupamento em classes com características específicas.

Ex: polinomiais, trigonométricas, etc.

e) tomada aleatória (= chute).

1.2 Refinamento de Raízes de $f(x) = 0$

As técnicas de refinamento de raízes isoladas de $f(x) = 0$, podem ser agrupadas em três grandes famílias:

1ª) *Quebras*:

Obter [a,b] | $\alpha \in$ [a,b] .

Particionar [a,b] em subintervalos $[a_i, b_i]$ | $\alpha \in [a_i, b_i]$ até que $\lim_{i \to \infty} |a_i - b_i| = 0$

2ª) *Tipo Newton*

Tomar $\alpha \simeq x_0$.

Construir uma seqüência de valores $\{X_i\}_{i=0}^{\infty}$ | $\lim_{i \to \infty} X_i = \alpha$

3ª) *Híbridos*

Mesclar as duas famílias anteriores.

Note-se que em todas estas famílias, em princípio, o resultado final é afetado por erro de truncamento.

1.2.1 Métodos de Quebras

Numa $f(x) = 0$, com $y = f(x)$ contínua em [a,b] e | $f(a) * f(b) < 0$, a solução $\alpha \in$ [a,b] pode ser obtida via:

Método da Bisseção

i) Obter XM $= \dfrac{a+b}{2}$

Se $f(XM) = 0 \Rightarrow \alpha = XM$

Senão

Se $f(a) * f(XM) < 0 \Rightarrow a = XM$

Senão b = XM

ii) Retornar ao passo i).

Daí, $\lim\limits_{i\to\infty} |b_i - a_i| = \lim\limits_{i\to\infty} \dfrac{|b-a|}{2^i} = 0 \Rightarrow XM = \alpha$

Ex.1: Aproxime por bisseção a raiz de $e^x \sin x - 1 = 0$ situada em [0;1]. Efetue cinco quebras.

Solução:

Temos $f(x) = e^x \sin x - 1$ e $f(0) = -1$ e $f(1) = 1,287 \Rightarrow f(0) * f(1) < 0$

Aplicando Bisseção resulta:

a	XM	b	$f(a)$	$f(XM)$	$f(b)$
0	0,5	1	−1	−0,209	1,287
0,5	0,75	1	−0,209	0,443	1,287
0,5	0,625	0,75	−0,209	0,093	0,443
0,5	0,5625	0,625	−0,209	−0,064	0,093
0,5625	0,59375	0,625	−0,064	0,013	0,093
$\alpha \simeq 0,59375$					

Análise

1ª) Quando sustar as quebras para assegurar a precisão ε da α ?

a)Obter o número de quebras k via:

$$Como \qquad \frac{|b-a|}{2^k} \simeq \varepsilon \Rightarrow k = \frac{(\ln \frac{|b-a|}{\varepsilon})}{ln2}$$

Ex.2: Quantas quebras por bisseção são necessárias para assegurar precisão $\varepsilon = 10^{-15}$ na equação do Ex.1.

Solução:

$a = 0;$

$b = 1;$

$\Rightarrow |b-a| = 1$

$\Rightarrow k = \dfrac{(\ln \frac{1}{10^{-15}})}{\ln 2} = 49,8282$

$\Rightarrow k = 50$ quebras.

b)Repetir até que $|f(X_m)| < \varepsilon \Rightarrow$ pode falhar.

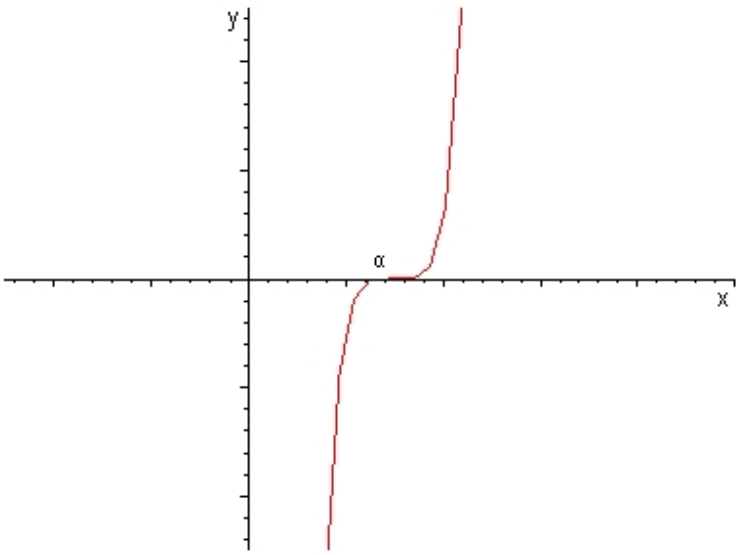


Figura 1.2: Quebras pode falhar

c)Repetir até que $|b_i - a_i| < \varepsilon$ OK

No Ex1 após 5 quebras $\Rightarrow |b_i - a_i| = |0,59375 - 0,56275| = 0,031$

2ª) Algoritmo(pseudocódigo)

Algoritmo 1 Bisseção

Entrada: $a, b, \varepsilon, f(x)$
Sada: Raiz de $f(x)$
  $c = f(a) : d = f(b) : erro = |b - a|$
  enquanto $erro > \varepsilon$ **faa**
    $xm = (a + b)/2 : fm = f(xm)$
    se $fm = 0$ **ento**
      $\alpha = xm : erro = 0$
      FIM
    fim se
    $prod = fm * c$
    se $prod < 0$ **ento**
      $b = xm : d = fm$
    seno
      $a = xm : c = fm$
    fim se
    $erro = |b - a|$
  fim enquanto

3ª) Bisseção obtém apenas $\alpha \in \mathbb{R}$ de $f(x) = 0$, não obtém todas as $\alpha \in \mathbb{R}$, e normalmente é lento (alto custo). Como acelerá-lo?

Método da Falsa Posição (FP)

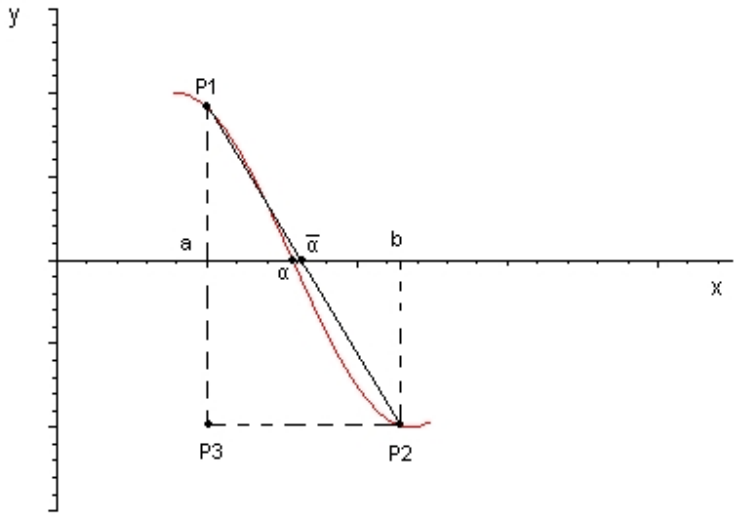


Figura 1.3: Método da Falsa Posição

Em $f(x) = 0$, com $f(x)$ contínua em $[a, b]$ e $f(a) * f(b) < 0$, proceder:

i) Obter $P_1 = (a, f(a))$ e $P_2 = (b, f(b))$.

Pela semelhança dos triângulos retângulos $P_1\hat{a}\alpha$ e $P_1\hat{P}_3P_2$, tem-se:

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

$$\Rightarrow \overline{x} = a - \frac{f(a) * (b - a)}{f(b) - f(a)}$$

Se $f(\overline{x}) = 0$

$\overline{x} = \alpha$

Senão (idem bisseção alterando-se apenas a forma das quebras)

ii) Retorna ao i)

Ex.3: $e^x * \sin x = 1$ em $[0; 1]$ por Falsa Posição com 3 quebras.

Solução:

a	$\overline{x}$	b	$f(a)$	$f(\overline{x})$	$f(b)$
0	0,437	1	−1	−0,344	1,287
0,437	0,556	1	−0,344	−0,079	1,287
0,556	0,582	1	−0,079	−0,016	1,287
$\alpha \simeq 0,582$					

Análise

1ª) Algoritmo FP (fazer)

2ª) Pelo exemplo anterior, nota-se que FP pode ser mais veloz que Bisseção, devido à forma das quebras. Contudo, ele pode ser até mais lento que o Bisseção quando ocorrer pontos de estagnação.

Ex.4: Aproxime uma raiz de $x^{10} - 2 = 0$ com $\varepsilon = 10^{-4}$ em $[0; 1, 3]$.

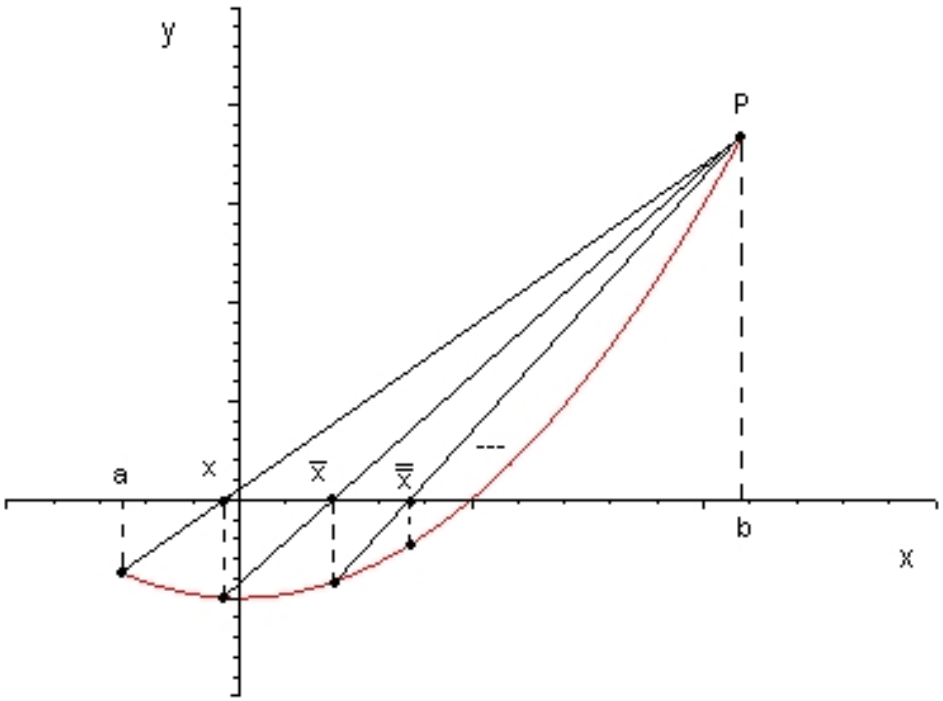


Figura 1.4: Ponto de estagnação P

Como libertar o FP do seu ponto de estagnação?

Usando o Falsa Posição Modificado (FPM), que consiste em se detectar permanências de sinal entre $f(\overline{x})$ e $f(a)$ ou $f(\overline{x})$ e $f(b)$ e tomar $\frac{f(a)}{2}$ ou $\frac{f(b)}{2}$

3ª) Algoritmo FPM

Algoritmo 2 Falsa Posição Modificado

Entrada: $a, b, \varepsilon, f(x)$
Sada: Raiz de $f(x)$
$fa = f(a)$
$fb = f(b)$
$aux = fa$
$erro = |b - a|$
enquanto $erro > \varepsilon$ **faa**
$xx = a - \dfrac{f(a) * (b - a)}{fb - fa}$
$fx = f(xx)$
se $fm == 0$ **ento**
$erro = 0$
fim se
se $fx * fa < 0$ **ento**
$b = xx$
$fb = fx$
se $fx * aux > 0$ **ento**
$fa = \dfrac{fa}{2}$
fim se
seno
$a = xx$
$fa = fx$
se $fx * aux > 0$ **ento**
$fb = \dfrac{fb}{2}$
fim se
fim se
$aux = fx$
$erro = |b - a|$
fim enquanto
Escreva raiz $= xx$
FIM

1.2.2 Métodos tipo Newton para $f(x) = 0$

Newton Geral

Este é o método mais abrangente e de melhor eficiência para as $f(x) = 0$ em geral.

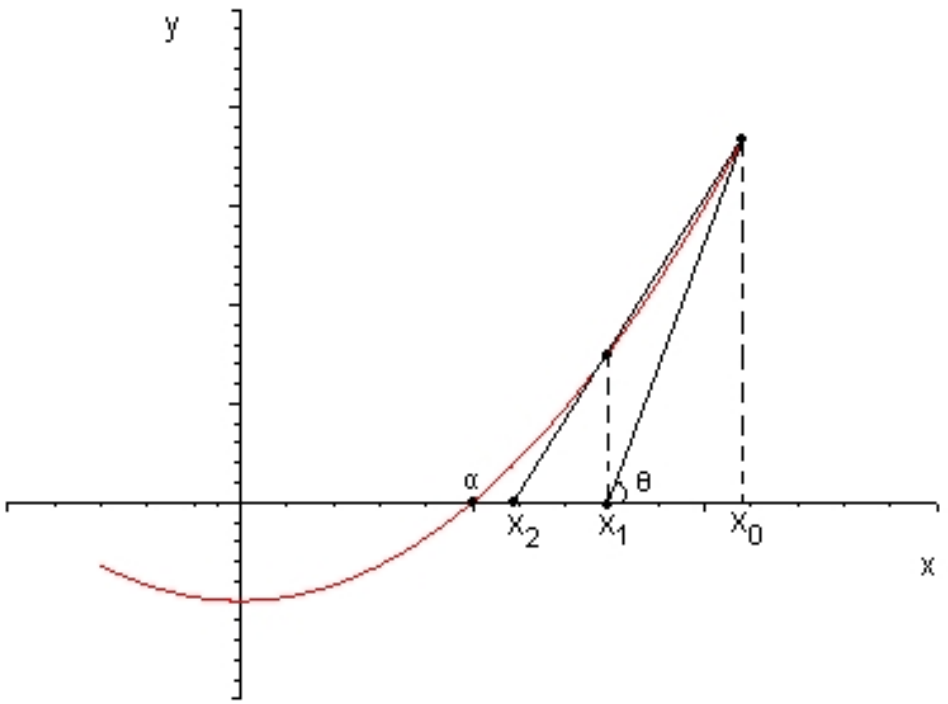


Figura 1.5: Método Newton Geral

Em $f(x) = 0$ se $y = f(x)$ e $y' = f'(x)$ forem contínuas, então uma raiz α pode ser obtida como segue:

1º) Tomar o valor inicial $x_0 (\simeq \alpha)$;

2º) Obter a tangente à $y = f(x)$ e que passa por (x_0, y_0) prolongando-a até o eixo $\overline{OX}$

$$\tan\theta = \frac{f(x_0)}{x_0 - x_1} \Rightarrow f'(x_0) = \frac{f(x_0)}{x_0 - x_1} \Rightarrow x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

3º) Repetir o 2º em $(x_1, f(x_1)) \Rightarrow x_2 = x_1 - \dfrac{f(x_1)}{f'(x_1)}$

Assim sucessivamente obtendo-se a sequência $\{x_k\}_{k=0}^{\infty}$ onde

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{1.1}$$

Daí se

$$\exists \lim_{k\to\infty} x_{k+1} = \xi \Rightarrow \lim_{k\to\infty} x_{k+1} = \lim_{k\to\infty} x_k - \frac{\lim_{k\to\infty} f(x_k)}{\lim_{k\to\infty} f'(x_k)}$$

e como $f(x)$ e $f'(x)$ são contínuas

$$\lim_{k\to\infty} x_{k+1} = \lim_{k\to\infty} x_k - \frac{f(\lim_{k\to\infty} x_k)}{f'(\lim_{k\to\infty} x_k)} \Rightarrow \xi = \xi - \frac{f(\xi)}{f'(\xi)}$$

$$\Rightarrow f(\xi) = 0 \Rightarrow \xi = \alpha \qquad se \qquad f'(\xi) \neq 0$$

Ex.1: Aproxime a raiz de $x + e^x - 2 = 0$ por Newton com $x_0 = 0$ e 4 iterações.

Solução:
Temos $x_0 = 0, f(x) = x + e^x - 2$ e $f'(x) = 1 + e^x$ Aplicando a equação 1.1 resulta:

$$x_0 = 0$$

$$x_1 = 0 - \frac{(-1)}{2} = 0,5$$

$$x_2 = 0,5 - \frac{0,1487}{2,6487} = 0,4438$$

$$x_3 = 0,4438 - \frac{0,0024}{2,5586} = 0,4429$$

$$x_4 = 0,4429 - \frac{0,0001}{0,2578} = 0,4429 \simeq \alpha.$$

Ex.2: Aproxime a raiz de $2^{-2x} - \sqrt{x} = 0$ por Newton com $x_0 = 1$ e $\varepsilon = 10^{-5}$

Solução:
$f(x) = 2^{-2x} - \sqrt{x} \Rightarrow f'(x) = -(2^{-2x} * 2 * \ln 2 + \frac{1}{2\sqrt{x}})$

$$x_0 = 1$$

$$x_1 = 1 - \frac{-0,75}{-0,84657359} = 0,1140758 \Rightarrow x_1 - x_0 = 0,8859$$

$$x_2 = 0,3077682 \Rightarrow x_2 - x_1 = 0,1936$$

$$x_3 = 0,3619839 \Rightarrow x_3 = x_2 = 0,0542$$

$$\vdots$$

$$x_6 = 0,36425$$

$$x_7 = 0,36425 \Rightarrow x_7 - x_6 = 0,00000 < \varepsilon \Rightarrow \alpha \simeq 0,36425$$

Ex.3: Em $\sqrt[3]{x} = 0$ aplicando Newton com $x_0 = 10$, resulta:

$$x_0 = 10$$

$$x_1 \simeq -20$$

$$x_2 \simeq 40$$

$$x_3 \simeq -80$$

$$x_4 \simeq 160 \ldots \Rightarrow \text{diverge.}$$

Considerações

1ª) Algoritmo Newton

Algoritmo 3 Newton

Entrada: $x_0, nit, \epsilon, f(x), df(x)$
Sada: Raiz de $f(x)$
  para $i = 1 \ldots \ nit$ **faa**
    $x_1 = x_0 - \dfrac{f(x_0)}{df(x_0)}$
    $erro = |x_1 - x_0|$
    se $erro < \epsilon$ **ento**
      $raiz = x_1$
      FIM
    fim se
    $x_0 = x_1$
  fim para
  Escreva: "Precisão não obtida"

2ª) Newton pode não convergir (patologias)

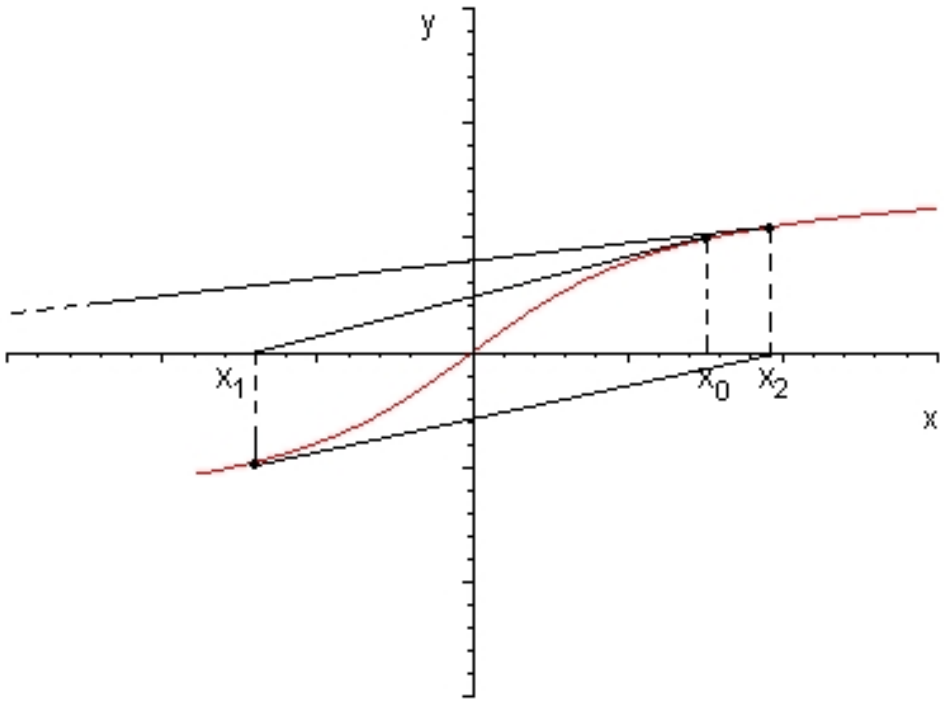


Figura 1.6: x_0 inadequado

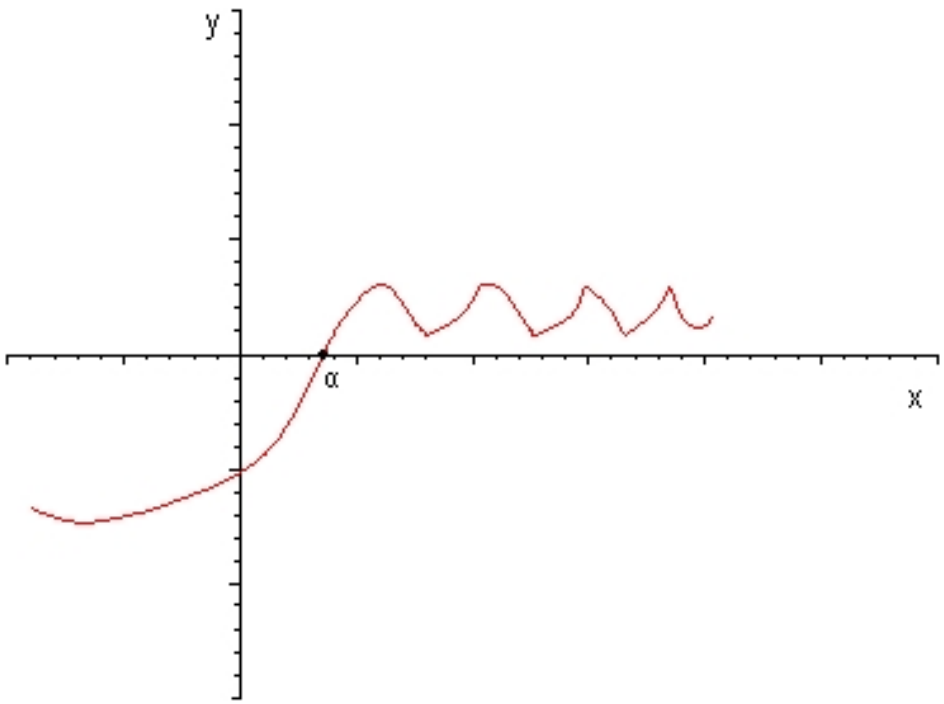


Figura 1.7: Ocorrer overflow

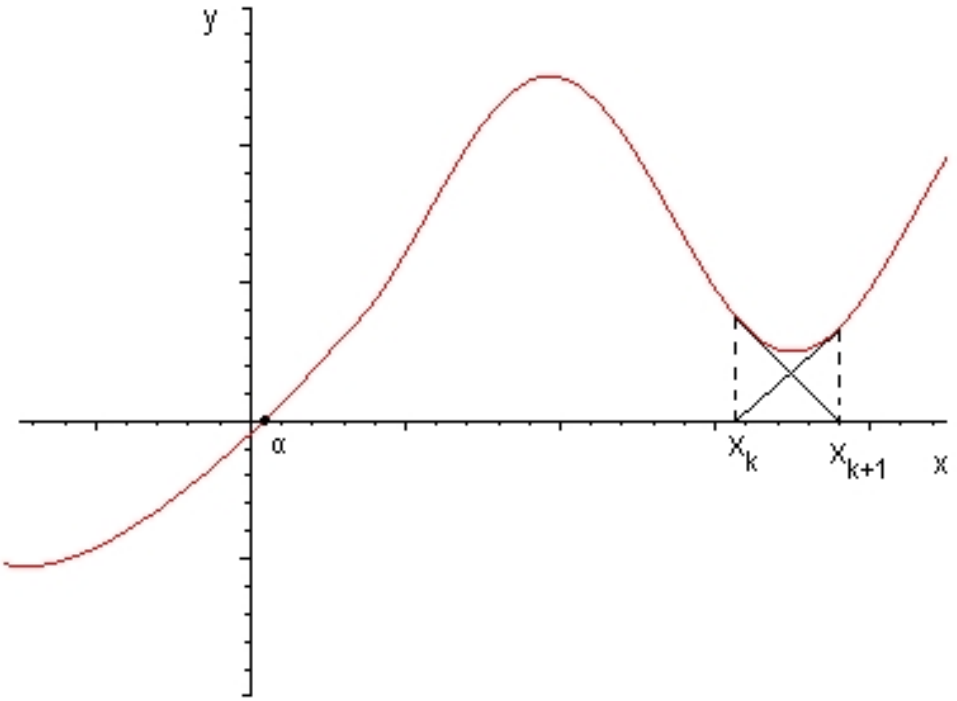


Figura 1.8: Ocorre laços repetitivos cíclicos

Soluções

i) Para resolver o problema da Figura 1.6, usar métodos mais consistentes para o x_0. Não abordaremos aqui.

ii) Para resolver o problema da Figura 1.7, usar simplificações do Newton, denominadas de *quasi*-Newton

iii) Para resolver o problema da Figura1.8, também usar simplificações do Newton, denominadas de *quasi*-Newton

Quasi-Newton1

Em 1.1 fixando-se a $f'(x)$ em $x = x_0 \Rightarrow$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_0)} \tag{1.2}$$

$$\Rightarrow \begin{cases} f'(x_0) \neq 0 \\ \nexists \text{ laços cíclicos;} \\ \text{É mais rápido para executar uma iteração} \\ \text{Perde-se na abrangência no que se refere a convergência} \end{cases}$$

Ex.4: $x + e^x - 2 = 0$ com $x_0 = 0$ e 4 iterações

$f(x) = x + e^x - 2, f'(x) = 1 + e^x \Rightarrow f'(0) = 2$

$x_0 = 0$

$x_1 = 0,5$

$x_2 = 0,4256$

$x_3 = 0,4475$

$x_4 = 0,4415\ldots$

$f(x_4) = 3 * 10^{-3}$

Obs.: pelo Newton Geral temos:

$f(x) = x + e^x - 2, f'(x) = 1 + e^x \Rightarrow f'(0) = 2$

$x_0 = 0$

$x_1 = 0,5$

$x_2 = 0,4438$

$x_3 = 0,4429$

$x_4 = 0,4429\ldots$

$f(x_4) = 3 * 10^{-4}$

3ª) Pode ser difícil ou "impossível"para o usuário a determinação da $y' = f'(x)$ para alimentar o algoritmo do NEWTON GERAL.

