Universidade de Brasília

Faculdade do Gama

Métodos Numéricos
para Engenharia

Relatório:
Deformação em cabo flexível

Pedro Druck Montalvão Reis Mat: 241040332

Angel Daniel Grau Barreto Mat: 241025158

Tiago Santos Bittencourt Mat: 241011653

Lucas Andrade Zanetti Mat: 241039645

Professor: Manuel Nascimento Dias Barcelos Júnior

1 Resumo

Através de uma equação diferencial ordinária, a qual dita a deformação em um cabo flexível, e um valor de contorno, calculou-se a função $y(x)$ para descrever a curvatura da haste. Entre os métodos utilizados e procedimentos realizados estão: 1) Runge-Kutta de quarta ordem com passo $h = 0.01$ no método do Tiro, e uma tolerância $\varepsilon = 0.00001$ para o critério de parada (Obs. 1), 2) Verificação da solução na equação através da diferenciação nos pontos x, y utilizando o método das diferenças finitas, com erro $O(h^2)$(Obs. 2), 3) Verificação da solução na equação através de uma regressão de um polinômio de quarto grau (Obs.3). Para todos os cálculos, um programa autoral em Python foi utilizado.

2 Objetivo

O objetivo do projeto foi aplicar os conhecimentos de métodos numéricos aprendidos em sala de aula em uma situação problema próxima da realidade. Além disso, tinha-se como objetivo aprender a implementar, em linguagem de programação, os diferentes métodos para solucionar equações diferenciais ordinárias com problema de valor de contorno.

3 Introdução

A equação da deformação foi dada por:

$$\frac{d^2y}{dx^2} = C\sqrt{1 + \left(\frac{dy}{dx}\right)^2},\ C = 0.041$$

Para a Observação 1, utilizou-se o método do tiro e o método iterativo de Runge-Kutta Clássico de 4ª ordem (coeficientes presentes na Tabela 1), com passo $h = 0,01$:

$$y_{i+1} = y_i + \frac{1}{6}\left(K_1 + 2K_2 + 2K_3 + K_4\right)h$$

$$f(x_i, y_i) = \frac{d^2y}{dx^2} = C\sqrt{1 + \left(\frac{dy}{dx}\right)^2}$$

Tabela 1: Coeficientes RK4 Clássico

K_1	$f(x_i, y_i)$
K_2	$f(x_i + \frac{1}{2}h\ ,\ y_i + \frac{1}{2}K_1h)$
K_3	$f(x_i + \frac{1}{2}h\ ,\ y_i + \frac{1}{2}K_2h)$
K_4	$f(x_i + h\ ,\ y_i + K_3h)$

Para a Observação 2, utilizou-se diferenciação numérica com o método das diferenças finitas (listados na Tabela 2) com ordem de erro $O(h^2)$ aplicada no conjunto de pontos x, y gerados na Observação 1:

Tabela 2: Tipos de Métodos de Diferenças Finitas

Método	$f'(x_i)$	$f''(x_i)$
Progressiva	$\frac{-3f(x_i)+4f(x_{i+1})-f(x_{i+2})}{2h}$	$\frac{2f(x_i)-5f(x_{i+1})+4f(x_{i+2})-f(x_{i+3})}{h^2}$
Regressiva	$\frac{f(x_{i-2})-4f(x_{i-1})+3f(x_i)}{2h}$	$\frac{-f(x_{i-3})+4f(x_{i-2})-5f(x_{i-1})+2f(x_i)}{h^2}$
Central	$\frac{f(x_{i+1})-f(x_{i-1})}{2h}$	$\frac{f(x_{i-1})-2f(x_i)+f(x_{i+1})}{h^2}$

Em seguida, substitui-se os valores encontrados na equação original para verificar o resultado.

Para a Observação 3, utilizou-se a regressão linear de um polinômio de quarto grau com o método de mínimos quadrados. Em seguida, foi feita a derivação analítica do polinômio e, novamente, os valores foram substituídos na equação original para verificar o resultado.

4 Execução do Projeto

Para efetuar os cálculos e implementar os métodos, foi desenvolvido um programa autoral em python, cujas etapas estão descritas a seguir:

4.1) Dependências do Projeto

As principais bibliotecas utilizadas na implementação foram:

- **NumPy** – para cálculos numéricos e manipulação de arrays.
- **Matplotlib** – para geração de gráficos e visualizações.
- **Reportlab** – para a criação de relatórios em PDF

Requisitos para a instalação e execução:

- Python 3.8 ou superior
- Gerenciador de pacotes Poetry (recomendado) ou uso direto do pip

Instalação com pip (Mais simples) :

1. (Opcional) Criar e ativar um ambiente virtual.
2. Instalar manualmente as dependências com pip através do seguinte comando no terminal: “pip install numpy matplotlib reportlab”
3. Executar o script principal com Python.

