Universidade Federal do Paraná - UFPR
Centro Politécnico

Disciplina: Análise Numerica II **Código:** CMI052 **Semestre letivo:** 2025/1
Professor: Luiz Carlos Matioli
Aluna: Kauane Batista e Mariana Mirela Ferreira

PROJETO 1

1 Polinômio Interpolador de Newton-Gregory

Podemos utilizar uma fórmula mais simples para o polinômio de interpolação quando os argumento x_i são equidistantes, e para isso precisa-se da definição de **Diferenças ordinárias** de uma função.

Seja $x_0, x_1, ..., x_n$, $n+1$ pontos distintos igualmente espaçados em um intervalo [a,b], isto é $x_{i+1} - x_i = h$, com $i = 0, 1, ..., n-1$. Define-se

$$
\begin{aligned}
\Delta^0 f(x) &= f(x), \\
\Delta^1 f(x) &= f(x+h) - f(x), \\
\Delta^2 f(x) &= \Delta f(x+h) - \Delta f(x), \\
&. \\
&. \\
&. \\
\Delta^n f(x) &= \Delta^{n-1} f(x+h) - \Delta^{n-1} f(x),
\end{aligned}
$$

Onde $\Delta^n f(x)$ é a **diferença ordinária** de ordem n em uma função $f(x)$ sobre os pontos $x_0, x_1, ..., x_n$ igualmente espaçados de h.

Com isso podemos construir a tabela de diferenças ordinárias:

x_i	$f(x_i)$	$\Delta f(x_i)$	$\Delta^2 f(x_i)$	$\cdots$	$\Delta^n f(x_i)$
x_0	$f(x_0)$				
x_1	$f(x_1)$	$\Delta f(x_0)$			
x_2	$f(x_2)$	$\Delta^f(x_1)$	$\Delta^2 f(x_0)$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
x_n	$f(x_n)$	$\Delta f(x_n)$	$\Delta^2 f(x_n)$	$\cdots$	$\Delta^n f(x_n)$

1.1 Teorema das diferenças ordinárias

Teorema 1

Se $x_k = x_0 + kh$; $k = 0, 1, 2..., n$, então:

$$f[x_0, x_1, ..., x_n] = \frac{\Delta^n f(x_0)}{h^n n!}$$

- **Demonstração por indução**

i) Para $n = 1$ temos

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{f(x_0 + h) - f(x_0)}{h} = \frac{\Delta^1 f(x_0)}{h},$$

desde que $x_1 = x_0 + h$

ii) Para $n = k$:

$$f[x_0, \ldots, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}$$
$$= \frac{1}{kh}\left[\frac{\Delta^{k-1} f(x_1)}{h^{k-1}(k-1)!} - \frac{\Delta^{k-1} f(x_0)}{h^{k-1}(k-1)!}\right] = \frac{\Delta^k f(x_0)}{h^k k!}$$

iii) Aplicando o *Teorema 1* supracitado no polinômio interpolador de newton, que é dado por

$$P_n(x) = f(x_0) + f(x_0, x_1)(x - x_0) + ... + f(x_0, x_1, ..., x_n)(x - x_0)...(x - x_{n-1})$$

para uma função $f(x)$ em um intervalo [a,b] onde os argumento x_i são igualmente espaçados de h, definimos o **polinômio de interpolação de Newton-Gregory** como

$$P_n(x) = f(x_0) + (x - x_0)\frac{\Delta f(x_0)}{h} + (x - x_0)(x - x_1)\frac{\Delta^2 f(x_0)}{h^2 2!} + \cdots + (x - x_0)(x - x_1)\cdots(x - x_{n-1})\frac{\Delta^n f(x_0)}{h^n n!}$$

■

1.2 Exercícios

1. Considere

x	x_0	x_1	$\ldots$	x_n
$f(x)$	$f(x_0)$	$f(x_1)$	$\ldots$	$f(x_n)$

onde os nós de interpolação são tais que: $x_{j+1} - x_j = h$, $j = 0, 1, \ldots, (n-1)$.

