Universidade Estadual de Maringá

Relatório Calculo Seno Taylor e Padé

Gabriel Colli Pavan RA: 109882
João Pedro Paes Landim Alkamim RA: 112648
Sarah Anduca de Oliveira RA: 115506
Vitor Grabski Gomes RA: RA: 98369

Matemática Computacional

08 de Março de 2023

Sumário

1 Introdução

Este relatório descreve um trabalho de matemática computacional que teve como objetivo implementar algoritmos para o cálculo do seno usando os métodos de Taylor e Padé. O trabalho incluiu o cálculo do erro associado a cada método e a medição de suas respectivas velocidades de execução. Todo os métodos estão considerando o intervalo $\frac{-\pi}{4} < x < \frac{\pi}{4}$ e um grau de polinômio de valor 11.

2 Desenvolvimento

2.1 Linguagem e métodos

Utilizamos a linguagem Python, na versão 3.9.1, para implementação do algoritmo junto com a biblioteca math e matplotlib.

2.2 Série de Taylor

A série de Taylor é uma representação de uma função como uma soma infinita de termos polinomiais. A ideia é que podemos aproximar qualquer função suave como um polinômio de grau infinito, centrado em um determinado ponto. A fórmula geral da série de Taylor é dada por:

$$f(x) = f(a) + f'(a)(x-a) + (f''(a)/2!)(x-a)^2 + (f'''(a)/3!)(x-a)^3 + ... \quad (1)$$

Onde f(a) é o valor da função no ponto de centro a, f'(a) é a derivada primeira da função em a, f"(a) é a segunda derivada da função em a, e assim por diante. O símbolo "!"representa o fatorial.

Cada termo na série de Taylor é obtido a partir da derivada da função em a, e a série pode ser truncada em qualquer grau desejado. Quanto mais termos da série forem incluídos, melhor será a aproximação da função original. No entanto, como a série tem um número infinito de termos, na prática precisamos escolher um grau de truncamento.

Para este trabalho, que usaremos apenas até o grau 11 da série de Taylor. Isso significa que a aproximação da função que estamos considerando será dada pelos primeiros 11 termos da série. Para uma função suave o suficiente, isso geralmente é suficiente para obter uma boa aproximação.

Aqui está a fórmula da série de Taylor truncada até o grau 11:

$$\begin{aligned} f(x) = f(a) + f'(a)(x-a) + (f''(a)/2!)(x-a)^2 + (f'''(a)/3!)(x-a)^3 + \\ (f^4(a)/4!)(x-a)^4 + (f^5(a)/5!)(x-a)^5 + (f^6(a)/6!)(x-a)^6 + \\ (f^7(a)/7!)(x-a)^7 + (f^8(a)/8!)(x-a)^8 + (f^9(a)/9!)(x-a)^9 + \\ (f^{10}(a)/10!)(x-a)^{10} + (f^{11}(a)/11!)(x-a)^{11} \end{aligned} \tag{2}$$

Onde $f^n(a)$ representa a n-ésima derivada da função em a. Para calcular o seno de um ângulo x usando a série de Taylor, podemos usar a seguinte fórmula:

$$\begin{aligned} sen(x) = x - (x^3)/3! + (x^5)/5! - (x^7)/7! \\ +(x^9)/9! - (x^{11})/11! \end{aligned} \tag{3}$$

Note que a série de Taylor para o seno é uma série alternada, o que significa que os termos alternam entre positivos e negativos. Além disso, o valor absoluto dos termos da série diminui rapidamente à medida que n aumenta. Isso significa que podemos calcular uma boa aproximação para o seno de um ângulo usando apenas os primeiros termos da série.

Abaixo um código em Python a qual o x é o ângulo para calcular até um grau escolhido. Lembrando que **todos** os códigos foram escritos utilizando o esquema de Horner para minimizar o número de operações de multiplicação.

2.2.1 Algoritmo do Seno em Taylor

```python
def taylor_seno(x, grau):
    f = x
    seno = x
    for n in range(1, grau+1):
        f = -f * x**2 / ((2*n) * (2*n+1))
        seno += f
    return seno
```

2.3 Método Padé

O método Padé é uma técnica matemática utilizada para aproximar funções complexas por meio de frações racionais. Ele permite representar uma função de forma fracionária, usando uma combinação de polinômios para aproximar a função original.

A ideia por trás do método Padé é que, para uma função analítica f(z), é possível escrevê-la como uma série de potências em torno de um ponto z=a:

$$f(z) = a_0 + a_1(z-a) + a_2(z-a)^2 + ... \tag{4}$$

No entanto, essa série pode ser muito difícil ou impossível de calcular para alguns valores de z, especialmente quando a função é muito complexa. O método Padé contorna esse problema aproximando a função original por meio de uma fração racional de grau menor do que a série original.

