Escola Politécnica da Universidade de São Paulo

PTC5725 – Introdução aos Métodos Espectrais

Relatório: Lista de Exercícios 1 (Questões 1 e 2)

Renan de Luca Avila

São Paulo, 25 de setembro de 2025

1 Introdução

Este relatório apresenta as soluções para as duas primeiras tarefas propostas na disciplina *Introdução aos Métodos Espectrais (PCS5029)*. As questões abordam a **interpolação de Lagrange** e seu papel na aproximação numérica, além da demonstração da *unicidade* do polinômio interpolador. Durante a elaboração do relatório duas referências principais foram utilizadas: [1] para fórmulas e teoria, e [2] para inspiração em relação à códigos e aplicações. Além disso, a escolha de linguagem para programação foi Python.

2 Fundamentação Teórica

Sejam $n+1$ nós distintos $x_0, x_1, \ldots, x_n$ e os valores $f(x_0), f(x_1), \ldots, f(x_n)$. O polinômio interpolador de Lagrange $P_n(x)$ é

$$P_n(x) = \sum_{j=0}^{n} f(x_j)\, \ell_j(x), \tag{1}$$

em que os polinômios base de Lagrange $\ell_j(x)$ são dados por

$$\ell_j(x) = \prod_{\substack{0 \le k \le n \\ k \neq j}} \frac{x - x_k}{x_j - x_k}. \tag{2}$$

Por construção, $P_n(x_j) = f(x_j)$ para todo $j = 0, \ldots, n$.

3 Enunciados das Questões

Conforme apresentado em aula, os dois primeiros exercícios são:

1. Escreva um código para obter os valores de $f(x_k)$ com a interpolação de Lagrange feita para os $(n+1)$ pontos $f(x_j)$.
2. Mostre que o polinômio interpolante de Lagrange para $n+1$ pontos é o único polinômio de grau $\le n$ que passa por todos esses pontos.

4 Resolução da Questão 1: Código de Interpolação de Lagrange

A seguir está o código em Python que constrói e *avalia* o polinômio interpolador de Lagrange em pontos arbitrários x_k.

```
import numpy as np

def lagrange_interpolation(x_nodes, y_nodes, x_eval):
    """
    Calcula os valores do polinômio interpolador de Lagrange em pontos x_eval.

    Parâmetros
    ----------
    x_nodes : array-like
```

```
        Nós conhecidos x_j (distintos).
    y_nodes : array-like
        Valores f(x_j) correspondentes.
    x_eval : array-like
        Pontos nos quais avaliar o interpolador.

    Retorno
    -------
    y_eval : np.ndarray
        Valores interpolados em x_eval.
    """
    x_nodes = np.array(x_nodes, dtype=float)
    y_nodes = np.array(y_nodes, dtype=float)
    x_eval  = np.array(x_eval,  dtype=float)
    n = len(x_nodes)

    y_eval = np.zeros_like(x_eval, dtype=float)

    # Loop principal: soma ponderada dos polinômios-base _j(x)
    for j in range(n):
        lj = np.ones_like(x_eval, dtype=float)  # _j(x), começa em 1 (produto vazio)
        for m in range(n):
            if m != j:
                # Fator do produto: (x - x_m) / (x_j - x_m)
                lj *= (x_eval - x_nodes[m]) / (x_nodes[j] - x_nodes[m])
        # Acumula f(x_j) * _j(x) para compor P_n(x)
        y_eval += y_nodes[j] * lj

    return y_eval

# Exemplo mínimo de uso:
if __name__ == "__main__":
    xj = np.linspace(-1, 1, 5)            # nós
    yj = np.sin(np.pi * xj)               # valores f(x_j)
    xk = np.linspace(-1, 1, 100)          # pontos de avaliação
    yk = lagrange_interpolation(xj, yj, xk)
```

Listing 1: Interpolação de Lagrange: função de avaliação

Explicação detalhada do loop principal

O código implementa exatamente a fórmula teórica. Para cada $j = 0, \ldots, n$, o bloco

$$\ell_j(x) \;=\; \prod_{\substack{0 \le k \le n \\ k \neq j}} \frac{x - x_k}{x_j - x_k}$$

é calculado no laço interno (índice `m`). Em seguida, soma-se a contribuição $f(x_j)\,\ell_j(x)$ ao acumulador `y_eval`, conforme

$$P_n(x) \;=\; \sum_{j=0}^{n} f(x_j)\,\ell_j(x).$$

No código:

- `lj` começa como um vetor de uns (produto vazio vale 1) e é multiplicado fator a fator por $(x - x_m)/(x_j - x_m)$ para cada $m \neq j$.

