

Interpolação de Newton

Igor Ferris Eduardo e Gabriel Pires da Cruz

Prof. Rogério Vargas – Cálculo Numérico

13 de junho de 2025

Sumário

- 1 Conceitos Teóricos
- 2 Exercícios resolvidos e as Aplicações em python
- 3 Conclusão e Referências

Introdução ao conceito

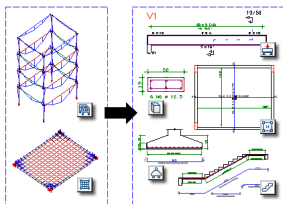
- A interpolação de Newton é um método matemático usado para construir um polinômio que passa por um conjunto de pontos conhecidos. Esse polinômio serve para estimar valores de uma função entre os pontos dados. O método utiliza diferenças divididas para calcular os coeficientes do polinômio, sendo eficiente e fácil de atualizar com novos dados. É útil quando os valores de entrada (x) não são igualmente espaçados.

Qual é a sua importância

- Apresentar o conceito de interpolação de Newton é importante porque permite estimar valores de forma precisa com base em dados limitados. Isso é essencial em situações em que não é possível obter medições contínuas. O método também é eficiente, pois não exige a repetição completa dos cálculos quando se adiciona um novo ponto. Além disso, fornece uma base para análises mais complexas, como modelagem computacional e simulações.

Aplicação para a Engenharia Civil

Na engenharia civil, a interpolação de Newton é aplicada em diversas áreas. É usada para prever deformações de estruturas com base em medições pontuais, estimar variações de cargas em terrenos, interpolar dados topográficos e calcular esforços em elementos de uma estrutura. Também aparece em softwares de análise estrutural, onde auxilia na criação de modelos aproximados com dados reais. Isso contribui para projetos mais seguros e precisos.



(a) Figura de análise de deformações de estruturas



(b) Figura da interpolação dos dados topográficos

Interpolação de Newton: Teoria

- **Definição:** Seja $f(x)$ contínua e com tantas derivadas contínuas quantas necessárias num intervalo $[a, b]$. Sejam $a = x_0 < x_1 < \dots < x_n = b$, com $n + 1$ pontos. A forma de Newton para o polinômio de grau $\leq n$ que interpola $f(x)$ em x_0, x_1, \dots, x_n é dada por:

Polinômio de Interpolação de Newton

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

Interpolação de Newton: Teoria

- **Teorema do Erro de Interpolação:**

Sejam x_0, \dots, x_n , $n + 1$ pontos reais distintos, e seja $p_n(x)$ o polinômio interpolador de grau até n da função $f : D \rightarrow \mathbb{R}$ nos pontos x_k . Então, para cada $x \in D$, o erro da interpolação é dado por:

Teorema do Erro de Interpolação

$$f(x) - p_n(x) = f[x_0, x_1, \dots, x_n, x] \cdot \prod_{j=0}^n (x - x_j)$$

Interpolação de Newton: Teoria

Pressuposição

Seja $f(x)$ função tabelada em $n + 1$ pontos distintos: x_0, x_1, \dots, x_n :

Logo, apresentará a $f(x)$ função em:

$f[x_1, x_2, x_3, \dots, x_n]$

• Operador Diferenças Divididas:

x	Ordem 0	Ordem 1	Ordem 2	Ordem 3	Ordem 4
x_0	$f[x_0]$				
		$f[x_0, x_1]$			
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$		
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$	
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$		$f[x_0, x_1, x_2, x_3, x_4]$
		$f[x_2, x_3]$		$f[x_1, x_2, x_3, x_4]$	
x_3	$f[x_3]$		$f[x_2, x_3, x_4]$		
		$f[x_3, x_4]$			
x_4	$f[x_4]$				

Interpolação de Newton: Teoria

Diz-se que $f[x_0, x_1, x_2, \dots, x_n]$ é a **diferença dividida de ordem n** da função $f(x)$ sobre os $(n + 1)$ pontos distintos: x_0, x_1, \dots, x_n . Se para cada um desses pontos $f(x)$ for conhecida, obtém-se:

Equação da Diferenças Divididas

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

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

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

Interpolação de Newton: Teoria

- **Diferenças Divididas:** São operadores definidos recursivamente que calculam os coeficientes do polinômio interpolador de Newton. A diferença dividida de ordem zero é simplesmente o valor da função no ponto:

$$f[x_i] = f(x_i)$$

As de ordem superior são definidas por:

$$f[x_i, x_{i+1}, \dots, x_{i+k}] = \frac{f[x_{i+1}, \dots, x_{i+k}] - f[x_i, \dots, x_{i+k-1}]}{x_{i+k} - x_i}$$

- **Polinômio de Newton:** Utiliza as diferenças divididas como coeficientes para construir o polinômio interpolador. A forma geral do polinômio de Newton para $n + 1$ pontos é:
$$P_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \dots + f[x_0, x_1, \dots, x_n](x - x_0)(x - x_1) \dots (x - x_{n-1})$$

Interpolação de Newton: Teoria

- **Método de Newton:**

- Estrutura incremental: permite adicionar novos pontos sem recalcular todo o polinômio.
- Usa diferenças divididas, facilitando a construção do polinômio em forma recursiva.
- Mais eficiente para atualizar ou estender a interpolação.