Solução

Simular a $y' = f'(x)$ via:

Por definição,

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \Rightarrow f'(x) \simeq \frac{f(x+h) - f(x)}{h}$$

se h for "pequeno".

Daí, tomando dois valores iniciais x_0 e x_1 e considerando $h = x_1 - x_0$:

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

e aplicando na equação 1.1 resulta:

$$x_2 = x_1 - \frac{f(x_1)}{\frac{f(x_1)-f(x_0)}{x_1-x_0}} \Rightarrow x_2 = x_1 - \frac{f(x_1) * (x_1 - x_0)}{f(x_1) - f(x_0)}$$

$$\Rightarrow x_{k+1} = x_k - \frac{f(x_k) * (x_k - x_{k-1})}{f(x_k) - f(x_{k-1})} \tag{1.3}$$

que é um *quasi*-NEWTON, denominado de MÉTODO DA SECANTE.

Geometricamente,

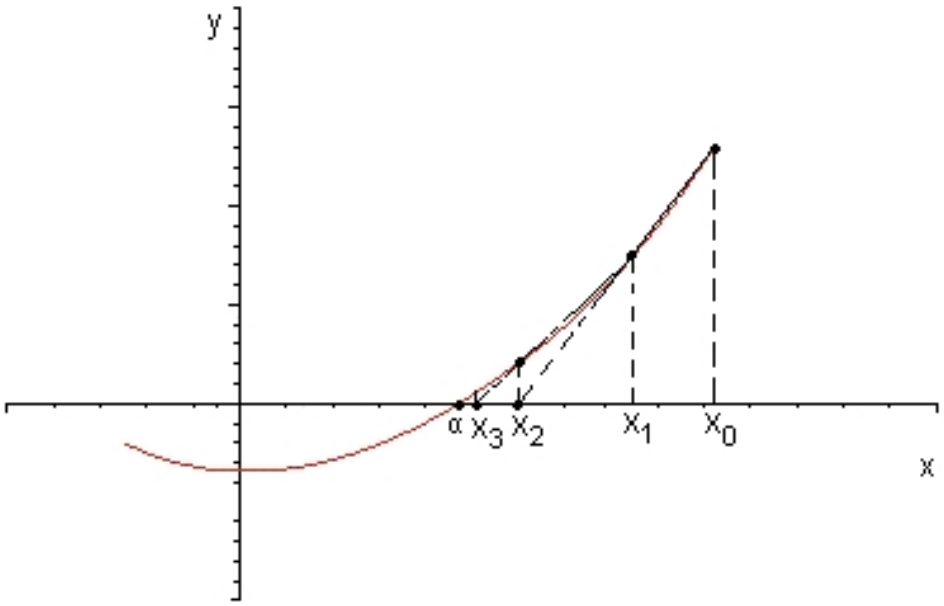


Figura 1.9: Método da Secante

VARIANTE: Método de Steffensen:

$$x_{k+1} = x_k - \frac{f(x_k)}{g(x_k)} \tag{1.4}$$

onde $g(x_k) = \dfrac{f(x_k + f(x_k)) - f(x_k)}{f(x_k)}$

Ex.5: Resolver $x + e^x - 2 = 0$ com $x_0 = 0$ e $x_1 = 1$ e aplicando o método da SECANTE com 4 iterações:

Solução:

$x_0 = 0$

$x_1 = 1$

$x_2 = 1 - \dfrac{1,7182}{2,718} = 0,3679$

$x_3 = 0,43$

$x_4 = 0,4431$

4ª) Para se obter mais de uma raiz de $f(x) = 0$, a reaplicação da equação 1.1 pode convergir para uma raiz α_1 já obtida.

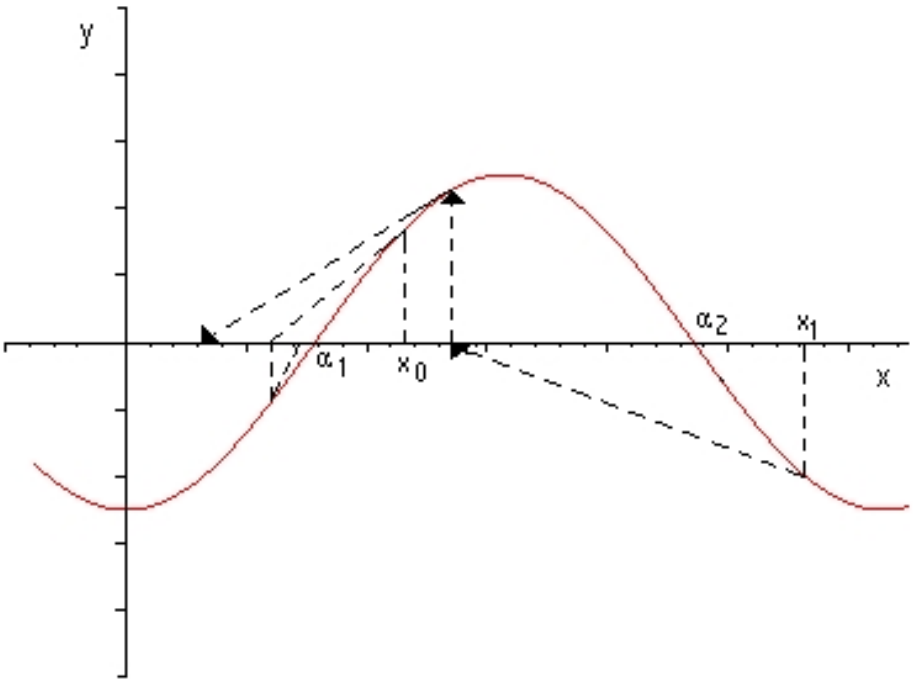


Figura 1.10: Convergência para raiz já obtida

Solução

Se α_1 é raiz de $f(x) = 0$, efetuando $\dfrac{f(x)}{x - \alpha_1}$, temos

$$\Rightarrow f(x) = (x - \alpha_1) * g(x) + R$$
$$\Rightarrow f(\alpha_1) = 0$$
$$\Rightarrow 0 * g(x) + R$$
$$\Rightarrow R = 0$$

Daí

$$f(x) = (x - \alpha_1) * g(x) \Rightarrow g(x) = \frac{f(x)}{x - \alpha_1}$$

que possui as mesmas raízes de f(x) = 0, exceto a α_1. Logo, aplicando 1.1 em $g(x) = 0$, temos:

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)} \Rightarrow x_{k+1} = x_k - \frac{\frac{f(x_k)}{x_k - \alpha_1}}{(\frac{f(x_k)}{x_k - \alpha_1})'} \Rightarrow$$

$$x_k + 1 = x_k - \frac{(x_k - \alpha_1) f(x_k)}{[(x_k - \alpha_1) f'(x_k) - f(x_k)]} \tag{1.5}$$

que converge para uma raiz $\alpha \neq \alpha_1$ e é outro *quasi*-NEWTON.

5ª) Sob condições normais, o NEWTON GERAL possui convergência quadrática, isto é, $E_{k+1} \simeq E_k^2$.

Ex.6: Em $3^{-x} - x = 0$ com $x_0 = 0,5$ a equação 1.1 fornece:

$$x_0 = 0,5$$
$$x_1 = 0,547327757 \Rightarrow x_1 - x_0 = 0,047$$
$$x_2 = 0,54780857 \Rightarrow x_2 - x_1 = 0,00048$$
$$x_3 = 0,54780862 \Rightarrow x_3 - x_2 = 0,00000004$$

Esta velocidade de convergência torna-se linear no caso de α ser uma raiz múltipla, isto é, $f(\alpha) =, f'(\alpha) = 0$, ...

Ex.7: Verifique isso em $e^x - x - 1 = 0$ com $x_0 = 1$

$$x_0 = 1$$
$$x_1 = 0,58198$$
$$x_2 = 0,31906$$
$$x_3 = 0,168$$
$$\vdots$$
$$x_{10} = 1,3 * 10^{-3}$$

$x_{11} = 6,9 * 10^{-4}$

$x_{12} = 3,4 * 10^{-4}$

$x_{13} = 1,7 * 10^{-4}$

Por quê? Se α é raiz

$$\Rightarrow f(x) = (x - \alpha) * g(x) \Rightarrow f'(x) = (x - \alpha) * g'(x) + g(x).$$

Daí se α também for raiz de $g(x) \Rightarrow f'(\alpha) = 0$ (DUPLA).

6ª) Como "acelerar"a convergência quadrática do NEWTON?

a)Usar o método de KINKAID

$$x_{k+1} = x_k - \frac{f(x_k)}{\sqrt{[f'(x_k)]^2 - f''(x_k) * f(x_k)}} \tag{1.6}$$

que possui convergência do tipo biquadrática $\Rightarrow E_{k+1} \simeq E_k^4$

Ex.8: Teste $x + e^x - 2 = 0$ com $x_0 = 0$, por 1.6, com 4 iterações

Solução

$x_0 = 0$

$x_1 = 0,447213595 \Rightarrow x_1 - x_0 \simeq 0,4472...$

$x_2 = 0,442854396 \Rightarrow x_2 - x_1 \simeq 10^{-3}$

$x_3 = 0,442854401 \Rightarrow x_3 - x_2 \simeq 10^{-8}$

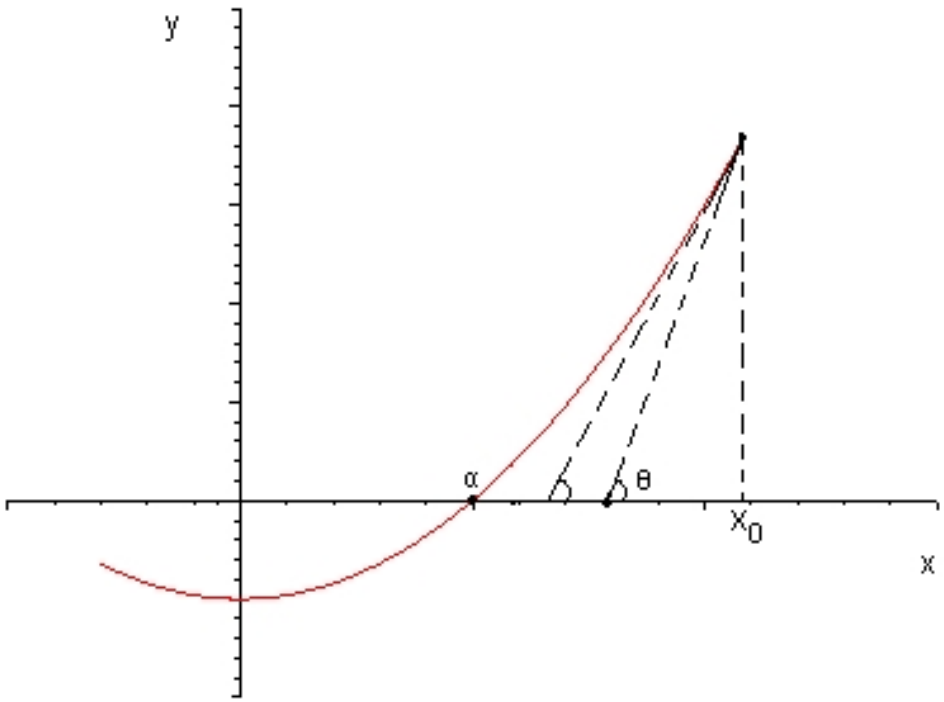
b)Usar coeficientes de relaxação

$$x_{k+1} = x_k - w * \frac{f(x_k)}{f'(x_k)} \tag{1.7}$$

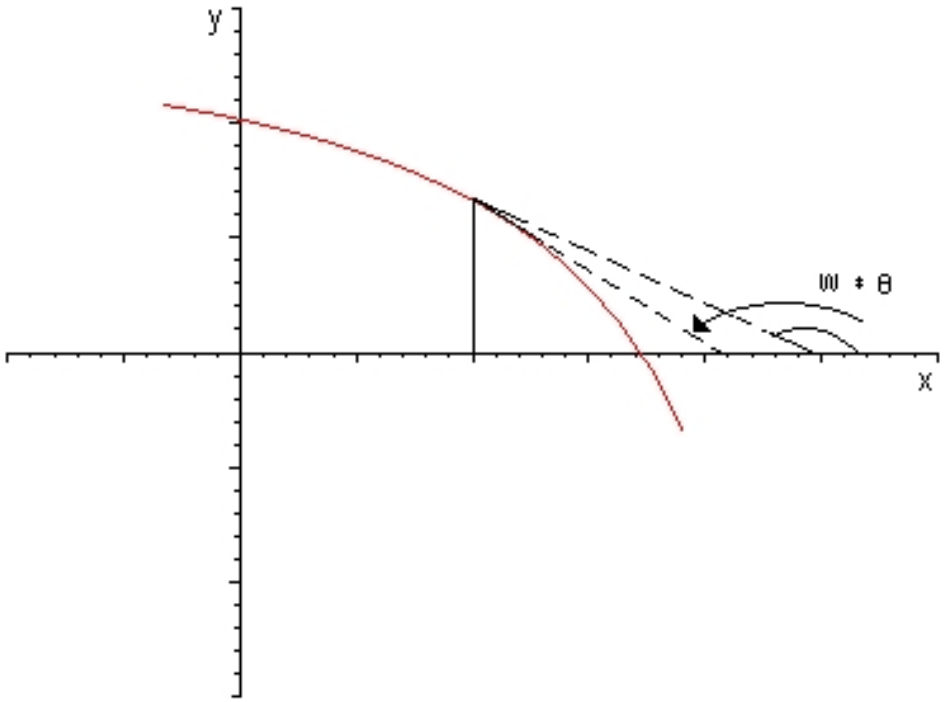
com $\quad 0 < w < 2 \quad e \quad w \simeq 1$

Como tomar o w? Heurísticamente.

1ª situação



2ª situação



1.2.3 Métodos Híbridos para $f(x) = 0$

Nestes métodos é feita uma mescla de QUEBRAS com os de NEWTON de modo a obter-se o maior proveito possível das vantagens de cada um.

Ex.1: Poli-Algoritmo Newton/Bisseção

- Obter $[a, b]$ que contenha todas as α desejadas
- Particionar $[a, b]$ em k subintervalos $[a_i, b_i]$ de comprimento $h = \left(\frac{b-a}{n}\right)$
- Em cada $[a_i, b_i] \mid f(a_i) * f(b_i) < 0$ aplicar Bisseção e posteriormente Newton com $x_0 = \frac{a_i + b_i}{2}$

Ex.2 Poli-Algoritmo de Dekker/Brent (Teses de doutorado da década 1970)

Combinação judiciosa do Falsa Posição, Secante e Bisseção (ver na literatura)

1.3 Solução de $f(x) = 0$ por agrupamento em famílias

Neste tipo de métodos é feita a abordagem restrita a uma família específica de funções geradoras de $f(x) = 0$. Por exemplo se $f(x) = P_n(x)$; onde $P_n(x) = a_1 * x^n + \ldots + a_n + 1, a_1 \neq 0 \Rightarrow$ grau n e $a_i \in \mathbb{R}$, pode-se ampliar a eficiência dos métodos iterativos. Como resolver $P_n(x) = 0$?

1.3.1 História

- $n = 1 \Rightarrow a_1 * x + a_2 = 0 \Rightarrow \alpha = \frac{-a_2}{a_1}$ (Euclides 300A.C.)
- $n = 2 \Rightarrow a_1 * x^2 + a_2 * x + a_3 = 0 \Rightarrow \alpha = \frac{-a_2 \pm \sqrt{a_2^2 - 4 * a_1 * a_3}}{2 * a_1}$ (Sridhara 1100D.C.)
- $n = 3 \Rightarrow a_1 * x^3 + \ldots + a_4 = 0$ (Nicolo Fontana resolveu em 10/02/1535) $\Rightarrow$

$$a_1 * x^3 + a_2 * x^2 + a_3 * x + a_4 = 0, x = y - \frac{a_2}{3 * a_1} \Rightarrow$$

$$y^3 + p * y + q = 0 \qquad ? \tag{1.8}$$

Se:

$$y = a + b \Rightarrow y^3 = a^3 + b^3 + 3 * a * b * (a + b)$$
$$y^3 = a^3 + b^3 + 3 * a * b * (y)$$

Então $y^3 - 3 * a * b * y - (a^3 + b^3) = 0$

Comparando com 1.8

$$\begin{cases} p = -3 * a * b \\ q = -(a^3 + b^3) \end{cases} \Rightarrow \begin{cases} a^3 * b^3 = -\dfrac{p^3}{27} \\ a^3 + b^3 = -q \end{cases}$$

Agora basta obter os valores de a e b, uma vez que conhecemos o seu produto e sua soma. Este é um problema clássico cjua solução pode ser obtida facilmente resolvendo-se uma equação de grau $n = 2$ (Shidhara)

$$\Rightarrow \quad \alpha = \sqrt[3]{-\frac{q}{2} + \sqrt{\frac{p^3}{27} - \frac{q^2}{4}}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\frac{p^3}{27} - \frac{q^2}{4}}}$$

- $n = 4 \Rightarrow a_1 * x^4 + \ldots + a_5 = 0$ Ferrari(1560) desenvolveu método de redução para $n = 3$
- $n \geqslant 5 \Rightarrow \nexists$ método geral por radiciação

$$\begin{cases} \text{Euler, Lagrange, Ruffini tentaram sem sucesso} \\ \text{Abel (1824) provou ser impossível} \\ \text{Galois (1832) classificou as possíveis e impossíveis} \end{cases}$$

- **SAÍDA:** $\forall P_n(x) = 0$**,** $n \geqslant 5$**, use métodos iterativos**

1.3.2 Como expressar o $P_n(x)$ para se obter $P_n(\beta) =$?

1. Usando a geral $P_n(x) = a_1 * x^n + a_2 * x^{n-1} + \ldots + a_{n-1}$, se # = número de operações

$$\Rightarrow \begin{cases} \# + = n \\ \# * = 2n - 1 \end{cases}$$

2. Expressando $P_n(x)$ na forma linear ou Horner

$$P_n(x) = (\ldots(((a_1 * x + a_2) * x + a_3) * x + a_4) \ldots + a_n) * x + a_{n+1}$$

$$\Rightarrow \begin{cases} \# + = n \\ \# * = n \end{cases}$$

Algoritmo 4 Horner

Entrada: $n, \beta, (a_i, i = 1, n+1)$
Sada: Valor do polinômio em β
  $VP = a_1$
  para $i = 1$ até n **faa**
    $VP = VP * \beta + a_{i+1}$
  fim para

3. Outra forma de se obter $P_n(x)$ equivalente à de Horner, consiste em se usar a divisão de polinômios.

   Lembrete: De $P_n(x) \div (x - \beta)$

$$\Rightarrow P_n(x) = (x - \beta) * Q_{n-1}(x) + R \tag{1.9}$$

   $\Rightarrow P_n(\beta) = R$

   $\Rightarrow$ *Prop1: "O valor numérico de* $P_n(x)$ *em* $x = \beta$ *é o resto de* $P_n(x) \div (x - \beta)$

   Como se obter R:

   Se $Q_{n-1}(x) = b_1 * x^{n-1} + b_2 * xn - 2 + \ldots + b^n$

$$\Rightarrow \begin{cases} b_1 = a_1 \\ b_2 = a_2 + \beta * b_1 \\ b_3 = a_3 + \beta * b_2 \\ \vdots \\ b_n = a_n + \beta * b_{n-1} \\ b_{n+1} = R = a_{n+1} + \beta * b_n \end{cases}$$

   Tomando $Q_{n-1}(x)$ e efetuando $Q_{n-1}(x) \div (x - \beta)$

$$\Rightarrow Q_{n-1} = (x - \beta) * Q_{n-2}(x) + \overline{R} \tag{1.10}$$

   Substituindo 1.10 em 1.9 $\Rightarrow$

$$P_n(x) = (x - \beta) * [(x - \beta) * Q_{n-2}(x) + \overline{R}] + R$$
$$P_n(x) = (x - \beta)^2 * Q_{n-2}(x) + (x - \beta) * \bar{(}R) + R$$
$$P'_n(x) = (x - \beta)^2 * Q'_{n-2}(x) + 2(x - \beta) * Qn - 2 + \overline{R}$$
$$P'_n(\beta) = \overline{R}$$

   Prop2: "O valor numérico de $P'_n(x)$ *em* $x = \beta$ *é o resto da segunda divisão sucessiva de* $P_n(x)$ *por* $x - \beta$*"*

   Prop2 Geral: "Em geral, valor numérico de $P^k(\beta) = k!R_k$*, onde* R_k *= resto da* $k + 1$*-ésima divisão sucessiva de* $P_n(x)$ *por* $x - \beta$*."*

Ex.1: Para $P_4(x) = 2 * x^4 + 3 * x - 2$, obtenha $P_4(2)$, $P_4'(2)$, $P_4''(2)$ e $P_4'''(2)$ sem derivar.

Solução: $(\beta = 2)$

$P_4(x) \div (x - 2)$

$$\begin{cases} b_1 = a_1 = 2 \\ b_2 = 0 + 2 * 2 = 4 \\ b_3 = 0 + 2 * 4 = 8 \\ b_4 = 3 + 2 * 8 = 19 \\ R = P_4(2) = -2 + 2 * 19 = 36 \end{cases} \Rightarrow \begin{cases} c_1 = b_1 = 2 \\ c_2 = b_2 + 2 * c_1 = 4 + 4 = 8 \\ c_3 = b_3 + 2 * c_2 = 8 + 16 = 24 \\ R_1 = 19 + 2 * 24 = 67 * 1! = P_4'(2) \end{cases}$$

$$\begin{cases} d_1 = c_1 = 2 \\ d_2 = c_2 + 2 * c_1 = 8 + 2 * 2 = 12 \\ R_2 = 24 + 2 * 12 = 48 \\ P_4''(2) = 48 * 2! = 96 \end{cases} \Rightarrow \begin{cases} e_1 = 2 \\ R_3 = e_2 = d_2 + 2 * 2 = 16 \\ P_4'''(2) = R_3 * 3! = 16 * 6 = 96 \end{cases}$$

Algoritmo 5 Derivadas

Entrada: $n, \beta, (a_i, i = 1, n + 1)$
Sada: Valores de $p^k(\beta)$, $k = 0, 1, \ldots, n$

```
m = n
aux = 1
para k = 0 até n faa
    para i = 1 até m faa
        a_{i+1} = a_{i+1} + β * a_i
    fim para
    v(k) = aux * a_{m+1}
    m = m + 1
    aux = aux * (k + 1)
fim para
```

1.3.3 Existência de raízes

Prop3:(Teorema Fundamental)"Toda $P_n(x) = 0$ possui pelo menos uma raiz $\alpha \in \mathbb{C}$."

Prop4:"Se α é raiz de $P_n(x) = 0 \Rightarrow P_n(x)$ é divisível por $(x - \alpha)$."

Prop5:(Corolário)"$P_n(x) = 0$ possui n raízes, distintas ou repetidas."

Prop6:"Se $\alpha = a + b_i$ é raiz de $P_n(x) = 0 \Rightarrow \overline{\alpha} = (a - b_i)$ também é raiz."
Corolário:"Se n for ímpar $\Rightarrow \exists \alpha \in \mathbb{R}$"

Prop7:"Em $P_n(x) = 0$ se n for par e $a_1 * a_{n+1} < 0 \Rightarrow \exists \alpha \in \mathbb{R}^+$ e $\exists \alpha \in \mathbb{R}^-$."

Prop8:"Em $P_n(x) = 0$ se $\exists a_i = a_{i+1} = 0$, $i = 2, \ldots, n \Rightarrow \exists \alpha \in \mathbb{C}$."

1.3.4 Multiplicidade de raízes

Prop9:"Em $P_n(x) = 0$, α *possui multiplicidade* $\mu = k \Leftrightarrow P_n^i(\alpha) = 0$, $i = 0, 1, 2, \ldots, k-1$ *e* $P_n^k(\alpha) \neq 0$*."*

Ex1. Em $x^5 - x^4 - 5 * x^3 + x^2 + 8 * x = 0$, verificar a multiplicidade da raiz $\alpha = -1$

Solução:

$P_5(-1) = 0$

$P_5'(-1) = 5 * x^4 - 4 * x^3 - 15 * x^2 + 2 * x + 8 = 0 \Rightarrow P_5'(-1) = 0$

$P_5''(-1) = 20 * x^3 - 12 * x^2 - 30 * x + 2 = 0 \Rightarrow P_5''(-1) = 0$

$60 * x^2 - 24 * x - 30 = 0 \Rightarrow P_5'''(-1) = 54 \neq 0$

Logo $\alpha = -1$ tem $\mu = 3$

Algoritmo 6 TestaMultiplicidade(Instável numericamente)

Entrada: α, n,ε,$(a_i, i = 1, n+1)$
Sada: Retorna o número de vezes que uma raiz é repetida
$m = n : \mu = 0 : resto = \varepsilon$
enquanto $resto \leq \varepsilon$ **faa**
para $i = 1$ até m **faa**
$a_{i+1} = a_{i+1} + \alpha * a_i$
fim para
$resto = |a_{m+1}| : \mu = \mu + 1 : m = m - 1$
fim enquanto

Algoritmo 7 TestaMultiplicidade(Estável)

Entrada: $n, \alpha, \varepsilon, (a_i, i = 1, n+1)$
Sada: Retorna o número de vezes que uma raiz é repetida
$m = n$
para $k = 1$ até n **faa**
para $i = 1$ até m **faa**
$a_i = a_i * (m - i + 1)$
fim para
$m = m - 1$
$executeHorner(m, \alpha, A \mid VP)$
se $(|VP| > \varepsilon)$ **ento**
$\mu = k$
FIM
fim se
fim para

1.3.5 Localização de raízes

Prop10:"Em $P_n(x) = 0$ *se* $M = máx\{|a_2|, |a_3|, \ldots, |a_{n+1}|\}$, *então* $\forall \alpha \in \mathbb{C} \Rightarrow |\alpha| < 1 + \dfrac{M}{|a_1|}$. *Se* $\alpha = a + bi \Rightarrow |\alpha| = \sqrt{a^2 + b^2}$."