Partindo da forma de Newton para $p_n(x)$ e usando o teorema anterior, verifique que:

$$p_n(x) = f(x_0) + (x - x_0)\frac{\Delta f(x_0)}{h} + (x - x_0)(x - x_1)\frac{\Delta^2 f(x_0)}{2h^2} + \cdots$$
$$+ (x - x_0)(x - x_1)\cdots(x - x_{n-1})\frac{\Delta^n f(x_0)}{h^n n!}$$

que é a forma de Newton-Gregory para o polinômio interpolador.

Resposta 1 - A forma geral do polinômio interpolador de Newton é dada por:

$$p_n(x) = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_n(x - x_0)(x - x_1)\dots(x - x_{n-1})$$

onde os coeficientes c_k são as diferenças divididas de f, ou seja:

$$c_k = f[x_0, x_1, \dots, x_k]$$

Seguindo o *Teorema 1*, as diferenças divididas podem ser expressas em termos de diferenças finitas ordinárias quando os argumentos são igualmente espaçados, como:

Para $k = 0$:

$$c_0 = f(x_0)$$

Para $k = 1$:

$$c_1 = f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

Dado que $x_1 - x_0 = h$, temos:

$$c_1 = \frac{\Delta f(x_0)}{h}$$

Para $k = 2$:

$$c_2 = f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$$

$$c_2 = \frac{\frac{\Delta f(x_1)}{h} - \frac{\Delta f(x_0)}{h}}{2h} = \frac{\Delta f(x_1) - \Delta f(x_0)}{2h^2} = \frac{\Delta^2 f(x_0)}{2h^2}$$

Generalizando para o coeficiente c_k:

$$c_k = f[x_0, x_1, \dots, x_k] = \frac{f[x_1, x_2, \dots, x_n] - f[x_0, x_1, \dots, x_{n-1}]}{x_n - x_0} = \frac{\Delta^k f(x_0)}{k!h^k}$$

Substituindo os coeficientes c_k na forma de Newton do polinômio interpolador, obtemos:

$$p_n(x) = f(x_0) + (x - x_0)\frac{\Delta f(x_0)}{h} + (x - x_0)(x - x_1)\frac{\Delta^2 f(x_0)}{2h^2} + \cdots + (x - x_0)(x - x_1)\dots(x - x_{n-1})\frac{\Delta^n f(x_0)}{h^n n!}$$

Mostrando assim a forma de Newton-Gregory para o polinômio interpolador. ■

2. Usando a forma de Newton-Gregory para $p_3(x)$ obtenha uma aproximação para $f(2,7)$, onde $f(x)$ é a função tabelada a seguir:

x	1	2	3	4	5
$f(x)$	0	1,3863	2,1972	2,7726	3,2189

Resposta 2 - Como os valores x_n são igualmente espaçados com $x_n - x_{n-1} = h \Rightarrow h = 1$, podemos usar

$$f[x_0, x_1, ..., x_n] = \frac{\Delta^n f(x_0)}{h^n n!}$$

para monta o polinômio interpolador de Newton $p_3(x)$ dado por

$$p_3(x) = f(x_0) + (x - x_0)\frac{\Delta f(x_0)}{h} + (x - x_0)(x - x_1)\frac{\Delta^2 f(x_0)}{2!h^2} + (x - x_0)(x - x_1)(x - x_2)\frac{\Delta^3 f(x_0)}{h^3 3!}$$

Calculamos os $\frac{\Delta^n f(x_0)}{h^n n!}$

$$\Delta f(x_0) = f(x_1) - f(x_0) = 1.3863 - 0 = 1.3863$$
$$\Delta f(x_1) = f(x_2) - f(x_1) = 2.1972 - 1.3863 = 0.8109$$
$$\Delta f(x_2) = f(x_3) - f(x_2) = 2.7726 - 2.1972 = 0.5754$$

$$\Delta^2 f(x_0) = \Delta f(x_1) - \Delta f(x_0) = 0.8109 - 1.3863 = -0.5754$$
$$\Delta^2 f(x_1) = \Delta f(x_2) - \Delta f(x_1) = 0.5754 - 0.8109 = -0.2355$$

$$\Delta^3 f(x_0) = \Delta^2 f(x_1) - \Delta^2 f(x_0) = -0.2355 - (-0.5754) = 0.3399$$