A fórmula para uma aproximação Padé de ordem (n,m) de f(z) é dada por:

$$f(z) = \frac{P_n(z)}{Q_m(z)} \tag{5}$$

Onde $P_n(z)$ e $Q_m(z)$ são polinômios de graus n e m, respectivamente. Esses polinômios são determinados a partir da expansão em série de potências de f(z) em torno de z = a, juntamente com as suas primeiras n + m + 1 derivadas avaliadas em z = a.

Para exemplificar, vamos considerar a aproximação Padé 7/4 para a função f(z) em torno de z=0:

$$f(z) = \frac{e^z}{1 + z + z^2/2} \tag{6}$$

Para calcular a aproximação Padé 7/4, precisamos calcular as primeiras 12 derivadas de f(z) em z = 0. Depois, utilizamos essas derivadas para encontrar os coeficientes dos polinômios $P_7(z)$ e $Q_4(z)$ na fórmula acima. Os resultados são:

$$\begin{aligned} P_7(z) = 1 + z + z^2/2 + z^3/6 + z^4/24 + z^5/120 \\ +z^6/720 + z^7/5040 \\ Q_4(z) = 1 + z + 5z^2/12 + z^3/3 + z^4/24 \end{aligned} \tag{7}$$

Agora, substituindo esses polinômios na fórmula Padé, obtemos a aproximação Padé 7/4 de f(z):

$$f(z) = \frac{1 + z + z^2/2 + z^3/6 + z^4/24 + z^5/120 + z^6/720 + z^7/5040}{1 + z + 5z^2/12 + z^3/3 + z^4/24} \tag{8}$$

Lembrando que a série de potências para a função seno é dada por:

$$sen(x) = x - x^3/3! + x^5/5! - x^7/7! + ... \tag{9}$$

Podemos usar o método Padé para obter uma aproximação da função seno com uma fração racional de grau menor do que a série original. Calculando as primeira 12 derivadas de seno(x) em x = 0 temos:

$$\begin{aligned} f(x) &= seno(x) \\ f'(x) &= cos(x) \\ f''(x) &= -seno(x) \\ f'''(x) &= -cos(x) \\ f''''(x) &= seno(x) \\ f'''''(x) &= cos(x) \\ f''''''(x) &= -seno(x) \\ f'''''''(x) &= -cos(x) \\ f''''''''(x) &= seno(x) \\ f'''''''''(x) &= cos(x) \\ f''''''''''(x) &= -seno(x) \\ f'''''''''''(x) &= -cos(x) \end{aligned} \tag{10}$$

Agora, usamos essas derivadas para encontrar os coeficientes dos polinômios $P_7(x)$ e $Q_4(x)$ na fórmula Padé. O grau do polinômio $P_7(x)$ é 7 e o grau do polinômio $Q_4(x)$ é 4, então precisamos calcular as primeiras 12 derivadas de seno(x) para encontrar os coeficientes. Os coeficientes de $P_7(x)$ são dados por:

$$\begin{aligned} a_0 &= seno(0) = 0 \\ a_1 &= cos(0) = 1 \\ a_2 &= -seno(0)/2! = 0 \\ a_3 &= -cos(0)/3! = -1/6 \\ a_4 &= seno(0)/4! = 0 \\ a_5 &= cos(0)/5! = 1/120 \\ a_6 &= -seno(0)/6! = 0 \\ a_7 &= -cos(0)/7! = -1/5040 \end{aligned} \tag{11}$$

Então, temos:

$$P_7(x) = 0 + x + 0 + (-1/6)x^3 + 0 + (1/120)x^5 + 0 + (-1/5040)x^7 \tag{12}$$

Os coeficientes de $Q_4(x)$ são dados por:

$$\begin{aligned} b_0 &= 1 \\ b_1 &= 0 \\ b_2 &= -1/3! \\ b_3 &= 0 \\ b_4 &= 1/5! \end{aligned} \tag{13}$$

Então, temos:

$$Q_4(x) = 1 + 0x + (-1/6)x^2 + 0x^3 + (1/120)x^4 \tag{14}$$

Substituindo esses polinômios na fórmula Padé, obtemos a aproximação Padé 7/4 de seno(x):

$$seno(x) = \frac{P_7(x)}{Q_4(x)} \tag{15}$$

$$P_7(x) = 71x - 12873/160x3 + 412801/26880 * x5 - 3416435/114688x * *7$$
$$Q_4(x) = 1 - 205/72x2 + 1365/160 * x4 - 8165/5376 * x * *6 \tag{16}$$