Ex.1 Localize as raízes de $2 * x^{35} - 4 * x^{10} - 8 * x^2 + 3 = 0$

Solução:
$|\alpha| < 1 + \dfrac{|-8|}{|2|} = 5$

$$|\alpha| < 5 \Rightarrow \begin{cases} \alpha \in \mathbb{R} \Rightarrow \alpha \in (-5; 5) \\ \alpha \in \mathbb{C} \Rightarrow \text{círculo de centro } (0,0) \text{ e raio } r = 5. \end{cases}$$

Prop11:"Em $P_n(x) = 0$ *sejam* $\beta = a + bi$ *qualquer* $|P_n'(\beta)| \neq 0$ *e* $r = n * \left|\dfrac{P_n(\beta)}{P_n'(\beta)}\right| \Rightarrow \exists \alpha$ *no círculo de centro* (a, b) *e raio* r"

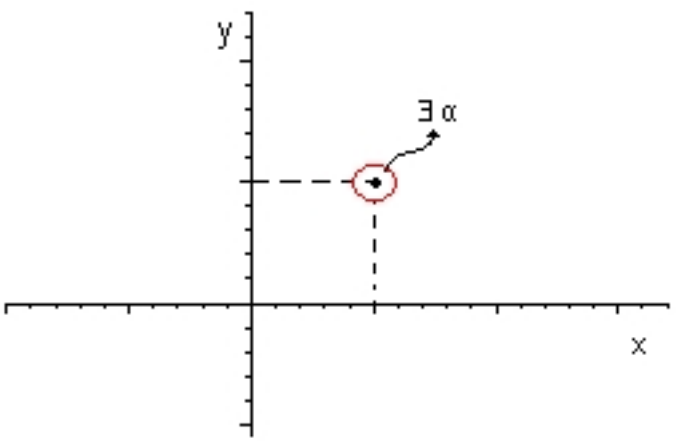
OBS: $\alpha = a + bi \Rightarrow \dfrac{1}{\alpha} = \dfrac{a - bi}{a^2 + b^2} \Rightarrow k = \dfrac{\alpha_1}{\alpha_2} = \dfrac{\alpha_1 * \overline{\alpha_2}}{|\alpha_2|^2}$

Ex.2 Localize as raízes de $x^4 + 2 * x - 3 = 0$

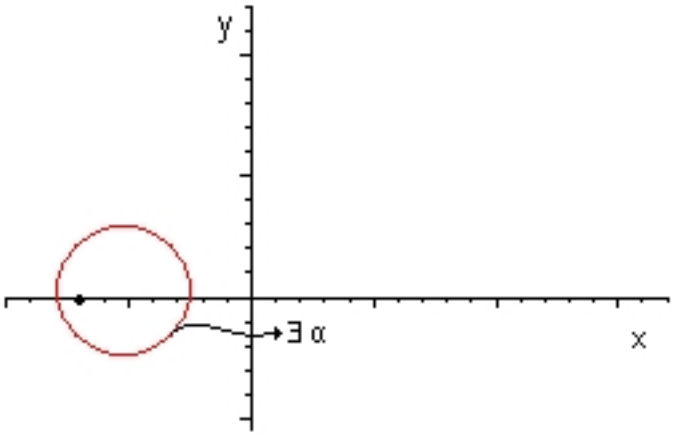
Solução:

Pela Prop10: $|\alpha| < 4$

Pela Prop11, tomando $\beta = 1 + i \Rightarrow r = 4 * 0.06 = 0.24$



Pela Prop11, tomando $\beta = -2 \Rightarrow r = 1.2$



Ex.3 Localize as raízes em $2 * x^5 - 8 * x + 3 = 0$

Solução:

Pela Prop10: $|\alpha| < 5$

Pela Prop11 com $\beta = 0$: $\Rightarrow P_5(0) = 3$ e $P_5'(0) = 8$. Assim:

$\mathrm{r} = 5 * \left| \dfrac{3}{-8} \right| = 1.875$

1.3.6 Refinamento de raízes de $P_n(x) = 0$

i) Newton

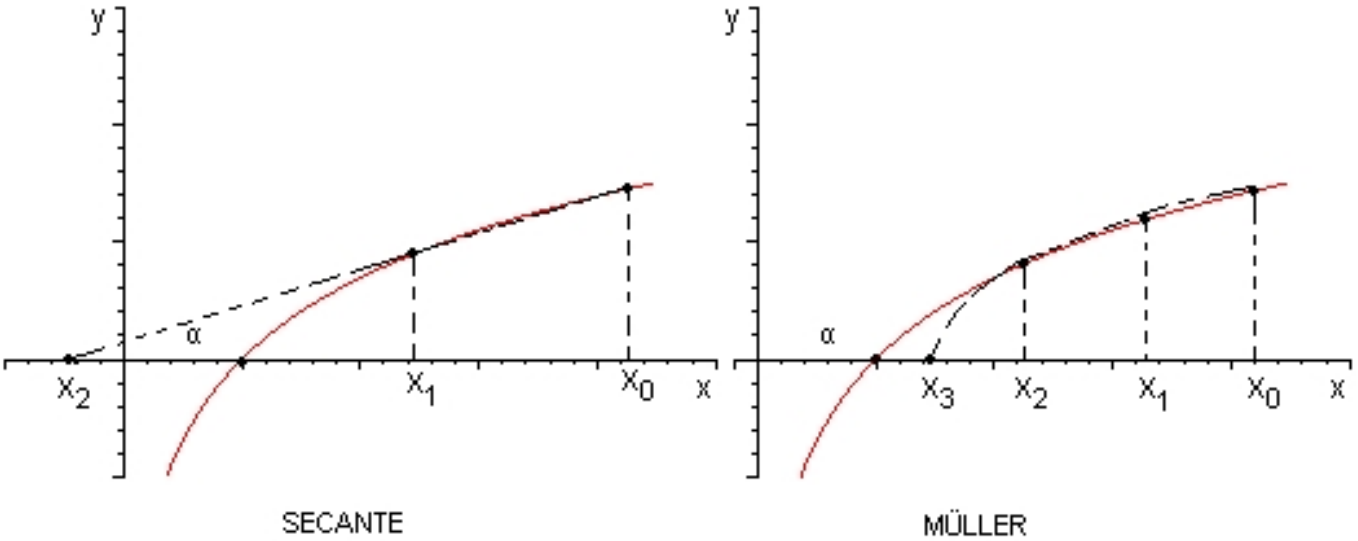
Em $P_n(x) = 0$ tomando x_0 com o auxílio das *Prop10* e/ou *Prop11* e aplicando os refinamentos de Newton 1.1 $\Rightarrow$

$$x_{k+1} = x_k - \frac{P_n(x)}{P_n'(x)} \Rightarrow x_{k+1} = x_k - \frac{R}{R_1} \Rightarrow \begin{cases} \text{não necessita de procedimento função} \\ \text{não necessita de derivada} \\ \text{Secante perde o sentido prático} \end{cases}$$

Algoritmo NewtonPol(fazer)

ii) Müller

Esta proposta é uma extensão do método da Secante.



Em $P_n(x) = 0$ para obter α, proceder:

1. Tomar três valores iniciais x_0, x_1, x_2 e gerar os três pontos (x_0, y_0); (x_1, y_1); (x_2, y_2), onde $y_i = P_n(x_i)$;

2. Obter a parábola:

$$P(x) = a * (x - x_2)^2 + b * (x - x_2) + c \tag{1.11}$$

que passa por estes três pontos:

$$\begin{cases} y_0 = p(x_0) = a * (x_0 - x_2)^2 + b * (x_0 - x_2) + c \\ y_1 = p(x_1) = a * (x_1 - x_2)^2 + b * (x_1 - x_2) + c \\ y_2 = p(x_2) = c \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} c = y_2 \\ b = \dfrac{(x_0 - x_2)^2 * (y_1 - y_2) - (x_1 - x_2)^2 * (y_0 - y_2)}{(x_0 - x_2) * (x_1 - x_2) * (x_0 - x_1)} \\ a = \dfrac{[(x_1 - x_2) * (y_0 - y_2)] - [(x_0 - x_2)(y_1 - y_2)]}{(x_0 - x_2) * (x_1 - x_2) * (x_0 - x_1)} \end{cases}$$

3. Obter por báscara as duas raízes de 1.11 $\Rightarrow$

$$\overline{x} - x_2 = \frac{-2 * c}{b \pm \sqrt{b^2 - 4 * a * c}} \Rightarrow \overline{x} = x_2 - \frac{2 * c}{b \pm \sqrt{b^2 - 4 * a * c}}$$

Aqui tem-se duas opções:

Qual de ambas devemos usar? Müller sugere que se use a opção:

$\overline{x} = x_2 - \dfrac{2 * c}{b + sinal(b)\sqrt{b^2 - 4 * a * c}}$. Se $b = 0$, o sinal é indiferente.

4. Retornar ao segundo passo com $x_0 \longleftarrow x_1$; $x_1 \longleftarrow x_2$; $x_2 \longleftarrow \overline{x}$.

Daí se $\exists$ convergência, $\lim\limits_{k \to \infty} \overline{x} = \alpha$

Ex.1 Aplique Müller em $16 * x^4 - 40 * x^3 + 5 * x^2 + 20 * x + 6 = 0$, com $x_0 = 0.5$; $x_1 = 1$; $x_2 = 1.5$ e $\varepsilon = 10^{-4}$

Solução:

$$k = 1 \left\{ \begin{array}{l} y_0 = 13.25 \\ y_1 = 7 \\ y_2 = -6.75 \end{array} \right. \Rightarrow \left\{ \begin{array}{l} c = -6.75 \\ b = -35 \\ a = -15 \end{array} \right. \Rightarrow \overline{x} = 1.28785$$

$k = 2 \Rightarrow \overline{x} = 1.23786$
$k = 3 \Rightarrow \overline{x} = 1.24160$
$k = 4 \Rightarrow \overline{x} = 1.24168$
$k = 5 \Rightarrow \overline{x} = 1.24164 \backsimeq \alpha$

$$Para \quad cada \quad k \Rightarrow \left\{ \begin{array}{l} \text{Gera sistema 2 x 2} \\ \text{Resolve sistema 2 x 2} \\ \text{Usa báskara} \end{array} \right\} Desvantagens$$

Vantagem: Dá resultados em Complexos

1.3.7 Condicionamento/Instabilidade

Numa $P_n(x) = 0$ quando se obtém uma raiz α_1 com precisão $\varepsilon \neq 0$ e reduz-se seu grau para $n - 1$ via $P_{n-1}(x) = \dfrac{P_n(x)}{x - \alpha_1}$ os coeficientes de $P_{n-1}(x)$ são afetados pelo Erro de Truncamento de α_1. Quando se obtém α_2 em $\overline{P}_{n-1}(x) = 0$ novamente repete-se a dose com patologia dobrada. No final, os resultados podem ficar totalmente deturpados devido a esta Instabilidade Numérica

Ex.1 $P_4(x) = x^4 - 11.101 * x^3 + 11.1111 * x^2 - 1.0111 * x + 0.001 = 0$, possui as raízes $\alpha_1 = 10$; $\alpha_2 = 1$; $\alpha_3 = 0.1$ e $\alpha_4 = 0.001$ com $P_4(\alpha_i) = 0$. Contudo se $\alpha_1 \backsimeq \overline{\alpha_1} = 10.000005$ e deflacionando-se $P_4(x) \div (x - \alpha_1)$ em um processador de precisão $t = 8 \Rightarrow$

$$\overline{P}_3(x) = x^3 - 1.0099 * x^2 + 0.101144 * x + 0.3405 \Rightarrow \left\{ \begin{array}{l} \overline{P}_3(1) = 0.431744! \\ \overline{P}_3(0.1) = 0.3415!! \end{array} \right\} \neq 0$$

Como resolver este problema ?

Usando o recurso da purificação de raízes obtidas após a primeira divisão.

Procedimento Purificação:

- Obter α_1 com precisão $\varepsilon \neq 0$;
- Efetuar $P_n(x) \div (x - \alpha_1) \Rightarrow \overline{P}_{n-1}(x) = 0$;
- Obter $\overline{\alpha}_2$ em $\overline{P}_{n-1}(x) = 0$ com precisão $\varepsilon \neq 0$;

  Considerar $x_0 = \overline{\alpha}_2$ e refiná-lo via Newton ou outro mais adequado em $P_n(x) = 0$ até a precisão ε para obter α_2;
- Efetuar $\overline{P}_{n-1}(x) \div (x - \alpha_2) \Rightarrow \overline{P}_{n-2}(x) = 0$

Capítulo 2

Solução de Sistemas de equações não lineares

2.1 Fundamentação

No capítulo I abordamos a solução de uma única equação a uma incógnita $f(x) = 0$. Contudo, nos modelos matemáticos é freqüente a necessidade de se considerar mais de uma variável independente, gerando por conseqüência a necessidade de se resolver sistemas de equações. Quando tais sistemas forem do tipo lineares, o assunto já foi objeto de estudo na Análise Numérica I. Aqui vamos nos ater apenas aos sistemas não lineares.

Definio 2.1 *Um sistema de n equações não lineares a n incógnitas é toda expressão do tipo:*

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases} \approx F(X) = 0, \quad onde \quad F = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} e \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Ex.1:

$$a) \begin{cases} x_1^2 + x_2^2 = 3 \\ x_1 * x_2 = 1 \end{cases} ; \quad b) \begin{cases} 3 * x_1 - \cos(x_1 * x_2) = 1 \\ x_1^2 - x_2^2 + \sin(x_3) = -1 \\ e^{-x_1 * x_2} + 20 * x_3 = -\pi. \end{cases}$$

Definio 2.2 *Solução de* $F(X) = 0$ *é toda matriz* $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T \mid F(\alpha) = 0$

Ex.2: Uma das soluções de

$$\begin{cases} x_1^2 + x_2^2 = 3 \\ x_1 * x_2 = 1 \end{cases}$$

é $\alpha = [1.618033989; 0.618033988]^T$.

Note-se que a primeira dificuldade de se resolver sistemas não lineares é a impossibilidade de se conhecer previamente se existem soluções no corpo de interesse, bem como, caso existam, quantas são. No Ex.1 a) existem quatro soluções reais, apesar da ordem do sistema de ser $n = 2$. Outra dificuldade é a inexistência de métodos abstrativos e genéricos de solução de $F(X) = 0$, restando-nos apenas os métodos construtivos.

2.2 Métodos de Solução de $F(X) = 0$

Devido a questões teóricas que situam-se além do escopo desta apostila, os métodos de quebras não serão utilizados para resolver sistemas não lineares.

Para resolver $F(X) = 0$ por Newton, procede-se:

1º) Tomar uma solução inicial $X^0 = [x_1^0, x_2^0, \ldots, x_n^0]^T$ (o índice dos expoentes é o contador das iterações)

2º) Aplicar o procedimento refinativo de Newton estendido para matrizes, resultando em:
$X^1 = X^0 - \dfrac{F(X^0)}{F'(X^0)}$; $X^2 = X^1 - \dfrac{F(X^1)}{F'(X^1)}$; $\ldots$; $X^{k+1} = X^k - \dfrac{F(X^k)}{F'(X^k)}$; onde:

$$F' = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

que é uma matriz n x n cujos elementos são $a_{ij} = \dfrac{\partial f_i}{\partial x_j}(x_1^k, \ldots, x_n^k)$. Esta matriz é denominada de Jacobiana da $F(X)$ e que denotaremos por $J(X)$.

Daí, como resultado das operações com matrizes tem-se que:

$$X^{k+1} = X^k - \frac{F(X^k)}{F'(X^k)} \simeq$$

$$\simeq X^{k+1} = X^k - J^{-1}(X^k) * F(X^k) \tag{2.1}$$

Assim cada iteração por Newton em $F(X) = 0$ necessita que seja gerada e invertida uma matriz n x n. Para minorar o custo de cada iteração, vamos reescrever (2.1) na forma:

$$\begin{bmatrix} x_1^{k+1} \\ x_2^{k+1} \\ \vdots \\ x_n^{k+1} \end{bmatrix} = \begin{bmatrix} x_1^k \\ x_2^k \\ \vdots \\ x_n^k \end{bmatrix} - \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}^{-1} * \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \simeq \begin{bmatrix} x_1^{k+1} \\ x_2^{k+1} \\ \vdots \\ x_n^{k+1} \end{bmatrix} - \begin{bmatrix} x_1^k \\ x_2^k \\ \vdots \\ x_n^k \end{bmatrix} =$$

$$= - \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} * \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n^(k) \end{bmatrix} \tag{2.2}$$

Em (2.2) considerando $\delta = X^{k+1} - X^k$, e multiplicando ambos os lados por $J(X^k)$ resulta em:

$$X^{k+1} - X^k = -J^{-1}(X^k) * F(X^k) \Rightarrow$$

$$\Rightarrow J(X^k) * \delta = -F(X^k) \tag{2.3}$$

Agora tem-se que (2.3) é um sistema linear e portanto cada iteração exigirá a geração e solução de um sistema linear cujo custo é o menor que o da inversão da matriz.

Daí, de (2.3) tem-se que $X^{k+1} = X^k + \delta$, se ocorrer a convergência da seqüência $\{X^k\}_{k=0}^{\infty} \Rightarrow \lim_{k\to\infty} X^{k+1} = \alpha$

Ex.3: Aproxime com 3 iterações a solução de $\begin{cases} e^{x_1} + x_2 - 1 = 0 \\ x_1^2 + x_2^2 = 4 \end{cases}$, tomando $X^0 = [1; -1]^T$.

Solução:
Tem-se que

$$F(X) = \begin{cases} f_1(x_1, x_2) = e^{x_1} + x_2 - 1 \\ f_2(x_1, x_2) = x_1^2 + x_2^2 - 4 \end{cases} \Rightarrow J(X) = \begin{bmatrix} e^{x_1} & 1 \\ 2 * x_1 & 2 * x_2 \end{bmatrix}$$

Aplicando (2.3) resulta em:

1ªiteração

$$\begin{bmatrix} 2.718 & 1 \\ 2 & -2 \end{bmatrix} * \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} \text{-0.718} \\ 2 \end{bmatrix} \Rightarrow \begin{cases} \delta_1 = 0.0758 \\ \delta_2 = -0.9241 \end{cases} \Rightarrow \begin{cases} x_1^1 = 1.0758 \\ x_2^1 = -1.9241 \end{cases}$$

2ªiteração

$$\begin{bmatrix} 2.933 & 1 \\ 2.152 & -3.848 \end{bmatrix} * \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} 0.0089 \\ \text{-0.859} \end{bmatrix} \Rightarrow \begin{cases} \delta_1 = -0.0598 \\ \delta_2 = 0.184 \end{cases} \Rightarrow \begin{cases} x_1^2 = 1.1016 \\ x_2^2 = -1.74 \end{cases}$$

Repetindo o mesmo procedimento resulta que $\begin{cases} x_1^3 = 1.003 \\ x_2^3 = -1.728 \end{cases}$

Portanto $\alpha \simeq [1.003; -1.728]^T$

2.3 Análise do Newton para $F(X) = 0$

Na aplicação do NEWTON em $F(X) = 0$, pelo menos três questões básicas devem ser consideradas:

1ª) Dados de entrada

Escolha do X^0;

Necessidade das n^2 derivadas parciais que formam a Jacobiana $J(X)$;

2ª) Lentidão de cada iteração

Necessidade de gerar e resolver um sistema linear n x n em cada iteração requerida;

3ª) Algoritmização

Elaboração de um algoritmo cuja implementação gere um programa eficaz, rápido e de interface amigável.

2.3.1 Dados de Entrada

Além da escolha adequada do X^0 (já comentada no capítulo I), ocorre a necessidade de se obter e implementar as n^2 derivadas parciais que compõem a Jacobiana, custo que pode se tornar proibitivo ao usuário se o n for moderado. Para evitar ste custo, vamos simular a Jacobiana.

Por definição de derivada parcial, tem-se que:

$$\frac{\partial f_i}{\partial x_j} = \lim_{h \to 0} \frac{f_i(x_1, x_2, ..., x_j + h, ..., x_n) - f_i(x_1, x_2, ..., x_n)}{h}.$$

Daí, se o h for *pequeno* $\Rightarrow \dfrac{\partial f_i}{\partial x_j} \simeq \dfrac{f_i(x_1, x_2, ..., x_j + h, ..., x_n) - f_i(x_1, x_2, ..., x_n)}{h}$

e por conseqüência cada elemento a_{ij} da Jacobiana será

$$\overline{a}_{ij} \simeq \frac{f_i(x_1, x_2, ..., x_j + h, ..., x_n) - f_i(x_1, ..., x_n)}{h}.$$

Note-se que desta forma, aproxima-se a Jacobiana sem necessidade de obter e implementar as n^2 derivadas parciais, com um custo que envolve $n^2 + n$ chamadas funcionais e os refinamentos do NEWTON são efetuados através de :

$$\overline{J}(X^k) * \delta = -F(X^k) \tag{2.4}$$

2.3.2 Lentidão das iterações

O procedimento iterativo (2.4) pode ser acelerado se forem efetuadas variações no mesmo, resultando em métodos denominados de quasi-NEWTON. A título de ilustração, apresentaremos a seguir dois destes métodos.

a) *Quasi NEWTON 1*

Para $F(X) = 0$, proceder:

1. Tomar uma solução inicial X^0;
2. Obter por simulação a $\overline{J}(X^0)$ e sua inversa $\overline{J}^{-1}(X^0)$;
3. Aplicar o procedimento iterativo

$$X^{k+1} = X^k - \overline{J}^{-1}(X^0) * F(X^k) \tag{2.5}$$

até que $||X^{k+1} - X^k|| < \varepsilon$, caso ocorra a convergência.

Note-se que a redução significativa no custo de cada iteração de (2.5) se comparada com a de (2.4). Entretanto, a abrangência de (2.5) no tocante à convergência é muito inferior à de (2.4).

b) *Quasi NEWTON 2 (Broyden)*

Para $F(X) = 0$, proceder:

1. Tomar uma solução inicial X^0;
2. Obter por simulação a $\overline{J}(X^0)$, a sua inversa $\overline{J}^{-1}(X^0)$ e $X^1 = X^0 - \overline{J}^{-1}(X^0) * F(X^0)$;
3. Para cada uma das demais iterações, fazer:
   Considerar $S = X^k - X^{k-1}$ e $U = F(X^k) - F(X^{k-1})$;
   Obter $\overline{J}^{-1}(X^k) = \overline{J}^{-1}(X^{k-1}) - \dfrac{[S - (\overline{J}^{-1}(X^{k-1}) * U) * (S^T)] * \overline{J}^{-1}(X^{k-1})}{S^T * \overline{J}^{-1}(X^{k-1}) * U}$;
   Determinar

$$X^{k+1} = X^k - \overline{J}^{-1}(X^k) * F(X^k); \tag{2.6}$$

4. Repetir o passo 3 até que $||X^{k+1} - X^k|| < \varepsilon$, caso ocorra a convergência, isto é, $\lim_{k\to\infty} X^k = \alpha$ e $\lim_{k\to\infty} \overline{J}^{-1}(X^k) = J^{-1}(\alpha)$.

Algoritmização

A seguir será apresentado o algoritmo referente ao processo (2.4) deixando-se os de (2.5) e (2.6) a cargo do leitor. Neste algoritmo será evocado um procedimento resolvedor de sistemas de equações lineares, assunto já abordado na Análise NuméricaI.

Algoritmo 8 Sistemas Não Lineares

Entrada: $n, \varepsilon, NIT, h, (x_i, i = 1, \ldots, n)$, Função
Sada: Solução $\alpha = (\alpha_i, i = 1, \ldots, n)$ na precisão ε

Procedimento Função (X|F)
$f_1 = \ldots$
$f_2 = \ldots$
$\vdots$
$f_n = \ldots$
Fim do Procedimento
para $k = 1$ até NIT **faa**
Execute Função (X|F)
para $i = 1$ até n **faa**
$ff_i = f_i$
fim para
para $j = 1$ até n **faa**
$x_j = x_j + h$
Execute Função (X|F)
para $i = 1$ até n **faa**
$$a_{ij} = \frac{(f_i - ff_i)}{h}$$
fim para
$a_{jn+1} = -ff_i$
$x_j = x_j - h$
fim para
Execute ResolveSisLin(n,A|Y)
$erro = 0$
para $i = 1$ até n **faa**
$x_i = x_i + y_i$
$erro = erro + |y_i|$
fim para
se $erro < \varepsilon$ **ento**
para $i = 1$ até n **faa**
$\alpha_i = x_i$
fim para
Escreva: 'Solução = 'α
FIM
fim se
fim para
Escreva: 'Solução não obtida.'

Capítulo 3

Teoria da Aproximação

3.1 Fundamentos

Aqui será abordado o tópico central da Análise Numérica, o qual trata da aproximação de uma função $y = f(x), x \in [a, b]$ com expressão conhecida ou desconhecida. Para fins de facilidades na apresentação e entendimento do leitor, sempre será abordada a aproximação de funções com uma única variável. Após o entendimento dos métodos e técnicas, os mesmos serão estendidos para funções com várias variáveis independentes.

A fundamentação e motivação para o estudo desta teoria, pode ser sintetizada em três questões básicas:

1ª) Por que aproximar uma $y = f(x)$?

Agruparemos a resposta a esta pergunta em seis itens:

1.a) Bibliotecas de funções pré-elaboradas

Como são obtidos os valores de $\sqrt{u}$, $\ln(u)$, e^u, $\tan(u)$, ... com processadores numéricos que efetuam apenas as operações básicas?

1.b) Sistemas de tempo real

Em determinado sistema dedicado, é possível se obter num tempo τ o valor de $f(u) = \sqrt[5]{e^{\ln|\cot u|} + \cosh(\sqrt{u})}$, dispondo-se apenas de um processador que obtém o valor de cada subfunção num tempo médio de $\frac{\tau}{3}$?

1.c) Computação gráfica

Nos sistemas de computação gráfica, o usuário indica alguns pontos de referência ou controle e o sistema preenche os caminhos intermediários destes pontos sem distorções. Como?

1.d) Filtros digitais

Numa variada gama de aplicações, obtém-se imagens, fotos, sinais, etc que são deturpados devido aos mais variados motivos (obtenção inadequada, ruídos de transmissão, etc). Posteriormente eles são reconstituídos ou filtrados digitalmente. Como?

1.e) Reconhecimento de padrões

Nos sistemas de reconhecimento de padrões são feitas as identificações dos proprietários de assinatura, voz, íris do olho, etc. Como?

1.f) Tratamento de dados obtidos em experimentos

Na quase totalidade dos experimentos nos quais é utilizado um método científico, são feitas as coletas dos valores amostrais e posteriormente extrapolados para todo o universo do experimento. E funciona. Por quê?

2ª) Com quem aproximar uma $y = f(x)$?