E a tabela de diferenças ordinárias é descrita como

x_i	$f(x_i)$	1ª ordem	2ª ordem	3ª ordem
1	$f_0 = 0.0000$			
2	$f_1 = 1.3863$	$\Delta f_0 = 1.3863$		
3	$f_2 = 2.1972$	$\Delta f_1 = 0.8109$	$\Delta^2 f_0 = -0.5754$	
4	$f_3 = 2.7726$	$\Delta f_2 = 0.5754$	$\Delta^2 f_1 = -0.2355$	$\Delta^3 f_0 = 0.3399$
5	$f_4 = 3.2189$			

Como $h = 1$, então $P_3(x)$ é dado por:

$$P_3(x) = 0 + 1.3863(x - 1) + \frac{(-0.5754)}{2}(x - 1)(x - 2) + \frac{0.3399}{6}(x - 1)(x - 2)(x - 3)$$
$$P_3(x) = 0.05665x^3 - 0.6276x^2 + 2.87255x - 2.3016$$

Para $x = 2.7$ obtemos

$$P_3(x) = 0.05665(2.7)^3 - 0.6276(2.7)^2 + 2.87255(2.7) - 2.3016$$

$\boxed{f(2.7) \approx P_3(2.7) = 1.9941}$

■

3. A forma de Newton-Gregory para $p_n(x)$ pode ser simplificada, se usarmos uma mudança de variáveis:

$$s = \frac{x - x_0}{h} \Rightarrow x = sh - x_0$$

Daí,

$$x - x_j = sh + x_0 - (x_0 + jh) = (s - j)h$$

Usando essa troca de variáveis, escrevemos a forma geral para $p_n(x)$.

Resposta 3 - A forma geral do polinômio de Newton-Gregory é dado por:

$$p_n(x) = f(x_0) + (x - x_0)\frac{\Delta f(x_0)}{h} + (x - x_0)(x - x_1)\frac{\Delta^2 f(x_0)}{2!h^2} + \cdots + \prod_{j=0}^{n-1}(x - x_j)\frac{\Delta^n f(x_0)}{n!h^n}$$

Fazendo a substituição:

$$s = \frac{x - x_0}{h} \Rightarrow x = x_0 + sh$$

Temos:

$$x - x_j = (x_0 + sh) - (x_0 + jh) = (s - j)h$$

Logo,

$$\prod_{j=0}^{n-1}(x - x_j) = \prod_{j=0}^{n-1}(s - j)h = h^n \prod_{j=0}^{n-1}(s - j)$$

Substituindo no polinômio, temos:

$$p_n(x) = f(x_0) + s \cdot \Delta f(x_0) + \frac{s(s-1)}{2!} \cdot \Delta^2 f(x_0) + \cdots + \frac{s(s-1)\cdots(s-n+1)}{n!} \cdot \Delta^n f(x_0)$$

■

2 Polinômio Interpolador de Hermite

A interpolação de Hermite abrange a interpolação de Lagrange. Não só interpola os valores da função, mas também as derivadas em certos pontos. Isso nos permite um ajuste mais suave especialmente quando a derivada da função é conhecida.

2.1 Teorema de Hermite

Teorema 2

Seja f uma função suficientemente diferenciável e $x_0, x_1, \ldots, x_n$ pontos distintos. Então, existe um único polinômio $H(x)$ de grau no máximo $2n+1$, tal que:

$$\begin{cases} H(x_i) = f(x_i) \\ H'(x_i) = f'(x_i) \end{cases} \quad \text{para } i = 0, 1, \ldots, n$$

O polinômio interpola os valores da função e valores de sua derivada nos pontos x_i. O número de condições é $2(n+1)$, pois há duas para cada ponto (função e derivada). O polinômio deve ter grau $\leq 2n+1$ para satisfazer todas as condições. Isso garante a existência e unicidade do polinômio.