Podemos implementar essa aproximação em python da seguinte forma:

2.3.1 Algoritmo do Seno em Padé

```
def pade_seno(x):
    p = [71/240, -71/1152, 71/8064, -71/72576, 71/798336]
    q = [1, -1/6, 1/120, -1/5040, 1/362880]
    u = x*x
    num = p[4]
    for i in range(3, -1, -1):
        num = p[i] + num * u
    den = q[4]
    for i in range(3, -1, -1):
        den = q[i] + den * u
    return x * num / den
```

2.4 Identificar os erros

Para observar a diferença entre a série de Taylor e a aproximação Padé na função seno, utilizamos a biblioteca matplotlib em Python. A ideia é plotar os valores de seno(x) calculados usando as duas técnicas em um mesmo gráfico, para que possamos comparar visualmente as curvas. Lembrando que o intervalo de ângulos usados é entre $\frac{-\pi}{4} < x < \frac{\pi}{4}$.

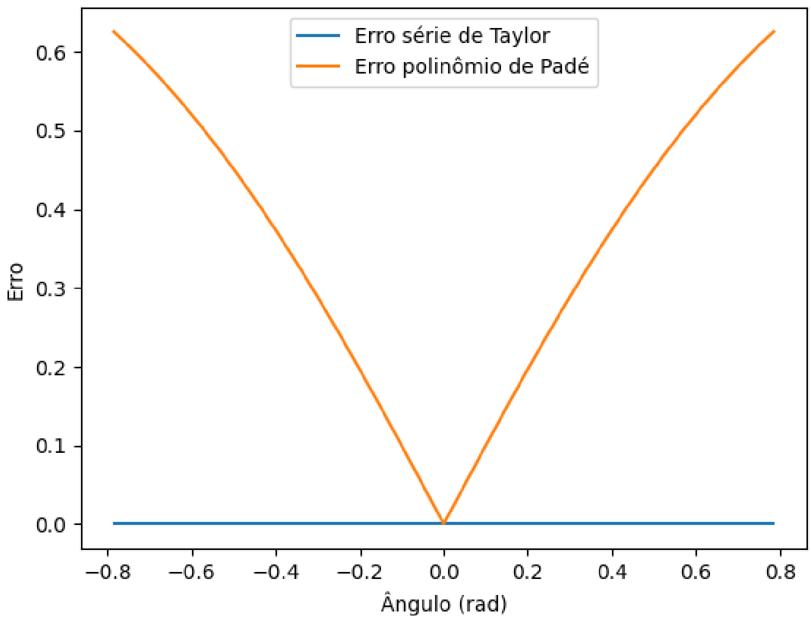
```
# Intervalo de ngulos de - /4 a  /4
x = np.linspace(-math.pi/4, math.pi/4, num=1000)

# Clculo dos valores do seno usando a funo seno, srie de
  Taylor e polinmio de Pad
seno = np.sin(x)
seno_taylor = np.array([taylor_seno(xi, 11) for xi in x])
seno_pade = pade_seno(x)
```

```
# Clculo dos erros
erro_taylor = np.abs(seno - seno_taylor)
erro_pade = np.abs(seno - seno_pade)

# Plotagem do grfico dos erros
plt.plot(x, erro_taylor, label='Erro srie de Taylor')
plt.plot(x, erro_pade, label='Erro polinmio de Pad')
plt.xlabel('ngulo (rad)')
plt.ylabel('Erro')
plt.legend()
plt.show()
```

Figura 1: Resultado do Matplotlib



Fonte: Autoria Própria

2.5 Calcular o Tempo

Para comparar o desempenho das funções de aproximação de seno por Padé e por série de Taylor, foi utilizado o módulo timeit da biblioteca padrão do Python. Esse módulo permite medir o tempo de execução de um trecho de código em Python, o que é especialmente útil para avaliar o desempenho de algoritmos.

Para o cálculo do tempo, foi definido um array com 1000 valores de $-\pi/4$ a $\pi/4$, usando a função linspace da biblioteca NumPy. Em seguida, foram medidos os tempos de execução da função pade_seno e da função taylor_seno para esses 1000 valores de entrada, usando a função timeit.timeit.

A função timeit.timeit recebe dois parâmetros: o primeiro é um objeto que representa o trecho de código que será executado e o segundo é o número de vezes que o trecho de código será executado. No caso do código acima, foram executadas 1000 vezes cada uma das funções de aproximação, para garantir que os resultados obtidos fossem estatisticamente significativos.

