

PYTHON PARA ENGENHARIA MECÂNICA

Preparado por: Prof. Gustavo Anjos
14 de Agosto de 2017

Resumo. Este texto de nível introdutório tem como objetivo familiarizar o aluno de graduação na criação de ferramentas e solução de problemas em engenharia mecânica com a utilização de uma moderna linguagem de computador - Python. Este texto não se restringe a futuros engenheiros mecânicos, mas também pode ser usado por alunos que desejam obter um conhecimento inicial de solução de problemas diferenciais e como introdução à construção de códigos numéricos mais elaborados.

Conteúdo

1	Introdução	2
1.1	Considerações iniciais	2
1.2	Importância da linguagem Python na Engenharia Mecânica	2
1.3	Orientação a objetos	2
1.4	Solução de problemas diferenciais	2
1.5	Importância da validação do código numérico	3
1.6	Controle de versões - GIT	4
1.7	Editor de texto - VIM	4
2	Mecânica de massas pontuais	5
2.1	Movimento Horizontal de um Carrinho	5
2.2	Velocidade terminal de uma gota	7
2.3	Lançamento de projétil	8
2.4	Sistema massa-mola	9
2.5	Sistema massa-mola dissipativo	10
2.6	Sistema massa-mola vertical	10
2.7	Pêndulo simples	11
3	Mecânica do contínuo	12
3.1	Geração de malha 1D	12
3.2	Geração de malha 2D	14
3.3	Solução de problema térmico permanente 1D	15
3.4	Solução de problema térmico permanente com geração de calor 1D	18
3.5	Solução de problema térmico transiente 1D	18
3.6	Solução de problema térmico transiente 1D com geração de calor	19
3.7	Solução de equação de transporte	19
3.8	Escoamento com partículas - solução numérica	21

1 Introdução

1.1 Considerações iniciais

Este texto foi elaborado para atender as necessidades de ensino de programação numérica para alunos de Transmissão de Calor II do 8º período de Engenharia Mecânica da Universidade do Estado do Rio de Janeiro. O texto está em contínuo desenvolvimento e ainda carece de informações essenciais para o desenvolvimento de todos os exemplos apresentados. Por isso, o autor pede a compreensão do estudante para falta de dados e exemplos mais ilustrativos ao longo do texto. Este texto está baseado livros Mecânica Clássica e outros.

1.2 Importância da linguagem Python na Engenharia Mecânica

O Python é uma linguagem de programação de alto nível, de script, que pode ser orientada a objetos. Esta linguagem foi lançada por Guido van Rossum em 1991 como uma linguagem acadêmica para ensino de programação numérica a alunos de graduação.

1.3 Orientação a objetos

A programação orientada a objetos é um modelo de projeto para desenvolvimento de projetos numéricos, baseado na integração dos diversos segmentos do programa chamados objetos. Diferentemente da programação estruturada, a orientação a objetos é conceituada no campo da abstração de conceitos do mundo real.

- classe:
- objeto/instância:
- atributo:
- método:
- mensagem:
- herança:
- associação:
- encapsulamento:
- abstração:
- polimorfismo:
- interface:
- pacotes (ou namespaces):

Descrever diagrama UML (Unified Modeling Language).

1.4 Solução de problemas diferenciais

As equações diferenciais parciais (EDP) de 2ª. ordem podem ser classificadas em três tipos distintos:

- Equação hiperbólica: usadas para estudo de vibrações, oscilações elétricas, acústica etc. É comumente conhecida como **Equação da Onda**.

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (1)$$

- Equação parabólica: usada no estudo de problemas de difusão de calor ou massa, propagação de calor etc.

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (2)$$

- Equação elíptica: usada para modelar campos magnéticos, hidrodinâmica e problemas de transferência de calor estacionários. Na matemática se $f(x, y) = 0$, esta equação recebe o nome de **Equação de Laplace**, em referência à Pierre Simon Laplace. Se $f(x, y) \neq 0$ esta equação recebe o nome de **Equação de Poisson**, em referência à Siméon-Denis Poisson.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y) \quad (3)$$

Um método simples para identificação do tipo de EPD pode ser obtido da seguinte forma. Se uma função $u(x, y)$ pode ser definida no espaço \mathbb{R}^2 , qualquer EPD de 2a. ordem pode ser escrita da seguinte forma:

$$a \frac{d^2 u(x, y)}{dx^2} + 2b \frac{d^2 u(x, y)}{dxdy} + c \frac{d^2 u}{dy^2} + d \frac{du}{dx} + e \frac{du}{dy} + fu = g \quad (4)$$

Considerando que os coeficientes a , b e c forem constantes e tais que:

$$a^2 + b^2 + c^2 \neq 0 \quad (5)$$

Podemos classificar as EPDs através da seguinte relação:

- **hiperbólicas:** $b^2 - 4ac > 0$, raízes reais e distintas;
- **parabólicas:** $b^2 - 4ac = 0$, raízes reais e idênticas;
- **elípticas:** $b^2 - 4ac < 0$, raízes conjugadas e complexas;

1.5 Importância da validação do código numérico

Validação de código é importante para se testar os limites de aplicação do código numérico bem como se certificar de o código está respondendo como planejado. Um código numérico deverá sempre ser validado e testado. Para o caso da engenharia, as validações devem cobrir de três campos importantes:

- modelo matemático
- física do problema
- lógica numérica

O modelo matemático adotado para reproduzir o fenômeno físico deve estar condizente com a realidade do problema, ou seja com a física do problema. Por exemplo, ao se tratar problemas de condução unidimensional de calor em sólidos sem fonte de calor, usa-se a equação de Laplace para encontrar a distribuição de temperatura. Através desta escolha, *modela-se* o problema físico através de uma equação matemática.

A lógica numérica trata das sequências lógicas que são criadas no código numérico para automatização de processos repetitivos. Esta lógica deve ser questionada durante todo o desenvolvimento do código para que o cientista/programador não acredite que um erro pode estar vindo daquele procedimento. Note que esta lógica pode abranger o código como um todo, bem como trechos envolvendo funções, loops, classes etc. É importante validar e verificar que o procedimento leva a resultados esperados.

Além destes campos, é importante verificar a abrangência da solução obtida, conhecendo o erro inserido nas discretizações e sua propagação através dos diversos procedimentos (ver [3]).

1.6 Controle de versões - GIT

O controle de versões é necessário no desenvolvimento de softwares para que o programador tenha um histórico de cada etapa relacionada a sua construção. Existem diversos controladores de versão que ainda são bastante utilizados pela comunidade: *SVN*, *CVS*, *Mercurial*, *Monotone*, *SVK*, *Rational ClearCase*, *Borland StarTeam* etc.