2.2 Construção Prática

Interpolação por Lagrage

O polinômio interpolador de Hermite pode ser expresso através dos polinômios de Lagrange.

$$H_{2n+1}(x) = \sum_{j=0}^{n} f(x_j) \cdot h_j(x) + \sum_{j=0}^{n} f'(x_j) \cdot \hat{h}_j(x)$$

$$h_j(x) = \left(1 - 2(x - x_j) \cdot \ell_j'(x_j)\right) \cdot [\ell_j(x)]^2$$

$$\hat{h}_j(x) = (x - x_j) \cdot [\ell_j(x)]^2$$

$$\ell_j(x) = \prod_{\substack{0 \leq m \leq n \\ m \neq j}} \frac{x - x_m}{x_j - x_m}$$

é o polinômio de Lagrange associado ao ponto x_j, e $\ell_j'(x_j)$ é a derivada de $\ell_j(x)$ avaliada em x_j.

- $h_j(x)$: garante que $H(x_j) = f(x_j)$ e que sua derivada também coincida com $f'(x_j)$.
- $\hat{h}_j(x)$: ajusta a inclinação de H para coincidir com $f'(x_j)$.

Características da interpolação de Hermite

- Usa mais informações que Newton ou Lagrange: valores de f e f'.
- Gera polinômio de grau até $2n+1$ para $n+1$ pontos.
- Utilizado para modelagem onde derivadas são conhecidas, como em soluções de EDOs ou ajustes suaves de curvas.

Observação prática (conforme instrução do professor)

Queremos interpolar $f(x)$ e também aproximar $f'(x)$. O polinômio de Hermite $H_{2n+1}(x)$ satisfaz:

$$H(x_i) = f(x_i)$$

$$H'(x_i) = f'(x_i) \approx P_n'(x_i)$$

onde $P_n(x)$ é o polinômio de Newton-Gregory. Mas como os $f'(x_i)$ não são dados, substituiremos por:

$$f'(x_i) \approx P_n'(x_i)$$

O polinômio interpolador de Newton-Gregory (com espaçamento h) é:

$$P_n(x) = f(x_0) + \frac{\Delta f(x_0)}{h}(x - x_0) + \frac{\Delta^2 f(x_0)}{2!h^2}(x - x_0)(x - x_1) + \cdots + \frac{\Delta^n f(x_0)}{n!h^n}(x - x_0)\cdots(x - x_{n-1})$$

A derivada $P_n'(x)$ é:

$$P_n'(x) = \frac{\Delta f(x_0)}{h} + \frac{\Delta^2 f(x_0)}{2!h^2}[(x - x_0) + (x - x_1)] + \cdots$$

Podemos usar a regra do produto e derivar cada termo do polinômio.

Avalie em $x = x_i$ para estimar:

$$f'(x_i) \approx P_n'(x_i)$$

Substituindo na forma de Hermite:

$$H(x) = \sum_{i=0}^{n} \left[\left(1 - 2L_i'(x_i)(x - x_i)\right) L_i(x)^2 f(x_i) + (x - x_i) L_i(x)^2 \cdot P_n'(x_i) \right]$$

Este é o polinômio de Hermite interpolador com derivadas estimadas a partir da derivada do polinômio de Newton-Gregory.

2.3 Exercício

Este exemplo corresponde ao exercício indicado pelo professor em aula, visando a uma melhor compreensão do polinômio de Hermite.