Os resultados da medição de tempo foram impressos na tela usando a função print. O tempo de execução da função pade_seno foi armazenado na variável tempo_pade e o tempo de execução da função taylor_seno foi armazenado na variável tempo_taylor. Os tempos foram formatados com a função f-string, que permite incluir valores de variáveis dentro de uma string formatada.

Com esses dados, podemos comparar o tempo de execução das duas funções e verificar qual delas é mais rápida. Esse tipo de análise é importante para determinar a eficiência de um algoritmo e para escolher o melhor método para uma determinada aplicação.

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

# Definir um array com 1000 valores de -/4 a /4
valores = np.linspace(-np.pi/4, np.pi/4, num=1000)

#  Medio  do tempo de  execuo  da  funo  pade_seno para 1000
   valores de entrada
tempo_pade = timeit.timeit(lambda: [pade_seno(x) for x in
   valores], number=1000)

#  Medio  do tempo de  execuo  da  funo  taylor_seno para
   1000 valores de entrada
tempo_taylor = timeit.timeit(lambda: [taylor_seno(x, 11) for x
   in valores], number=1000)

print(f"Tempo de  execuo  polinmio  de  Pad : {tempo_pade:.5f}")
print(f"Tempo de  execuo  srie  de Taylor: {tempo_taylor:.5f}")
```

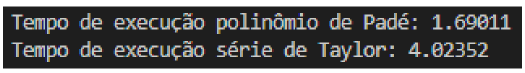
3 Resultados

O resultado da Figura 1 indica que a aproximação de Padé é mais precisa em torno do ponto em que é calculada (neste caso, em torno de zero), mas pode se tornar menos precisa à medida que nos afastamos desse ponto. Por outro lado, a série de Taylor pode se tornar menos precisa à medida que nos afastamos do ponto de expansão também, mas em uma taxa muito mais lenta do que a aproximação de Padé.

No caso específico da função seno, a série de Taylor converge para a função seno em todos os pontos do seu domínio, enquanto a aproximação de Padé tem maior precisão em torno de zero. Portanto, a série de Taylor é geralmente preferível ao calcular o seno em um intervalo mais amplo, enquanto a aproximação de Padé pode ser mais útil em aplicações em que é necessário um alto grau de precisão em torno de zero.

Utilizando os conceitos explicados na Seção 2.5 mostram que o método do polinômio de Padé é muito mais eficiente do que a série de Taylor para calcular o seno de valores em um determinado intervalo. O tempo de execução como mostra na Figura 2 do método de Padé (1.69011) é muito menor do que o tempo de execução do método de Taylor (4.02352) para calcular o seno de 1000 valores no intervalo de $-\pi/4$ a $\pi/4$. Isso significa que o método de Padé é muito mais rápido e eficiente do que o método de Taylor para esses cálculos, o que pode ser importante em aplicações onde a velocidade de cálculo é crítica. Vale lembrar que o tempo de execução pode variar de acordo com a máquina utilizada para o cálculo e outros fatores, mas, nesse caso, a diferença de tempo entre os dois métodos é significativa.

Figura 2: Resultado do Cálculo do Tempo



Fonte: Autoria Própria

4 Conclusão

Com tudo o que fora apresentado e trabalhado por nós, pudemos notar que dependendo do contexto e da precisão desejada, a escolha entre a aproximação de Padé e a série de Taylor pode variar. No caso do seno, a série de Taylor é geralmente preferível para cálculos em um intervalo mais amplo, enquanto a aproximação de Padé pode ser mais útil para aplicações que exigem alta precisão em torno de zero. Além disso, o método do polinômio de Padé é mais eficiente e rápido do que a série de Taylor para calcular o seno de valores em um determinado intervalo, o que pode ser importante em aplicações onde a velocidade de cálculo é crítica. No entanto, é importante lembrar que o tempo de execução pode variar dependendo da máquina utilizada para o cálculo e outros fatores, e a escolha entre os dois métodos deve ser feita com base nas necessidades específicas da aplicação em questão.

5 Referências

Anton, H., Bivens, I. (2002). Cálculo - Uma Variável. Bookman Editora.

Boas, R. P. (1954). Rational Approximation of Real Functions. American Mathematical Society.

Corless, R. M., Stewart, G. W., Jeffrey, D. J. (1996). A practical guide to Pade approximants. SIAM.

Folland, G. B. (1999). Real Analysis: Modern Techniques and Their Applications. Bookman Editora.

Lorentz, G. G., Rvachev, L. S. (1996). Rational Quadratic and Cubic Approximations. Academic Press.

Stewart, J. (2013). Cálculo - Volume 1. Cengage Learning.

Trefethen, L. N. (2013). Approximation Theory and Approximation Practice. SIAM.