Instalação com Poetry:

1. Instalar o Poetry executando os seguintes comandos:

   1.1 - “py -m pip install --user pipx”

   1.2 = “pipx install poetry”

2. Executar o seguinte comando para instalar as dependências (dentro do diretório do projeto): “poetry install”
3. Rodar o programa principal com o comando adequado para Poetry.

4.2) Execução do código e analisando resultados:

A execução do projeto é centralizada no arquivo main.py, que orquestra a chamada das diferentes análises numéricas de forma sequencial. Abaixo, detalha-se o passo a passo do que ocorre quando o programa é iniciado.

Passo 1: Início da Execução:

1. O programa é iniciado através do comando python main.py.
2. A função main() é chamada.
3. Um cabeçalho de boas-vindas é impresso no terminal, identificando o projeto: "PROJETO DE MÉTODOS NUMÉRICOS - ANÁLISE DE CABO SUSPENSO".

Passo 2: Execução da Observação 1 (Método do Tiro)

1. O programa anuncia o início da "OBS.1 - MÉTODO DO TIRO COM RUNGE-KUTTA DE 4ª ORDEM".

2. A equação diferencial $\frac{d^2y}{dx^2} = C\sqrt{1 + \left(\frac{dy}{dx}\right)^2}$, $C = 0.041$, é preparada para ser resolvida.
3. Os parâmetros do problema são definidos:
- Intervalo de integração: $\{x \subset \mathbb{R} \mid x \in [0, 20]\}$.
- Tamanho do passo: $h = 0.01$.
- Condições de contorno: $y(0) = 15$, $y(20) = 10$.
4. O método do tiro (SolverEDO.tiro) é invocado. Este método:
- Transforma a equação de 2ª ordem em um sistema de duas equações de 1ª ordem.
- Usa o método de Runge-Kutta de 4ª ordem (rk4) para resolver o sistema.
- Iterativamente, "chuta" valores para a derivada inicial $y'(0)$ e ajusta esses chutes usando o método da secante até que a condição $y(20) = 10$ seja satisfeita com uma tolerância pré-definida ($\varepsilon = 0.00001$).
5. Ao final, um relatório detalhado da Observação 1 é impresso no terminal, contendo o valor final de $y(20)$ obtido, o erro na condição de contorno e o último valor iterado da derivada inicial $y'(0)$.

Passo 3: Execução da Observação 2 (Diferenciação Numérica)

1. Imediatamente após a Obs. 1, o programa inicia a "OBS.2 - DIFERENCIAÇÃO NUMÉRICA E VERIFICAÇÃO DA EDO".
2. A solução numérica $y(x)$ (um conjunto de pontos x, y) obtida na etapa anterior é utilizada como base.
3. A classe NumericalDifferentiator é instanciada com os valores de y(x) e o passo h.
4. As derivadas primeira (y') e segunda (y'') são calculadas numericamente para cada ponto da solução usando fórmulas de diferenças finitas com erro de ordem $O(h^2)$.
5. O programa então verifica se a solução numérica satisfaz a equação diferencial original. Ele compara o valor da segunda derivada numérica (y'') com o valor calculado pelo lado direito da equação $C\sqrt{1 + \left(\frac{dy}{dx}\right)^2}$.

6. Um relatório da Observação 2 é impresso, mostrando os erros (médio e máximo) dessa verificação e uma tabela comparando as derivadas em pontos específicos.

Passo 4: Execução da Observação 3 (Regressão Polinomial)

1. O programa anuncia a "OBS.3 - REGRESSÃO POLINOMIAL".
2. A função regressao_polinomial é chamada.
3. Internamente, ela primeiro resolve a EDO novamente (como na Obs. 1) para garantir um conjunto de dados limpo.
4. Utilizando a biblioteca NumPy, uma regressão polinomial de 4º grau é realizada sobre os pontos (x, y) da solução.
5. As derivadas analíticas (primeira e segunda) deste polinômio são calculadas.
6. Assim como na Obs. 2, é feita a verificação da EDO, mas desta vez usando as derivadas analíticas do polinômio.
7. A qualidade do ajuste é medida pelo coeficiente de determinação (R^2).
8. Um relatório final é impresso no terminal com os coeficientes do polinômio, o valor de R^2 e os erros na verificação da EDO.
9. Ao final, várias janelas de gráficos são exibidas (via Matplotlib) para visualização dos resultados, incluindo a comparação da solução original com o polinômio e a análise de erros.