1. Verifique que o polinômio de Hermite $H_{2n+1}(x)$ satisfaz:

$$H_{2n+1}(x_i) = f(x_i) \quad \text{e} \quad H'_{2n+1}(x_i) = f'(x_i), \quad \text{para todo } i = 0, 1, \ldots, n$$

Resposta 1 - Usando a fórmula:

$$H_{2n+1}(x) = \sum_{j=0}^{n} f_j \cdot h_j(x) + \sum_{j=0}^{n} f'_j \cdot \hat{h}_j(x)$$

$$h_j(x) = \left[1 - 2(x - x_j)\ell'_j(x_j)\right] \cdot \ell_j(x)^2$$

$$\hat{h}_j(x) = (x - x_j) \cdot \ell_j(x)^2$$

$$\ell_j(x_i) = \begin{cases} 1, & \text{se } i = j \\ 0, & \text{se } i \neq j \end{cases}$$

1° verificação: $H_{2n+1}(x_i) = f(x_i)$

Vamos substituir $x = x_i$ na fórmula:

$$H_{2n+1}(x_i) = \sum_{j=0}^{n} f_j \cdot h_j(x_i) + \sum_{j=0}^{n} f'_j \cdot \hat{h}_j(x_i)$$

Avaliando $h_j(x_i)$:

- Quando $j \neq i$: $\ell_j(x_i) = 0 \Rightarrow h_j(x_i) = 0$
- Quando $j = i$: $\ell_i(x_i) = 1$ e $\ell'_i(x_i) = \ell'_i(x_i)$, então:

$$h_i(x_i) = 1^2 = 1$$

Avaliando $\hat{h}_j(x_i)$:

- Quando $j \neq i$: $\ell_j(x_i) = 0 \Rightarrow \hat{h}_j(x_i) = 0$
- Quando $j = i$: $x_i - x_i = 0 \Rightarrow \hat{h}_i(x_i) = 0$

$$H_{2n+1}(x_i) = f_i \cdot 1 + \sum_{j \neq i} f_j \cdot 0 + \sum_{j=0}^{n} f'_j \cdot 0 = f_i$$

Verificado que $H_{2n+1}(x_i) = f(x_i)$

2° verificação: $H'_{2n+1}(x_i) = f'(x_i)$

Derivando a fórmula:

$$H'_{2n+1}(x) = \sum_{j=0}^{n} f_j \cdot h'_j(x) + \sum_{j=0}^{n} f'_j \cdot \hat{h}'_j(x)$$

Avaliando $x = x_i$ em $h'_j(x_i)$

- Quando $j \neq i$: $\ell_j(x_i) = 0 \Rightarrow h_j(x_i) = 0 \Rightarrow h'_j(x_i) = 0$
- Quando $j = i$:

  Derivando:

  $$h_i(x) = \left[1 - 2(x - x_i)\ell'_i(x_i)\right] \cdot \ell_i(x)^2$$

  Como $\ell_i(x_i) = 1$ e $\ell'_i(x_i)$ é constante, a derivada de $h_i(x)$ em x_i é:

  $$h'_i(x_i) = 0 \quad \text{(conhecido do resultado)}$$

Avaliando $\hat{h}'_j(x_i)$

- Quando $j \neq i$: $\ell_j(x_i) = 0 \Rightarrow \hat{h}_j(x_i) = 0 \Rightarrow \hat{h}'_j(x_i) = 0$
- Quando $j = i$:

  $$\hat{h}_i(x) = (x - x_i) \cdot \ell_i(x)^2$$

  $$\hat{h}'_i(x) = \ell_i(x)^2 + 2(x - x_i)\ell_i(x)\ell'_i(x)$$

  Avaliando em $x = x_i$:

  $$\hat{h}'_i(x_i) = 1^2 + 2 \cdot 0 \cdot \ell'_i(x_i) = 1$$

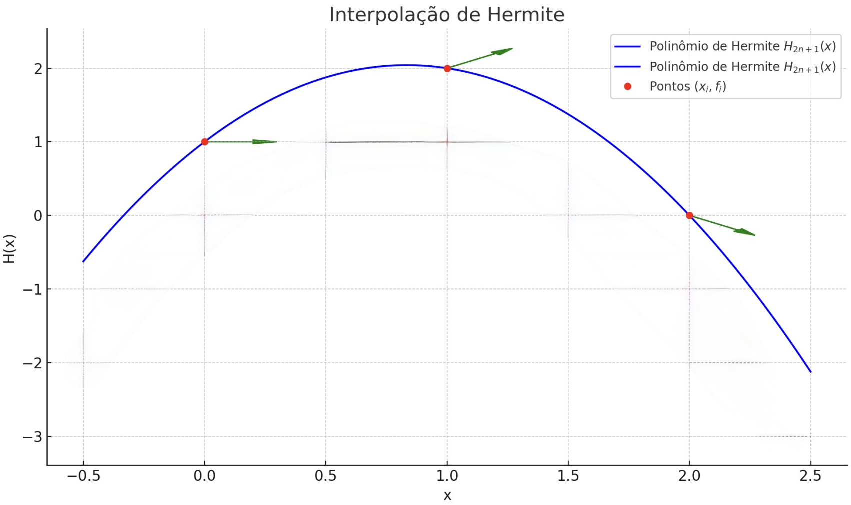
$$H'_{2n+1}(x_i) = f_i \cdot 0 + f'_i \cdot 1 + \sum_{j \neq i} (f_j \cdot 0 + f'_j \cdot 0) = f'_i$$

3° verificação: $H'_{2n+1}(x_i) = f'(x_i)$

Portanto, o polinômio $H_{2n+1}(x)$ satisfaz todas as condições de interpolação de Hermite.$\blacksquare$

Para a plotagem do gráfico foi utilizado o seguinte código Aplicação da Interpolação de Hermite

A imagem abaixo mostra a construção do polinômio de Hermite $H_{2n+1}(x)$ que interpola os dados:



$$f(0) = 1, f'(0) = 0; f(1) = 2, f'(1) = 1; f(2) = 0, f'(2) = -1$$

Detalhes da figura

- Os pontos vermelhos são os dados (x_i, f_i).
- As setas verdes indicam as derivadas $f'(x_i)$ nos respectivos pontos.
- A curva azul representa o polinômio interpolador de Hermite que satisfaz simultaneamente as condições de valor e derivada.

■

2.4 Pseudocódigo

Pseudocódigo: Polinômio de Hermite $H_{2n+1}(x)$

Entrada:

Vetores com nós: $x = [x_0, x_1, \ldots, x_n]$

Valores da função: $f = [f_0, f_1, \ldots, f_n]$

Derivadas da função: $f' = [f'_0, f'_1, \ldots, f'_n]$

Passo 1: Construir os polinômios de Lagrange $\ell_j(x)$

Para cada $j = 0$ até n:

$$\ell_j(x) = \prod_{\substack{m=0 \\ m \neq j}}^{n} \frac{x - x_m}{x_j - x_m}$$

Passo 2: Construir os polinômios auxiliares

Para cada $j = 0$ até n:

$$h_j(x) = \left[1 - 2(x - x_j)\ell'_j(x_j)\right] \cdot [\ell_j(x)]^2$$

$$\hat{h}_j(x) = (x - x_j) \cdot [\ell_j(x)]^2$$

onde $\ell'_j(x_j)$ é a derivada de $\ell_j(x)$ avaliada em x_j.

Passo 3: Construir o polinômio interpolador de Hermite

$$H(x) = \sum_{j=0}^{n} f_j \cdot h_j(x) + f'_j \cdot \hat{h}_j(x)$$

Saída:

O valor $H(x)$ do polinômio de Hermite em um ponto x.

Representação gráfica de $H(x)$ em um intervalo.