A ferramenta *Git* é um sistema de controle de versão distribuído e também um sistema de gerenciamento de código fonte. O *Git* foi projetado pelo criador do sistema Linux (Linus Torvalds) e é adotado por diversas empresas que desenvolvem softwares comerciais de pequeno e grande porte. O *Git* é um software livre e pode ser instalado em todos os sistemas operacionais através de download na página (<https://git-scm.com>)

É importante notar que o *Git* não é um sistema dedicado à linguagem de programação, podendo ser usando em qualquer tipo de código fonte, como por exemplo em documentos Word ou Notepad, além de LaTeX e outras linguagens de programação como Bash, C/C++ e Perl.

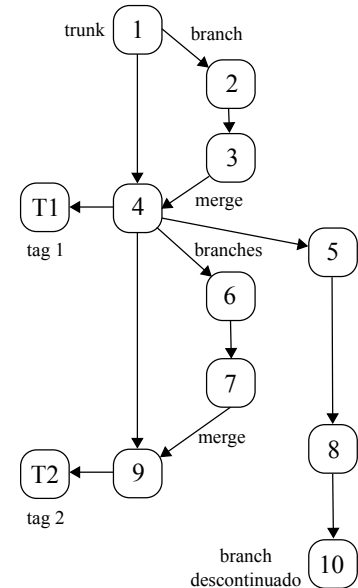


Figura 1: Exemplo da visualização do histórico de um projeto usando um sistema de controle de versões

1.7 Editor de texto - VIM

Existem diversos editores de texto são aplicativos essenciais em qualquer ambiente de trabalho e estão disponíveis para instalação nos diversos sistemas operacionais existentes. Alguns deles já são pré-instalados pelo sistema, enquanto que alguns outros devem ser comprados ou transferidos gratuitamente de páginas na internet. Alguns exemplos de editores de texto conhecidos no sistema *Windows* são Microsoft Office e Notepad, no sistema *Mac OS X* pode-se citar o TextEdit e o Notes. Alguns editores estão disponíveis para os diversos sistemas operacionais, como é o caso do Microsoft Word, encontrado nos sistemas *Windows* e *Mac OS X* e o OpenOffice que pode ser instalados nos sistemas *Windows*, *Mac OS X* e *Linux*. Um editor de texto pode ser projetado para criar/editar diversos tipos de texto, entretanto também pode ser especializado em uma determinada função, como por exemplo em textos de programação numérica.



O *VIM* é um editor de texto poderoso que utiliza baixo recurso gráfico, tornando-o flexível, versátil e de baixíssimo consumo de processador. Apesar de ser um editor que opera em modo texto, é possível encontrar versões deste editor em modo gráfico (gvim), acompanhado de botões para abrir arquivo, gravar, editar, imprimir etc. É interessante notar que a curva de aprendizagem deste editor necessita do programador um esforço inicial para que ele consiga dominar as funções básicas de edição, entretanto o próprio vim oferece uma ferramenta de tutorial que vem acompanhada do editor: *vimtutor*. Abaixo encontra-se alguns comandos de edição bastante utilizados:

- :w gravar documento
- :q sair do editor
- :qa sair do editor caso esteja editando mais de um documento
- :wq gravar o documento e sair do editor

- `:wq` gravar todos os documentos e sair do editor
- `:saveas` gravar como novo documento
- `:e .` abrir gerenciador de arquivos no VIM
- `:args` lista de arquivos abertos no editor
- `:e nome-do-arquivo` editar outro arquivo
- `:split nome-do-arquivo` separar janelas horizontalmente e carregar outro arquivo
- `:vsplit nome-do-arquivo` separar janelas verticalmente e carregar outro arquivo
- `:sview nome-do-arquivo` abrir arquivo em modo de leitura
- `:set backup` criar arquivo de backup
- `:ggVG` selecionar todo documento
- `:make` rodar script Makefile no editor
- `:32` ir para a linha 32

O editor de texto VIM suporta a instalações de aditivos para facilitar o desenvolvimento de projetos. Alguns exemplos são:

- `taglist.vim` - gerenciador de arquivos fonte
- The Nerd Commenter - plugin para facilitar comentários em diversos tipos de arquivos
- Align - alinhamento de textos automáticos
- BlockComment.vim - automatização de comentários de blocos de código
- DirDiff.vim - plugin para manipulação de diretórios

Os aditivos devem ser instalados na pasta de instalação do editor VIM. Estes aditivos estão centralizados no seguinte endereço eletrônico:

<http://www.vim.org/scripts/>

Este editor de texto é compatível com os mais importantes sistemas operacionais.

2 Mecânica de massas pontuais

2.1 Movimento Horizontal de um Carrinho

Neste exemplo deseja-se calcular como a força de atrito linear $F_{drag} = -bv$ atua em um carrinho com massa m a fim de desacelerá-lo até sua completa parada na direção x . Desconsidere o atrito dos mancais no eixo de rodas do carrinho em Fig. (3).

$$m \frac{dv}{dt} = -bv \quad (6)$$

Para encontrarmos a solução numérica deste problema, usaremos diferenças finitas progressiva para a discretização da derivada da velocidade em relação ao tempo:

$$m \frac{v^{n+1} - v^n}{dt} = -bv \quad (7)$$

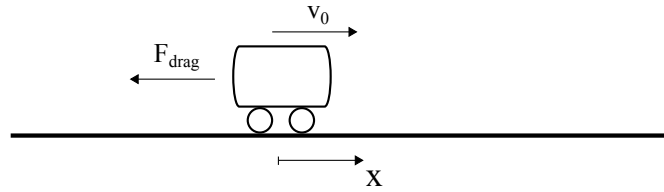


Figura 2: Desaceleração de carrinho por atrito do ar linear $F_{drag} = -bv$.

Note que o método de discretização acima recebe também o nome de *Método de Euler* para solução de equação diferencial. Este método é o tipo de método explícito mais básico para integração numérica. Como todo método numérico, o Método de Euler é uma aproximação da solução exata, por isso, a solução numérica pode se afastar da solução exata dependendo da escolha do tamanho do passo de tempo dt . No limite onde dt tende a zero, a solução numérica coincide com a solução exata. Em problemas simples, pode-se escolher um passo de tempo extremamente pequeno, de modo que o erro gerado pela solução numérica seja praticamente desprezível. Entretanto, para problemas de grande porte, o uso de passos de tempo pequenos inviabilizam a solução devido à duração total de simulação, que pode levar dias, meses ou até mesmo anos para que se chegue na solução desejada. Outros métodos de aproximação podem ser usados para obtenção de resultados mais precisos e mais rápidos. No caso de equações ordinárias, pode-se usar o Método de Runge-Kutta.

Uma vez encontrada a equação discreta da equação ordinária, pode-se determinar o valor da velocidade no próximo passo de tempo $n + 1$ fazendo:

$$v^{n+1} = v^n - dt \frac{bv}{m} \quad (8)$$

Este procedimento pode então ser repetido sucessivamente para que se obtenha a velocidade em diversos tempos, ou seja, $n + 1, n + 2, n + 3, n + 4 \dots$ e com isso determine a distribuição da velocidade em função do tempo. Para o problema em questão, observa-se que a velocidade diminuirá exponencialmente até um valor próximo a zero, concluindo-se que a velocidade do carrinho foi reduzida até sua parada completa. Note que *à priori* não se sabe quantas repetições devem ser realizadas no cálculo de v^{n+1} para que se obtenha o valor zero para a velocidade. Este cálculo dependerá dos dados do problema e do tamanho escolhido para o passo de tempo dt .

