Aula 01 - Introdução ao método de diferenças finitas*

Daniel Coelho†

Universidade do Estado do Rio de Janeiro (UERJ)
Depto. de Engenharia Mecânica (MECAN)
Programa de Pós-Graduação em Engenharia Mecânica (PPG-EM)

16 de setembro de 2020

*FEN03-05100: Tópicos Especiais em Engenharia Mecânica I
†Aluno de mestrado (PPG-EM)

Índice

Introdução

Modelagem Computacional

- Diversos fenômenos físicos podem ser descritos através de **modelos matemáticos** a partir dos quais podemos obter informações e predições do que ocorre experimentalmente;
- Em casos mais complexos, a dinâmica desses sistemas físicos pode ser descrita por **equações diferenciais**, das quais nem sempre podemos obter facilmente uma **solução analítica** (a partir de uma integração direta);
- Portanto, é necessário **aproximar** essas equações (contínuas) por outras (discretas) que podemos resolver computacionalmente, obtendo assim uma **solução numérica**.

Métodos Numéricos

- Existem diferentes maneiras de **discretizar** essas **equações diferenciais**, que podem ser ordinárias (**EDO**) ou parciais (**EDP**);
- Essas maneiras são conhecidas como **métodos numéricos** (método de diferenças finitas, método dos volumes finitos, método dos elementos finitos, métodos espectrais, etc);
- Na primeira parte desse curso serão apresentadas algumas das técnicas envolvidas no **Método de Diferenças Finitas (MDF)** juntamente à algumas das suas aplicações.

Ferramenta: *Python*

- Para a finalidade do curso, iremos utilizar a linguagem de programação *Python* como **ferramenta computacional**.
- É uma linguagem interpretada (*script*) baseada em uma linguagem de mais baixo nível (**C**).
- Quanto mais alto o nível de sintaxe da linguagem de programação, mais facilmente ela pode ser entendida por programadores humanos.

Método das Diferenças Finitas (MDF)

Método das Diferenças Finitas (MDF)

Em análise numérica, o **Método das Diferenças Finitas (MDF)** é um método de resolução de equações diferenciais que se baseia na aproximação de derivadas por **diferenças finitas**.

Relembrando a própria definição de derivada:

Se $y = u(x)$ é uma função real de uma variável real, sua derivada é definida por

$$\frac{d}{dx}u(x) = u'(x) = \lim_{h \to 0} \frac{u(x+h) - u(x)}{h}, \tag{1}$$

onde $h = \Delta x$ e $u'(x)$ pode ser interpretada como a taxa de variação de y em relação a x.

Método das Diferenças Finitas (MDF)

Exemplo:
Considere $u(x) = x^4 - 10x^2 - 4x - 1$. Analiticamente,

$$u'(x) = 4x^3 - 20x - 4. \tag{2}$$

No entanto, como podemos calcular $u'(x)$ numericamente?

- *Repare que x, $u(x)$ e $u'(x)$ são **contínuas**, mas se quisermos representá-las manualmente ou computacionalmente devemos efetuar cálculos **discretos**.*

- *Porém, $h \to 0$ significa armazenarmos $N \to \infty$ números, o que é inviável na prática. Por isso, devemos atribuir um valor **finito** a h ($h = 0.01$, $h = 0.001$, por exemplo).*

Método das Diferenças Finitas (MDF)

Portanto, vamos aproximar a definição de derivada para

$$u'(x) \approx \frac{u(x+h) - u(x)}{h}. \qquad (3)$$

O que é equivalente a definirmos como

$$u'(x) = \frac{u(x+h) - u(x)}{h} + \mathcal{O}(h^p), \qquad (4)$$

onde o termo $\mathcal{O}(h^p)^*$ representa o erro cometido nessa **discretização**. Repare que $u(x+h) - u(x)$ é uma **diferença** e é tão menor quanto nós desejamos (afinal, h é arbitrário).

Hipótese: O erro é proporcional a alguma potência p de h.

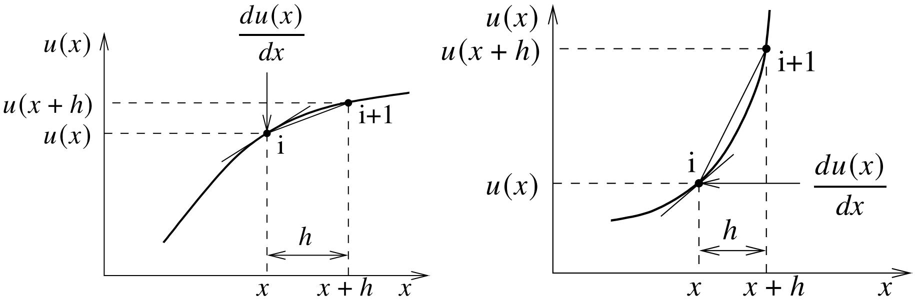
---

*O símbolo $\mathcal{O}(...)$ significa "da ordem de (...)"

Método das Diferenças Finitas (MDF)

- *Comumente escrevemos $u(x+h)$ e $u(x)$ como u_{i+1} e u_i, respectivamente.*

Graficamente:



Método das Diferenças Finitas (MDF)

Da Eq.(2) temos a seguinte EDO:

$$u'(x) = b(x) \tag{5}$$

onde $b(x) = 4x^3 - 20x - 4$. Usando a Eq.(3) para $u'(x)$, obtemos:

$$\frac{u(x+h) - u(x)}{h} = b(x) + \mathcal{O}(h^p) \tag{6}$$

Exercício 1

Reescrevendo a Eq.(6),

$$u(x+h) = u(x) + h\,b(x) + \mathcal{O}(h^{p+1}) \tag{7}$$

Dessa forma, teremos que partir de uma condição inicial arbitrária $u(x_0)$.