3 Fenômeno de Runge

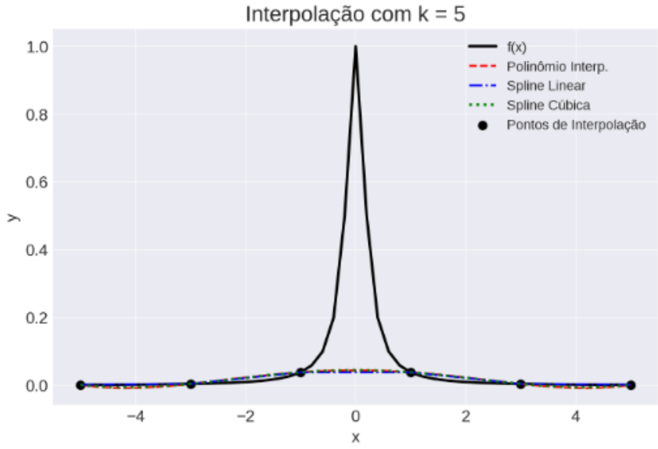
Para a resolução do exercício abaixo, foi utilizado o código Aplicação do Fenômeno de Runge

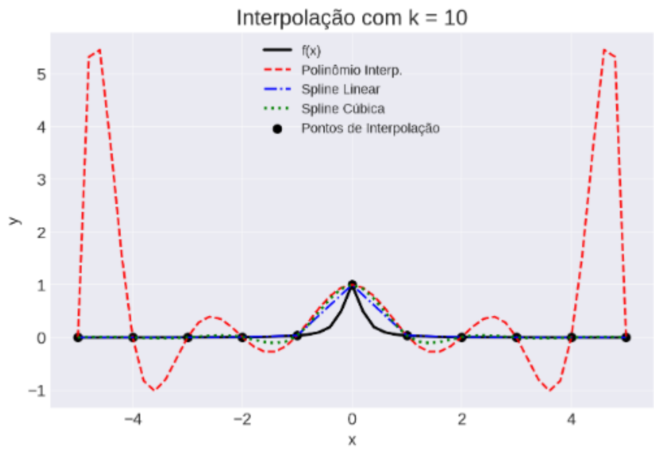
Para os valores de $k = 5, 10, 20$, temos os erros máximos:

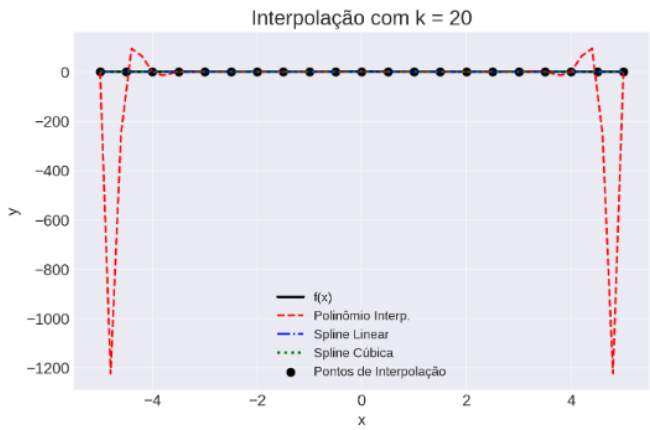
- **k = 5**
  - Erro máx. Polinômio: 0,956
  - Erro máx. Spline Linear: 0,962
  - Erro máx. Spline Cúbica: 0,956
- **k = 10**
  - Erro máx. Polinômio: 5,454
  - Erro máx. Spline Linear: 0,415
  - Erro máx. Spline Cúbica: 0,537
- **k = 20**
  - Erro máx. Polinômio: 1223,084 **(Fenômeno de Runge muito evidente!)**
  - Erro máx. Spline Linear: 0,155
  - Erro máx. Spline Cúbica: 0,268

Ao analisar os erros obtidos nos métodos de interpolação aplicados à função de Runge, podemos tirar conclusões bastante relevantes. Para $k = 20$, o Fenômeno de Runge torna-se evidente, com um erro explosivo especialmente nas extremidades do intervalo. Esse comportamento ocorre devido ao uso de polinômios de grau elevado sobre uma malha de pontos uniformemente espaçados, o que, conforme apontado nos vídeos do professor Ricardo Biloti, pode levar à divergência da interpolação polinomial, mesmo quando a função original $f(x)$ é analítica.