Para efeitos de validação do método de cálculo numérico apresentado, pode-se usar a solução analítica da Eq. (6) para $v(t)$. Note que nem sempre é possível obter solução analítica de problemas para validação, entretanto buscase, sempre que possível, o uso desta ferramenta para comprovação do código numérico elaborado. A solução analítica toma a seguinte forma:

$$v(t) = v_0 e^{-bt/m} \quad (9)$$

onde v_0 é a velocidade inicial do carrinho, t é o tempo medido em segundos e b é o coeficiente de atrito linear. Ao integrarmos novamente a equação de $v(t)$ obtemos a solução da posição, já que $dx/dt = v$:

$$x(t) = \int_0^t v(t) dt = x_\infty (1 - e^{-bt/m}) \quad (10)$$

onde $x_\infty = v_0 m / b$

Dados da simulação:

Condições iniciais:

Tabela 1: Dados da simulação do carrinho submetido à força de atrito linear do ar.

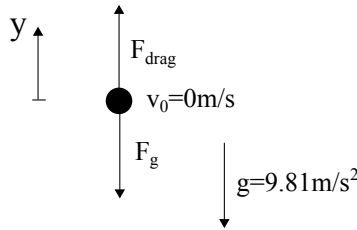
massa da partícula	coeficiente de atrito	passo de tempo
$[kg]$	$[kg]/[s]$	$[s]$
$m = 1.0$	$b = 0.1$	$dt = 0.1$

Tabela 2: Condições iniciais da da simulação do carrinho submetido à força de atrito linear do ar.

tempo inicial	velocidade inicial	posição inicial
$[s]$	$[m]/[s]$	$[m]$
$time = 0.0$	$v_0 = 10.0$	$x = 0.0$

2.2 Velocidade terminal de uma gota

Neste problema deseja-se calcular como a força de atrito linear $F_{drag} = -bv$ atua em uma partícula com massa m sob uma força de gravidade $F_{grav} = mg$ a fim de desacelerá-la até o equilíbrio de forças, onde a aceleração seja igual a 0 na direção y .


 Figura 3: Aceleração de uma gota caindo sob efeito gravitacional $F_g = mg$ e atrito do ar linear $F_{drag} = -bv$.

Através da 2a. Lei de Newton, a equação matemática para modelar este problema pode ser escrito como:

$$m \frac{dv}{dt} = F_{drag} + F_g \quad (11)$$

onde v , que é uma função do tempo t é a incógnita que queremos resolver.

$$m \frac{v^{n+1} - v^n}{dt} = bv - mg \quad (12)$$

$$v^{n+1} = v^n + dt \frac{bv}{m} - dtg \quad (13)$$

Este procedimento deverá ser repetido sucessivamente para que se obtenha a velocidade nos diversos tempos, ou seja, $n+1, n+2, n+3, n+4 \dots$ e com isso determine a distribuição da velocidade em função do tempo. Para o problema em questão, observa-se que a velocidade aumentará até que a gota esteja em equilíbrio dinâmica, ou seja, a força de atrito fique igual à força de gravidade. Novamente, não se sabe quantas repetições devem ser realizadas no cálculo de v^{n+1} para que se obtenha o valor de velocidade terminal para a gota. Este cálculo dependerá dos dados do problema e do tamanho escolhido para o passo de tempo dt .

A Eq. (11) pode ser resolvida analiticamente para $v(t)$ e sua solução analítica toma a seguinte forma:

$$v(t) = v_0 e^{-bt/m} + v_{lim}(1 - e^{-bt/m}) \quad (14)$$

onde V_0 é a velocidade inicial do carrinho, t é o tempo medido em segundos e b é o coeficiente de atrito. Ao integrarmos novamente a equação de $v(t)$ obtemos a solução da posição, já que $dx/dt = v$:

$$x(t) = \int_0^t v(t) dt \quad (15)$$

Tabela 3: Dados da simulação da gota submetida à força de atrito linear do ar.

diâmetro da gota óleo	densidade do líquido	aceleração da gravidade	volume da gota
[m]	[kg]/[m ³]	[m]/[s ²]	[m ³]
$D = 1.5e - 06$	$\rho = 840.0$	$g = 9.81$	$V = \pi D^3/6.0$

Tabela 4: Continuação de dados da simulação do carrinho submetido à força de atrito linear do ar.

massa da partícula	viscosidade dinâmica do ar	coeficiente de atrito linear	passo de tempo
[kg]	[kg]/[m][s]	[kg]/[s]	[s]
$m = \rho V$	$\beta = 1.6e - 04$	$b = \beta D$	$dt = 1e - 7$

2.3 Lançamento de projétil

Neste exemplo deseja-se calcular como a força de atrito linear $F = bv$, ou a força de atrito quadrática $F = cv^2$ ou ausência de forças de atrito atuam em uma partícula com massa m sob uma força de gravidade $F = mg$.

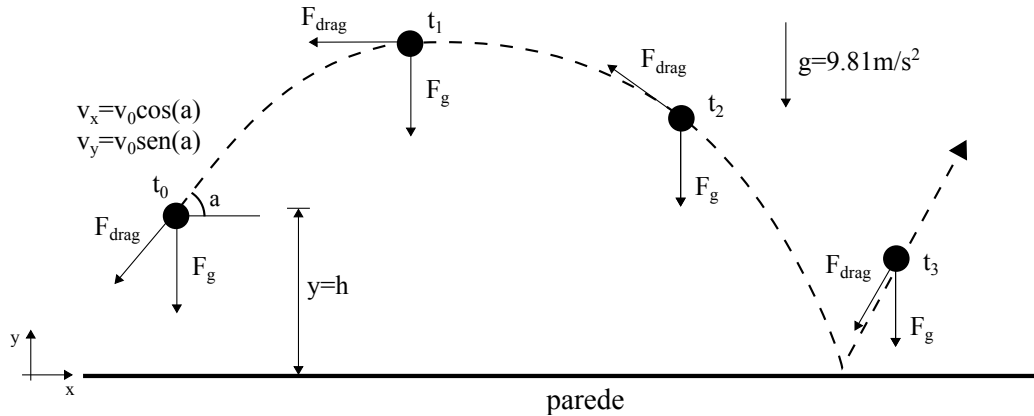


Figura 4: Desaceleração de um projétil por atrito do ar $F_{drag} = -bv$ sob efeito de gravidade.