Por motivos óbvios, a premissa básica é que a aproximadora $z = g(x)$ deve ser uma função simples. Por conseqüência, o universo de busca de g(x) restringe-se às funções do tipo:

2.a) Polinomiais

$$g(x) = p_n(x) = \sum_{i=0}^{n} a_i * x^i \Rightarrow f(x) \simeq p_n(x)$$

2.b) Racionais

$$g(x) = R_{nm}(x) = \frac{p_n(x)}{q_m(x)} \Rightarrow f(x) \simeq R_{nm}(x)$$

2.c) Polinomiais Trigonométricas

$$g(x) = P_m(x) = \sum_{i=0}^{m} (ai * \sin(ix) + bi * \cos(ix)) \Rightarrow f(x) \simeq P_m(x)$$

Devido às suas características e suas propriedades algébricas, as polinomiais são as funções mais utilizadas como aproximadoras de $y = f(x)$. A mais importante destas propriedades é o teorema de Weierstrass: "Se $y = f(x)$ for contínua em $[a, b]$, então $\forall \varepsilon > 0, \exists p_n(x) / |f(x) - p_n(x)| < \varepsilon, \forall x \in [a, b]$". Por estar além do escopo desta apostila, este teorema não será demonstrado.

3ª) Como aproximar?

Todos os tópicos a serem abordados nesta teoria terão como objetivo responder de maneira eficaz a esta ultima questão.

3.2 Aproximação polinomial de $y = f(x)$ por interpolação

A técnica mais simples, nem sempre a mais eficiente, de se aproximar uma $y = f(x), x \in [a, b]$, consiste em:

1ª) Obter $n + 1$ valores funcionais da aproximanda na forma discreta (ou tabelada) $\begin{array}{c|c||c|c|c|c} x & x_0 & x_1 & \dots & x_n \\ \hline y & y_0 & y_1 & \dots & y_n \end{array}$, onde $x_0 = a, x_n = b$ e $y_i = f(x_i)$, com os x_i escolhidos adequadamente;

2ª) Determinar um polinômio de grau n, $p_n(x) = a_0 + a_1 * x + \ldots + a_n * x^n$ tal que $p_n(x_i) = y_i$; $i = 0, 1, \ldots, n$. Este polinômio aproxima a $f(x)$, sendo denominado de seu interpolador. A questão agora é só obter os coeficientes a_i do interpolador.

Aplicando-se a condição de aproximação, $p_n(x_i) = y_i$; resulta:

$$\begin{cases} p_n(x_0) = a_0 + a_1 * x_0 + \ldots + a_n * x_0^n \\ p_n(x_1) = a_0 + a_1 * x_1 + \ldots + a_n * x_1^n \\ \vdots \\ p_n(x_n) = a_0 + a_1 * x_n + \ldots + a_n * x_n^n \end{cases} \simeq \begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{bmatrix} * \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} \quad (3.1)$$

Note-se que (3.1) é um sistema linear $n + 1$ x $n + 1$, cuja solução se existir, fornece os coeficientes do interpolador da $y = f(x)$. Para fins de simplificação, vamos representar (3.1) na forma reduzida $U * A = Y$, onde os elementos da matriz dos coeficientes U podem ser gerados via $u_{ij} = x_i^{j-1}$.

Ex.1: Aproximar a $f(x) = e^x, x \in [0.5; 1]$ através do seu interpolador de grau $n = 2$ e estimar o valor de $f(0.8)$

Solução:

Como n=2 e $[a, b] = [0.5; 1]$, discretizando $[0.5; 1]$ em partes iguais resulta

x	0.5	0.75	1
$f(x) = e^x$	1.649	2.117	2.718

Aplicando (3.1):

$$\begin{bmatrix} 1 & 0.5 & 0.25 \\ 1 & 0.75 & 0.5625 \\ 1 & 1 & 1 \end{bmatrix} * \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1.649 \\ 2.117 \\ 2.718 \end{bmatrix}$$

Resolvendo o sistema linear, tem-se que $a_0 = 1.512$; $a_1 = -0.658$ e $a_2 = 1.864$.

Daí, $f(x) = e^x \simeq p_2(x) = 1.512 - 0.658 * x + 1.864 * x^2$

Estimando $e^{0.8}$, tem-se que $e^{0.8} \simeq p_2(0.8) = 2.178$

Análise

Sob a ótica do analista, a aproximação pela técnica da interpolação polinomial, suscita pelo menos três questões:

1ª) Será que (3.1) tem solução sempre? Se possuir solução, ela é única?

2ª) É possível se obter $p_n(x)$ com maior eficiência computacional?

3ª) Qual é a precisão de $p_n(x)$ quando $p_n(x) \simeq f(x)$?

3.2.1 Existência e Unicidade do interpolador

Para assegurar a existência e unicidade do interpolador basta provar que (3.1) possui solução única, ou de maneira equivalente, que o determinante $detU \neq 0$.

Como U é uma matriz específica ($u_{ij} = x_i^{j-1}$), Vandermond provou que para este tipo de matrizes, $detU = \prod_{i>j}(x_i - x_j)$

Ora, como a tabela é uma função, isto é, $x_i \neq x_j \Rightarrow \prod_{i>j}(x_i - x_j) \neq 0 \Rightarrow detU \neq 0 \Rightarrow$ Existe um único interpolador.

3.2.2 Determinação eficiente do interpolador

Note-se que para obter o interpolador via (3.1), tem-se que:

- Gerar o sistema linear $U * A = Y$ ($\Rightarrow n^2 - 2 * n$ operações)
- Resolver o sistema linear (3.1) ($\Rightarrow \frac{n^3}{2}$ operações se for usada a eliminação Gaussiana)

Como tornar este processo mais eficiente?

Aqui existem pelo menos duas alternativas clássicas para a solução deste problema:

1ª) Usar o interpolador na forma de LAGRANGE

Definio 3.1 *Para uma função discreta* $\begin{array}{c|c|c|c|c} x & x_0 & x_1 & \dots & x_n \\ \hline y & y_0 & y_1 & \dots & y_n \end{array}$*,seu polinômio de grau n na forma de LAGRANGE é a expressão:*

$$Lp_n(x) = \sum_{i=0}^{n} y_i * \left[\prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \right] \tag{3.2}$$

Daí, tem-se que $Lp_n(x_0) = y_0$; $Lp_n(x_1) = y_1$; ...; $Lp_n(x_n) = y_n$ e portanto $Lp_n(x)$ é o interpolador.

2ª) Usar Interpolador de NEWTON com Diferenças Divididas

Definio 3.2 *Para a função* $\begin{array}{c|c|c|c|c} x & x_0 & x_1 & \dots & x_n \\ \hline y & y_0 & y_1 & \dots & y_n \end{array}$*,tem-se que suas diferenças divididas* (Δ) *são:*

$\Delta y_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$ *(1ª Ordem), (i=0, n-1)*

$\Delta^2 y_i = \Delta(\Delta y_i) = \frac{\Delta y_{i+1} - \Delta y_i}{x_{i+2} - x_i}$ *(2ª Ordem)*

$\vdots$

$\Delta^k y_i = \frac{\Delta^{k-1} y_{i+1} - \Delta^{k-1} y_i}{x_{i+k} - x_i}$ *(k-ésima ordem)*

Note que as diferenças divididas para uma $y = f(x)$ na forma tabelada, são adaptações da definição de derivadas para $y = f(x)$ com expressão conhecida, e que a determinação das $\Delta^k y_i$ é um processo recursivo.

Definio 3.3 *Para a função* $\begin{array}{c|c||c|c|c|c} x & & x_0 & x_1 & \dots & x_n \\ \hline y & & y_0 & y_1 & \dots & y_n \end{array}$*, seu polinômio de Newton com diferenças dividas de grau n é:*

$$Np_n(x) = y_0 + \sum_{k=1}^{n} \Delta^k y_0 * \left[\prod_{j=0}^{k-1} (x - x_j) \right] \tag{3.3}$$

Daí tem-se que:

$Np_n(x_0) = y_0; Np_n(x_1) = y_0 + \Delta y_0(x_1 - x_0) = y_0 + \dfrac{y_1 - y_0(x - x_0)}{x_1 - x_0} = y_1$

$Np_n(x_2) = y_2; \dots; Np_n(x_n) = y_n$(pode ser comprovado por indução finita). Portanto $Np_n(x)$ é o interpolador da tabela.

Ex2: Na função $\begin{array}{c|c||c|c|c|c} x & & 1 & 3 & 4 & 7 \\ \hline y = f(x) & & 2 & 0 & 1 & 3 \end{array}$, estime os valores de $f(5)$ e $f(8)$ usando os interpoladores de LAGRANGE e de NEWTON com Δ.

Solução

Tem-se que $n + 1 = 4 \Rightarrow n = 3$

Aplicando o interpolador de LAGRANGE (3.2), resulta:

$$Lp_3(x) = 2 * \frac{(x-3)*(x-4)*(x-7)}{(1-3)*(1-4)*(1-7)} + 0 * \frac{(x-1)*(x-4)*(x-7)}{(3-1)*(3-4)*(3-7)} + 1 * \frac{(x-1)*(x-3)*(x-7)}{(4-1)*(4-3)*(4-7)} + 3 * \frac{(x-1)*(x-3)*(x-4)}{(7-1)*(4-3)*(7-4)}$$

$$Lp_3(x) = 2 * \frac{(x-3)*(x-4)*(x-7)}{-36} + 0 * \frac{(x-1)*(x-4)*(x-7)}{-8} + 1 * \frac{(x-1)*(x-3)*(x-7)}{-9} + 3 * \frac{(x-1)*(x-3)*(x-4)}{72}$$

Daí,

$f(5) \simeq Lp_3(5) = 2.333$

$f(8) \simeq Lp_3(8) = 0.833$

Aplicando o interpolador de NEWTON com Δ (3.3), resulta:

i) Diferenças

x	y	Δ	Δ^2	Δ^3
1	2	-1	$\frac{2}{3}$	$-\frac{1}{8}$
3	0	1	$-\frac{1}{12}$	
4	1	$\frac{2}{3}$		
7	3			

ii) Interpolador

$Np_3(x) = y_0 + (\Delta y_0) * (x - x_0) + (\Delta^2 y_2) * (x - x_0) * (x - x_1) + (\Delta^3 y_0) * (x - x_0) * (x - x_1) * (x - x_2)$

$$Np_3(x) = 2 - 1 * (x - 1) + \frac{2}{2} * (x - 1) * (x - 3) - \frac{1}{8} * (x - 1) * (x - 3) * (x - 4)$$

Daí,

$f(5) \simeq Np_3(5) = 2.333$
$f(8) \simeq Np_3(8) = 0.833$

Note-se que em ambos os interpoladores obteve-se os mesmos resultados, conforme previsto pela unicidade do interpolador.

Ex.3: Algoritmos

a) Algoritmo interpolador de LAGRANGE

Algoritmo 9 Interpolador de LAGRANGE

Entrada: n, u, $((x_i, y_i), i = 0, \ldots, n)$
Sada: $f(u) \simeq Lp_n(u)$
num = 1
para $i = 0$ até n **faa**
$num = num * (u - x_i)$
fim para
$S = 0$
para $i = 0$ até n **faa**
den = 1
para $j = 0$ até n **faa**
se $i \neq j$ **ento**
$den = den * (x_i - x_j)$
fim se
fim para
$S = S + \frac{(y_i * num)}{(den * (x - x_i))}$
fim para
Escreva: 'Valor interpolado' S
FIM

b) Algoritmo interpolador de NEWTON Δ

Deixamos este algoritmo ao leitor.

Ex.4: Verificação do interpolador computacionalmente mais eficiente.

Efetuando-se contagem do número de operações executadas em cada interpolador para se estimar $f(u)$, resulta:
Em $Lp_n(u) \Rightarrow 2 * n^2 + 7 * n$ operações
Em $Np_n(u) \Rightarrow \frac{3 * n^2 + 11 * n}{2}$ operações.

3.2.3 Precisão do interpolador

Na aproximação de $y = f(x)$ com expressão conhecida e $x \in [a, b]$ através de um interpolador (Ex.1), tem-se que:

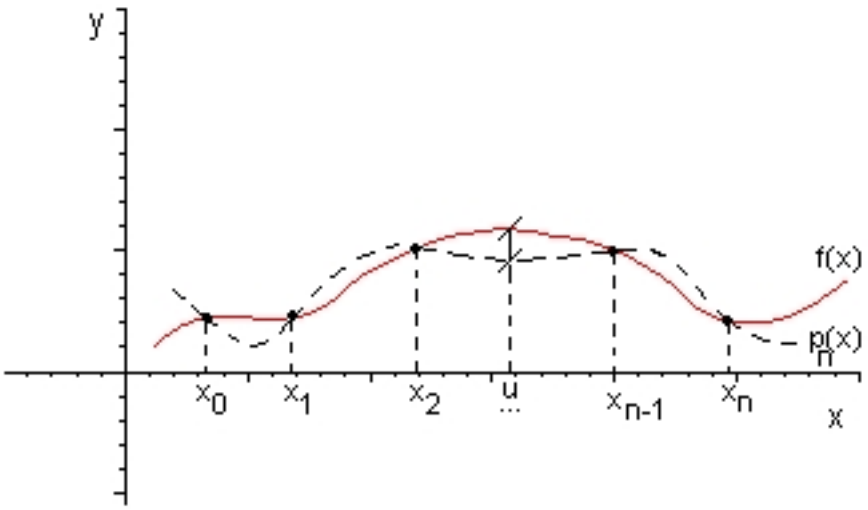


Figura 3.1: Interpolação

$\Rightarrow E(u) = |p_n(u) - f(u)|$.

A delimitação do erro de truncamento de cada valor u a ser estimado, indicará o grau de confiança a ser depositado nas aproximações $f(u) \simeq p_n(u)$.

O erro $E(u)$ pode ser estimando utilizando-se o seguinte teorema, o qual novamente não será demonstrado:

Teorema 3.1 *Se $y = f(x)$ for contínua e diferenciável em todas as ordens em $[a, b]$ e $p_n(x)$ o seu interpolador, então $\forall u \in [a, b], \exists \varepsilon \in [a, b]$ tal que:*

$$E(u) = |f(u) - p_n(u)| = \frac{f^{n+1}(\varepsilon) * \prod_{i=0}^{n}(u - x_i)}{(n+1)!} \tag{3.4}$$

Note-se que a aplicação prática de (3.4) é dificultada devido a que:

i) Nem sempre é fácil de se obter todas as derivadas $f^{n+1}(x)$;

ii) Não se sabe o valor do ε, apenas sua localização;

iii) $\forall u \Rightarrow$ reaplicar (3.4) (EFEITO LOCAL)

Para amenizar estas dificuldades, pode-se usar o:

Corolrio 3.1 *"Se além das condições do (Teorema 3.1), os x_i da tabela estiverem ordenados ($x_{i+1} > x_i$) e igualmente espaçados ($x_{i+1} - x_i = h$), então:*

$$|E(u)| < \frac{h^{n+1} * M}{4 * (n+1)}, \quad onde \quad M = máx_{x \in [a,b]} \left| f^{n+1}(x) \right| ." \tag{3.5}$$

Note-se que (3.5) é uma expressão de EFEITO GLOBAL no domínio $[a, b]$ da aproximanda.

Ex.5: Determinar o erro de truncamento máximo cometido ao se aproximar $f(x) = \ln(x), x \in [1; 2]$ através de um interpolador de grau $n = 20$

Solução
Aplicando (3.5) tem-se que:
$n = 20 \Rightarrow h = \dfrac{b-a}{n} = \dfrac{1}{20} = 0.05$
$f(x) = \ln x;\ f'(x) = x^{-1};\ f''(x) = -x^{-2};\ \ldots;\ f^{n+1}(x) = n! * x^{(-n+1)}$

Daí,
$M = máx_{x\in[1;2]} \left|f^{n+1}(x)\right| = 20!$ e $|E(u)| < \dfrac{(0.05)^{21} * 20!}{4 * 21} = 1.3 * 10^{-11}$

e portanto, tem-se que erro máximo de $\varepsilon = 1.3 * 10^{-11}$

Ex.6: Determinar o grau n do interpolador de $f(x) = \sqrt{e^x}$, o qual deverá aproximá-la com precisão $\varepsilon = 10^{-10}$ no domínio $[0; 1]$.

Solução:
Tem-se que:
$f(x) = \sqrt{e^x} \Rightarrow f'(x) = \dfrac{1}{2} * e^{\frac{x}{2}};\ f''(x) = \dfrac{1}{2^2} * e^{\frac{x}{2}};\ f'''(x) = \dfrac{1}{2^3} * e^{\frac{x}{2}};\ \ldots;$
$f^{n+1}(x) = \dfrac{1}{2^{n+1}} * e^{\frac{x}{2}}$.

Daí $M = máx_{x\in[0;1]} \left|f^{n+1}(x)\right| = f^{n+1}(1) = \dfrac{\sqrt{e}}{2^{n+1}}$.

Aplicando (3.5), resulta em:

$10^{-10} \simeq \dfrac{h^{n+1} * M}{4 * (n+1)} \Rightarrow 10^{-10} \simeq \dfrac{(\frac{1}{n})^{n+1} * M}{4 * (n+1)} \Rightarrow 10^{-10} \simeq \dfrac{\sqrt{e}}{n^{n+1} * 2^{n+1} * 4 * (n+1)} \Rightarrow$
$n = 8$.

3.2.4 Interpolação de funções com várias variáveis

Conforme já comentado no início do Capítulo 2, o estudo de determinado fenômeno, não pode ser completo se for considerada apenas uma variável influenciante. Tem-se que, via de regra, considerar várias variáveis independentes. Por exemplo, como estimar $f(u; v)$ quando a aproximanda é uma função

$$\begin{cases} f : \mathbb{R}x\mathbb{R} \to \mathbb{R} \\ (x, y) \mapsto z = f(x, y) \end{cases} \quad se \quad x \in [a, b] \quad e \quad y \in [c, d] \quad for \quad sua \quad região \quad de \quad domínio?$$

Para solucionar este problema pode-se:

1º) Dividir $[a, b]$ em n partes e $[c, d]$ em m partes, isto é, discretizar o domínio;

2º) Obter $k = n * m$ valores funcionais da aproximanda $z = f(x, y)$;

$y \setminus x$	x_0	x_1	$\ldots$	x_n
y_0	z_{00}	z_{10}		z_{n0}
y_1	z_{01}	z_{11}		z_{n1}
$\vdots$				$\vdots$
y_m	z_{0m}	z_{1m}	$\ldots$	z_{nm}

3º) Fixar uma das variáveis (por convenção será fixada a variável x);

4º) Para cada x_i fixado, obter a tabela $\begin{array}{c\|\|c|c|c|c} y & y_0 & y_1 & \ldots & y_m \\ \hline z & z_{i0} & z_{i1} & \ldots & z_{im} \end{array}$, o seu interpolador $pY_m(y)$ e estimar $pY_m(v) = t_i$, $i = 0, 1, \ldots, m$;

5º) Do passo anterior resulta a tabela $\begin{array}{c\|\|c|c|c|c} x & x_0 & x_1 & \ldots & x_n \\ \hline t & t_0 & t_1 & \ldots & t_n \end{array}$. Agora basta obter o seu interpolador $pX_n(x)$ e estimar $pX_n(v) \simeq f(u; v)$.

Observe-se que a estimativa via interpolação de um único $f(u; v)$ envolve $n+2$ interpolações simples, e que este mesmo raciocínio pode ser estendido para três, quatro, ... variáveis. Por exemplo, a estimativa de $f(u, v, w)$ envolve $(n + 1)$ interpolações duplas e uma simples $\Rightarrow n^2 + 3 * n + 3$ simples. Conseqüentemente, o crescimento no volume de operações a serem executadas cresce exponencialmente.

Ex.6: Algoritmo interpolação dupla de LAGRANGE.

Deixamos a cargo do leitor.

3.3 Aproximação de $y = f(x)$ por splines cúbicas

Quando aproxima-se por interpolação uma $y = f(x), x \in [a, b]$ e que possua gráfico com alterações de comportamento (caminhos), obtém-se resultados aproximadores pobres, pois se o grau n for pequeno resultam erros de truncamento elevados, se por outro lado, o grau n for grande resultam muitas oscilações da aproximadora e por conseqüência deturpações no caminho.

Ex.1: Aproxime por interpolação a função

$$f(x) = \begin{cases} \left|x^3 - 1\right| \text{ se } |x| \geqslant 1 \\ \\ \left|x^2 - 1\right| \text{ se } -1 < x < 1 \end{cases} \Rightarrow$$

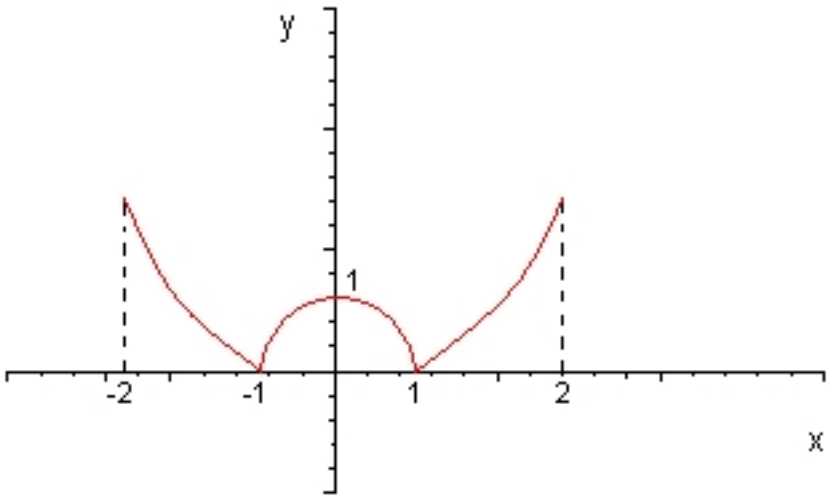


Figura 3.2: Função a ser interpolada

Solução:

Deixamos a cargo do leitor a tarefa de esboçar os gráficos dos interpoladores de grau $n = 2$ e $n = 4$ desta função e verificar o comportamento dos mesmos.

Este tipo de aproximação é feito com técnicas nas quais a aproximanda deve se adaptar ao comportamento do caminho, isto é, ser flexível.

Definio 3.4 *Aproximação spline (=haste flexível) é uma técnica que consiste em se dividir o domínio $[a, b]$ em k partes convenientemente espaçadas e aproximar separadamente em cada parte, efetuando-se as junções da maneira mais suave possível.*

Definio 3.5 *Aproximação de $y = f(x), x \in [a, b]$ por splines cúbicas consiste em:*

1º) Dividir $[a, b]$ em k partes $[x_i; x_{i+1}] \mid x_{i+1} > x_i$ e $h_i = x_{i+1} - x_i$, $(i = 1, 2, \ldots, k)$;

2º) Para cada $[x_i; x_{i+1}]$ obter uma aproximadora cúbica $sp_i(x) = a_i * (x - x_i)^3 + b_i * (x - x_i)^2 + c_i * (x - x_i) + d_i$, tal que:

2.a) $sp_i(x_i) = y_i, i = 1, 2, \ldots, k$

2.b) $sp_{i-1}(x_i) = sp_i(x_i), i = 2, \ldots, k$ *(*CONTINUIDADE*)*

2.c) $sp'_{i-1}(x_i) = sp'_i(x_i), i = 2, \ldots, k$ *(*SUAVIDADE*)*

2.d) $sp''_{i-1}(x_i) = sp''_i(x_i), i = 2, \ldots, k$ *(*ENCURVAMENTOS IGUAIS*)*

Agora tem-se que obter os coeficientes a_i, b_i, c_i, d_i de cada uma das k splines $sp_i(x)$, utilizando-se das quatro condições anteriores.

Aplicando 2.a) em $sp_i(x) \Rightarrow sp_i(x_i) = y_i = d_i \Rightarrow$

$$d_i = y_i \quad i = 1, 2, \ldots, k \tag{3.6}$$

Derivando $sp_i(x) \Rightarrow$

$$sp_i'(x) = 3 * a_i * (x - x_i)^2 + 2 * b_i * (x - x_i) + c_i \tag{3.7}$$

e

$$sp_i''(x) = 6 * a_i * (x - x_i) + 2 * b_i \tag{3.8}$$

Agora considerando as auxiliares $S_i = sp_i''(x_i)$ e $S_{i+1} = sp_i''(x_{i+1})$ e aplicando (3.8) $\Rightarrow 2 * b_i = S_i \Rightarrow$

$$b_i = \frac{S_i}{2} \tag{3.9}$$

e $S_{i+1} = 6 * a_i * h_i + 2 * b_i \Rightarrow$

$$a_i = \frac{S_{i+1} - S_i}{6 * h_i} \tag{3.10}$$

Como $sp_i(x_{i+1}) = y_{i+1} = a_i * h_i^3 + b_i * h_i^2 + c_i * h_i + d_i$ e substituindo (3.6), (3.9) e (3.10) nesta expressão resulta:

$$y_{i+1} = \left(\frac{S_{i+1} - S_i}{6 * h_i}\right) * h_i^3 + \left(\frac{S_i}{2}\right) * h_i^2 + c_i * h_i + y_i \Rightarrow$$

$$c_i = \frac{y_{i+1} - y_i}{h_i} - \left(\frac{S_{i+1} + 2 * S_i}{6}\right) * h_i \tag{3.11}$$

Agora tem-se todos os quatro coeficientes de cada spline expressos em função das y_i, h_i, S_i e S_{i+1}, isto é:

$$(*) \begin{cases} a_i = \dfrac{(S_{i+1} - S_i)}{6 * h_i} \\ b_i = \dfrac{S_i}{2} \\ \\ c_i = \dfrac{y_{i+1} - y_i}{h_i} - \left(\dfrac{S_{i+1} + 2 * S_i}{6}\right) * h_i \\ d_i = y_i \end{cases}$$

Portanto, para se obter as $sp_i(x)$ basta determinar os valores de S_i e S_{i+1}.
Como $sp_i'(x_i) = sp_{i-1}'(x_i)$ e aplicando (3.7) resulta:

$$c_i = 3 * a_{i-1} * h_{i-1}^2 + 2 * b_i * h_{i-1} + c_{i-1} \tag{3.12}$$

Substituindo as expressões de (*) em (3.12) e agrupando $\Rightarrow$

$$h_{i-1}*S_{i-1}+2*(h_{i-1}+h_i)*S_i+h_i*S_{i+1}=6*\left[\frac{(y_{i+1}-y_i)}{h_i}-\frac{y_i-y_{i-1}}{h_{i-1}}\right] \quad i=2,\ldots,k \tag{3.13}$$

Desenvolvendo (3.13) $\Rightarrow$

$$(**)\begin{cases} h_1*S_1+2*(h_1+h_2)*S_2+h_2*S_3=6*\left[\dfrac{y_3-y_2}{h_2}-\dfrac{y_2-y_1}{h_1}\right] \\ h_2*S_2+2*(h_2+h_3)*S_3+h_3*S_4=6*\left[\dfrac{y_4-y_3}{h_3}-\dfrac{y_3-y_2}{h_2}\right] \\ \vdots \\ h_{k-1}*S_{k-1}+2*(h_{k-1}+h_k)*S_k+h_k*S_{k+1}=6*\left[\dfrac{y_{k+1}-y_k}{h_k}-\dfrac{y_k-y_{k-1}}{h_{k-1}}\right] \end{cases} ,i=2,\ldots,k$$

Note-se que (**) é um sistema linear de ordem $k-1$x$k+1$, tendo por conseqüência infinitas soluções. Para que este sistema possua solução única, isto é, seja não ambíguo, tem-se que impor duas novas condições de aproximação. Estas condições são centradas nos pontos extremos do caminho, uma vez que os mesmos estão "soltos"ou livres. Daí, para cada tipo de condição imposta nos extremos resulta um tipo de spline cúbica. A seguir serão explicitadas dois tipos de splines cúbicas.