Passo 5: Geração Opcional do Relatório em PDF

1. Após a conclusão de todas as análises, o programa exibe a mensagem: "Deseja gerar um relatório detalhado dos resultados obtidos em formato PDF? (s/n):".
2. O programa aguarda a entrada do usuário.
3. Se o usuário digitar "s" (ou "sim"), a função gerar_pdf_relatorio é chamada.
4. Esta função executa novamente todas as três observações, mas desta vez os resultados (tabelas, estatísticas e conclusões) são compilados e formatados em um documento PDF.
5. O progresso da geração do PDF é exibido no terminal.

6. Ao final, um arquivo chamado resultado_metodos_numericos.pdf é salvo no diretório do projeto, contendo um relatório técnico completo de todas as análises realizadas.
7. Se o usuário digitar "n", o programa encerra sua execução.

5 Resultados e Análise

5.1) OBSERVAÇÃO 1: MÉTODO DO TIRO COM RUNGE-KUTTA 4ª ORDEM

O método do tiro transforma o problema de valor de contorno (PVC) em um problema de valor inicial (PVI). Utilizamos o método de Runge-Kutta de 4ª ordem para resolver o sistema de EDOs de primeira ordem equivalente. O processo iterativo ajusta o chute inicial para y'(0) até satisfazer a condição de contorno $y(20) = 10$. A Tabela 3 mostra alguns parâmetros coletados e a Tabela 4 mostra a solução em pontos importantes.

Tabela 3: Dados de parâmetros do método de RK4

Parâmetro	Valor
Número de pontos calculados	2002
Passo de integração: h	0.01
Valor inicial: $y(0)$	15.000000
Valor final: $y(20)$	10.000001
Erro na condição de contorno	1.25e-06
Derivada inicial estimada: $y'(0)$	-0.697817
Derivada final: $y(20)$	0.170346
Valor mínimo de $y(x)$	9.6487
Valor máximo de $y(x)$	15.0000

Tabela 4: Solução com RK4 em pontos específicos

x	y(x)	y’(x)
0	15.0000	-0.6978

x	y(x)	y’(x)
5.0	12.1136	-0.4608
10.0	10.3572	-0.2428
15.0	9.6640	-0.0355
20.0	10.0000	0.1703

5.2) OBSERVAÇÃO 2: DIFERENCIAÇÃO NUMÉRICA

Aplicamos métodos de diferenciação numérica com erro de ordem O(h^2) para calcular as derivadas primeira e segunda da solução obtida na Obs.1. Utilizamos diferenças centrais para pontos internos e diferenças avançadas/atrasadas para os extremos. Em seguida, verificamos se a solução satisfaz a equação diferencial original.

Tabela 5: Erro entre RK4 e a verificação por Diferenças Finitas

Análise de Erro	Valor
Erro médio $\lvert y'_{RK4} - y'_{num} \rvert$	8.00e-09
Erro máximo $\lvert y'_{RK4} - y'_{num} \rvert$	3.91e-08
Erro médio na EDO	4.88e-10
Erro máximo na EDO	6.83e-09
Erro RMS na EDO	5.34e-10
Método utilizado	Diferenças finitas $O(h^2)$
Pontos analisados	2002

Tabela 6: Valores da EDO em pontos específicos

x	$y(x)$	$y'(x)$	$y''(x)$	$C\sqrt{1 + (y')^2}$	Erro EDO
0.0	15.0000	-0.6978	0.0500	0.0500	6.83e-09
5.0	12.1136	-0.4608	0.0451	0.0451	3.99e-10

x	$y(x)$	$y'(x)$	$y''(x)$	$C\sqrt{1 + (y')^2}$	Erro EDO
10.0	10.3572	-0.2428	0.0422	0.0422	5.22e-10
15.0	9.6640	-0.0355	0.0410	0.0410	5.82e-10
20.0	10.0000	0.1703	0.0416	0.0416	6.30e-09