A equação vetorial (em X e Y) toma a seguinte forma:

(16)

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_g \quad \begin{cases} m \frac{dv_x}{dt} = F_{drag_x} \\ m \frac{dv_y}{dt} = F_{drag_y} + F_{g_y} \end{cases} \quad (17)$$

Dados da simulação;

- $\#D = 1.5e - 06$ $[m]$ diâmetro do projétil
- $D = 7.0e - 02$ $[m]$ diâmetro da gota de neblina
- $g = 9.81$ $[m/s^2]$ aceleração da gravidade
- $V = \pi D^3 / 6.0$ $[m^3]$ volume da gota
- $m = 0.15$ $[kg]$ massa da partícula
- $\beta = 0.25$ $[kg/ms]$ viscosidade dinâmica do ar
- $b = \beta D$ $[kg/s]$ coeficiente de atrito linear
- $\gamma = 0.25$ $[Ns^2/m^4]$
- $c = \gamma D^2$ coeficiente de atrito linear
- $dt = 0.01$ $[s]$ passo de tempo

Condições iniciais

- $time = 0.0$ $[s]$ tempo
- $x = 0.0$ $[m]$ posição
- $y = 0.0$ $[m]$ posição
- $vx = 19.3$ $[m/s]$ velocidade
- $vy = 23.0$ $[m/s]$ velocidade

2.4 Sistema massa-mola

Neste exemplo deseja-se calcular como a força de mola $F_{spring} = -kx$ atua em uma partícula com massa m sem dissipação.

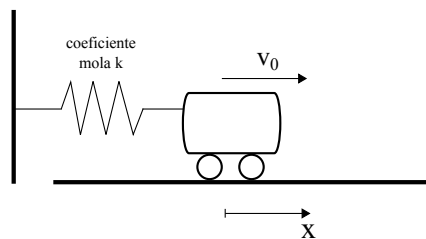


Figura 5: Sistema massa-mola sem dissipação $F_{spring} = -kx$

A solução analítica da equação:

$$m \frac{dv}{dt} = -kx \quad (18)$$

$$v(t) = v_0 e^{-bt/m} \quad (19)$$

$$x(t) = \int_0^t v(t) dt = x_\infty (1 - e^{-bt/m}) \quad (20)$$

onde $x_{inf} = v_0 m / b$

Dados da simulação

- $m = 1.0$ massa da partícula
- $k = 0.1$ coeficiente da mola
- $dt = 0.1$ passo de tempo

Condições iniciais:

- $time = 0.0$ tempo total da simulação
- $x = 0.0$ posição inicial da partícula
- $v_x = 10.0$ velocidade inicial da partícula

2.5 Sistema massa-mola dissipativo

Neste exemplo deseja-se calcular como a força de atrito linear $F = -bv$ atua em uma partícula com massa m sob uma força elástica (de mola) linear $F = -kx$ a fim de desacelerá-la até o equilíbrio de forças, onde a aceleração seja igual a 0 na direção y .

A equação em X:

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{spring} + \mathbf{F}_{friction} \quad (21)$$

Dados da simulação:

- $m = 1.0$ massa da partícula
- $k = 0.1$ coeficiente da mola
- $b = 0.1$ coeficiente de atrito linear
- $dt = 0.1$ passo de tempo

Condições iniciais:

- $time = 0.0$ tempo total da simulação
- $x = 0.0$ posição inicial da partícula
- $v_0 = 10.0$ velocidade inicial da partícula

2.6 Sistema massa-mola vertical

Neste exemplo deseja-se calcular como a força de gravidade $F = -mg$ atua em uma partícula com massa m sob uma força elástica (de mola) $F = -ky$ a fim de mantê-la oscilando.

A equação em X:

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{spring} + \mathbf{F}_{grav} \quad (22)$$

$$\mathbf{F}_{spring} = F_{spring_y} - ky \text{ e } \mathbf{F}_{grav} = F_{grav_y}mg$$

Dados da simulação:

- $m = 1.0$ massa da partícula
- $k = 0.1$ coeficiente da mola
- $b = 0.1$ coeficiente de atrito linear
- $y = 0.0$ posição inicial da partícula
- $v_0 = 10.0$ velocidade inicial da partícula
- $g = 9.81$ $[m/s^2]$ aceleração da gravidade
- $D = 7.0e - 02$ $[m]$ diâmetro da partícula
- $\gamma = 0.25$ $[Ns^2/m^4]$
- $c = \gamma * D * D$ coeficiente de atrito linear
- $dt = 0.1$ $[s]$ passo de tempo
- $time = 0.0$ tempo total da simulação
- $\#D = 1.5e - 06$ $[m]$ diâmetro do projétil
- $m = 0.15$ $[kg]$ massa da partícula
- $\beta = 0.25$ $[kg/ms]$ viscosidade dinâmica do ar
- $b = \beta D$ $[kg/s]$ coeficiente de atrito linear

2.7 Pêndulo simples

Deseja-se resolver o problema do pêndulo simples sob efeito de força gravitacional $F_g = mg$:

$$m \frac{d^2\theta}{dt^2} + \frac{mg}{l} \sin \theta = 0 \quad (23)$$

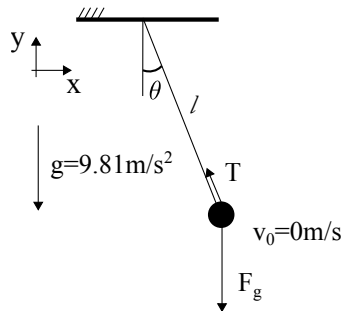


Figura 6: Pêndulo simples sob efeito de força gravitacional $F_g = mg$.

Esta equação pode ser resolvida de diferentes formas. A mais simples é tornar esta equação diferencial de segunda ordem em duas equações de ordem mais baixa:

$$\frac{d\theta}{dt} = v \quad (24)$$

e

$$m \frac{dv}{dt} + \frac{mg}{l} \sin \theta = 0 \quad (25)$$

Com isso, resolve-se a equação para encontrar θ no tempo $n + 1$:

$$\frac{\theta^{n+1} - \theta^n}{\Delta t} = v^n \longrightarrow \theta^{n+1} = \theta^n + \Delta t v^n \quad (26)$$

E depois a Eq. (25) para encontrar v no tempo $n + 1$:

$$\frac{v^{n+1} - v^n}{\Delta t} = -\frac{g}{l} \sin \theta \longrightarrow v^{n+1} = v^n - \Delta t \frac{g}{l} \sin \theta^{n+1} \quad (27)$$

Este processo deve ser repetido sucessivamente em um *loop* para encontrar a evolução de v e θ no tempo t . Como não há adição ou perda de energia no sistema, espera-se um comportamento de movimento repetitivo com o tempo.