- **Método de Lagrange:**

- Requer o reprocessamento completo ao adicionar novos pontos.
- A forma do polinômio é explícita, mas não reutilizável de forma incremental.
- Simples para pequenos conjuntos de dados fixos.

- **Vantagens comparativas:**

- Newton é preferido em situações com dados que podem ser atualizados dinamicamente.
- Lagrange é útil em casos de aplicação direta e pontual, com poucos dados.

Interpolação de Newton: Teoria

• Vantagens do Método de Newton

- Flexível e adaptável a diferentes conjuntos de dados
- Implementação computacional eficiente, utilizando diferenças divididas
- Ideal para atualizações frequentes: novos pontos podem ser adicionados de forma incremental, sem reprocessar todo o polinômio
- Vantajoso para aplicações dinâmicas, com inserção progressiva de dados

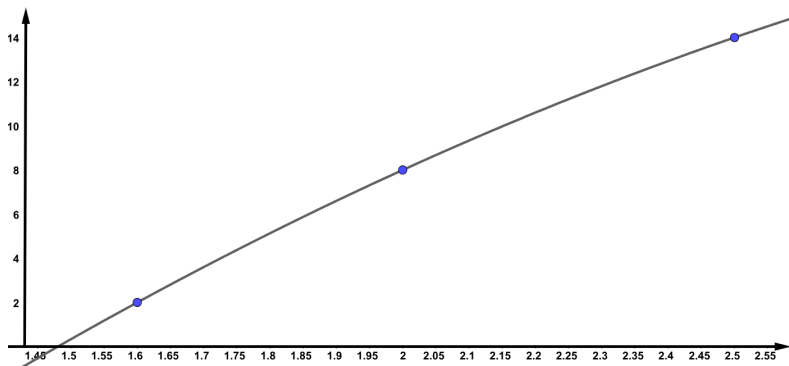


Figura: Visualização Gráfica de Interpolação de Newton

Exemplo 1: Tabela

Dados os pontos determine o polinômio que atende a seguinte estrutura:

x	F(x)
-1	4
0	1
2	-1

Exemplo 1: Resolução

x	Ordem 0	Ordem 1	Ordem 2
$x_0 = -1$	$f[x_0] = 4$		
		$f[x_0, x_1] = -3$	
$x_1 = 0$	$f[x_1] = 1$		$f[x_0, x_1, x_2] = 2/3$
		$f[x_1, x_2] = -1$	
$x_2 = 2$	$f[x_2] = -1$		

Resolução

$$P_2(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$

$$P_2(x) = 1 - \frac{7}{3}x + \frac{2}{3}x^2$$

Exemplo 1: Gráfico

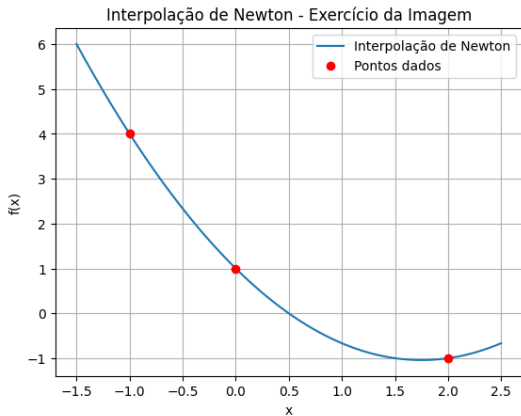


Figura: Exemplo 1

Exemplo 2: Pontos

Um experimento produziu os seguintes resultados para uma função desconhecida: $(0,1), (1,2), (2,0), (3,7)$

Exemplo 2: Código Simplificado

```
1 x = [0, 1, 2, 3]
2 y = [1, 2, 0, 7]
3 n = len(x)
4 div = [y[i] for i in range(n)] # primeira coluna (y)
5 coef = [div[0]] # primeiro coeficiente sempre y[0]
6 for j in range(1, n):
7     for i in range(n - 1, j - 1, -1):
8         div[i] = (div[i] - div[i - 1]) / (x[i] - x[i - j])
9         coef.append(div[j])
10 polinomio = f"{coef[0]:.2f}"
11 termo = ""
12 for i in range(1, n):
13     termo += f"(x - {x[i - 1]})"
14     sinal = "+" if coef[i] >= 0 else "-"
15     polinomio += f"{sinal} {abs(coef[i]):.2f}{termo}"
16 print("Polinomio de Newton:")
17 print("P(x) =", polinomio)
```