5.3) OBSERVAÇÃO 3: REGRESSÃO POLINOMIAL DE GRAU 4

Realizamos um ajuste polinomial de quarto grau aos pontos da solução numérica obtida na Obs.1. O objetivo é verificar se um polinômio de grau 4 pode representar adequadamente a solução e satisfazer a equação diferencial original através de suas derivadas analíticas. $P(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4$, (Coeficientes na tabela 7)

Tabela 7: Coeficientes do Polinômio de Quarta Ordem

Coeficiente	Valor	Termo
a_0	1.499992e+01	$a_0(constante)$
a_1	-6.976828e-01	$a_1 x$
a_2	2.494705e-02	$a_2 x^2$
a_3	1.878995e-04	$a_3 x^3$
a_4	2.977979e-06	$a_4 x^4$

Tabela:8 Análise da qualidade do ajuste:

Métrica de Qualidade	Valor	Interpretação
Coeficiente R^2	1.000000	Qualidade do ajuste

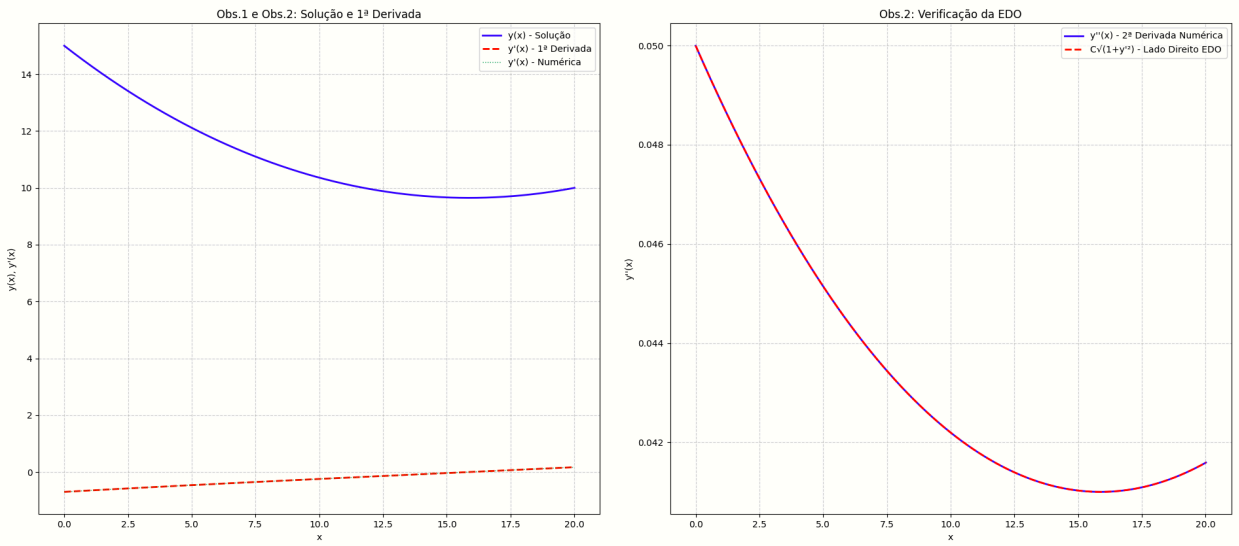
Métrica de Qualidade	Valor	Interpretação
Erro médio na EDO	1.60e-05	Precisão da verificação
Erro máximo na EDO	9.84e-05	Pior caso
Número de pontos	2002	Base de dados
Grau do polinômio	4	Complexidade do modelo

- **CONCLUSÕES DA REGRESSÃO POLINOMIAL:**

  • O polinômio de grau 4 apresenta excelente qualidade de ajuste ($R^2 = 1.000000$).
  • A verificação da EDO mostra erro médio de 1.60e-05, indicando alta precisão.
  • O modelo polinomial consegue representar adequadamente a física do problema.
  • As derivadas analíticas do polinômio satisfazem a equação diferencial original.