Para pequenas perturbações esta equação tem solução analítica, com isso aproxima-se $\sin \theta \approx \theta$, chegando à seguinte equação:

$$m \frac{d^2 \theta}{dt^2} + \frac{mg}{l} \theta = 0 \quad (28)$$

A solução analítica é então escrita da seguinte forma para pequenos ângulos ($< 10^\circ$):

$$\theta(t) = \theta_0 \cos(\sqrt{g/l}t) \quad (29)$$

Dados da simulação:

Tabela 5: Dados da simulação do pêndulo submetido à força de gravidade.

massa da partícula	aceleração da gravidade	passo de tempo	comprimento da haste
$[kg]$	$[m]/[s^2]$	$[s]$	$[m]$
$m = 1.0$	$g = 9.81$	$dt = 0.01$	$l = 1.0$

Condições iniciais:

Tabela 6: Condições iniciais da simulação do pêndulo simples.

tempo inicial	velocidade inicial	ângulo inicial	posição inicial
$[s]$	$[m]/[s]$	$^\circ$	$[rad]$
$time = 0.0$	$v_0 = 0.0$	10	$\theta_0 = \frac{2grad\pi}{360}$

3 Mecânica do contínuo

3.1 Geração de malha 1D

Criação de malha 1D para o método de elementos finitos com dx variando conforme as seguintes equações:

- constante: $dx = cte$
- quadrática: x^2
- cúbica: x^3
- exponencial: $exp(x)$

Observe cada caso ilustrado na Fig. (7), onde a função escolhida fornece o espaçamento entre nós da malha. Para o caso linear Fig. (7a), o espaçamento dx é constante. Para os outros casos, o espaçamento varia conforme a função adotada.

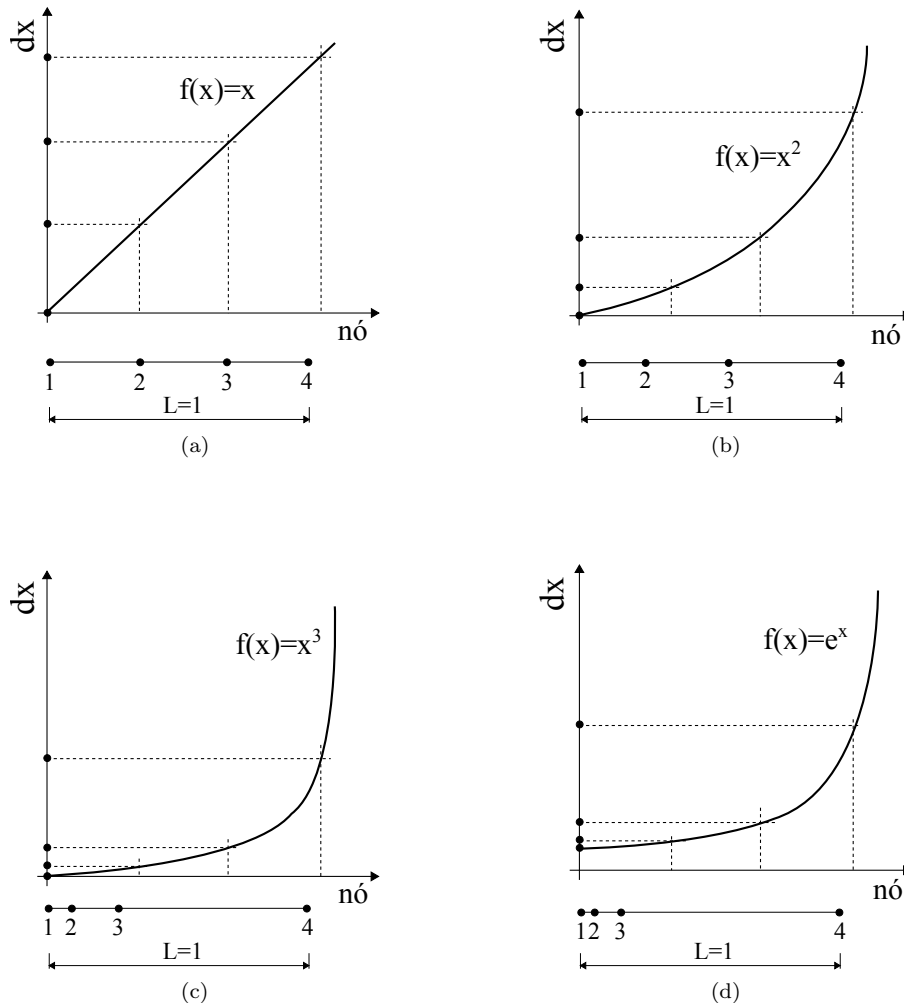


Figura 7: Distribuição de espaçamento de malha dx e representação final da malha unidimensional com dimensão $L = 1$ usando função (a) linear, (b) quadrática, (c) cúbica e (d) exponencial.

Parâmetros da malha:

- $L = 1.0$ comprimento total da malha
- $nx = 10$ número total de nós
- $ne = nx - 1$ número total de elementos

Dica: para criação de malha computacional, 2 estruturas são necessárias: um vetor de coordenadas dos nós da malha onde cada elemento do vetor é a posição do nó, e uma matriz de conectividade de nós onde a linha da matriz representa o elemento e as colunas representam os nós daquele elemento.

	v_1	v_2	v_3
1	1	5	4
2	1	2	5
3	2	6	5
4	2	3	6
5	4	8	7
6	4	5	8
7	5	9	8
8	5	6	9

Tabela 7: Estrutura da matriz de conectividade IEN_{ij} . Note que a numeração do primeiro vértice e do primeiro elemento começam com 1. Em algumas linguagens de programação, como o Python e C/C++, é conveniente começar a numeração com 0.

3.2 Geração de malha 2D

A criação de malha em duas dimensões é um processo matemático sofisticado e requer especial atenção na escolha de formação dos elementos. Neste problema, deseja-se criar uma malha computacional 2D simplificada, onde a distribuição de pontos é regular e uniforme e a triangulação é estruturada, com isso simplificando o processo de criação de malha 2D.

Parâmetros da malha:

- $L_x = 1.0$ comprimento total da malha na direção x
- $L_y = 1.0$ comprimento total da malha na direção y
- $nx \times ny = 4 \times 3 = 12$ número total de nós
- $ne = 2(nx - 1)(ny - 1)$ número total de elementos

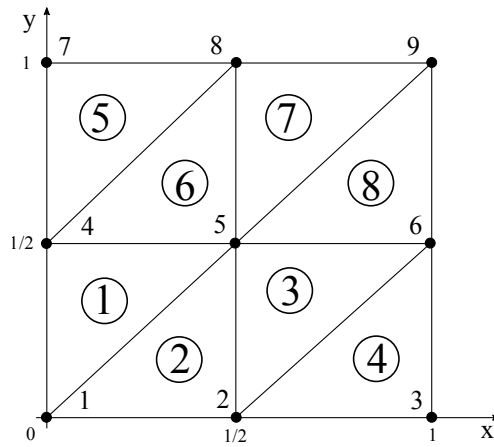


Figura 8: Distribuição de pontos e elementos em malha bidimensional. A numeração dos elementos é definida pelos números com círculos, enquanto os índices dos nós estão localizados ao lado nó. Note que a figura não está em escala.