Em contraste, as splines — tanto a linear quanto a cúbica — mantêm um erro baixo e estável em todos os casos analisados, demonstrando maior robustez e fidelidade à função. Para valores menores de k, como $k = 5$, o polinômio ainda consegue aproximar bem a função, sem apresentar oscilações significativas. No entanto, à medida que o número de pontos aumenta, como no caso $k = 10$, começa-se a perceber oscilações nas bordas da interpolação polinomial, enquanto as splines continuam mais próximas da função original.


Interpolação com k = 5
f(x)
Polinômio Interp.
Spline Linear
Spline Cúbica
Pontos de Interpolação
y
x


Interpolação com k = 10
f(x)
Polinômio Interp.
Spline Linear
Spline Cúbica
Pontos de Interpolação
y
x


Interpolação com k = 20
f(x)
Polinômio Interp.
Spline Linear
Spline Cúbica
Pontos de Interpolação
y
x

4 Códigos utilizados

Listing 1: Bibliotecas Python Necessárias

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate import interp1d, CubicSpline,BarycentricInterpolator,
    KroghInterpolator
```

Listing 2: Aplicação da Interpolação de Hermite (PARTE II) realizada em Python

```


x = np.array([0.0, 1.0, 2.0])
f = np.array([1.0, 2.0, 0.0])
df = np.array([0.0, 1.0, -1.0])

interp = KroghInterpolator(x, np.vstack((f, df)).T)

x_vals = np.linspace(min(x) - 0.5, max(x) + 0.5, 400)
y_vals = interp(x_vals)

plt.figure(figsize=(10, 6))
plt.plot(x_vals, y_vals, label="Polin mio de Hermite $H_{2n+1}(x)$", color='blue')
plt.plot(x, f, 'ro', label="Pontos $(x_i, f_i)$")
for i in range(len(x)):
    plt.arrow(x[i], f[i], 0.2, df[i]*0.2, head_width=0.05, head_length=0.1, color='green')
plt.title("Interpola  o de Hermite")
plt.xlabel("x")
plt.ylabel("H(x)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Listing 3: Aplicação do Fenômeno de Runge (PARTE III) realizada em Python

```python

def f(x):
    return 1 / (1 + 25 * x ** 2)


z = np.linspace(-5, 5, 51)
f_z = f(z)


ks = [5, 10, 20]
erros = {}

for k in ks:

    x = np.linspace(-5, 5, k + 1)
    y = f(x)

    p = BarycentricInterpolator(x, y)
    p_z = p(z)

    S1 = interp1d(x, y, kind='linear')
    S1_z = S1(z)

    S3 = CubicSpline(x, y)
    S3_z = S3(z)

    erro_pk = np.max(np.abs(f_z - p_z))
    erro_s1 = np.max(np.abs(f_z - S1_z))
    erro_s3 = np.max(np.abs(f_z - S3_z))

    erros[k] = {'polin mio': erro_pk, 'spline_linear': erro_s1, 'spline_cubica': erro_s3}

    plt.figure(figsize=(10, 6))
    plt.plot(z, f_z, 'k-', label='f(x)')
    plt.plot(z, p_z, 'r--', label='Polin mio Interp.')
    plt.plot(z, S1_z, 'b-.', label='Spline Linear')
    plt.plot(z, S3_z, 'g:', label='Spline C bica')
    plt.scatter(x, y, c='k', marker='o', label='Pontos de Interpola  o')
    plt.title(f'Interpola  o com k={k}')
    plt.legend()
    plt.grid()
    plt.show()

for k in ks:
    print(f'\nk = {k}:')
    print(f'Erro  m x. polin mio: {erros[k]["polin mio"]:.6f}')
    print(f'Erro  m x. spline linear: {erros[k]["spline_linear"]:.6f}')
    print(f'Erro  m x. spline c bica: {erros[k]["spline_cubica"]:.6f}')
```