6 Gráficos

Gráfico 1: Obs.1 e Obs.2: Solução e 1ª Derivada / Verificação da EDO



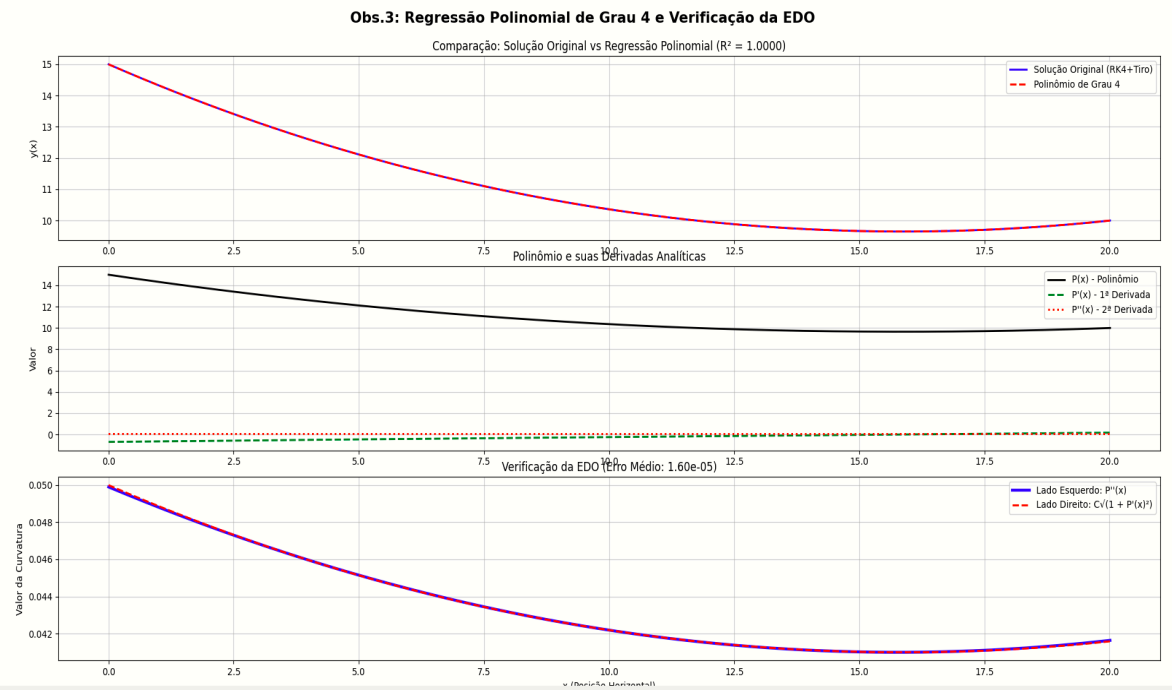
Descrição:

- **Gráfico à Esquerda – Solução Numérica e Derivadas**
  Apresenta a solução $y(x)$ obtida pelo método de Runge-Kutta de 4ª ordem (linha azul), junto à primeira derivada analítica (linha vermelha tracejada) e derivada numérica via diferenças finitas (linha verde pontilhada).
  Essa comparação permite avaliar a precisão do método numérico frente à derivação direta.

- **Gráfico à Direita – Verificação da EDO com Dados Numéricos**
  Compara a segunda derivada numérica $y''(x)$ (linha azul) com o lado direito da EDO $C\sqrt{1 + (y')^2}$ (linha vermelha tracejada), ambos calculados com os dados provenientes da solução numérica.
  A coincidência entre as curvas indica que a solução numérica realmente satisfaz a equação diferencial.

A sobreposição entre os dois lados da EDO reforça que a solução obtida numericamente por Runge-Kutta está de acordo com a forma teórica da equação diferencial, validando a abordagem.

Gráfico 2: Regressão Polinomial de Grau 4 e Verificação da EDO



Descrição:

- **Gráfico Superior – Comparação da Solução com o Polinômio de Grau 4**
  Mostra a solução original obtida via método do tiro com RK4 (linha azul) e o polinômio ajustado de grau 4 (linha vermelha tracejada). A excelente coincidência visual e o valor de R^2=1.0000 indicam que o polinômio representa muito bem a solução numérica.