Dica: para criação de malha computacional bidimensional, 3 estruturas são agora necessárias: dois vetores de coordenadas dos nós da malha onde cada elemento do vetor é a posição do nó no plano $x - y$, e uma matriz

de conectividade de nós onde a linha da matriz representa o elemento e as colunas representam os nós daquele elemento.

Perturbe a parte de malha onde $Y = Y_{max}$ usando a seguinte uma função de onda senoidal do tipo:

$$y(x) = A \sin\left(\frac{2\pi}{\lambda}x - \phi\right) \quad (30)$$

onde A representa a amplitude da onda, λ o comprimento de onda que equivale a $\lambda = L/n$, n é o número de ondas, x é a coordenada do ponto ao longo do eixo x e ϕ é o deslocamento de fase. Adote os seguintes valores:

- $A = 0.07$
- $\phi = 2\pi/4.0$
- $\lambda = 24/6$
- $k = 2\pi/\lambda$
- $Lx = 10.0$
- $Ly = 0.5$
- $nx = 100$
- $ny = 120$

Depois de completada esta tarefa, propague a perturbação da onda gerada em $Y = Y_{max}$ suavemente para o restante da malha, ou seja, variando y , de modo que a distribuição final tenha perturbação nula para $Y = 0$.

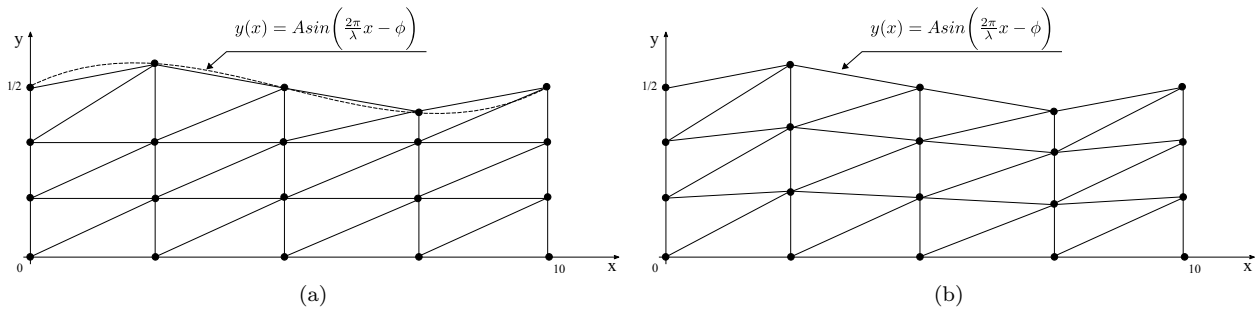


Figura 9: Malha bidimensional de triângulos com perturbação senoidal no (a) topo e (b) propagada para o interior com amortecimento da função.

3.3 Solução de problema térmico permanente 1D

Neste exemplo deseja-se calcular a distribuição de temperatura unidimensional em regime permanente em uma barra com dimensão adimensional $L = 1$ e temperaturas constantes $T(x = 0) = 0$ e $T(x = L) = 1$ nas extremidades da barra.

A equação de interesse:

$$\frac{d^2 T}{dx^2} = 0 \quad (31)$$

para $T(x = 0) = 0$ e $T(x = L) = 1$

Execução do programa:

- $nx = 4$ número de pontos em x
- $L = 1.0$ comprimento total
- $dx = L/nx$ intervalo dx
- $T_i = 0.0$ condição de contorno do primeiro nó
- $T_f = 1.0$ condição de contorno do último nó

Neste problema, a discretização da derivada é em relação ao espaço e não ao tempo como nos casos do anteriores. É comum impregar os índices n para designação temporal e i para espacial. Além disto, a equação apresenta uma derivada de segunda ordem que pode ser discretizada de diversas formas (diferenças centradas, diferenças adiantadas e diferenças atrasadas), além da ordem de discretização (primeira ordem, segunda ordem, terceira ordem), tornando assim o procedimento mais preciso (ver [2]). Para a discretização temporal de primeira ordem, busca-se o valor de $v_{n+1} = v_n + \Delta T * F/m$. No caso da derivada no espaço deste problema, é necessário construir um sistema linear de equações, onde cada linha da matriz é a equação do ponto i , com isso a Eq. (31) em sua forma discreta utilizando diferenças finitas centradas, é reescrita para o nó da malha i como:

$$\frac{T_{i-1} - 2T_i + T_{i+1}}{dx^2} = 0 \quad (32)$$

Note que nesta equação a derivada no tempo não está presente. Dizemos então a equação é um modelo de condução de calor em modo permanente.

Se usarmos uma malha com 3 elementos lineares e 4 nós igualmente espaçados, a equação do nó 2 se escreve:

$$\frac{T_1 - 2T_2 + T_3}{dx^2} = 0 \quad (33)$$

Com isso, podemos também obter as equações dos outros nós 1, 3 e 4. Entretanto, é de se notar que encontraremos um problema com o cálculo discreto da derivada segunda nos nós de contorno, pois o nó $i - 1$ cairia fora da malha computacional (levando em consideração que a malha tem nós 1 - 4). Diversas soluções podem ser encontradas na literatura ([referencia para livro](#)), porém pode-se sugerir o uso de derivadas de ordem mais baixas que não necessitem da informação de nós fora da malha em combinação e/ou com esquemas do tipo diferenças progressivas e atrasadas. É importante notar que ao se usar aproximações de ordem mais baixa, a precisão do cálculo estará comprometida pois o erro será da ordem da aproximação menos precisa. Para fins ilustrativos, usaremos uma aproximação de primeira ordem nos nós de contorno com diferenças progressivas para o nó 1 e diferenças atrasadas para o nó 4 respectivamente.

$$\frac{T_1 - 2T_2 + T_3}{dx^2} = 0 \quad (34)$$

$$\frac{T_4 - 2T_3 + T_2}{dx^2} = 0 \quad (35)$$

Ao final do procedimento para todos os pontos i da malha, monta-se a matriz de rigidez K_{ij} , o vetor de incógnitas T_1, T_2, T_3, T_4 e o vetor do lado direito - chamaremos aqui de b - e que para o caso da equação de Laplace $b = 0$.

Este procedimento resultará em um sistema linear de equações do tipo $Ax = b$ que poderá ser resolvido para obtenção dos valores da temperatura nos nós da malha. Entretanto, o aluno deve ainda se atentar ao fato de que de alguma maneira as condições de contorno do problema sejam passadas para o sistema linear e a solução das temperaturas forneça os valores corretos do problema particular, ao invés do problema geral. Uma discussão mais profunda deste procedimento pode ser encontrada em ([referencia para livro](#)).

K	1	2	3	4	T	b
1	1	-2	1	0	T_1	0
2	1	-2	1	0	T_2	0
3	0	1	-2	1	T_3	0
4	0	1	-2	1	T_4	0

 Tabela 8: Estrutura da matriz K_{ij} e vetor de carregamento b_j antes da imposição de condições de contorno.