- Ao final do laço interno, `lj` contém $\ell_j(x)$ avaliado em todos os pontos `x_eval`.
- A linha `y_eval += y_nodes[j] * lj` realiza a soma $\sum_j f(x_j)\,\ell_j(x)$ ponto a ponto.

Assim, o *loop externo* percorre os índices j (termos da soma) e o *loop interno* constrói o produto que define cada base $\ell_j(x)$.

5 Resolução da Questão 2: Unicidade do Polinômio Interpolador

Enunciado

Mostrar que o polinômio interpolante de Lagrange para $n+1$ pontos é o *único* polinômio de grau $\leq n$ que passa por todos esses pontos.

Existência

Dados nós distintos $x_0, \ldots, x_n$ e valores $f(x_0), \ldots, f(x_n)$, o polinômio de Lagrange

$$P_n(x) \;=\; \sum_{j=0}^{n} f(x_j)\,\ell_j(x), \qquad \ell_j(x) \;=\; \prod_{\substack{0\leq k\leq n\\ k\neq j}} \frac{x - x_k}{x_j - x_k},$$

satisfaz, por construção,

$$P_n(x_j) \;=\; f(x_j), \qquad j = 0, \ldots, n,$$

logo *existe* pelo menos um polinômio de grau $\leq n$ que interpola os dados.

Unicidade: Prova 1 (contagem de raízes)

Suponha que existam P e Q, ambos de grau $\leq n$, tais que

$$P(x_j) \;=\; Q(x_j) \;=\; f(x_j), \qquad j = 0, \ldots, n.$$

Considere a diferença

$$R(x) \;=\; P(x) - Q(x).$$

Então R é um polinômio de grau $\leq n$ com

$$R(x_j) \;=\; 0, \qquad j = 0, \ldots, n,$$

isto é, R possui ao menos $n+1$ raízes *distintas*. Mas um polinômio não nulo de grau $\leq n$ não pode ter mais do que n raízes distintas. Portanto, necessariamente $R \equiv 0$ e, assim, $P \equiv Q$. Logo, o interpolador é *único*. □

Unicidade: Prova 2 (matriz de Vandermonde)

Outra via é escrever o problema como um sistema linear para os coeficientes de $P(x)$. Suponha

$$P(x) \;=\; a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n.$$

Impor $P(x_j) = f(x_j)$ para $j = 0, \ldots, n$ gera

$$\underbrace{\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{bmatrix}}_{V(x_0, \ldots, x_n)} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} \;=\; \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}.$$

A matriz V é a *Vandermonde*. Para nós *distintos*, seu determinante é

$$\det V \;=\; \prod_{0 \le i < j \le n} (x_j - x_i) \;\neq\; 0,$$

portanto V é invertível e o sistema tem *solução única* para $(a_0, \ldots, a_n)^\top$. Consequentemente, o polinômio interpolador é único. □

Toy case (verificação manual com $n = 2$)

Considere três nós distintos no intervalo $[-1, 1]$:

$$x_0 = -1, \qquad x_1 = 0, \qquad x_2 = 1,$$

e valores genéricos

$$f(x_0) = a, \qquad f(x_1) = b, \qquad f(x_2) = c.$$

Buscamos um polinômio de grau ≤ 2,

$$P(x) \;=\; \alpha x^2 + \beta x + \gamma,$$

tal que

$$\begin{aligned} P(-1) = \alpha(-1)^2 + \beta(-1) + \gamma \;&=\; \alpha - \beta + \gamma \;=\; a, \\ P(0) = \gamma \;&=\; b, \\ P(1) = \alpha(1)^2 + \beta(1) + \gamma \;&=\; \alpha + \beta + \gamma \;=\; c. \end{aligned}$$