Gráfico Comparativo

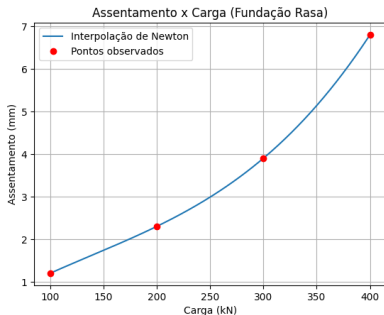


Figura: Gráfico Exemplo 1

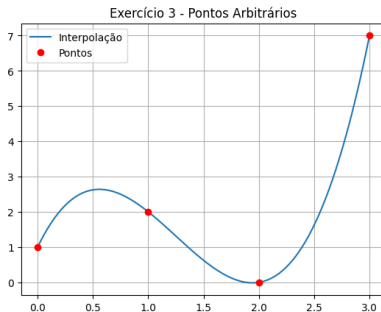


Figura: Gráfico Exemplo 2

Durante um ensaio de carregamento de uma fundação rasa, foram registrados os seguintes deslocamentos verticais (assentamentos) em função da carga aplicada:

Carga (kN)	Assentamento (mm)
100	1,2
200	2,3
300	3,9
400	6,8

- a) Utilizando interpolação de Newton com diferenças divididas, determine o polinômio interpolador $P(x)$ que representa a relação entre carga x e o assentamento.
- b) Escreva o polinômio na forma de Newton:
- c) Represente graficamente os pontos e o polinômio interpolador.

Exemplo 3: Código Simplificado

```
1 def newton_divided_diff(x, y):
2     n = len(x)
3     coef = y[:] # c pia dos valores de y
4     for j in range(1, n):
5         for i in range(n - 1, j - 1, -1):
6             coef[i] = (coef[i] - coef[i - 1]) / (x[i] - x[i - j])
7     return coef
8 def newton_poly(coef, x_data, x):
9     n = len(coef)
10    p = coef[n - 1]
11    for k in range(n - 2, -1, -1):
12        p = coef[k] + (x - x_data[k]) * p
13    return p
14 x4 = [100, 200, 300, 400] # Carga (kN)
15 y4 = [1.2, 2.3, 3.9, 6.8] # Assentamento (mm)
```

Exemplo 3: Código Simplificado

```
1 coef4 = newton_divided_diff(x4, y4)
2 n = len(x4)
3 polinomio = f"{coef4[0]:.4f}"
4 termo = ""
5 for i in range(1, n):
6     termo += f"(x - {x4[i - 1]})"
7     sinal = "+" if coef4[i] >= 0 else "-"
8     polinomio += f" {sinal} {abs(coef4[i]):.4f}{termo}"
9     termo = ""
9 print("P(x) =", polinomio)
```

Gráfico Comparativo

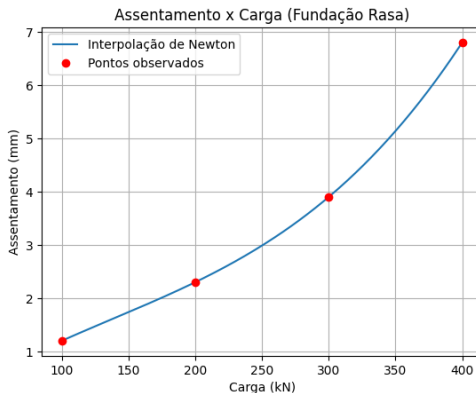


Figura: Grafico Exemplo 3

Análise de Comparação

A interpolação de Newton se destaca pela sua estrutura incremental, permitindo a adição de novos pontos ao conjunto de dados sem a necessidade de reconstruir todo o polinômio, graças ao uso das diferenças divididas. Isso a torna especialmente eficiente em contextos dinâmicos, onde os dados são atualizados com frequência. Em comparação, o método de Lagrange exige a reinterpretação completa da interpolação sempre que um novo ponto é inserido, o que o torna menos prático para grandes volumes de dados ou atualizações constantes. Assim, o método de Newton oferece maior flexibilidade computacional e estabilidade numérica, sendo amplamente preferido em aplicações de engenharia e modelagem computacional.

Conclusão Final

A interpolação de Newton é uma ferramenta poderosa, destacando-se pela sua flexibilidade na inclusão de novos pontos e pela eficiência computacional proporcionada pelas diferenças divididas. Comparada à interpolação de Lagrange, Newton é mais eficiente em atualizações. Em relação à interpolação spline, Newton pode ser menos precisa para conjuntos extensos de dados. Já métodos baseados em mínimos quadrados são preferíveis quando há ruído nos dados, pois ajustam a tendência geral e não passam exatamente por todos os pontos. Assim, Newton é ideal para dados limpos, de tamanho moderado e aplicações que exigem atualização contínua, sendo uma escolha equilibrada entre simplicidade e desempenho.

Referências

- Fernandes, R. K., Kirnev, D. C. B., & Boni, K. T. (2016). *Cálculo numérico*. Londrina: Editora e Distribuidora Educacional S.A.
- Roman, J. (2020). *Interpolação Polinomial – Parte II*. Instituto de Matemática, Estatística e Computação Científica, UNICAMP. Disponível em: <https://www.ime.unicamp.br/~roman/courses/MS211/1s2020/interpolacao2.pdf>
- Vídeo: Interpolação Polinomial - Exemplo com Newton. YouTube. Disponível em: <https://www.youtube.com/watch?v=6D7psrPApGU&t=259s>
- Arquivo de apoio (Drive). Disponível em: <https://drive.google.com/file/d/1X0Ejnt5eERoMZoFUy9HxKati90wznp7l/view?usp=sharing>