K	1	2	3	4	T	b
1	1	0	0	0	T_1	0
2	0	-2	1	0	T_2	$0 - 1T_1$
3	0	1	-2	1	T_3	$0 - 1T_4$
4	0	0	0	1	T_4	1

 Tabela 9: Estrutura da matriz K_{ij} e vetor de carregamento b_j depois da imposição de condições de contorno do tipo Dirichlet nas duas extremidades ($T_1 = 0, T_4 = 1$).

O problema a ser resolvido, impõe condições de contorno de temperatura definida nos nós de contorno (1 e 4), com isso pode-se reescrever as equações deste nós e substituir nas linhas 1 e 4 da matriz K_{ij} por equações triviais, ou seja:

$$\frac{T_1 - 2T_2 + T_3}{dx^2} = 0 \longrightarrow 1T_1 = 0.0 \quad (36)$$

e

$$\frac{T_4 - 2T_3 + T_2}{dx^2} = 0 \longrightarrow 1T_4 = 1.0 \quad (37)$$

Note que a matriz K_{ij} não será alterada no seu tamanho, ou seja, permanecendo uma matriz 4×4 . Outras técnicas podem ser usadas, como por exemplo a eliminação das linhas 1 e 4, e a consequente redução das dimensões da matriz, já que não há necessidade de calcular os valores de T_1 e T_4 (condições de contorno do problema). O método de penalidade também pode ser usado para imposição de condições de contorno (ver [1]). Entretanto, mantenhamo-nos no procedimento de substituição das equações e, ao final, verifica-se que as linhas 2 e 3 são dependentes dos valores de T_1 e T_4 e uma atualização deve ser procedida. Para isso, passamos os valores diferentes de zero da coluna 1 e multiplicado pela condição de contorno no nó 1 e subtraindo do vetor b . Repetimos o procedimento para o nó 4, passando os valores diferentes de zero da coluna 4 e multiplicado pela condição de contorno no nó 4 e subtraindo do vetor b do vetor b .

Após a obtenção do novo sistema linear representativo da equação de Laplace com as condições de contorno do problema, resolve-se o sistema linear invertendo a matriz de rigidez K_{ij} e multiplicando-a pelo vetor resultante b .

Tente resolver o mesmo problema com um número de pontos maior. Observe se há diferença na solução do problema. Mude os esquemas de diferenças finitas e verifique o grau de facilidade de implementação e o compromisso com a ordem de aproximação.

3.4 Solução de problema térmico permanente com geração de calor 1D

Neste exemplo deseja-se calcular a distribuição de temperatura unidimensional em regime permanente em uma barra com dimensão adimensional $L = 1$ e temperaturas adimensionais constantes $T(x = 0) = 0$ e $T(x = L) = 0$ nas extremidades da barra e com geração de calor Q .

A equação de interesse:

$$\alpha \frac{d^2 T}{dx^2} = Q \quad (38)$$

para $T(x = 0) = 0$ e $T(x = L) = 0$

Execução do programa:

- $nx = 40$ número de pontos em x
- $L = 1.0$ comprimento total
- $dx = L/nx$ intervalo dx
- $T_i = 0.0$ condição de contorno do primeiro nó
- $T_f = 0.0$ condição de contorno do último nó
- $\alpha = 1.0$ difusividade térmica do material
- $Q = 2.0$ fonte de calor

3.5 Solução de problema térmico transiente 1D

Neste exemplo deseja-se calcular a evolução temporal da distribuição de temperatura unidimensional em uma barra com dimensão $L = 1$ e temperaturas constantes $T(x = 0) = 0$ e $T(x = L) = 1$ nas extremidades da barra.

A equação de interesse:

$$\frac{dT}{dt} = \alpha \frac{d^2 T}{dx^2} \quad (39)$$

para $T(x = 0) = 0$ e $T(x = L) = 1$

A solução da equação em estado permanente se escreve:

$$T(x) = c_1 x + c_2 \quad (40)$$

Com constantes c_1 e c_2 a serem determinadas através da aplicação das condições de contorno. Para condições de contorno arbitrárias com temperatura fixa nas extremidades (condição de contorno do tipo Dirichlet), as constantes assumem os seguintes valores:

$$c_1 = \frac{T_{s2} - T_{s1}}{L} \quad c_2 = T_{s1} \quad (41)$$

onde T_{s1} é a temperatura da superfície 1 e T_{s2} é a temperatura da superfície 2. Com isso, a distribuição de temperatura para o caso de duas condições de contorno do tipo Dirichlet toma a forma:

$$T(x) = \frac{T_{s2} - T_{s1}}{L} x + T_{s1} \quad (42)$$

Para o caso do problema sugerido, a solução final toma a forma de:

$$T(x) = \frac{x}{L} \quad (43)$$

3.6 Solução de problema térmico transiente 1D com geração de calor

Neste exemplo deseja-se calcular a evolução temporal da distribuição de temperatura unidimensional em uma barra com dimensão $L = 1m$ e temperaturas constantes $T(x = 0) = 0$ e $T(x = L) = 1$ nas extremidades da barra. A forma forte do problema se escreve:

$$\frac{dT}{dt} = \alpha \frac{d^2T}{dx^2} + Q \quad (44)$$

para $T(x = 0) = 0$ e $T(x = L) = 1$

A solução da equação em estado permanente se escreve:

$$T(x) = -\frac{QL}{2k} \left(x - \frac{x^2}{L} \right) + \frac{T_{s2}}{L}x + T_{s1} \quad (45)$$

Esta é a solução particular para temperatura da superfície 1 igual a T_{s1} e temperatura da superfície 2 igual a T_{s2} no caso permanente ($\partial/\partial t = 0$). No caso de tomarmos as condições de contorno do problema $T_{s1} = T_{s2} = 0$, a solução da distribuição de temperatura $T(x)$ fica:

$$T(x) = -\frac{QL}{2k} \left(x - \frac{x^2}{L} \right) \quad (46)$$

3.7 Solução de equação de transporte

Neste exemplo deseja-se calcular a posição do perfil de temperatura T em função do tempo através da equação diferencial parcial usando o método de diferenças finitas com as seguintes metodologias:

- diferenças centradas
- upwind 1a. ordem
- upwind 2a. ordem
- semi-lagrangiano 1a. ordem
- semi-lagrangiano 2a. ordem
- lagrangiano

A equação diferencial parcial para $T(t, x)$:

$$\frac{dT}{dt} + v \frac{dT}{dx} = 0 \quad (47)$$

O número de Courant (CFL) é definido como:

$$CFL = v \frac{dt}{dx} \quad (48)$$

Tabela 10: Condições iniciais da simulação para equação de transporte.

tempo inicial	velocidade inicial	CFL	perfil inicial
[s]	[m]/[s]	—	[m]
$time = 0.0$	$a = 1.0$	0.5	$T[i] = \sin(\pi * (X[i] - L/6.0)/(L/6.0))$

Condições iniciais:

Usando o método de diferenças centradas para o termo espacial e um esquema explícito de derivação temporal, obtém-se a seguinte expressão para a equação:

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{T_{i+1}^n - T_{i-1}^n}{2dx} = 0 \quad (49)$$

note que nesta equação o índice i representa a variação espacial, enquanto que n representa a variação temporal.