Da segunda equação, obtemos imediatamente

$$\gamma \;=\; b.$$

Somando a primeira e a terceira equações:

$$(\alpha - \beta + \gamma) + (\alpha + \beta + \gamma) \;=\; a + c \;\;\Longrightarrow\;\; 2\alpha + 2\gamma \;=\; a + c \;\;\Longrightarrow\;\; \alpha \;=\; \frac{a + c - 2b}{2}.$$

Subtraindo a primeira da terceira:

$$(\alpha + \beta + \gamma) - (\alpha - \beta + \gamma) \;=\; c - a \;\;\Longrightarrow\;\; 2\beta \;=\; c - a \;\;\Longrightarrow\;\; \beta \;=\; \frac{c - a}{2}.$$

Logo,

$$\boxed{\alpha \;=\; \frac{a+c-2b}{2}, \qquad \beta \;=\; \frac{c-a}{2}, \qquad \gamma \;=\; b}$$

são *determinados univocamente* pelos dados (a, b, c). Assim, existe um *único* $P(x) = \alpha x^2 + \beta x + \gamma$ que interpola os três pontos. Como verificação rápida, se tomarmos, por exemplo,

$$(a, b, c) \;=\; \big(0,\, 1,\, 0\big),$$

então

$$\alpha \;=\; \frac{0+0-2\cdot 1}{2} \;=\; -1, \qquad \beta \;=\; \frac{0-0}{2} \;=\; 0, \qquad \gamma \;=\; 1,$$

e obtemos

$$P(x) \;=\; 1 - x^2,$$

o que de fato satisfaz $P(-1) = 0$, $P(0) = 1$, $P(1) = 0$ de forma *única*.

6 Bônus: Experimentação com pontos de Chebyshev

Nesta seção comparamos a interpolação usando nós *equidistantes* e nós de *Chebyshev* no intervalo $[-1, 1]$, para a função de Runge

$$f(x) \;=\; \frac{1}{1+16x^2}.$$

É sabido que nós equidistantes podem induzir grandes oscilações nas extremidades (*fenômeno de Runge*), enquanto nós de Chebyshev reduzem esse efeito.

Comparação nós equidistantes Vs. Nós de chebyshev

O código abaixo: (i) define a função f, (ii) gera nós equidistantes e de Chebyshev, (iii) avalia o interpolador em uma malha fina.

A tabela 1 evidencia o impacto decisivo da escolha dos nós na interpolação polinomial. Com **nós equidistantes**, observa-se um erro máximo (L^∞) substancialmente maior ($\approx$ 1.18), típico do *fenômeno de Runge*, além de erros médios elevados (RMS $\approx$ 0.356 e MSE $\approx$ 0.127). Ao adotar **nós de Chebyshev-Lobatto**, a distribuição se concentra nas vizinhanças das extremidades do intervalo, reduzindo severamente as oscilações e, por consequência, *todas* as métricas de erro: o erro máximo cai em mais de uma ordem de grandeza ($\approx$ 0.075), o erro quadrático médio (RMS) e a MSE tornam-se duas a três ordens de magnitude menores, e a diferença de área (L^1) também reduz de ≈ 0.354 para ≈ 0.063. Esses resultados confirmam a recomendação clássica de empregar pontos de Chebyshev (ou formulações baricêntricas) para obter interpolação mais estável e acurada, especialmente próxima às bordas do domínio.

As duas figuras ilustram o efeito da escolha dos nós na qualidade da interpolação de Lagrange em $[-1, 1]$ para a função de Runge $f(x) = \frac{1}{1+16x^2}$. Na Figura 1, com nós equidistantes, observa-se o fenômeno de Runge: grandes oscilações nas extremidades do intervalo e degradação global da aproximação. Já a Figura 2, com nós de Chebyshev-Lobatto, concentra mais pontos perto das bordas e reduz significativamente as oscilações, produzindo uma aproximação muito mais fiel.

Tabela 1: Erros de interpolação em $[-1, 1]$ para $f(x) = \frac{1}{1+16x^2}$ com $N = 10$ (11 nós). Métricas: norma L^∞ (erro máximo), L^2 (RMS), L^1 (integral do erro absoluto) e MSE.