3.3.1 Splines cúbicas do tipo natural

Considerando que a aproximanda $y=f(x)$ possui tendência linear nos extremos, resulta que $S_1=0$ e $S_{k+1}=0$ e de (**) $\Rightarrow$

$$\begin{bmatrix} 2(h_1+h_2) & h_2 & 0 & \ldots & \ldots & 0 \\ h_2 & 2(h_2+h_3) & h_3 & \ldots & \ldots & 0 \\ 0 & h_3 & 2(h_3+h_4) & h_4 & \ldots & 0 \\ \vdots & \vdots & \vdots & & & \vdots \\ 0 & 0 & \ldots & \ldots & h_{k-1} & 2(h_{k-1}+h_k) \end{bmatrix} * \begin{bmatrix} S_2 \\ S_3 \\ S_4 \\ \vdots \\ S_k \end{bmatrix} =$$

$$= 6 * \begin{bmatrix} \dfrac{y_3-y_2}{h_2}-\dfrac{y_2-y_1}{h_1} \\ \dfrac{y_4-y_3}{h_3}-\dfrac{y_3-y_2}{h_2} \\ \vdots \\ \dfrac{y_{k+1}-y_k}{h_k}-\dfrac{y_k-y_{k-1}}{h_{k-1}} \end{bmatrix}$$

que é um sistema linear de ordem kxk, do tipo tridiagonal e cuja solução fornece os $S_i, i=2,...,k$ que juntamente com o S_0 e S_{k+1} quando substituídos em (*) geram todos os coeficientes das k splines cúbicas.

3.3.2 Splines cúbicas com extremos quadráticos

Se $y = f(x)$ possuir tendência parabólica nos extremos, então $S_1 = S_2$ e $S_k = S_{k+1}$ e (**) torna-se:

$$\begin{bmatrix} 3h_1+2h_2 & h_2 & 0 & \dots & \dots & 0 \\ h_2 & 2(h_2+h_3) & h_3 & \dots & \dots & 0 \\ 0 & h_3 & 2(h_3+h_4) & h_4 & \dots & 0 \\ \vdots & \vdots & \vdots & & & \vdots \\ 0 & 0 & \dots & \dots & h_{k-1} & 2h_{k-1}+3h_k \end{bmatrix} * \begin{bmatrix} S_2 \\ S_3 \\ S_4 \\ \vdots \\ S_k \end{bmatrix} =$$

$$= 6 * \begin{bmatrix} \frac{y_3-y_2}{h_2} - \frac{y_2-y_1}{h_1} \\ \frac{y_4-y_3}{h_3} - \frac{y_3-y_2}{h_2} \\ \vdots \\ \frac{y_{k+1}-y_k}{h_k} - \frac{y_k-y_{k-1}}{h_{k-1}} \end{bmatrix}$$

que novamente é um sistema linear tridiagonal e cuja solução após substituída em (*) fornece os coeficientes das splines cúbicas.

Ex.1: Estimar o valor de $f(2.5)$ em $\begin{array}{c|c||c|c|c|c|c} x & 0 & 1 & 2 & 3 & 4 \\ \hline y & -3 & -2 & 5 & 24 & 61 \end{array}$, usando splines cúbicas do tipo natural.

Solução:
Tem-se que:
$k+1 = 5 \Rightarrow k = 4$ e $h_i = x_{i+1} - x_i = 1$.

Aplicando estes dados para gerar o sistema linear das splines naturais, resulta:

$$\begin{bmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{bmatrix} * \begin{bmatrix} S_2 \\ S_3 \\ S_4 \end{bmatrix} = \begin{bmatrix} 36 \\ 72 \\ 108 \end{bmatrix}, \quad \textit{Resolvendo} \quad \Rightarrow \begin{cases} S_2 = 6.428 \\ S_3 = 10.285 \\ S_4 = 24.428 \end{cases}$$

Como $S_1 = S_5 = 0$ e o $u = 2.5$ está situado no $k = 3$ (3^o subintervalo), de (*) resulta que:

$$\begin{cases} a_3 = \dfrac{S_4 - S_3}{6 * h_3} = 2.357 \\ b_3 = \dfrac{S_3}{2} = 5.1428 \\ c_3 = 11.5 \\ d_3 = Y_3 = 5 \end{cases} \qquad \textit{Daí} \quad f(2.5) \simeq sp_3(2.5) = 12.33$$

Ex.2: Deixamos a cargo do leitor a tarefa de obter o mesmo $f(2.5)$ usando splines cúbicas com extremos quadráticos.

Ex.3: Splines Cúbicas Naturais

Algoritmo 10 Splines Cúbicas Naturais

Entrada: $k, v, ((x_i, y_i), i = 1, 2, \ldots, k+1)$

Sada: A, B, C, D, que conterão respectivamente os coeficientes das k splines

para i até k **faa**

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

fim para

$m = k - 1$

$r_1 = 2*(h_1+h_2) : d_1 = h_2 : au1 = \dfrac{y_2 - y_1}{h_1} : au2 = \dfrac{y_3 - y_2}{h_2} : g_1 = 6*(au2+au1)$

para $i = 2$ até $m - 1$ **faa**

$t_i = h_i : r_i = 2 * (h_{i+1} - h_i) : d_i = h_{i+1}$

$au1 = au2 : au2 = \dfrac{(y_{i+2} - y_{i+1})}{h_{i+1}} : g_i = (au1 - au2) * 6$

fim para

$t_m = h_m : r_m = 2 * (h_m + h_k) : g_m = \left(\dfrac{(y_{k+1} - y_k)}{h_k} - au2\right) * 6$

para $i = 2$ até m **faa**

$r_i = r_i - \left(\dfrac{t_i}{r_{i-1}}\right) * d_{i-1} : g_i = g_i - \left(\dfrac{t_i}{r_{i-1}}\right) * g_{i-1}$

fim para

$x_m = \dfrac{g_m}{r_m}$

para $i = m - 1$ até 1 **faa**

$x_i = \dfrac{(g_i - d_i * x_{i+1})}{r_i}$

fim para

para $i = 2$ até k **faa**

$S_i = x_{i-1}$

fim para

$in = 1 : fim = k + 1$

enquanto $|fim - in| \neq 1$ **faa**

$meio = parteInteira\left(\dfrac{(fim + in)}{2}\right)$

se $x_{meio} < v$ **ento**

$in = meio$

seno

$fim = meio$

fim se

fim enquanto

$j = in$

$a_j = \dfrac{(S_{j+1} - S_j)}{6 * h_j} : b_j = \dfrac{S_j}{2}$

$c_j = \dfrac{(y_{j+1} - y_j)}{h_j} - \dfrac{(S_{j+1} + 2 * S_j * h_j)}{6} : d_j = y_j$

$aux = v - x_j$

$P = d_j + aux * (c_j + aux * (b_j + aux * a_j))$

$Escreva' f(v) =' P$

FIM

3.4 Aproximação de não funções via parametrização

As duas técnicas de aproximação abordadas até o momento, possibilitam o traçado de caminhos que sejam funções. Se o caminho ou curva for não funcional, como por exemplo o esboçado abaixo.

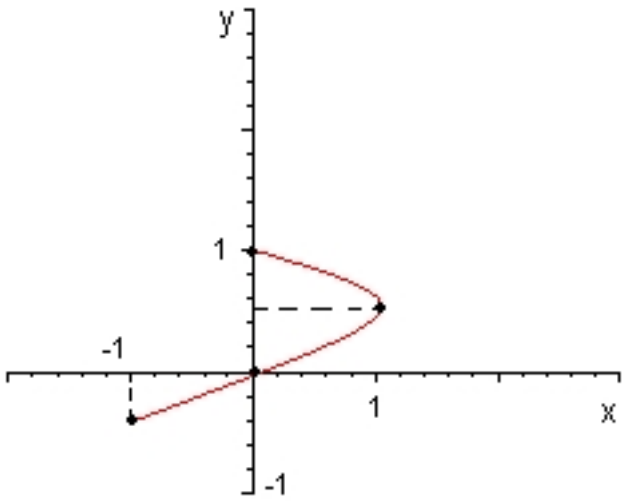


Figura 3.3: Aproximação de não funções via parametrização

Para aproximá-lo, procede-se como segue:

1º) Escolher $n+1$ pontos de referência do caminho $\begin{array}{c|c|c|c|c} x & x_0 & x_1 & \dots & x_n \\ \hline y & y_0 & y_1 & \dots & y_n \end{array}$. Na Figura (3.3) $\Rightarrow$ $\begin{array}{c|c|c|c|c} x & 0 & 1 & 0 & -1 \\ \hline y & 1 & \frac{1}{2} & 0 & -\frac{1}{2} \end{array}$;

2º) Adotar uma nova variável independente (= parâmetro) $t \in [0;1]$ e tomar $n+1$ valores $t_i \mid t_i < t_{i+1}, i = 0, 1, \dots, n$ e gerar as duas tabelas:

$$\left\{ \begin{array}{c|c|c|c|c} t & t_0 & t_1 & \dots & t_n \\ \hline x & x_0 & x_1 & \dots & x_n \end{array} \right. \qquad \left. \begin{array}{c|c|c|c|c} t & t_0 & t_1 & \dots & t_n \\ \hline y & y_0 & y_1 & \dots & y_n \end{array} \right\}$$

$$Na \quad figura \quad (3.3) \Rightarrow \left\{ \begin{array}{c|c|c|c|c} t & 0 & \frac{1}{3} & \frac{2}{3} & 1 \\ \hline x & 0 & 1 & 0 & -1 \end{array} \right. \qquad \left. \begin{array}{c|c|c|c|c} t & 0 & \frac{1}{3} & \frac{2}{3} & 1 \\ \hline y & 1 & \frac{1}{2} & 0 & -\frac{1}{2} \end{array} \right\}$$

Note-se que estas duas tabelas agora são funções, isto é, cada par (x, y) do caminho é $(x, y) = (x(t), y(t))$;

3º) Obter pela técnica mais adequada ao problema as duas aproximadoras de $x(t)$ e $y(t)$. Por exemplo, na figura (3.3), aplicando interpolação de Newton com Δ resulta:

$$\begin{cases} x(t) \simeq pX_3(t) = 3*t - 3*t*(3*t-1) + t*(3*t-1)*(3*t-2) \\ y(t) \simeq pY_3(t) = 1 - \dfrac{3*t}{2} \end{cases}$$

4º) Plotar os pontos de interesse $P_k = (pX_n(t_k); pY_n(t_k))$para gerar o caminho desejado. Na figura (3.3), usando os interpoladores do passo anterior, tem-se que para $t_k = \dfrac{1}{2} \Rightarrow P_k = (0.625; 0.25)$.

Deixaremos a cargo do leitor a tarefa de estender os algoritmos das técnicas de aproximação por interpolação e splines cúbicas para funções, para que também aproximem caminhos não funcionais.

3.5 Aproximação de Bézier

Todas as técnicas de aproximação abordadas nesse capítulo são fundamentadas no paradigma do CONTROLE GLOBAL, isto é, qualquer alteração em parte da aproximadora só é possível alterando-a no todo. Isto pode causar alto custo e pouca versatilidade nos desenhos de figuras, esboços de formas, etc.

No início da década de 1970, foi proposta por Bézier uma técnica de aproximação fundamentada no paradigma do CONTROLE LOCAL, e que consiste em:

1º) Tomar $n+1$ pontos de controle $p_i = (x_i, y_i), i = 0, 1, \ldots, n$ do caminho e expressá-los na forma parametrizada $p_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} x_i(t) \\ y_i(t) \end{bmatrix}, t \in [0; 1]$;

2º) Para cada grupo de $k+1$ pontos, escolhidos de acordo com a conveniência do usuário, obter o aproximador polinomial de Bézier de grau k, definido por:

$$B_k(t) = \sum_{i=0}^{k} C_k^i (i-t)^{k-i} * t^i * p_i, \quad onde \quad C_k^i = \frac{k!}{(k-i)! * i!} \quad e \quad t \in [0; 1]$$

Ex.1: Obter a expressão dos polinômios $B_2(t)$ e $B_3(t)$

Solução:
Desenvolvendo $B_k(t)$ para $k = 2$, resulta em:

$$B_2(t) = 1*(1-t)^2*p_0 + 2*(1-t)*t*p_1 + t^2*p_2 \simeq \begin{cases} BX_2(t) = (1-t)^2 * x_0 + 2*(1-t)*t*x_1 + t^2*x_2 \\ BY_2(t) = (1-t)^2 * y_0 + 2*(1-t)*t*y_1 + t^2*y_2 \end{cases}$$

Desenvolvendo $B_k(t)$ para $k = 3$, resulta em:

$$B_3(t) = (1-t)^3 * p_0 + 3*(1-t)^2 * t * p_1 + 3*(1-t)*t^2*p_2 + t^3*p_3 \simeq$$

$$\begin{cases} B_3(t) = (1-t)^3 * x_0 + 3*(1-t)^2 * t * x_1 + 3*(1-t)*t^2*x_2 + t^3*x_3 \\ B_3(t) = (1-t)^3 * y_0 + 3*(1-t)^2 * t * y_1 + 3*(1-t)*t^2*y_2 + t^3*y_3 \end{cases}$$

Geometricamente, o gráfico de $B_k(t)$ é uma curva parametrizada com as seguintes propriedades:

Teorema 3.2

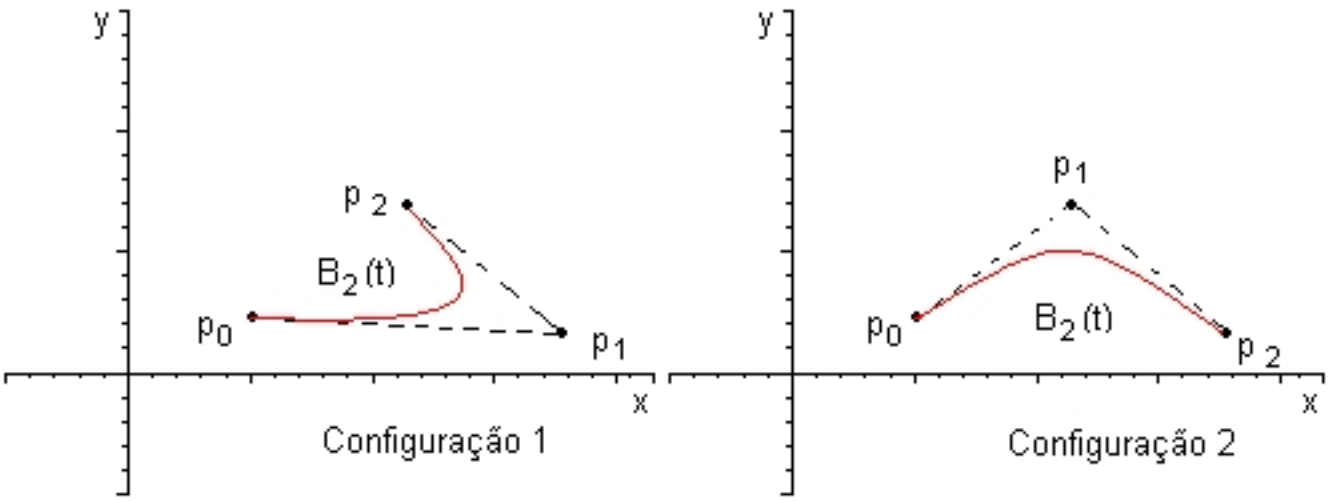
$$B_k(0) = \begin{cases} BX_k(0) = x_0 \\ BY_k(0) = y_0 \end{cases} \quad e \quad B_k(1) = \begin{cases} BX_k(1) = X_k \\ BY_k(1) = Y_k \end{cases}$$

Note-se que esta propriedade, facilmente verificável, assegura que o aproximador $B_k(t)$ passa pelo primeiro e pelo último dos pontos de referência do caminho.

Teorema 3.3 *A reta tangente a $B_k(t)$ em $t = 0$ é a reta definida pelos pontos p_0 e p_1. A reta tangente a $B_k(t)$ em $t = 1$ é a reta definida pelos pontos p_{k-1} e p_k.*

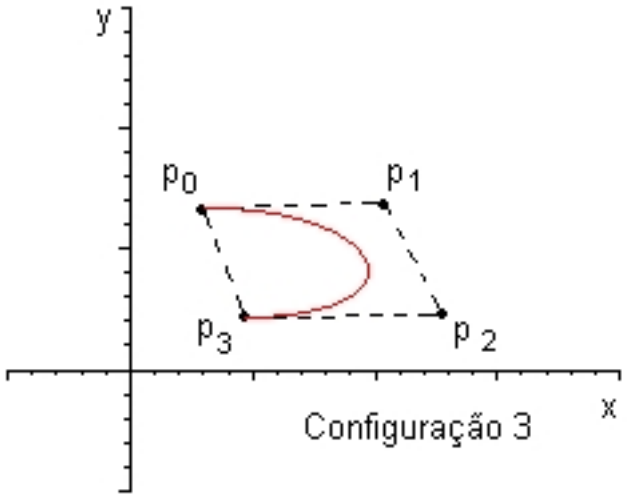
Ex.2: Tomando $k+1 = 3$ esboçar dois gráficos de $B_2(t)$, alterando os seus pontos de controle.

Solução:



Teorema 3.4 *Toda $B_k(t)$ está sempre contida no menor polígono convexo definido pelos $k + 1$ pontos.*

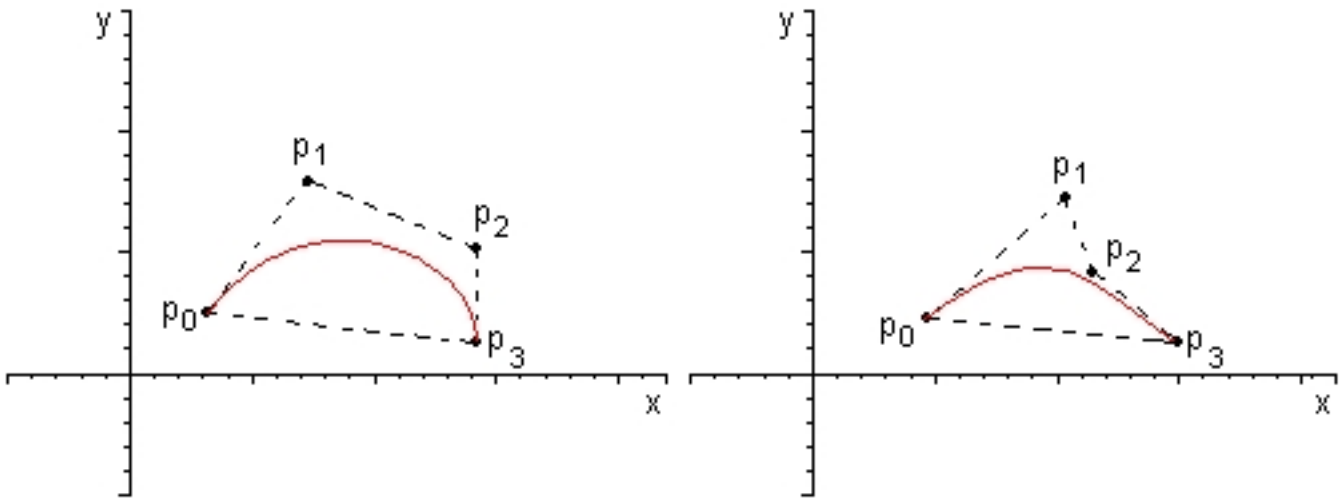
Ex.3: A $B_3(t)$ definida pelos quatro pontos da configuração 3 está contida no quadrilátero.

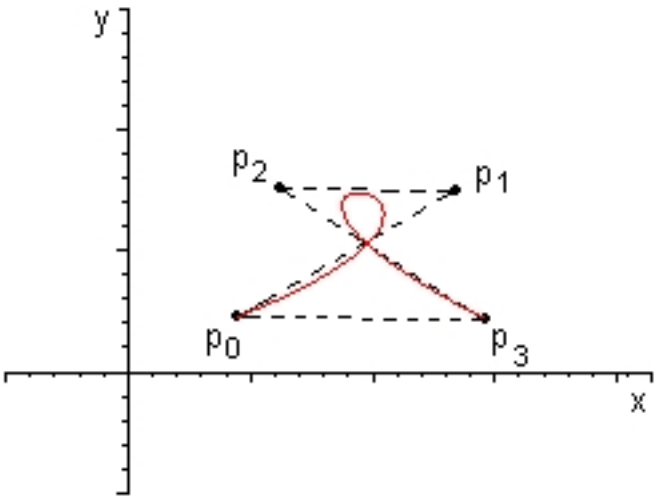


Teorema 3.5 *A $B_k(t)$ não passa necessariamente pelos pontos intermediários $p_i, i = 1, 2 \ldots, k-1$. Tais pontos são denominados de pontos de controle.*

Ex.4: Dados quatro pontos $p_i, i = 0, \ldots, 4$, apresente três curvas de Bézier reconfigurando os pontos de controle p_1 e p_2.

Solução:





Ex.5: Elabore um algoritmo eficiente para obter valores de $B_k(t_j), t_j \in [0;1]$

Solução:
Fica a cargo do leitor.

Capítulo 4

Integração Numérica

4.1 Fundamentos

Neste capítulo será utilizada a teoria da aproximação de funções para efetuar via processamento numérico uma operação clássica do Cálculo Diferencial e Integral, que é a de obter $I = \int_a^b f(x)dx$, $I = \int_{a_1}^{b_1}\int_{a_2}^{b_2} f(x_1, x_2)dx_2dx_1$, ..., $I = \int_{a_1}^{b_1}\int_{a_2}^{b_2}\ldots\int_{a_k}^{b_k} f(x_1, x_2, \ldots, x_k)dx_k \ldots dx_1$.

Por definição, obter $I = \int_a^b f(x)dx$ é uma operação que pode ser efetivada através de duas formas distintas:

1ª) $\overline{I} = \int f(x)dx$ equivale a obter uma $F(x) \mid F'(x) = f(x)$. Daí, $\overline{I} = \int f(x)dx = F(x) + C$, onde a $F(x)$ é denominada de primitiva ou anti-derivada da integranda $y = f(x)$;

2ª) $I = \int_a^b f(x)dx$ consiste em:

2.1 - Dividir $[a, b]$ em n partes de comprimento h_i

2.2 - Efetuar $S_n = \sum_{i=1}^{n} f(x_i) * h_i$

2.3 - Daí, $I = \int_a^b f(x)dx = \lim_{n\to\infty} S_n$

Note-se que nesta segunda forma, $I = \int_a^b f(x)dx$ representa a área subentendida por $y = f(x)$, $x = a$, $x = b$ e $y = 0$ e que pode ser efetivada construtivamente, enquanto que a primeira é essencialmente abstrativa.

A conexão entre ambas as definições de $I = \int_a^b f(x)dx$ é feita pelo Teorema Fundamental do Cálculo (TFC), do qual tem-se que:

Teorema 4.1 *TFC - " A área I subentendida por $y = f(x)$, $x = a$, $x = b$ e $y = 0$ é exatamente $I = \int_a^b f(x)dx = F(b) - F(a)$, onde $F(x)$ é a primitiva de y=f(x)."*

Entretanto, a aplicação generalizada deste Teorema em uma $I = \int_a^b f(x)dx$ qualquer pode ser difícil ou impossível, uma vez que:

1º) A $y = f(x)$ pode ser uma função discreta (= tabela);

2º) A $y = f(x)$ pode ter primitiva com expressão difícil de ser utilizada.

Ex.1: $\int_a^b x^m\sqrt{x^2+u^2}dx = x^{m-1} * \frac{\sqrt{(x^2+u^2)^3}}{m+\frac{3}{2}} - \frac{u^2*(m-1)}{m+\frac{3}{2}} * \int_a^b x^{m-2}\sqrt{x^2+u^2}dx$

3º) A $y = f(x)$ pode ter primitiva com expressão desconhecida.

Ex.2: $I = \int_1^5 e^{-x^2}dx$; $I = \int_2^4 \sqrt{\ln x}dx$; ...

Definio 4.1 *Um método de integração numérica fundamenta-se na aplicação do TFC, não na integranda $y = f(x)$, mas em aproximadora(s) $g(x)$ para a $f(x)$, $x \in [a, b]$.*

Fundamentalmente, os métodos de integração numérica podem ser agrupados em duas famílias básicas: NEWTONIANOS ou GAUSSIANOS.

4.2 Métodos de Newton para $I = \int_a^b f(x)dx$

4.2.1 Método dos Trapézios

Para obter $I = \int_a^b f(x)dx$ proceder:

- Dividir $[a, b]$ em n partes iguais de comprimento $h = \frac{b-a}{n}$;
- Obter os $n + 1$ pontos (x_i, y_i) onde $y_i = f(x_i)$, $x_1 = a$ e $x_{i+1} = x_i + h$;
- Para cada par de pontos sucessivos (x_i, y_i) e (x_{i+1}, y_{i+1}), obter o seu interpolador de NEWTON com Δ (Capítulo II)

$$p_i(x) = y_i + \frac{y_{i+1} - y_i}{h} * (x - x_i)$$

- Em cada $p_i(x)$ aplicar o TFC para obter $T_i = \int_{x_i}^{x_{i+1}} p_i(x)dx = \frac{y_{i+1}+y_i}{2} * h \simeq \int_{x_i}^{x_{i+1}} f(x)dx$

Daí, $\int_a^b f(x)dx \simeq \sum_{i=1}^{n} T_i$

$$\simeq \left(\frac{y_1+y_2}{2}\right) * h + \left(\frac{y_2+y_3}{2}\right) * h + \ldots + \left(\frac{y_n+y_{n+1}}{2}\right) * h$$

$$\simeq \frac{h}{2} * \left[y_1 + 2 * \sum_{i=2}^{n} y_i + y_{n+1}\right] = \tau_n \qquad (4.1)$$

Teorema 4.2 *"Em* $I = \int_a^b f(x)dx$*, se a* $f(x)$ *for contínua em* $[a,b]$ *então* $\lim_{n\to\infty} \tau_n = I$*."*

Ex.3: Efetue $I = \int_1^2 \ln x dx$ por Trapézios com $n = 4$, $n = 8$, $n = 10$.