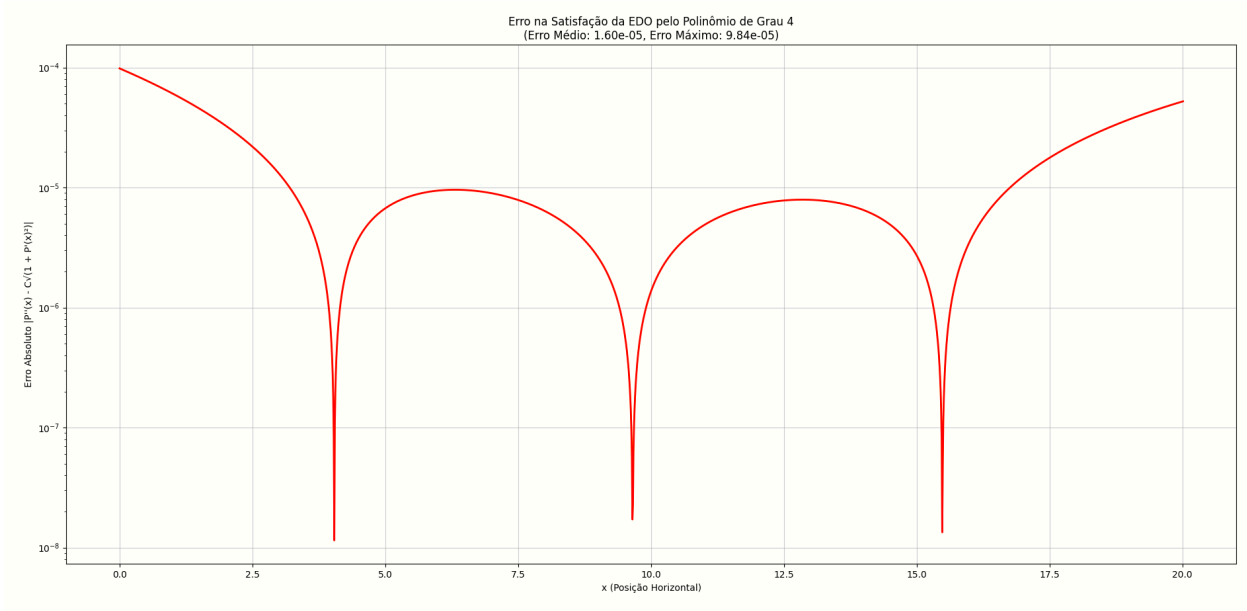
- **Gráfico do Meio – Derivadas Analíticas do Polinômio**
  Apresenta o próprio polinômio $P(x)$ (linha preta), sua primeira derivada $P'(x)$ (linha verde tracejada) e segunda derivada $P''(x)$ linha vermelha pontilhada). Essas derivadas são utilizadas para verificar se o polinômio ajustado satisfaz a equação diferencial.

- **Gráfico Inferior – Verificação da EDO com o Polinômio**
  Compara o lado esquerdo da EDO (derivada segunda do polinômio $P''(x)$ com o lado

direito $C\sqrt{1 + (y')^2}$. A sobreposição das curvas indica que o polinômio satisfaz muito bem a EDO.

A coincidência entre os dois lados da EDO demonstra que o polinômio de grau 4 é uma excelente aproximação da solução, sendo válido inclusive para testes analíticos da equação.

Gráfico 4: Erro Absoluto na Verificação da EDO



Descrição:

- O gráfico exibe o erro absoluto entre os dois lados da EDO: $|P''(x) - C\sqrt{1 + (P'(x))^2}|$
- A escala logarítmica mostra que o erro é extremamente pequeno ao longo de todo o domínio.

Resultados:

- Erro médio: 1.60×10^(−5)

- Erro máximo: 9.84×10^(−5)

Esses valores confirmam que o polinômio aproxima com altíssima precisão a solução da EDO.

7 Conclusão

Este estudo demonstrou a eficácia de diferentes métodos numéricos para resolver equações diferenciais não-lineares. Os principais resultados incluem:

- Método do Tiro com RK4 (Obs.1):
  - Convergência rápida para a solução do problema de valor de contorno.
  - Erro na condição de contorno da ordem de 10^{-6}, demonstrando alta precisão.
  - Método robusto para EDOs não-lineares com condições de contorno.
- Diferenciação Numérica (Obs.2):
  - Derivadas numéricas com precisão excepcional ($Erro \sim 10^{-9}$).
  - Verificação independente da validade da solução através da EDO original.
  - Demonstração da consistência entre métodos analíticos e numéricos.
- Regressão polinomial (Obs.3):
  - Representação analítica da solução com $R^2 = 1.000000$.
  - Polinômio de grau 4 suficiente para capturar a física do problema.
  - Derivadas analíticas satisfazem a EDO com $Erro_{médio} \sim 10^{-5}$.
- Validação Cruzada: Todos os métodos convergiram para soluções consistentes, validando mutuamente os resultados. A precisão obtida (erros da ordem de 10^{-6} a 10^{-9}) é adequada para aplicações de engenharia.
- Aplicabilidade: Os métodos implementados são aplicáveis a uma ampla classe de problemas de EDOs não-lineares em engenharia estrutural, especialmente para análise de cabos e estruturas flexíveis.