Nós	$\|f - P\|_\infty$	RMS	$\int_{-1}^{1} \|f - P\|\, dx$	MSE
Equidistantes	1.176894	0.355940	0.353778	0.126694
Chebyshev	0.074761	0.038331	0.062573	0.001469

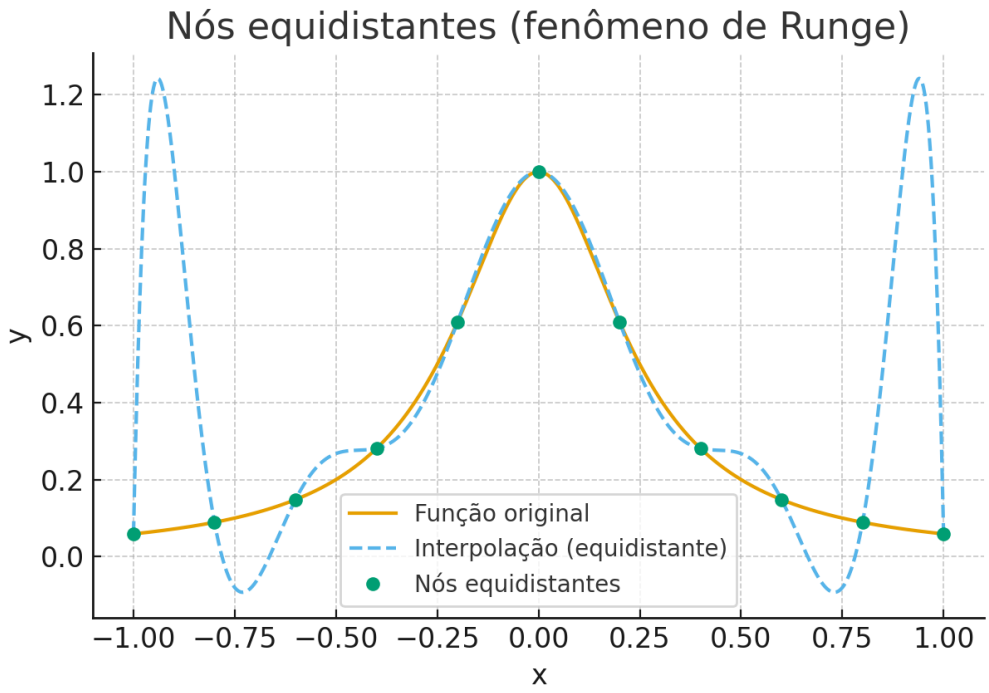

Figura 1: Interpolação com nós equidistantes em $[-1, 1]$: nota-se maior oscilação nas extremidades (fenômeno de Runge).

```python
import numpy as np
import matplotlib.pyplot as plt

from pathlib import Path
Path(".").mkdir(exist_ok=True)

def lagrange_interpolation(x_nodes, y_nodes, x_eval):
    x_nodes = np.array(x_nodes, dtype=float)
    y_nodes = np.array(y_nodes, dtype=float)
    x_eval  = np.array(x_eval,  dtype=float)
    n = len(x_nodes)
    y_eval = np.zeros_like(x_eval, dtype=float)
    for j in range(n):
        lj = np.ones_like(x_eval, dtype=float)
        for m in range(n):
            if m != j:
                lj *= (x_eval - x_nodes[m]) / (x_nodes[j] - x_nodes[m])
        y_eval += y_nodes[j] * lj
    return y_eval
```