Solução:
Aplicando (4.1) e o Teorema 4.2 resulta:

$n = 4 \Rightarrow h = 0.25 \Rightarrow \tau_4 = \frac{0.25}{4}\left[0+2*(0.223+0.405+0.56)+0.693\right] = 0.3836$

$n = 8 \Rightarrow h = 0.125 \Rightarrow \tau_8 = 0.3856$

$n = 10 \Rightarrow h = 0.1 \Rightarrow \tau_{10} = 0.3859$

Daí, tem-se que $I = \int_1^2 \ln x dx \simeq 0.3858$.

Ex.4: Efetue $I = \int_2^4 \frac{dx}{x \ln x}$ por Trapézios.

Solução:
Aplicando o Teorema 4.2, resulta:

$n = 2^3 = 8 \Rightarrow \tau_8 = 0.70943262...$

$n = 2^{20} = 1048576 \Rightarrow \tau = 0.69314718056$

$n = 2^{22} = 4194304 \Rightarrow \tau = 0.693147180559940$

$n = 2^{23} = 8388608 \Rightarrow \tau = 0.693147180559908$

Daí, tem-se que $I = \int_2^4 \frac{dx}{x \ln x} \simeq 0.693147180559908$.

Note-se que, para efeito de comparação, o valor "exato"desta integral é $I = 0.693147180559945$, valor este que está mais próximo do resultado obtido com $n = 2^{22}$ do que o fornecido com $n = 2^{23}$, contradizendo aparentemente o Teorema 4.2!

Análise do método dos Trapézios

1ª) Algoritmização

A primeira questão relacionada à obtenção de $I = \int_a^b f(x)dx$ via Trapézios, refere-se à algoritmização do Teorema 4.2. Para tanto, vamos elaborar um algoritmo que efetue NT tentativas de aproximação à integral sempre usando hp (próximo h) como sendo $hp = \frac{hv}{2}$ (hv = h anterior). Deixamos a cargo do leitor a tarefa de justificar os prós e contras desta estratégia.

Algoritmo 11 Trapézios

Entrada: $a, b, NT, f(x)(= integranda)$
Sada: I (aproximanda)

$hv = \frac{b-a}{2}$
$n = 2$
$E = f(a) + f(b)$
$meio = f(a + hv)$
$INT(1) = \left(\frac{hv}{2}\right) * (E + 2 * meio)$
para $k = 2$ até NT **faa**
  $hp = \frac{hv}{2}$
  $xx = a + hp$
  $meio = meio + f(xx)$
  para $i = 1$ até $n - 1$ **faa**
    $xx = xx + hv$
    $meio = meio + f(xx)$
  fim para
  $INT(k) = (E + 2 * meio) * \frac{hp}{2}$
  $n = n + n$
  $hv = hp$
fim para
FIM

2ª) Controle da instabilidade

Pelo comportamento apresentado no Ex.4, nota-se que a aplicação pura e simples do Teorema 4.2 para efetuar $I = \int_a^b f(x)dx$ padece de instabilidade numérica. Esta instabilidade deve-se ao fato de que quando o n cresce o erro de truncamento diminui; porém os erros de arredondamento podem acumular-se de forma a sobrepujar o ganho com a diminuição dos erros de truncamento deturpando o resultado final. Por conseqüência, na implementação do Algoritmo Trapézios tem-se que determinar um n ótimo resultante da precisão esperada para a I.

Tal n ótimo pode ser obtido através de duas vias distintas:

a) Teorema do valor médio para integrais

Efetuando-se $I = \int_a^b f(x)dx$ com $h = \frac{b-a}{n}$, pode-se obter do Teorema do valor médio que o erro de truncamento resultante é $E_t = -\frac{h^2 * (b-a) * f''(\varepsilon)}{12}$, $\varepsilon \in [a,b]$. Daí usando-se o majorante $M = máx_{x\in[a,b]} |f''(x)| \Rightarrow$

$$|E_t| \leqslant \frac{h^2 * (b-a) * M}{12} \tag{4.2}$$

Portanto, se for estipulado o E_t, de (4.2) pode-se obter o h e por conseqüência o n ótimo.

Ex.5: Obter o n ótimo para aproximar por Trapézios $I = \int_1^3 \frac{dx}{\sqrt{1+x}}$, na precisão $\varepsilon = 10^{-10}$.

Solução:

De $f(x) = (\sqrt{1+x})^{-1} \Rightarrow f''(x) = \frac{3}{4} * (1+x)^{-\frac{5}{2}} \Rightarrow M = máx_{x\in[1;3]} |f''(x)| = f''(1) \Rightarrow M = 0.132582521$.

Aplicando em (4.2) resulta:

$$10^{-10} \simeq \frac{h^2 * 2 * 0.132582521}{12} \Rightarrow h \simeq 6.727 * 10^{-5}.$$

Logo $n \simeq 29730 \Rightarrow n = 32000$.

Atente-se para as dificuldades de se algoritmizar esta forma de se obter o n ótimo.

b) Extrapolação ao Limite de Romberg

Esta técnica consiste em:

- Efetuar NT aproximações de I por Trapézios com $h_p = \frac{h_v}{2}$, resultando do Algoritmo as $INT(k), k = 1, 2, \ldots, NT$;
- Considerar cada $INT(k) = R_{k0}, k = 1, 2, \ldots, NT$ e obter as

$$R_{kj} = \frac{4^j * R_{k+1,j-1} - R_{kj-1}}{4^j - 1}, \quad j = 1, 2, \ldots, NT-1; \quad k = 1, 2, \ldots, NT-j; \tag{4.3}$$

Teorema 4.3 *(Romberg)* "$\lim_{j\to\infty} R_{1j} = \int_a^b f(x)dx$ *se* $y = f(x)$ *for contínua em* $[a,b]$"

Note-se que se $NT = 4$ a (4.3) resulta na matriz triangular:

R_{10}			
R_{20}	R_{11}		
R_{30}	R_{21}	R_{12}	
R_{40}	R_{31}	R_{22}	R_{13}

Ex.6: Efetue $I = \int_2^4 \frac{dx}{x \ln x}$ por Trapézios/Romberg com $NT = 3$.

Solução:

Aplicando (4.1) com $n = 2 \Rightarrow I \simeq 0,754255275$; com $n = 4 \Rightarrow I \simeq 0,709432628$ e com $n = 8 \Rightarrow I \simeq 0,697301126$.

Aplicando (4.3) resulta em:

0,754255275		
0,709432628	0,694491745	
0,697301126	0,693257292	0,693174995

e portanto $I \simeq 0,693174995$, a qual possui quatro dígitos exatos.

Algoritmo Trapézios/Romberg:

Fica a cargo do leitor a elaboração de um algoritmo para efetuar (4.3) até que $|R_{1j+1} - R_{1j}| \leq \varepsilon$, sendo ε a precisão do resultado da integral.

4.2.2 Método de Simpson

Para efetuar $I = \int_a^b f(x)dx$ proceder:

- Dividir $[a, b]$ em n(par) partes iguais de comprimento $h = \frac{b-a}{n}$;
- Obter os $n + 1$ pontos (x_i, y_i), onde $x_1 = a$, $x_{i+1} = x_i + h$ e $y_i = f(x_i)$;
- Para cada três pontos sucessivos (x_{i-1}, y_{i-1}), (x_i, y_i) e (x_{i+1}, y_{i+1}) determinar o seu polinômio interpolador de Newton com $\Delta \Rightarrow p(x) = y_{i-1} + (\Delta y_{i-1})(x - x_{i-1}) + (\Delta^2 y_{i-1})(x - x_{i-1})(x - x_i)$;
- Aplicar o TFC para obter $\int_{x_{i-1}}^{x_i} p(x)dx = \frac{h}{3} * [y_{i-1} + 4 * y_i + y_{i+1}] \simeq \int_{x_{i-1}}^{x_i} f(x)dx$

Daí, $\int_a^b f(x)dx \simeq \int_{x_1}^{x_3} p(x)dx + \int_{x_3}^{x_5} p(x)dx + \ldots + \int_{x_{n-1}}^{x_{n+1}} p(x)dx$

$$\simeq \frac{h}{3} * [y_1 + 4 * y_2 + y_3] + \frac{h}{3} * [y_3 + 4 * y_4 + y_5] + \ldots + \frac{h}{3} * [y_{n-1} + 4 * y_n + y_{n+1}]$$

$$\simeq \frac{h}{3} * [y_1 + 4 * \sum_{i=2}^{n,2} y_i + \sum_{i=3}^{n-1,2} y_i + y_{n+1}] = S_n \tag{4.4}$$

Teorema 4.4 *"Se $y = f(x)$ for contínua em $[a, b]$, então $\lim_{n\to\infty} S_n = I$."*

Note-se que a diferença entre o Trapézios e o Simpson é apenas na aproximação via segmentos lineares (retas) no primeiro e aproximação via segmentos parabólicos no segundo.

Ex.7: - Efetue $I = \int_1^2 \ln x dx$ por Simpson com $n = 4$ e $n = 8$.

Solução:
Aplicando (4.4) resulta em:
$n = 4 \Rightarrow h = 0,25 \Rightarrow \int_1^2 \ln x dx \simeq \frac{0,25}{3} * [\ln 1 + 4 * (\ln 1,25 + \ln 1,75) + 2 * \ln 1,5 + \ln 2] \simeq 0,386259562.$
$n = 8 \Rightarrow h = 0,125 \Rightarrow \int_1^2 \ln x dx \simeq \frac{0,125}{3} * [\ln 1 + 4 * (\ln 1,125 + \ln 1,375 + \ln 1,625 + \ln 1,875) + 2 * (\ln 1,25 + \ln 1,5 + \ln 1,75) + \ln 2] \simeq 0,386292043$

Logo a precisão do último resultado é $|S_8 - S_4| = 0,00003247$, tendo-se por conseqüência aproximadamente quatro dígitos da exata. Convida-se o leitor a obter diretamente pelo TFC o valor de $I = \int_1^2 \ln x dx$ e confirmar esta conclusão.

Análise do Método de Simpson

1ª) Algoritmização

De maneira semelhante e utilizando-se da mesma estratégia, a algoritmização do Teorema 4.4, resulta em:

Algoritmo 12 Simpson

Entrada: $a, b, NS, f(x)$
Sada: I
$hv = \frac{b-a}{4}$
$n = 4$
$E = f(a) + f(b)$
$SP = f(a + h) + f(a + 3 * hv)$
$SI = f(a + 2 * hv)$
$INT(1) = (EX + 4 * SP + 2 * SI) * \left(\frac{hv}{3}\right)$
para $k = 2$ até NS **faa**
$SI = SI + SP : hp = \frac{hv}{2}$
$c = a + hp : SP = f(c)$
para $i = 1$ até $n - 1$ **faa**
$c = c + hv$
$SP = SP + f(c)$
fim para
$n = n + n$
$hv = hp$
$INT(k) = (EX + 4 * SP + 2 * SI) * \left(\frac{hp}{3}\right)$
fim para
FIM

2ª) Controle da instabilidade

Pelo mesmos motivos já citados no Trapézios, a execução do algoritmo Simpson com $NS \to \infty$ pode gerar uma instabilidade numérica.

Como não existe um método de extrapolação ao limite para o Simpson, as alternativas de controle desta instabilidade são:

a) Teorema do valor médio para integrais

Efetuando-se $I = \int_a^b f(x)dx$ por Simpson, do Teorema do Valor Médio resulta que o erro de truncamento será:

$E_s = -\frac{h^4 * (b-a) * f^{iv}(\varepsilon)}{180}$, $\varepsilon \in [a,b] \Rightarrow$

$$|E_s| \leq \frac{h^4 * M * (b-a)}{180}, M = máx_{x\in[a,b]} \left|f^{iv}(x)\right| \tag{4.5}$$

.

Ex.8: Obter o n ótimo para aproximar $I = \int_1^3 \frac{dx}{\sqrt{1+x}}$ por Simpson na precisão $\varepsilon = 10^{-10}$.

Solução:

De $f(x) = (1+x)^{\frac{-1}{2}} \Rightarrow f^{iv}(x) = \frac{105}{16} * (1+x)^{-\frac{9}{2}} \Rightarrow M = máx_{x\in[1;3]} \left|f^{iv}(x)\right| = f^{iv}(1) = 0,044194173$.

Aplicando em (4.5) resulta:

$$10^{-10} \simeq \frac{h^4 * 2 * 0,044194173}{180} \Rightarrow h \simeq 0,021243 \Rightarrow h = 0,02 \Rightarrow n = 100.$$

b) Simpson Adaptativo

Aqui usa-se uma busca em profundidade para minimizar a quantidade de operações a serem efetuadas, através da seguinte técnica adaptativa:

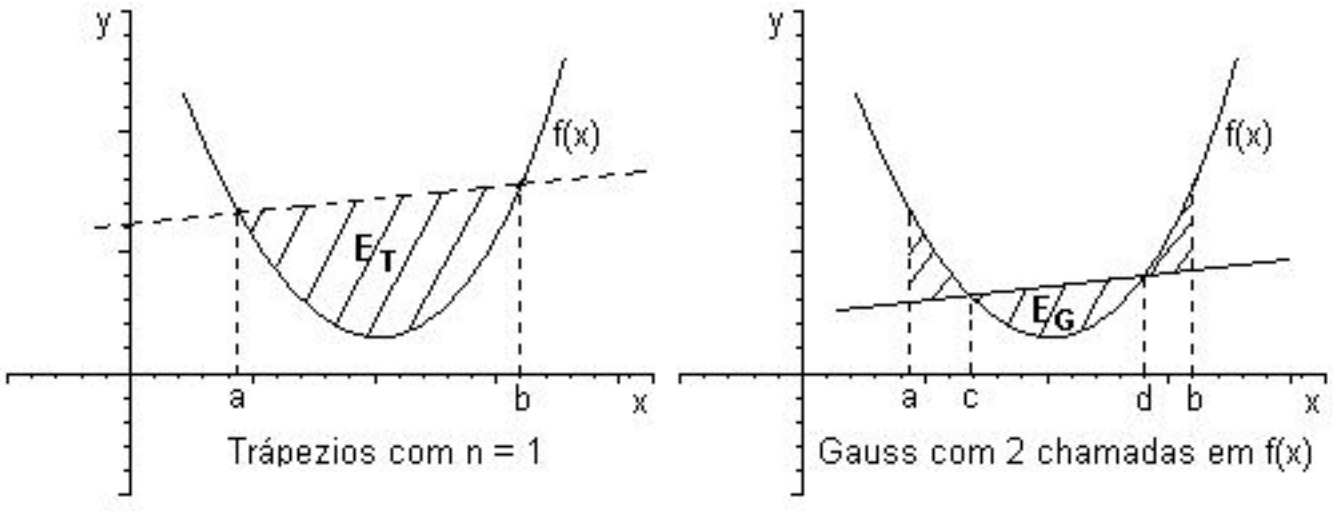
- Dividir $[a,b]$ em k partes convenientes de comprimento $h_i = x_{i+1} - x_i$;
- Em cada parte $[x_i; x_{i+1}]$ fazer:
  - Obter $P_i = \frac{h_i}{6} * \left[f(x_i) + 4 * f\left(x_i + \frac{h_i}{2}\right) + f(x_i + h_i)\right]$ $\left(\text{Simpson com } h = \frac{h_i}{2}\right)$
  - Obter $Q_i = \frac{h_i}{12} * \left\{f(x_i) + 4 * \left[f\left(x_i + \frac{h_i}{4}\right) + f\left(x_i + \frac{3h_i}{4}\right)\right] + 2 * f\left(x_i + \frac{h_i}{2}\right) + f(x_i + h_i)\right\}$ $\left(\text{Simpson com } h = \frac{h_i}{4}\right)$
  - Se $|P_i - Q_i| < \frac{16 * h_i * \varepsilon}{b-a}$, então $\int_{x_i}^{x_{i+1}} f(x)dx \simeq Q_i$;

- Senão, fazer $h_i = \dfrac{h_i}{2}$ e aplicar o mesmo procedimento em $[x_i; \overline{x_i}]$ e $[\overline{x_i}; x_{i+1}]$

4.3 Método de Gauss para $I = \displaystyle\int_a^b f(x)dx$

Ao investigar as causas dos erros de truncamento dos métodos de Newton, Gauss concluiu que os mesmos poderiam ser diminuídos sem aumento do custo desde que o intervalo de integração $[a, b]$ não fosse dividido em partes iguais, conforme preconizado pelos métodos de Newton.

Ex.9: Na configuração abaixo tem-se que:



Note-se que em cada uma duas configurações acima foram efetuadas duas chamadas $y = f(x)$. Entretanto $E_G < E_T$. Utilizando-se deste fundamento, Gauss desenvolveu um método de integração numérica no qual os valores de $y = f(x)$ são obtidos em pontos previamente definidos e que independem de $[a, b]$.

Para efetuar $I = \displaystyle\int_a^b f(x)dx$ por Gauss com m chamadas em $f(x)$, procede-se:

1º) Padronizar o $[a, b]$ transformando-o sempre em $[-1; 1]$. Isto é, algebricamente efetivado através da mudança de variáveis:

$x = \dfrac{b-a}{2} * t + \dfrac{b+a}{2} \Rightarrow dx = \dfrac{b-a}{2}dt \Rightarrow I = \displaystyle\int_a^b f(x)dx = \frac{b-a}{2} * \int_{-1}^{1} g(t)dt,$

onde $g(t) = f\left(\dfrac{b-a}{2} * t + \dfrac{b+a}{2}\right)$;

2º) Efetuar $\overline{I} = \displaystyle\int_{-1}^{1} g(t)dt = \sum_{i=1}^{m} a_i * g(t_i) + E_G$, onde

i) $E_G = \dfrac{2^{2m-1} * (m!)^4 * g^{(2m)}(\varepsilon)}{(2m+1) * [(2m)!]^3}$, $\varepsilon \in [-1;1]$

Teorema 4.5 *"Se $g(t)$ for contínua em $[-1;1]$, então $\lim_{m\to\infty} \sum_{i=1}^{m} a_i * g(t_i) = \overline{I}$, isto é, $E_G = 0$"*

Daí,

$$I = \int_a^b f(x)dx \simeq \frac{b-a}{2} * \overline{I} \tag{4.6}$$

ii) Os parâmetros $a_i, t_i, i = 1, 2, \ldots, m$ são fixos para cada um dos m e independem da integranda. Como tais parâmetros são fixos, não necessita-se obtê-los, mas apenas copiá-los na forma de um arquivo. A seguir, apresentam-se os a_i, t_i para $m = 2, 3$ e 4. Os demais podem ser obtidos na bibliografia recomendada.

m	a_i	t_i
2	$a_1 = a_2 = 1$	$-t_1 = t_2 = 0,577350269$
3	$a_1 = a_2 = 0,555555555$ $a_3 = 0,888888889$	$-t_1 = t_2 = 0,774596667$ $t_3 = 0$
4	$a_1 = a_4 = 0,34785484$ $a_2 = a_3 = 0,65214516$	$-t_1 = t_4 = 0,86113631$ $-t_2 = t_3 = 0,33998104$

Ex.10: Aproximar $I = \int_1^2 lnxdx$ por Gauss com $m = 3$.

Solução:

Aplicando (4.6) resulta em:

$x = \dfrac{b-a}{2} * t + \dfrac{b+a}{2} \Rightarrow x = 0,5 * t + 1,5 \Rightarrow dx = 0,5dt$

$g(t) = ln(0,5 * t + 1,5)$. Daí,

$\int_1^2 lnxdx \simeq 0,5*[0,555...g(-0,774...)+0,555...g(0,774...)+0,888...g(0)]$

$\int_1^2 lnxdx \simeq 0,38630042.$

Ex.11: Aproximar $I = \int_1^2 \sqrt{lnx}dx$.

Solução:

Aplicando (4.6) resulta em:

$m = 2 \Rightarrow I = 0,60019878$

$m = 3 \Rightarrow I = 0,59515774$

$m = 4 \Rightarrow I = 0,59376474$

1ª) Algoritmização

A seguir apresenta-se um algoritmo para efetuar uma única aproximação de $I = \int_a^b f(x)dx$ por Gauss. Na entrada dos dados, o usuário fornecerá apenas a, b e $y = f(x)$, ficando por conta do algoritmo as transformações requeridas pelo método.

Algoritmo 13 Gauss

Entrada: $a, b, m, f(x)$
Sada: I
Ler, $((t_{mi}, a_{mi}), i = 1, 2, \ldots, m)$
$aux1 = \dfrac{(a+b)}{2} : aux2 = \dfrac{a-b}{2}$
$S = 0$
para $i = 1$ até m **faa**
$xx = aux1 + aux2 * t_{mi}$
$S = S + a_{mi} * f(xx)$
fim para
$I = aux2 * S$
FIM

2ª) Pela expressão algébrica do erro de truncamento do método de Gauss, nota-se que $E_G = 0$ se $y = f(x)$ for um polinômio de grau $n \leq 2m - 1$;

3ª) Devido às suas características, Gauss só é aplicável se $y = f(x)$ tiver expressão conhecida (não for uma tabela) e fornece resultados ruins se $y = f(x)$ possuir descontinuidades em (a, b);

4ª) Gauss é de natureza intrinsecamente aberta, isto é, a integranda $y = f(x)$ não é obtida nos extremos a e b. Conseqüências:

i) Pode-se efetuar integrais impróprias do tipo $I = \int_a^b f(x)dx$ com descontinuidades em a ou em b.

Ex.12: Efetuar $I = \int_0^1 \ln x dx$ por Gauss com $m = 3$

Solução:
Aplicando (4.6) resulta:
$I = \int_0^1 \ln x dx \simeq 0,5 * [0,555\ldots g(-0,774\ldots) + 0,555\ldots g(0,774\ldots) + 0,888\ldots g(0)] \simeq -0,94767237.$

ii) Pode-se efetuar integrais impróprias do tipo $\int_a^\infty f(x)dx$. Basta utilizar a mudança de variáveis $x = \dfrac{1}{z} \Rightarrow dx = -\dfrac{dz}{z2} \Rightarrow$

$I = \int_a^{\infty} f(x)dx = -\int_{\frac{1}{a}}^{0} \frac{f(\frac{1}{z})dz}{z2} = \int_0^{\frac{1}{a}} \frac{f(\frac{1}{z})}{z2}dz$, que possui descontinuidade no extremo inferior.

Convida-se o leitor a comprovar algebricamente que

$I = \int_{-\infty}^{\infty} f(x)dx = \int_{-c}^{\infty} f(-x)dx + \int_c^{\infty} f(x)dx$ e a algoritmizar estas integrais impróprias.

4.4 Integração Múltipla

Como na interpolação, para se efetuar uma integral dupla, tripla, ... basta estender os métodos de integração para integrais simples. Novamente o preço a ser pago será um crescimento exponencial no tempo de processamento. Por exemplo, para se efetuar a integral dupla $I = \int_a^b \int_c^d f(x,y)dydx$ pelos métodos de:

a) Simpson

1°) Dividir $[a,b]$ em n partes iguais $\Rightarrow h = \frac{b-a}{n}$ e $[c,d]$ em $\overline{n}$ partes iguais $\Rightarrow \overline{h} = \frac{d-c}{\overline{n}}$; ($n$ e $\overline{n}$ devem ser pares)

2°) Aplicar a extensão

$$I = \int_a^b \left[\int_c^d f(x,y)dy\right]dx \simeq$$

$$\int_a^b \left\{\frac{\overline{h}}{3} * \left[f(x,c) + 4 * \sum_{i=2}^{\overline{n},2} f(x,y_i) + 2 * \sum_{i=3}^{\overline{n}-1,2} f(x,y_i) + f(x,d)\right]\right\}dx \simeq$$

$$\frac{\overline{h}}{3} * \left[\int_a^b f(x,c)dx + 4 * \sum \int_a^b f(x,y_i)dx + 2\sum \int_a^b f(x,y_i)dx + \int_a^b f(x,d)dx\right]$$

Note-se que uma integral dupla equivale à $\overline{n}+1$ integrais simples.

b) Gauss

1°) Transformar $[a,b]$ em $[-1,1] \Rightarrow x = \frac{b-a}{2} * t + \frac{b+a}{2}$ e $dx = \frac{b-a}{2}dt$ e $[c,d]$ em $[-1;1] \Rightarrow y = \frac{d-c}{2} * \overline{t} + \frac{d+c}{2}$ e $dy = \frac{d-c}{2}d\overline{t}$

2°) Aplicar a extensão

$$I = \int_a^b \int_c^d f(x,y)dydx = \frac{b-a}{2} * \frac{d-c}{2} \int_{-1}^{1} \left[g(t,\overline{t})d\overline{t}\right]dt$$

$$\simeq \frac{b-a}{2} * \frac{d-c}{2} * \int_{-1}^{1} \left[\sum_{i=1}^{m} a_{im} g(t,t_{mi})\right]dt$$

$$\simeq \frac{b-a}{2} * \frac{d-c}{2} * \left[\sum_{i=1}^{m} a_{im} \int_{-1}^{1} g(t,t_{mi})dt\right]$$

$$\simeq \frac{b-a}{2} * \frac{d-c}{2} * \left[\sum_{i=1}^{m} a_{im} \sum_{j=1}^{m} a_{jm} g(t_{mj},t_{mi})\right]$$

Algoritmo 14 Gauss2

Entrada: $a, b, c, d, m, f(x, y)$

Sada: I

Ler, $((t_{mi}, a_{mi}), i = 1, 2, \ldots, m)$

$aux1 = \dfrac{b-a}{2}$

$aux2 = \dfrac{b+a}{2}$

$auy1 = \dfrac{d-c}{2}$

$auy2 = \dfrac{d+c}{2}$

$S = 0$

para $i = 1$ até m **faa**

$SS = 0$

$xx = aux1 * t_{mi} + aux2$

para $j = 1$ até m **faa**

$yy = auy1 * t_{mj} + auy2$

$SS = SS + a_{mj} * f(xx, yy)$

fim para

$S = S + a_{mi} * SS$

fim para

$I = aux1 * auy1 * S$

FIM

Capítulo 5

Aproximação de funções

Até o presente momento, foram estudadas três técnicas de aproximação de uma $y = f(x), x \in [a, b]$ tais que:

1º] Conhecia-se ou interessavam apenas alguns dos valores funcionais da $y = f(x)$; isto é, a aproximanda era discretizada na forma $\begin{array}{c|c|c|c|c} x & x_0 & x_1 & \dots & x_n \\ \hline y & y_0 & y_1 & \dots & y_n \end{array}$

2º] Tomavam-se alguns ou todos os pontos discretos como controle ou referência para se obter a aproximadora $z = g(x), x \in [a, b]$.