Para o método de upwind de 1a. ordem, a seguinte expressão é usada:

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{T_{i+1}^n - T_i^n}{dx} = 0 \quad (50)$$

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{T_i^n - T_{i-1}^n}{dx} = 0 \quad (51)$$

Para o método de upwind de 2a. ordem, a seguinte expressão é usada:

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{-1T_{i+2}^n + 4T_{i+1}^n - 3T_i^n}{2dx} = 0 \quad (52)$$

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{3T_i^n - 4T_{i-1}^n + T_{i-2}^n}{2dx} = 0 \quad (53)$$

No método Semi-lagrangiano, discretiza-se a derivada material $DT(x, t)/Dt$ diretamente aon invés do termo transiente e do termo convectivo (representação euleriana) como feito anteriormente. Para tal, é necessário encontrar o valor de $T(x, t)$ no passo de tempo anterior fazendo-se uma busca na malha e interpolando o valor de $T(x, t)$ na posição x_d^{n-1} :

$$\frac{DT}{Dt} \approx \frac{T^{n+1}(x^n) - T^n(x_d^{n-1})}{\Delta t} = 0 \quad (54)$$

Para sabermos qual é o valor de T na posição x_d^{n-1} precisamos calcular x_d através da seguinte equação:

$$\frac{x^n - x_d^{n-1}}{\Delta t} = v^n \quad (55)$$

de modo que o valor final de x_d^{n-1} possa ser encontrado:

$$x_d^{n-1} = x^n - \Delta t v^n \quad (56)$$

Este procedimento define a malha computacional do tempo anterior $n - 1$, a qual chamaremos aqui de x_d . Esta malha é então usada para encontrar o valor de $T^n(x_d)$ através de uma interpolação dos valores de T^n em todos

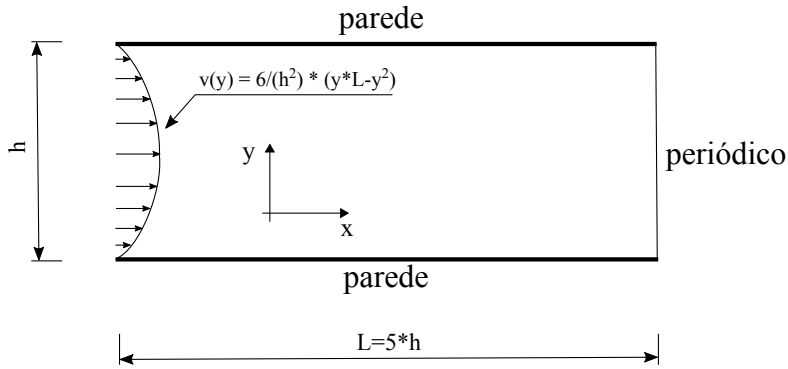


Figura 10: Domínio numérico e condições de contorno para partículas e perfil de velocidade.

os pontos da mala x_d . Esta interpolação é necessária pois os valores de T geralmente não coincidem com os pontos da malha x_d . Uma vez encontrados os valores de T^n , calcula-se a derivada material para encontrar o valor de T^{n+1} :

$$\frac{T^{n+1}(x^n) - T^n(x_d^{n-1})}{\Delta t} = 0 \quad (57)$$

$$T^{n+1}(x^n) = T^n(x_d^{n-1}) \quad (58)$$

3.8 Escoamento com partículas - solução numérica

Deseja-se calcular a trajetória de uma partícula em um canal sob efeito de um perfil de velocidades conhecido (dado por uma função em y) e submetida às forças de gravidade F_g , de arrasto F_{drag} e de sustentação F_{lift} .

A equação vetorial (x e y):

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{lift} + \mathbf{F}_g \quad (59)$$

onde:

$$\mathbf{F}_{drag} = 3\pi\mu d_p v \quad (60)$$

$$\mathbf{F}_{lift} = 1.61\sqrt{\mu\rho_d} dp^2 |u - v| \frac{du}{dy} \sqrt{\left| \frac{du}{dy} \right|} \quad (61)$$

$$\mathbf{F}_g = mg \quad (62)$$

Note que esta equação vetorial tem solução analítica, porém neste problema usaremos uma aproximação numérica explícita para encontrarmos v^{n+1} e, conseqüentemente, a posição das partículas x^{n+1} .

O perfil de velocidade $v(y)$ é conhecido e da forma:

$$v(y) = \frac{6}{L^2}(yL - y^2) \quad (63)$$

As condições de contorno deste problema estão ilustradas na Fig. (10):

Dados da simulação:

fluido (fase contínua)

Tabela 11: Propriedades de algumas fases contínuas.

fluido	viscosidade	densidade	temperatura
	$[N][s]/[m^2]$	$[kg]/[m^3]$	$[^\circ]$
ar	$\mu = 17.2e - 6$	$\rho = 1.225$	$T = 25$
água	$\mu = 1.003e - 3$	$\rho = 997.0$	$T = 25$
azeite	$\mu = 81.0e - 3$	$\rho = 703.0$	$T = 15$

partícula (fase dispersa)

Tabela 12: Propriedades de algumas fases dispersas. O volume pode ser calculado a partir de $V_p = (1.0/6.0)\pi d_p^3$, enquanto que a massa é dada por $m = \rho_p V_p$.

material	diâmetro	densidade
	$[m]$	$[kg]/[m^3]$
madeira	$d_p = 1.0e - 3$	$\rho = 785.0$
alumínio	$d_p = 1.0e - 3$	$\rho = 2700.0$

condição inicial

- $numParticles = 100$ número total de partículas
- $\#v_x = np.zeros((numParticles,1),dtype=float)$
- $g = 9.81$ $[m]/[s^2]$ gravidade
- $v_x = np.random.uniform(-0.1,0.1,(numParticles,1))$
- $\#v_y = np.zeros((numParticles,1),dtype=float)$
- $v_y = np.random.uniform(-0.1,0.1,(numParticles,1))$
- $time = 0.0$ $[s]$ tempo
- $x = np.random.uniform(2.0,1.0,(numParticles, 1))$
- $\#x = np.zeros((numParticles,1),dtype=float)$
- $y = np.random.uniform(0.8h,0.2h,(numParticles, 1))$
- $\#y = np.zeros((numParticles,1),dtype=float)$

passo de tempo

- $dt = 0.01$ $[s]$ passo de tempo

Referências

- [1] J. Fish and T. Belytschko. *A First Course in Finite Elements*. John Wiley & Sons, Ltd, 1 edition, 2007.
- [2] A.O. Fortuna. *Técnicas Computacionais para Dinâmica dos Fluidos*. edUSP, 1a. edition, 2000.
- [3] M.T Heath. *Scientific Computing - An Introductory Survey*. McGraw Hill, 2a. edition, 2002.