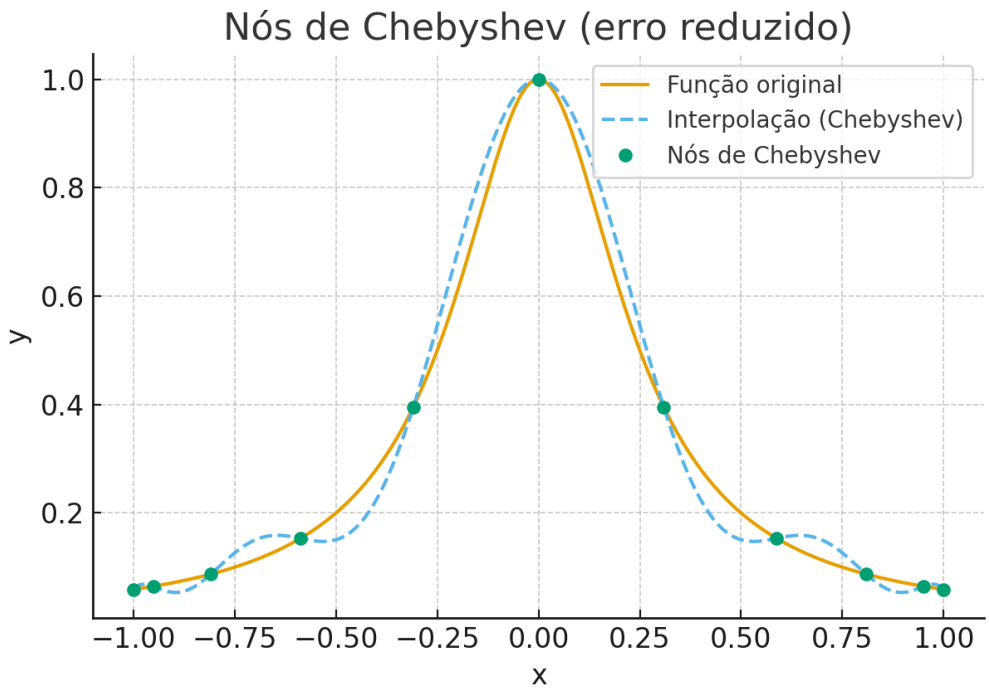

Figura 2: Interpolação com nós de Chebyshev-Lobatto: a aproximação melhora sensivelmente nas extremidades.

```
# Função-alvo (Runge)
f = lambda x: 1.0/(1.0 + 16.0*x**2)

# Malha de avaliação
x_eval = np.linspace(-1.0, 1.0, 800)

# Caso A: nós equidistantes
N_eq = 10  # número de segmentos => N_eq+1 nós
x_eq = np.linspace(-1.0, 1.0, N_eq+1)
y_eq = f(x_eq)
y_interp_eq = lagrange_interpolation(x_eq, y_eq, x_eval)

plt.figure(figsize=(6.2, 4.4))
plt.plot(x_eval, f(x_eval), label="Função original")
plt.plot(x_eval, y_interp_eq, linestyle="--", label="Interpolação (equidistante)")
plt.plot(x_eq, y_eq, "o", label="Nós equidistantes")
plt.title("Nós equidistantes (fenômeno de Runge)")
plt.xlabel("x")
plt.ylabel("y")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.savefig("fig_equidistante.png", dpi=200)
plt.close()

# Caso B: nós de Chebyshev (Chebyshev-Lobatto)
N_ch = 10
k = np.arange(N_ch+1)
x_ch = np.cos(np.pi * k / N_ch)   # em [-1,1]
y_ch = f(x_ch)
```

```
y_interp_ch = lagrange_interpolation(x_ch, y_ch, x_eval)

plt.figure(figsize=(6.2, 4.4))
plt.plot(x_eval, f(x_eval), label="Função original")
plt.plot(x_eval, y_interp_ch, linestyle="--", label="Interpolação (Chebyshev)")
plt.plot(x_ch, y_ch, "o", label="Nós de Chebyshev")
plt.title("Nós de Chebyshev (erro reduzido)")
plt.xlabel("x")
plt.ylabel("y")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.savefig("fig_chebyshev.png", dpi=200)
plt.close()
```

Listing 2: Código para gerar e salvar as figuras de comparação

7 Conclusão

Foi implementada a interpolação de Lagrange e explicada a correspondência direta entre o código e a fórmula teórica. Em seguida, por meio de experimentos, verificou-se a diferença qualitativa entre usar nós equidistantes (sujeitos ao fenômeno de Runge) e nós de Chebyshev (que reduzem o erro nas extremidades).

Reconhecimento de uso de LLM

Este relatório contou com o apoio de uma ferramenta de *Large Language Model* (LLM) para auxiliar na redação do texto, estruturação do relatório, geração de trechos em LaTeX, implementação de códigos em Python e produção de gráficos ilustrativos. Todo o material foi revisado e validado pelo autor antes da entrega final.

Referências

[1] L. N. Trefethen, *Spectral Methods in MATLAB*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2000.

[2] S. Ameh, "A quick visual guide to lagrange interpolation in numerical methods." https://medium.com/@amehsunday178/a-quick-visual-guide-to-lagrange-interpolation-in-numerical-methods-1fcc47ca218b, 2023. Medium article.