Dando continuidade ao estudo da teoria da aproximação de funções, aqui serão abordadas mais três técnicas, caracterizadas por:

1ª] Tem-se uma $y = f(x)$ com expressão conhecida e deve-se obter uma aproximadora $z = g(x)$ também com expressão conhecida e que seja mais simples de ser utilizada. Este tópico possui larga aplicação na construção de funções pré-definidas para uso em sistemas dedicados;

2ª] Coleta-se uma base de dados em determinado experimento científico no qual os valores amostrais obtidos podem estar afetados por erros inerentes (observacionais, calibração de equipamentos, etc), e deseja-se obter uma função $z = g(x)$ que seja imune a estes erros e que capte a tendência do fenômeno em estudo;

3ª] Dispõe-se de imagens coletadas, sinais recebidos, etc, os quais são deturpados por ineficiência dos equipamentos, ruídos de transmissão, etc e deseja-se reconstruí-los e/ou reconhecer o seu padrão de comportamento.

5.1 Aproximação de funções com expressão conhecida

Se uma $y = f(x), x \in [a, b]$ possuir expressão conhecida porém difícil ou ineficiente de ser utilizada, então deve-se encontrar uma aproximadora $z = g(x)$, a qual idealmente possua as seguintes características:

a) Os erros $E(\gamma) = |f(\gamma) - g(\gamma)|$ devem ser mínimos, uniformemente distribuídos $\forall \gamma \in [a, b]$ e possíveis de ser mensurados;

b) O tempo de resposta nos cálculos de $g(\beta) \simeq f(\beta)$ devem ser mínimos;

c) O custo de armazenamento dos parâmetros da $z = g(x)$ deve ser mínimo.

Note-se que nestas condições a aproximação de $y = f(x)$ via técnica da interporlação polinomial, já abordada no capítulo anterior, não seria adequada por não satisfazer a pelo menos dois dos quesitos da aproximadora ideal.

Ex.1 Para aproximar a $f(x) = ln\sqrt[5]{1+x}, x \in [0, 2]$ via um polinômio interpolador $p_n(x)$ com precisão $\varepsilon = 10^{-8}$, seria necessário que o $n = 40 \Rightarrow 41$ parâmetros a serem armazenados e 80 operações em cada cálculo de $f(\beta) \simeq p_{40}(\beta)$.

Para fins de padronização de uso e facilidade na delimitação da precisão da aproximadora $z = g(x)$, aqui o domínio $[a, b]$ será fixo e utilizado sempre o $[-1; 1]$. Isto é possível utilizando-se a seguinte mudança de variáveis:

$$x = \frac{b-a}{2} * t + \frac{b+a}{2} \Rightarrow \begin{cases} t = 1 \Rightarrow x = a \\ t = -1 \Rightarrow x = b \end{cases}$$

$$t = \frac{2x}{b-a} + \frac{b+a}{b-a} \Rightarrow \begin{cases} x = a \Rightarrow t = -1 \\ x = b \Rightarrow t = 1 \end{cases} \tag{5.1}$$

Ex.2 Em $f(x) = ln(x+3), x \in [1, 11] \Rightarrow f(5) = ln8$.

Efetuando-se a mudança do domínio via (5.1) resulta:
$x = 5t + 6 \Rightarrow \overline{f}(t) = ln(5t + 9)$.
Daí, se $x = 5 \Rightarrow t = -0.2 \Rightarrow$
$\overline{f}(t) = \overline{f}(-0.2) = ln8$

5.1.1 Aproximação com séries de Taylor

Pelo teorema de Taylor, se $y = f(x)$ for continuamente diferenciável em $[a, b]$ e $\beta \in [a, b]$, então:

$$f(x) = f(\beta) + \frac{f'(\beta)(x-\beta)}{1!} + \frac{f''(\beta)(x-\beta)^2}{2!} + \ldots + \frac{f^n(\beta)(x-\beta)^n}{n!} + \ldots \tag{5.2}$$

Daí, pelo teorema do Resto, tem-se que:

$$f(x) = f(\beta) + \frac{f'(\beta)(x-\beta)}{1!} + \frac{f''(\beta)(x-\beta)^2}{2!} + \ldots + \frac{f^n(\beta)(x-\beta)^n}{n!} + \frac{f^{n+1}(\varepsilon)(x-\beta)^{n+1}}{(n+1)!}, \varepsilon \in [\beta, x) \tag{5.3}$$

Note-se que de (5.3) resulta:

1º] Os $n+1$ primeiros termos desta série truncada, geram um polinômio $p_n(x)$ de grau n;

2º] O último termo é denominado de resto e será representado por $R_n(x) = \frac{f^{n+1}(\varepsilon)(x-\beta)^{n+1}}{(n+1)!}$. Daí $f(x) = p_n(x) + R_n(x)$, resultando portanto em uma aproximação polinomial se $R_n(x) \to 0$;

3º] Se o $\beta \in [a, b]$ for fixado em $\beta = 0$, de (5.3), resulta:

$$f(x) = f(0) + \frac{f'(0)x}{1!} + \frac{f''(0)x^2}{2!} + \ldots + \frac{f^n(0)x^n}{n!} + \frac{f^{n+1}(\varepsilon)x^{n+1}}{(n+1)!}, \varepsilon \in [0, x) \quad (5.4)$$

que é a denominada série de Maclaurin da $y = f(x)$

Como o assunto de séries de Taylor já foi abordado no Cálculo C, aqui apenas faremos uso do mesmo sem ater-se nas questões de obtenção da série, testes de convergência, delimitação do resto, etc.

Ex.3 Expandir em série de Taylor/Maclaurin as seguintes funções:

i) $f(x) = e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!} + \ldots$

ii) $f(x) = cosx = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots + (-1)^n * \frac{x^{2n}}{(2n)!} + \ldots$

iii) $f(x) = ln(1 + x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \ldots + (-1)^{n+1} * \frac{x^n}{n} + \ldots$

iv) $f(x) = \frac{1}{\sqrt{e^x}} = 1 - \frac{x}{2 * 1!} + \frac{x^2}{2^2 * 2!} - \frac{x^3}{2^3 * 3!} + \ldots + (-1)^{n+1} * \frac{x^n}{2^n * n!} + \ldots$

v) $f(x) = \int_0^x e^{-y^2} dy = x - \frac{x^3}{1 * 3} + \frac{x^5}{2! * 5} - \frac{x^7}{3! * 7} + (-1)^n * \frac{x^{2n+1}}{n! * (2n + 1)} + \ldots$

vi) $f(x) = \int_0^x cos\sqrt{y} dy = x - \frac{x^2}{2 * 2!} + \frac{x^3}{3 * 4!} - \frac{x^5}{5 * 6!} + \frac{x^7}{7 * 8!} - \ldots$

vii) $f(x) = \int_0^x \frac{sent}{t} dt = x - \frac{x^3}{3 * 3!} + \frac{x^5}{5 * 5!} - \frac{x^7}{7 * 7!} + \ldots + \frac{(-1)^n * x^{2n+1}}{(2n + 1) * (2n + 1)!} + \ldots$

Desta forma, a aproximação de uma função com expressão conhecida via série de Taylor/Maclaurin, consiste em:

1º] Obter a sua série;

2º] Delimitar o resto $R_n(x)$ em função da precisão ε desejada;

3º] Truncar a série em função do resto obtido, resultando em um polinômio aproximador $p_n(x) \simeq f(x)$.

Sempre é bom lembrar que a delimitação do resto $R_n(x)$ poderá ser efetivada através do majorante, artifício já utilizado duas vezes nesta disciplina. Mais ainda, do Cálculo C, sabe-se que: *"Se a série de Taylor/Maclaurin for convergente e alternada nos sinais, então* $R_n(x) \simeq \dfrac{f^{(n+1)}(\beta) * (x-\beta)^{n+1}}{(n+1)!}$*, isto é, o resto é aproximadamente o primeiro termo da série abandonado no truncamento".*

Ex.4 Determine o grau do aproximador de Taylor/Maclaurin que assegure precisão $\varepsilon = 10^{-8}$ para as seguintes funções:

a) $f(x) = e^x, x \in [-1, 1]$

Solução:

Pelo resto da série de Maclaurin, tem-se que $R_n(x) = \dfrac{f^{(n+1)}(\varepsilon) * x^{n+1}}{(n+1)!} \Rightarrow 10^{-8} \simeq \dfrac{e^{\varepsilon} * x^{n+1}}{(n+1)!}$.

Tomando-se $\varepsilon = 1$ e $x = 1$ (funções crescentes) resulta que:

$10^{-8} \simeq \dfrac{e^1 * 1^{n+1}}{(n+1)!} \Rightarrow (n+1)! \simeq 2,71828 * 10^8 \Rightarrow n = 11$

b) $f(x) = \dfrac{1}{\sqrt{e^x}}, x \in [-1; 1]$

Solução:

Como a série de $\dfrac{1}{\sqrt{e^x}}$ é convergente e alternada nos sinais, então:

$R_n(x) \simeq (-1)^{n+2} * \dfrac{x^{n+1}}{2^{n+1} * (n+1)!} \Rightarrow 10^{-8} \simeq \dfrac{1}{2^{n+1} * (n+1)!} \Rightarrow n = 9$

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

Solução:

$ln\sqrt{1+x} = \dfrac{x}{2} - \dfrac{x^2}{2*2} + \dfrac{x^3}{2*3} - \dfrac{x^4}{2*4} + \ldots + (-1)^{n+1} * \dfrac{x^n}{2*n} + \ldots$

Como a série é convergente e alternada nos sinais, então:

$R_n(x) \simeq (-1)^{n+2} * \dfrac{x^{n+1}}{2*(n+1)} \Rightarrow 10^{-8} \simeq \dfrac{1}{2*(n+1)} \Rightarrow n = \dfrac{1}{2} * 10^8 \Rightarrow n =$ 50.000.000

Análise da Aproximação de Taylor/Maclaurin

Pelos resultados advindos dos exemplos das aproximações por Taylor/Maclaurin e comparando-se com as condições de aproximadora ideal de $y = f(x)$, tem-se que:

a) Custo de Armazenamento

Este custo é ideal por ser nulo, uma vez que apenas segue-se a lei de formação da série;

b) Distribuição dos Erros de Truncamento

Neste quesito esta aproximação não cumpre a condição de aproximadora ideal, pois se o valor a ser estimado for igual ao $\beta \Rightarrow$ erro nulo. Porém, na medida em que este valor se distancia do β seu erro aumenta. Por exemplo, para a $f(x) = e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \ldots + (-1)^n * \frac{x^n}{n!} + \ldots, x \in [-1, 1]$ e $n = 4 \Rightarrow R_n(x) \simeq 0,0083, \forall x \in [-1; 1]$. Contudo, estes erros estão distribuídos conforme a tabela a seguir:

x	-1	$-0,5$	0	$0,5$	1
$e^{-x} - p_4(x)$	$0,0099$	$0,00028$	0	$0,00024$	$0,0071$

c) Custo das Chamadas

Pelos resultados do Ex.4, nota-se que para certas aproximandas o custo pode ser baixo, para outras regular e para algumas este custo pode ser proibitivo tornando a técnica inútil (Ex.4 c).

5.1.2 Aproximação com Polinômios de Tchebyshev

Teorema 5.1 *Um polinômio de Tchebyshev de grau n é toda expressão algébrica do tipo* $T_n(x) = cos(narcosx), x \in [-1; 1]$.

Para se comprovar que $T_n(x)$ é efetivamente um polinômio, note-se que como $arcosx = \Theta(\hat{a}ngulo), \Theta \in [-\pi; \pi] \Rightarrow x = cos\Theta$. Daí, $T_n(x) = cos(n\Theta)$. Desenvolvendo-se esta expressão e fazendo uso de identidades trigonométricas elementares, resulta em:

$$\begin{cases} T_0(x) = cos0 = 1 \\ T_1(x) = cos\Theta = x \\ T_2(x) = cos2\Theta = 2cos^2\Theta - 1 = 2x^2 - 1 \\ T_3(x) = cos3\Theta = 4x^3 - 3x \\ T_4(x) = cos4\Theta = 8x^4 - 8x^2 + 1 \\ T_5(x) = cos5\Theta = 16x^5 - 20x^3 + 5x \\ \vdots \end{cases} \tag{5.5}$$

Para uso posterior, isolando-se cada potência de x^i no respectivo $T_i(x)$ em (5.5), resulta em:

$$\begin{cases} x^0 = T_0 \\ x^1 = T_1 \\ x^2 = \frac{T_2 + T_0}{2} \\ x^3 = \frac{T_3 + 3T_1}{4} \\ x^4 = \frac{T_4 + 4T_2 + 3T_0}{8} \\ x^5 = \frac{T_5 + 5T_3 + 10T_1}{16} \\ \vdots \end{cases} \tag{5.6}$$

Os $T_n(x)$ são funções polinomiais com as seguintes propriedades:

Prop1: "$T_n(x)$ é um polinômio de grau n. Só existe um único $T_n(x)$ para cada n e o coeficiente de x^n é sempre igual a 2^{n-1}."

Prop2: "As expressões de $T_n(x), n \geq 2$ de (5.5) podem ser obtidas da fórmula recursiva $T_{n+1}(x) = 2x * T_n(x) - T_{n-1}(x)$."

Prop3: "$\forall x \in [-1;1] \Rightarrow |T_n(x)| \leq 1$. O máximo máx $|T_n(x)| = 1$ e $\forall$raiz α_k de $T_n(x) = 0 \Rightarrow \alpha_k = cos\left\{\dfrac{[2k-1]*\pi}{2n}\right\}$, $k = 1, 2, \ldots, n$."

Prop4: "Sejam $\overline{T}_n(x) = 2^{1-n} * T_n(x)$ e $\overline{p}_n(x) = a_n^{-1} p_n(x)$ os respectivos monômios de Tchebyshev e dos polinômios quaisquer. Então $2^{1-n} = máx\left|\overline{T}_n(x)\right| \leq máx\left|\overline{p}_n(x)\right|, \forall x \in [-1;1]$. A igualdade só ocorre se $\overline{p}_n(x) = \overline{T}_n(x)$."

Para aproximar $y = f(x), x \in [-1;1]$, procede-se como segue:

1º] Obter um aproximador polinomial $p_n(x)$ para a $y = f(x)$ na precisão ε desejada. Daí,

$$f(x) = p_n(x) + ET_1 \Rightarrow f(x) = a_0 + a_1 x + \ldots + a_n x^n + E_t \quad (1);$$

2º] Substituir os $x^i, i = 0, 1, \ldots, n$ de (1) pelas respectivas expressões de (5.6) e agrupá-los. Daí,

$$f(T) = b_0 + b_1 T_1 + b_2 T_2 + \ldots + b_n T_n + ET_1 \quad (2), \quad onde \quad b_n = \frac{a_n}{2^{n-1}};$$

3º] Truncar (2) a partir de $b_{k+1}T_{k+1}, k < n \Rightarrow$

$$f(T) = b_0 + b_1 T_1 + \ldots + b_k T_k + (ET_1 + ET_2) \quad (3) \quad de \quad modo \quad que \quad (\mid ET_1 \mid + \mid ET_2 \mid) \leq \varepsilon;$$

4º] Substituir em (3) os T_i pelas correspondentes expressões em x^i de (5.5) e agrupá-los. Daí

$$f(x) \simeq c_0 + c_1 x + \ldots + c_k x^k, \quad k < n \qquad (5.7)$$

O polinômio $p_k(x) = c_0 + c_1 x + \ldots + c_k x^k$ é denominado de aproximador de Tchebyshev da $y = f(x)$.

Ex.5 Aproximar a $f(x) = \dfrac{1}{\sqrt{e^x}}, x \in [-1;1]$ por Tchebyshev de grau $k = 3$, partindo do Taylor/Maclaurin de grau $n = 4$

Solução:

1º] Do Taylor/Maclaurin tem-se que:

$$f(x) = \frac{1}{\sqrt{e^x}} = 1 - \frac{x}{2*1!} + \frac{x^2}{2^2*2!} - \frac{x^3}{2^3*3!} + \left.\frac{x^4}{2^4*4!}\right| + ET_1, \quad onde \quad ET_1 \simeq \frac{1}{2^5*5!} = 0,00026$$

2º] De (5.6) resulta:

$$\frac{1}{\sqrt{e^x}} \simeq 1 + \frac{T_1}{2} + \left(\frac{T_2 + T_0}{2*8}\right) - \left(\frac{T_3 + 3T_1}{4*48}\right) + \left(\frac{T_4 + 4T_2 + 3T_0}{8*384}\right)$$

$$\simeq \left(T_0 + \frac{T_0}{16} + \frac{3T_0}{3072}\right) - \left(\frac{T_1}{2} + \frac{3T_1}{192}\right) + \left(\frac{T_2}{16} + \frac{4T_2}{3072}\right) - \frac{T_3}{192} + \frac{T_4}{3072}$$

3º] Como $T_4 \leq 1$ e $\dfrac{T_4}{3072} = ET_2 = 0,00032$

$$\frac{1}{\sqrt{e^x}} \simeq \frac{3267T_0}{3072} - \frac{99T_1}{192} + \frac{196T_2}{3072} - \frac{T_3}{192} + (ET_1 + ET_2),$$

onde $ET_1 + ET_2 \simeq 0,00026 + 0,00032 = 0,00058$

4º]

$$\frac{1}{\sqrt{e^x}} \simeq \frac{3267}{3072} - \frac{99x}{192} + \frac{196(2x^2-1)}{3072} - \frac{(4x^3-3x)}{192} \simeq \frac{3071}{3072} - \frac{x}{2} + \frac{49x^2}{384} - \frac{x^3}{48}$$

Daí $f(x) = \dfrac{1}{\sqrt{e^x}} \simeq \dfrac{3071}{3072} - \dfrac{x}{2} + \dfrac{49x^2}{384} - \dfrac{x^3}{48}$ com precisão $\varepsilon \simeq 0,00058$

Análise da Aproximação de Tchebyshev

1º] Distribuição dos Erros

Tomando-se como referência a aproximação do Ex.5 e obtendo a tabela de distribuição dos erros de truncamento em $[-1;1]$, resulta:

x	-1	$-0,5$	0	$0,5$	1
$\left\|\frac{1}{\sqrt{e^x}} - p_3(x)\right\|$	$0,00071$	$0,00015$	$0,00032$	$0,00017$	$0,00008$

Atente-se para a distribuição uniforme dos erros no domínio. Este fenômeno é característico desta aproximação e extensível para todas as aproximandas.

2º] Tempo de Resposta

O tempo de resposta, ou grau k do aproximador de Tchebyshev com precisão ε será "sempre menor"que o do equivalente de Taylor/Maclaurin. Isto é uma consequência da *Prop1.* Esta vantagem é denominada de Efeito Telescópico. Tal efeito torna-se tanto mais acentuado, quanto mais lenta

for a convergência da série inicial de $y = f(x)$. Por exemplo, para a $f(x) = arctgx, x \in [-1; 1]$ tem-se que:

$f(x) = arctgx = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \ldots$

Truncando-a para obter aproximador de Taylor/Maclaurin de grau $n = 7$ resulta em:

Truncando-a para obter aproximador de Taylor/Maclaurin de grau $n = 7$ resulta em:

$arctgx \simeq x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7}$ com precisão $\varepsilon \simeq \frac{1}{9} = 0,111$

Obtendo o seu aproximador de Tchebyshev de grau $k = 3$ a partir do grau $n = 7$ (efeito telescópico de j=4), resulta em:

$f(x) = arctgx \simeq x - \frac{91x^3}{336}$, onde $|p_3(1) - arctg1| = 0,0562$, com precisão destacadamente melhor que o de Taylor/Maclaurin de grau $n = 7$

3º] Pelo exemplo anterior, verifica-se que pode ser trabalhosa a determinação do aproximador de Tchebyshev, especialmente quando a série for lenta, caso em que esta aproximação é mais vantajosa. Daí a necessidade de uma técnica mais eficiente de geração de coeficientes. Tal técnica existe e os coeficientes b_i do polinômio do passo 2º] podem ser gerados "diretamente"via:

$$b_0 = \frac{1}{\pi}\int_{-1}^{1} \frac{f(x)dx}{\sqrt{1-x^2}} \quad e \quad b_i = \frac{2}{\pi}\int_{-1}^{1} \frac{T_i(x)f(x)dx}{\sqrt{1-x^2}}, i = 1, 2, \ldots, k$$

O problema é que tais integrandas não têm primitivas conhecidas. Por consequência, elas só podem ser obtidas via integração numérica.

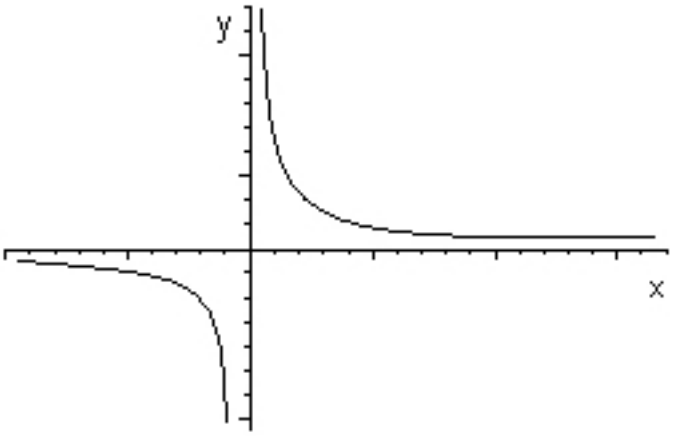
5.1.3 Aproximação racional de Padé

Apesar de todas as vantagens, já destacadas, do uso de aproximadores polinomiais, esta família de funções tem pelo menos uma desvantagem.

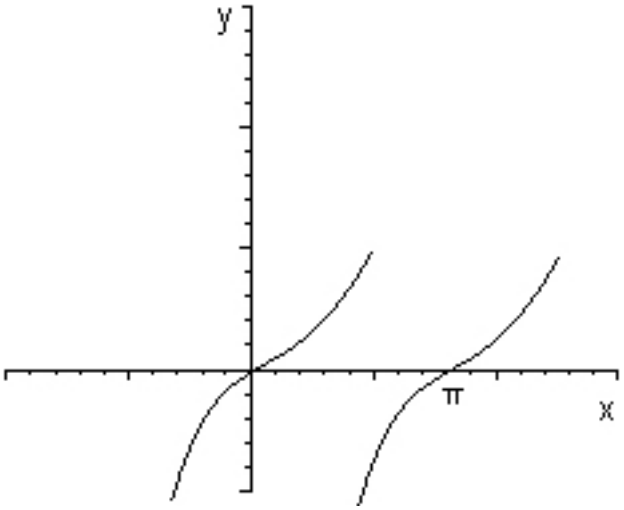
O gráfico de $z = p_n(x), n > 1$ oscila. Por consequência para funções $y = f(x)$ que sejam assíntotas o grau n do aproximador será elevado.

Por exemplo, esboçando os gráficos de $y = \frac{1}{x}$ e $y = tanx$, resulta:

a) $f(x) = \frac{1}{x}$



b) $f(x) = tanx$



Teorema 5.2 *Uma função racional de grau $M = m + n$ é toda expressão do tipo $R_{nm}(x) = \dfrac{p_n(x)}{q_m(x)}$, onde $p_n(x)$ e $q_m(x)$ são polinômios.*

Teorema 5.3 *A aproximação racional de Padé consiste em se obter uma aproximadora racional $R_{nm}(x)$ para a $y = f(x), x \in [-1; 1]$, tal que:*
$f^k(0) = R^k_{nm}(0), k = 0, 1, 2, \ldots, M$.

Note-se que esta aproximação é uma extensão da de Taylor/Maclaurin e para efetuá-la procede-se como segue:

1º] Obter o aproximador de Maclaurin de grau $M = m + n$ para a $y = f(x)$. Daí $f(x) \simeq c_0 + c_1x + c_2x^2 + \ldots + c_Mx^M$;

2º] Tomar $p_n(x) = a_0+a_1x+\ldots+a_nx^n$ e $q_m(x) = 1+b_1x+\ldots+b_mx^m$ e considerar que $f(x) \simeq c_0 + c_1x + c_2x^2 + \ldots + c_Mx^M = \dfrac{a_0 + a_1x + \ldots + a_nx^n}{1 + b_1x + \ldots + b_mx^m}(1)$;

3º] Em (1) aplicando-se a condição de aproximação $f^k(0) = R^k_{nm}(0)$, e explicitando-se os coeficientes desconhecidos, resulta em:

$$\begin{cases} a_0 = c_0 \\ a_1 = c_1 + b_1 c_0 \\ a_2 = c_2 + b_1 c_1 + b_2 c_0 \\ \vdots \\ a_n = c_n + b_1 c_{n-1} + \ldots + b_n c_0 \end{cases} \tag{5.8}$$

e

$$\begin{bmatrix} c_{n-m+1} & c_{n-m+2} & \ldots & c_n \\ c_{n-m+2} & c_{n-m+3} & \ldots & c_{n+1} \\ c_{n-m+3} & c_{n-m+4} & \ldots & c_{n+2} \\ \vdots & & & \vdots \\ c_n & c_{n+1} & \ldots & c_{n+m-1} \end{bmatrix} * \begin{bmatrix} b_m \\ b_{m-1} \\ b_{m-2} \\ \vdots \\ b_1 \end{bmatrix} = - \begin{bmatrix} c_{n+1} \\ c_{n+2} \\ c_{n+3} \\ \vdots \\ c_{n+m} \end{bmatrix} \tag{5.9}$$

Note-se que o (5.9) é um sistema simétrico de ordem mxm. Caso em (5.8) e/ou (5.9) ocorra situações em que b_k e/ou c_k possuam índice $k < 0$, então nestes casos considerar $b_k = 0$ e/ou $c_k = 0$.

Ex.6 Aproximar a $f(x) = ln\sqrt[5]{1+x}, x \in [0;1]$ via Padé $R_{32}(x)$. Avaliar a precisão do resultado obtido.

Solução:

Expandindo a $ln\sqrt[5]{1+x}$ em série de Maclaurin, resulta: $ln\sqrt[5]{1+x} = \frac{x}{5*1} - \frac{x^2}{5*2} + \frac{x^3}{5*3} - \frac{x^4}{5*4} + \ldots + (-1)^{n+1} * \frac{x^n}{5*n}$.