Objetivo:

- Obter a solução numérica de $u(x)$ e comparar com a solução analítica;
- Calcular o erro relativo entre elas.

Exercício 1 – *Solução*

Computacionalmente, a Eq.(7) pode ser descrita da seguinte forma,

$$u_{i+1} = u_i + h * b_i \ , \tag{8}$$

onde h tem um valor positivo finito a ser escolhido e u, b são vetores a serem armazenados.

- *Portanto, devemos construir um loop (de tamanho n), com o cálculo de novos valores de u, que parta da condição inicial arbitrária* u_1*:*

$$\begin{aligned} u_2 &= u_1 + h * b_1 \\ u_3 &= u_2 + h * b_2 \\ &\vdots \\ u_n &= u_{N-1} + h * b_{N-1} \end{aligned}$$

- *Ao final, teremos u, que é a solução numérica do Exercício 1.*

Exercício 2

Reescrevendo a Eq.(6) de outra forma,

$$\Lambda_x u(x) = b(x) \tag{9}$$

obtendo assim o seguinte sistema de equações lineares:

$$\mathbf{A}_{n\times n}\mathbf{u}_n = \mathbf{b}_n \tag{10}$$

onde $\mathbf{A}_{n\times n}$ é a representação discreta do operador Λ_x e n é o número de pontos. Como se trata de uma EDO de 1^a ordem, basta a informação $u_1 = b(x=0) = b_1$ ser concedida ao sistema.

Objetivo:

- Obter a solução numérica de $u(x)$ e comparar com a solução analítica;
- Calcular o erro relativo entre elas.

Exercício 2 – *Solução*

Reescrevendo a Eq.(6) de outra forma,

$$\Lambda_x u(x) = b(x) \tag{11}$$

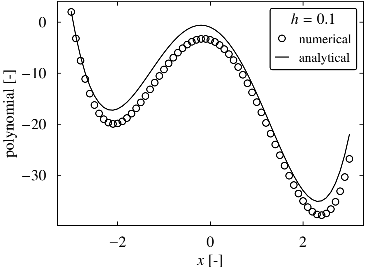
- *O sistema de equações lineares $n \times n$, $\boldsymbol{Au} = \boldsymbol{b}$, é representado da seguinte forma:*

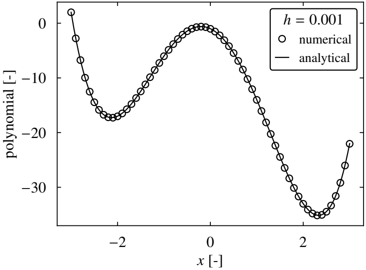
$$\frac{1}{h}\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-2} \\ u_{n-1} \\ u_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{bmatrix}$$

- *Para obter a solução deste sistema, $\boldsymbol{u} = \boldsymbol{A}^{-1}\boldsymbol{b}$, é necessária uma rotina de método direto ou iterativo. No python, utilizaremos o* `linalg.solve()` *(decomposição LU)*

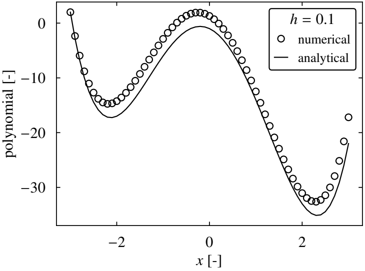
Resultados – *Soluções numéricas*

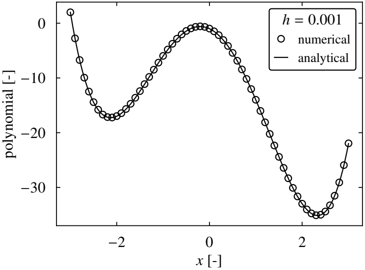
- Exercício 1:





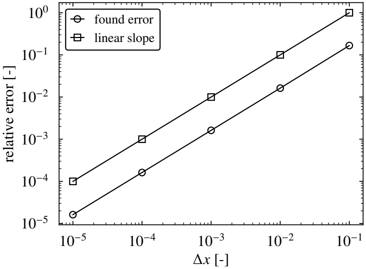
- Exercício 2:



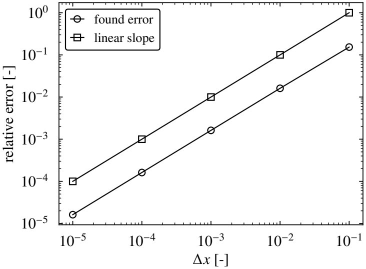


Resultados – *Erro relativo*

- Exercício 1:



- Exercício 2:



Conclusões

Conclusões

O MDF é um método para aproximação de derivadas e nessa disciplina, estará no contexto de discretização de equações diferenciais;

Nos exercícios, a solução numérica foi obtida com "duas abordagens":

- Exercício 1: **Problema de Valor Inicial (PVI)**;
- Exercício 2: **Problema de Valor de Contorno (PVC)**.

Em ambos, observamos que:

- A **solução numérica** se aproxima da **solução analítica** quando $h \longrightarrow 0$ ($N \longrightarrow \infty$);
- O **erro de discretização** decai a uma taxa p de h: $\mathcal{O}(h^p)$.

dúvidas, comentários, *feedbacks*,...

Contate-nos!

Daniel Coelho
danielcoelho.uerj@gmail.com
coelho.daniel@posgraduacao.uerj.br

Luís H. Carnevale
lh.carnevale@gmail.com

website
dancoelho.github.io

GESAR
Group for Environmental Studies in Reservoirs

www.gesar.uerj.br