Truncando esta série para obter um polinômio aproximador de grau $M = n + m = 3 + 2 = 5$, obtém-se:

$$ln\sqrt[5]{1+x} \simeq \frac{x}{5} - \frac{x^2}{10} + \frac{x^3}{15} - \frac{x^4}{20} + \frac{x^5}{25} \Rightarrow \begin{cases} c_0 = 0 \\ c_1 = \frac{1}{5} \\ c_2 = \frac{-1}{10} \\ c_3 = \frac{1}{15} \\ c_4 = \frac{-1}{20} \\ c_5 = \frac{1}{25} \end{cases}$$

Tomando $R_{32}(x) = \frac{a_0 + a_1 x + a_2 x^2 + a_3 x^3}{1 + b_1 x + b_2 x^2}$ e aplicando (5.9) resulta:

$$\begin{bmatrix} c_2 & c_3 \\ c_3 & c_4 \end{bmatrix} * \begin{bmatrix} b_2 \\ b_1 \end{bmatrix} = - \begin{bmatrix} c_4 \\ c_5 \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{-1}{10} & \frac{1}{15} \\ \frac{1}{15} & \frac{-1}{20} \end{bmatrix} * \begin{bmatrix} b_2 \\ b_1 \end{bmatrix} = \begin{bmatrix} \frac{1}{20} \\ \frac{-1}{25} \end{bmatrix} \Rightarrow \begin{cases} b_2 = \frac{3}{10} \\ b_1 = \frac{6}{5} \end{cases}$$

Aplicando (5.8) obtém-se

$$\begin{cases} a_0 = c_0 = 0 \\ a_1 = c_1 + b_1 c_0 = \dfrac{1}{5} \\ a_2 = c_2 + b_1 c_1 + b_2 c_0 = \dfrac{7}{50} \\ a_3 = c_3 + b_1 c_2 + b_2 c_1 + b_3 c_0 = \dfrac{1}{150} \end{cases}$$

Daí, $R_{32}(x) = \dfrac{\frac{x}{5} + \frac{7x^2}{50} + \frac{x^3}{150}}{1 + \frac{6x}{5} + \frac{3x^2}{10}}$ e $ln\sqrt[5]{1+x} \simeq \dfrac{30x + 21x^2 + x^3}{150 + 180x + 45x^2}$

Testando a precisão do $R_{32}(x)$ obtido por comparação com a exata, resulta que:

x	0	0,5	1
$R_{32}(x) - f(x)$	0	0,0000015	0,000037

Note-se que o custo de armazenamento desta $R_{32}(x)$ é de 5 parâmetros naturais e o custo de cada chamada da aproximanda é de 9 operações básicas, para se obter uma precisão mínima de $\varepsilon = 0,000037$. Esta mesma precisão só seria obtida num Maclaurin de grau n = 5045!

Análise da Aproximação de Padé

1º] Pelas características desta técnica, existem vários aproximadores de Padé de grau M para uma mesma $y = f(x)$. Basta combinar o n e m para se obter $M = n + m$. Por exemplo, para $f(x) = ln(1+x) \simeq x - \dfrac{x^2}{2} + \dfrac{x^3}{3} - \dfrac{x^4}{4}$.

Aqui todos os autores sugerem que se tome o $n = m$ ou $n = m + 1$, que são as situações de melhores resultados;

2º] Algoritmo Padé

A determinação dos coeficientes a_i e b_j de $R_{nm}(x)$ pode ser obtida via computar. Para os casos em que $n = m$ ou $n = m + 1$, tem-se:

Algoritmo 15 Aproximação de Padé

Entrada: $n, m, (c_i, i = 0, 1, \ldots, n + m)$
Sada: $(b_i, i = 1, m)$, $(a_j, j = 0, n)$
  Resolução equação (5.9) ↓
  $k = n - m$
  para $i = 1$ até m **faa**
    para $j = 1$ até i **faa**
      $a_{ij} = c_{k+i+j-1}$
      $a_{ji} = a_{ij}$
    fim para
    $a_{im+1} = -c_{n+i}$
  fim para
  $ExecuteLU(A, m|X)$
  para $i = 1$ até m **faa**
    $b_i = x_{m-i+1}$
  fim para
  Resolução equação (5.8) ↓
  $a_0 = c_0$
  para $i = 1$ até n **faa**
    $S = c_i$
    para $j = 1$ até i **faa**
      $S = S + b_j * c_{i-j}$
    fim para
    $a_i = S$
  fim para
  FIM

3º] Como Padé é uma extensão do Maclaurin, ele também não distribui uniformemente os erros de truncamento no domínio. Para sanar este problema, pode-se compor Padé com Tchebyshev via:

- Obter o passo 3º] do Tchebyshev de grau $M = n+m$ para a $y = f(x) \Rightarrow f(x) \simeq b_0T_0 + b_1T_1 + \ldots + b_MT_M$ (1);
- Aplicar (5.6) e (5.5) em (1) resultando em $f(x) \simeq \dfrac{\overline{a_0} + \overline{a_1}T_1 + \ldots + \overline{a_n}T_n}{T_0 + \overline{b_1}T_1 + \ldots + \overline{b_m}T_m}$ (2);
- Substituir os T_i de (2) pelas expressões em x^i, resultando em $f(x) \simeq R_{nm}(x) = \dfrac{a_0 + a_1x + \ldots + a_nx^n}{1 + b_1x + \ldots + b_mx^m}$

5.2 Aproximação pela Tendência

Na elaboração e análise de experimentos em geral quando se usa um método científico, o fluxo normal dos procedimentos a serem seguidos, consiste em:

1º]Coletar uma "grande"quantidade de dados amostrais e armazená-los. Caso seja considerada apenas uma variável influente, cada dado será representado algebricamente pelo par (x_k, y_k);

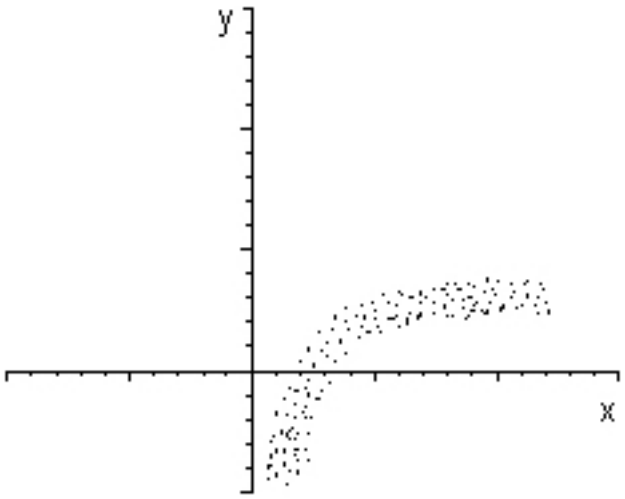
2º]Efetuar o tratamento destes dados, que tradicionalmente consiste em:

2.1) Expurgar ou minimizar a influência dos valores y_k que estejam afetados por erros inerentes (observacionais, descalibração de equipamentos, etc);

2.2) Plotar os pontos (x_k, y_k) gerando um gráfico formado pela nuvem de pontos que pode configurar uma função tendência do problema estudado;

2.3) Obter a função tendência representativa do problema.

Por exemplo, o experimento hipotético plotado abaixo configura uma tendência de fenômeno do tipo logarítmico.



Definio 5.1 *Para uma base de dados* $(x_k, y_k)_{k=1}^n$ *denomina-se função de ajuste a uma* $z = g(x)$ *com expressão conhecida e que mais se aproxime de todos os pontos da base, e não necessariamente que os contenha.*

Para se tentar quantificar algebricamente o "mais se aproxima de todos os pontos"da definição anterior, pode-se considerar o $d_k = g(x_k - y_k), k = 1, 2, \ldots, n$ como o desvio de cada ponto em relação à tendência do fenômeno e tentar obter a $z = g(x)$ de modo que:

a) $\sum_{k=1}^{n} d_k$ seja mínima.

Neste caso, pode-se obter ambigüidades devido ao sinal dos d_k e também não se pode perceber a $z = g(x)$ bem ajustada;

b) $\sum_{k=1}^{n} \mid d_k \mid$ seja mínima.

Neste caso, a ambigüidade devida ao sinal dos d_k desaparece, contudo ainda contínua a impossibilidade da percepção da $z = g(x)$ bem ajustada;

c) $\sum_{k=1}^{n} d_k^2$ seja mínima.

Finalmente, aqui desapareceu os sinais, enfatizam-se os grandes desvios, desprezam-se os pequenos desvios e consegue-se a $z = g(x)$ bem ajustada. Este será o critério que fundamentará este tipo de aproximação.

5.2.1 Ajuste com Polinômios Usando o Método do Mínimos Quadrados

Para se obter o polinômio $p_m(x) = a_0 + a_1x + \ldots + a_mx^m$ que melhor se ajusta a uma $\begin{array}{c|c|c|c|c} x & x_1 & x_2 & \ldots & x_n \\ \hline y & y_1 & y_2 & \ldots & x_n \end{array}$, tem-se que $\sum_{k=1}^{n} d^2k = \sum_{k=1}^{n}[p_m(x_k) - y_k]^2 = \sum_{k=1}^{n}[a_0 + a_1x_k + \ldots + a_mx_k^m - y_k]^2 = \varphi(a_0, a_1, \ldots, a_m)$. Agora basta determinar o mínimo global da $\varphi : \mathbb{R}^n \to \mathbb{R}$. Para tanto $\frac{\partial\varphi}{\partial a_0} = \frac{\partial\varphi}{\partial a_1} = \ldots = \frac{\partial\varphi}{\partial a_m} = 0$(pontos críticos). Daí efetuando as derivadas parciais, resulta:

$$\begin{cases} \frac{\partial\varphi}{\partial a_0} = 2\sum_{k=1}^{n}[a_0 + a_1x_k + \ldots + a_mx_m^k - y_k] * 1 = 0 \\ \frac{\partial\varphi}{\partial a_1} = 2\sum_{k=1}^{n}[a_0 + a_1x_k + \ldots + a_mx_m^k - y_k] * x_k = 0 \\ \vdots \\ \frac{\partial\varphi}{\partial a_m} = 2\sum_{k=1}^{n}[a_0 + a_1x_k + \ldots + a_mx_m^k - y_k] * x_k^m = 0 \end{cases}$$

expressão que desenvolvida e reescrita na forma matricial torna-se:

$$\begin{bmatrix} n & \sum_{k=1}^{n} x_k & \sum_{k=1}^{n} x_k^2 & \ldots & \sum_{k=1}^{n} x_k^m \\ \sum_{k=1}^{n} x_k & \sum_{k=1}^{n} x_k^2 & \sum_{k=1}^{n} x_k^3 & \ldots & \sum_{k=1}^{n} x_k^{m+1} \\ \vdots & & & & \vdots \\ \sum_{k=1}^{n} x_k^m & \sum_{k=1}^{n} x_k^{m+1} & \sum_{k=1}^{n} x_k^{m+2} & \ldots & \sum_{k=1}^{n} x_k^{2m} \end{bmatrix} * \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{n} y_k \\ \sum_{k=1}^{n} x_k y_k \\ \vdots \\ \sum_{k=1}^{n} x_k^m y_k \end{bmatrix} \tag{5.10}$$

Note-se que (5.10) é um sistema linear de ordem m+1 x m+1 cuja única (?) solução fornece a ponto mínimo de $\varphi(a_0, a_1, \ldots, a_m)$ e por conseqüência os coeficientes do melhor polinômio de ajuste. Convida-se o leitor a verificar que o grau m do polinômio de ajuste a uma base de dados com n valores, deve obrigatoriamente ser $m < n$.

Ex.7 Para $\begin{array}{c|c|c|c|c|c} x & 2 & 4 & 6 & 7 & 10 \\ \hline y = f(x) & 1 & 3 & 7 & 5 & 2 \end{array}$, obtenha a melhor reta e a melhor parábola de ajuste. Decida qual de ambas é a mais bem ajustada e estime $f(5)$ e $f(12)$.

Solução:

i) RETA $\Rightarrow m = 1 \Rightarrow p_1(x) = a_0 + a_1 x$. Daí aplicando (5.10), tem-se:

$$\begin{bmatrix} 5 & 29 \\ 29 & 205 \end{bmatrix} * \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 18 \\ 111 \end{bmatrix} \Rightarrow \begin{cases} a_0 = 2,56 \\ a_1 = 0,179 \end{cases} \Rightarrow p_1(x) = 2,56 + 0,179x;$$

ii) PARÁBOLA $\Rightarrow m = 2 \Rightarrow p_2(x)a_0 + a_1 x + a_2 x^2$. Aplicando (5.10), tem-se:

$$\begin{bmatrix} 5 & 29 & 205 \\ 29 & 205 & 1631 \\ 205 & 1631 & 13969 \end{bmatrix} * \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 18 \\ 111 \\ 749 \end{bmatrix} \Rightarrow \begin{cases} a_0 = -5,108 \\ a_1 = 3,393 \\ a_2 = -0,267 \end{cases}$$

$\Rightarrow p_2(x) = -5,108 + 3,393x - 0,267x^2$

Para se verificar qual de ambas é a melhor função de ajuste, tem-se que em $p_1(x) \Rightarrow \sum_{k=1}^{5} d^2k = 22,016$. Já em $p_2(x) \Rightarrow \sum_{k=1}^{5} d^2k = 3,756$. Portanto o melhor polinômio de ajuste entre os dois obtidos é a parábola. Daí, $f(5) \simeq p_2(5) = 5,182$ e $f(12) \simeq p_2(12) = -2,84$.

Análise do Ajuste Polinomial

1º] Como pode-se obter várias $z = g(x)$ ajustadas a uma mesma base de dados $\{(x_k, y_k)\}_{k=1}^{n}$ e o critério de decisão usado no exemplo anterior pode ser relativo, a qualidade da $z = g(x)$ pode ser melhor aferida obtendo-se o seu coeficiente de determinação r^2, onde $r^2 = \dfrac{\sum_{k=1}^{n}[g(x_k) - \overline{y}_m]^2}{\sum_{k=1}^{n} y_k - \overline{y}_m}$, onde $\overline{y}_m = \dfrac{\sum_{k=1}^{n} y_k}{n}$. Daí da estatística, tem-se que quanto mais próximo de 1 estiver o r^2, mais bem ajustada está a $z = g(x)$;

2º] O sistema linear (5.10) do ajuste a polinômios, além de simétrico e positivo definido, é também mal-condicionado (no Ex.7 testando o condicionamento da $U = \begin{bmatrix} 5 & 29 \\ 29 & 205 \end{bmatrix}$, tem-se que cond U $= \dfrac{detU}{\beta_1 \beta_2} = 0,0302!$). Por conseqüência, sugere-se o método de CHOLESKY para a resolução destes sistemas lineares;

3º]Algoritmo Ajuste a Polinomiais

Denotando por U a matriz aumentada de (5.7), ela pode ser gerada via:

Algoritmo 16 Ajuste a Polinomiais

Entrada: n, m, X, Y
Sada: $((a_i, i = 0, 1, , \ldots, m)$
  $S = 0$
  para $i = 1$ até n **faa**
    $v_i = 1$
    $S = S + y_i$
  fim para
  $S_0 = n$
  para $i = 1$ até $m + m$ **faa**
    $S_i = 0$
    para $k = 1$ até n **faa**
      $v_k = v_k * x_k$
      $S_i = S_i + v_k$
    fim para
  fim para
  para $i = 1$ até $m + 1$ **faa**
    para $j = 1$ até i **faa**
      $U_{ij} = S_{i+j-2}$
      $U_{ji} = U_{ij}$
    fim para
  fim para
  $U_{1,m+2} = S$
  para $i = 2$ até $m + 1$ **faa**
    S=0
    para $j = 1$ até n **faa**
      $y_j = y_j * x_j$
      $S = S + y_j$
    fim para
    $U_{i,m+2} = S$
  fim para
  $ExecuteCholesky(m, U|A)$
  FIM

5.2.2 Ajuste com não polinomiais usando Mínimos Quadrados

Quando a plotagem da base $\{(x_k, y_k)\}_{k=1}^{n}$ sugerir tendência não polinomial, tem-se duas alternativas para se efetuar o ajuste ao tipo de função desejada.

1º] Transformação

Através de artifícios algébricos, transformar a não polinomial em uma polinomial, resolver via (5.10) e após retornar à família de origem.

Ex.8

i) Ajuste a exponenciais $y = ab^x$
Solução:
De $y = ab^x \Rightarrow lny = lna + (lnb)x$, e se $lna = a_0$, $lnb = a_1$ e $z = lny$ $\Rightarrow z = a_0 + a_1x$, que resolvida por (5.10) fornece a_0 e a_1. Daí como $a_0 = lna \Rightarrow a = e^{a_0}$ e $a_1 = lnb \Rightarrow b = e^{a_1}$

ii) Ajuste a logarítmicas $y = ln(a + bx)$
Solução:
De $y = ln(a + bx) \Rightarrow e^y = a + bx$, e se $z = e^y, a = a_0 eb = a_1 \Rightarrow z = a_0 + a_1x$, que resolvida por (5.10) fornece a_0 e a_1. Daí como $a = a_0$ e $b = a_1$.

iii) Ajuste a geométricas $y = ax^b, b \in \mathbb{R}$
Solução:
De $y = ax^b \Rightarrow lny = lna + blnx$, e se $lny = z, lna = a_0, b = a_1$ e $lnx = t \Rightarrow z = a_0 + a_1t$, que resolvida por (5.10) fornece a_0 e a_1. Daí como $a_0 = lna \Rightarrow a = e^{a_0}$ e $b = a_1$

2º] Dedução

Quando não for possível transformar em polinomial, tem-se que deduzir o sistema resultante da aplicação de $\sum_{k=1}^{n} d^2k = \sum_{k=1}^{n}[g(x_k) - y_k]^2$, conforme feito no ítem 5.2.1.

Ex.9 Ajustar uma base de dados $\{(x_k, y_k)\}_{k=1}^{n}$ a uma função logarítmica do tipo $y = alnbx$

Solução:

De $\sum_{k=1}^{n} d^2k = \sum_{k=1}^{n}[g(x_k) - y_k]^2 = \sum_{k=1}^{n}[(alnbx_k) - y_k]^2 = \varphi(a, b)$

Minimizando a $\varphi(a, b) \Rightarrow$

$$\begin{cases} \dfrac{\delta\varphi}{\delta a} = \displaystyle\sum_{k=1}^{n} 2 * [(alnbx_k) - y_k] * (lnbx_k) = 0 \\ \dfrac{\delta\varphi}{\delta b} = \displaystyle\sum_{k=1}^{n} 2 * [(alnbx_k) - y_k] * \dfrac{a}{b} = 0 \end{cases}$$

Daí tem-se:

$$\begin{cases} a\displaystyle\sum_{k=1}^{n}(lnbx_k)^2 - \sum_{k=1}^{n}(y_k lnbx_k) = 0 \\ a\displaystyle\sum_{k=1}^{n} lnbx_k = \sum_{k=1}^{n} y_k \end{cases}$$

que é um sistema não linear de ordem 2x2, cuja solução fornece os parâmetros da $y = alnbx$ desejada.

5.3 Aproximação pelo padrão de comportamento via séries de Fourier

Neste item serão apresentados os fundamentos da teoria de reconhecimento de padrões (voz, íris, imagens, etc). Esta teoria possui ampla aplicação em diversos sistemas computacionais.

Uma função $y = f(x)$ é dita periódica se existir uma constante $\tau \in \mathbb{R}$ tal que $f(x+\tau) = f(x)$. Por exemplo, a $f(x) = senx$ é uma função periódica com $\tau = 2\pi$.

Definio 5.2 *Um polinômio trigonométrico de grau m é toda expressão do tipo:* $P_m(x) = \dfrac{a_0}{2} + \displaystyle\sum_{k=1}^{m-1}[a_k coskx + b_k senk_x] + a_m cosmx$

Por exemplo, a função $P_2(x) = \dfrac{a_0}{2} + a_1 cosx + b_1 senx + a_2 cosx$ é um polinômio trigonométrico genérico de grau $m = 2$

Definio 5.3 *Para uma função* $y = f(x), x \in [-\pi; \pi]$ *com expressão conhecida, sua série de Fourier é a função* $y(x) = \lim\limits_{m\to\infty} P_m(x) = \dfrac{a_0}{2} + \displaystyle\sum_{k=1}^{\infty}[a_k coskx + b_k senkx]$, *onde os coeficientes do polinômio trigonométrico são obtidos via:*

$$\begin{cases} a_k = \dfrac{1}{\sqrt{\pi}} \displaystyle\int_{-\pi}^{\pi} (f(x)coskx)dx, k = 0, 1, \ldots \\ b_k = \dfrac{1}{\sqrt{\pi}} \displaystyle\int_{-\pi}^{\pi} (f(x)senkx)dx, k = 1, 2, \ldots \end{cases} \tag{5.11}$$

Ex.10 Obter a série de Fourier de $f(x) = \mid x \mid, x \in [-\pi; \pi]$.

Solução:
Aplicando (5.11) resulta em:

$$f(x) =\mid x \mid= \frac{\pi}{2} + \frac{2}{\pi} * \sum_{k=1}^{\infty} \frac{[(-1)^k - 1]}{k^2} * coskx \tag{5.12}$$

Convida-se o leitor a verificar que o comportamento do gráfico dos $P_m(x), m \to \infty$, é $\lim_{m\to\infty} P_m(x) =\mid x \mid$ em (5.12).

Entretanto, o problema é que na prática não se dispõe da $y = f(x)$, mas apenas de pontos discretos (pixels) das imagens, dos gráficos de voz, etc. Daí para captar o padrão de comportamento das mesmas, pode-se ajustá-los por Mínimos Quadrados a um polinômio trigonométrico $P_m(x)$.

Definio 5.4 *Para uma base de dados com* $2n$ *elementos* $\{(x_j, y_j)\}_{j=0}^{2n-1}, x_j \in [-\pi; \pi]$, *seu ajuste de Fourier consiste em se tomar um* $P_m(x) = \frac{a_0}{2} + \sum_{k=1}^{m}[a_k coskx + b_k senkx] + a_m cosmx$, *aplicar-lhe os Mínimos Quadrados, resultando de (5.10) que:*

$$\begin{cases} a_k = \frac{1}{n} \sum_{j=0}^{2n-1} y_j coskx_j, k = 0, 1, \ldots, m \\ b_k = \frac{1}{n} \sum_{j=0}^{2n-1} y_j senkx_j, k = 1, 2, \ldots, m-1 \end{cases} \tag{5.13}$$

Este $P_m(x)$ obtido via (5.13) representa o padrão de comportamento do problema em estudo, sendo a base da teoria de reconhecimento de padrões, flitros digitais, etc.

Ex.11 Ajustar via Fourier a função

x	$-3,141$	$-1,885$	$-0,629$	$0,629$	$1,885$	$3,141$
y	$10,28$	$3,15$	$0,15$	$0,64$	$3,96$	$9,4$

a um polinômio trigonométrico de grau $m = 2$

Solução:
Como $x \in [-\pi; \pi], 2n = 6$ e $m = 2 \Rightarrow P_2(x) = \frac{a_0}{2} + a_1 cosx + a_2 cos2x + b_1 senx$, aplicando (5.12) obtém-se: $a_0 = 9,2; a_1 = -7,096; a_2 = 4,74$ e $b_1 = 0,352$. Daí $P_2(x) = 4,6 - 7,096 cosx + 4,74 cos2x + 0,352 senx$ é o ajuste proposto.

Análise do Ajuste de Fourier

1ª] O ajuste de Fourier aproximações adequadas do padrão de comportamento da $y = f(x)$ se a mesma for periódica. Caso a $y = f(x)$ não seja periódica os resultados podem ser ruins (fenômeno de Gibbs);

2ª] Para uma base de dados com $2n$ elementos os melhores resultados de captação do padrão de comportamento são obtidos quando se ajusta-a a um polinômio trigonométrico de grau $m = n$. A conseqüência disto é que o custo para aplicar (5.13) torna-se de aproximadamente $8n^2$ operações. Como o n

deve ser elevado ($n = 10^5, n = 10^6, n = 10^8$, etc) tem-se que para $n = 10^6 \Rightarrow$ custo de $8 * 10^{12}$ operações $\Rightarrow$ computador de 1 Gigaflop levaria $t = 8.000$ segundos $= 2,2$ horas de processamento. Já para $n = 10^8 \Rightarrow 8 * 10^{16}$ operações $\Rightarrow 2,5$ anos de processamento!;

3ª] Devido ao custo proibitivo do ajuste de Fourier usando diretamente a (5.13), seu uso era impraticável. Em 1965 Cooley/Tuckey desenvolveram uma técnica denominada de transformada rápida de Fourier (FFT) para acelerar este processo. Para uma base de dados com $2n = 2^k$(potência de dois) o custo do seu ajuste a uma $P_m(x)$ com $m = n$, usando FFT reduz-se para $n * log_2 n$ operações. Daí se $n = 10^6 \Rightarrow n * log_2 n = 2 * 10^7$ operações $\Rightarrow$ computador de 1Gigaflop executa em $0,02$ segundos. Já para $n = 10^8 \Rightarrow n * log_2 n = 27 * 10^8$ operações $\Rightarrow 2,7$ segundos de processamento pelo mesmo computador que levaria 2,5 anos de processamento ao ajuste normal.

Como é possível tal redução drástica?

4ª] Suporte técnico das FFT

Uma forma alternativa dos polinômios trigonométricos, consiste em se expressá-los na forma de exponenciais complexas. Por definição, tem-se que $i^2 = -1$. Desenvolvendo em série de Taylor/Maclaurin a $e^{ix} = 1 + ix - \frac{x^2}{2!} - \frac{ix^3}{3!} + \frac{x^4}{4!} + \frac{ix^5}{5!} - \frac{x^6}{6!} - \frac{ix^7}{7!} + \ldots$

$= (1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots) + i(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots)$

$= cosx + isenx$

Como $coskx = \frac{e^{ikx} + e^{-ikx}}{2}$ e $isenx = \frac{e^{ikx} - e^{-ikx}}{2} \Rightarrow$

$P_m(x) = \frac{a_0}{2} + \sum_{k=1}^{m}[a_k coskx + b_k senkx] = \frac{a_0}{2} + \sum_{k=1}^{m}[\alpha_k e^{ikx} + \beta_k e^{-ikx}]$, onde:

$\alpha_k = \frac{(a_k - ib_k)}{2}$, $\beta_k = \frac{a_k + ib_k}{2}$ e $\alpha_0 = \frac{a_0}{2}$

Considerando $\beta_k = \alpha_k \Rightarrow P_m(x) = \sum_{k=-m}^{m} \alpha_k * e^{ikx}$ que é a forma exponencial complexa, onde $\beta_k = \alpha_k$.

Após mergulhar o leitor no mundo das exponenciais complexas que fundamentam as FFT, vamos encerrar a disciplina. Fica a cargo do leitor interessado prosseguir no entendimento da estratégia de Cooley/Tuckey, a qual pode ser encontrada na bibliografia da Análise Numérica.