

INTRODUÇÃO À PROGRAMAÇÃO NUMÉRICA PARA ENGENHEIROS MECÂNICOS

Preparado por: Prof. Gustavo Anjos
6 de Setembro de 2017

Resumo. Este texto de nível introdutório tem como objetivo familiarizar o estudante de graduação e pós-graduação na criação de ferramentas e solução de problemas reais encontrados em engenharia mecânica que podem ser modelados através de equações diferenciais. A linguagem de programação é usada para processamento de dados e também para visualização de resultados, mostrando-se versátil e uma ferramenta indispensável para o futuro profissional. Este texto não se restringe a alunos de engenharia mecânica, mas também pode ser usado por alunos que desejam obter conhecimento em programação numérica de solução de problemas diferenciais. Este texto também pode oferecer base para a construção de códigos numéricos elaborados.

Conteúdo

1	Introdução	2	3.4	Lista de nós de contorno em malha 2d	15
1.1	Considerações iniciais	2	3.5	Curvatura 2D	16
1.2	Importância da linguagem Python na Engenharia Mecânica	2	3.6	Curvatura 3D	18
1.3	Orientação a objetos	2	3.7	Qualidade de malha	20
1.4	Solução de problemas diferenciais	2	4	Mecânica do contínuo	20
1.5	Como medir erros	3	4.1	Método de Diferenças Finitas	20
1.6	Importância da validação do código numérico	4	4.1.1	Introdução	20
1.7	Controle de versões - GIT	4	4.1.2	Solução de problema térmico permanente 1D	21
1.8	Editor de texto - VIM	5	4.1.3	Solução de problema térmico permanente com geração de calor 1D	23
1.9	Visualização de resultados	6	4.1.4	Solução de problema térmico transiente 1D	23
2	Mecânica de massas pontuais	6	4.1.5	Solução de problema térmico transiente 1D com geração de calor	24
2.1	Movimento Horizontal de um Carrinho	6	4.1.6	Solução de equação de transporte transiente	24
2.2	Velocidade terminal de uma gota	8	4.2	Método de Elementos Finitos	26
2.3	Lançamento de projétil	8	4.2.1	Introdução	26
2.4	Sistema massa-mola	10	4.2.2	Solução de problema térmico permanente 1D	29
2.5	Sistema massa-mola dissipativo	11	5	Escoamento multifásico	30
2.6	Sistema massa-mola vertical	11	5.1	Introdução	30
2.7	Pêndulo simples	12	5.2	Escoamento com partícula - solução numérica	30
3	Geração de Malha	13	5.3	Escoamento com múltiplas partículas	32
3.1	Classificação	13	5.3.1	Acoplamento <i>One-way</i> sem choque entre partículas	32
3.2	Geração de malha 1D	13			
3.3	Geração de malha 2D	14			

1 Introdução

1.1 Considerações iniciais

Este texto foi elaborado para atender as necessidades de ensino de programação numérica a alunos de Transmissão de Calor II do 8º período de Engenharia Mecânica da Universidade do Estado do Rio de Janeiro. Nas primeiras seções o estudante pode obter informações sobre ferramentas usadas em programação numérica, como o editor de texto, controle de versões e o paradigma de orientação a objetos. Seguindo o conteúdo do texto, diversos exercícios são propostos para solução de problemas encontrados em engenharia, como a mecânica de massas pontuais, onde a segunda lei de Newton é utilizada para modelagem do problema do tipo massa-mola ou ainda lançamento de um projétil. Outros exercícios são sugeridos para solução de problema térmico unidimensional e a criação de malhas sofisticadas para o uso no método de elementos finitos. Este texto é finalizado com problemas envolvendo escoamentos multifásicos onde duas fases distintas (contínua e dispersa) coexistem no mesmo problema.

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 no livro Mecânica Clássica [23] 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 (ver [2]).

Abaixo encontra-se alguns conceitos relacionados à programação orientada a objetos.

- classe: estrutura para abstração de um conjunto de objetos com características em comum através da utilização de métodos e atributos;
- objeto/instância: elemento computacional que representa uma entidade abstrata ou concreta;
- atributo: são elementos que definem as partes (ou estruturas) de uma classe. Os valores associados aos atributos definem os objetos.
- método: é a função que é executada pelo objeto ao receber instrução;
- herança: é um princípio que permite que classes compartilham métodos e atributos de outra classe;
- encapsulamento: conceito de tornar o programa mais flexível, fácil de modificar e de usar, escondendo detalhes do programa desnecessários para o conhecimento do usuário final;
- polimorfismo: uso do mesmo nome para múltiplos métodos com lista de argumentos diferentes;

Descrever diagrama UML (Unified Modeling Language).

1.4 Solução de problemas diferenciais

- **Solução geral:** solução que apresenta n constantes independentes entre si, onde n é a ordem da EDO. Essas constantes, de acordo com a conveniência, podem ser escritas como c , $2c$, c^2 , $\ln c$;
- **Solução Particular:** solução obtida a partir da solução geral do problema, mediante a imposição das condições dadas (chamadas condições iniciais ou condições de contorno).

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 Como medir erros

Toda vez que não conseguimos representar um valor exato, adicionamos um erro no sistema. Dependendo do algoritmo, este erro pode ser propagado até o final do processo, correndo o risco de acumular e fazer com que haja convergência para a solução do problema proposto.

- **erro de truncamento:** é a diferença entre o resultado real e o resultado que deveria ser produzido por um algoritmo dado usando aritmética exata. Este erro é devido a aproximações de truncamento de séries infinitas, como por exemplo substituindo uma derivada por uma diferença finita, ou substituindo um dízima periódica por um número com 3 casas decimais, ou pela substituição de uma função arbitrária por um polinômio, ou ainda pela parada de um processo iterativo antes da convergência.
- **erro de arredondamento:** é a diferença entre o resultado produzido por um algoritmo dado usando aritmética exata e o resultado do mesmo algoritmo usando precisão finita, aritmética de arredondamento. É devido ao falta de exatidão na representação de um número real e operações aritméticas em cima destes números.
- **erro absoluto:** valor aproximado - valor exato
- **erro relativo:** $\frac{\text{erro absoluto}}{\text{valor exato}}$

1.6 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 [9]).

1.7 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. Este controle digital se assemelha a um caderno de anotações, onde o estudante anota o desenvolvimento de suas atividades em cada dia de trabalho quando executa uma tarefa qualquer. A qualquer momento o estudante pode voltar nas anotações e saber o que foi modificado ou realizado na data em questão. Analisando mais profundamente esta mesma situação, poderíamos envolver mais um estudante para realizar o mesmo trabalho, onde cada pessoa ficaria encarregada de um turno no dia. Se as anotações são feitas rigorosamente, o estudante seguinte toma conhecimento das ações e decisões tomadas pelo anterior. Existem diversos controladores de versão que ainda são bastante utilizados pela comunidade: *SVN*, *CVS*, *Git*, *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 (VLC, Reddit, rsync, Samba, VTK). O *Git* é um software livre e pode ser instalado em todos os sistemas operacionais através de download na página (<https://git-scm.com>).

Abaixo encontra-se alguns comandos do git bastante utilizados:

- `git add "nome-do-arquivo"` adiciona um arquivo no git
- `git commit` submete as modificações marcadas por add para o git
- `git commit -a` submete todas as modificações para o git
- `git log` mostra descrição do desenvolvimento do projeto
- `git shortlog` mostra descrição curta do desenvolvimento do projeto
- `git status` verifica modificações no diretório do projeto

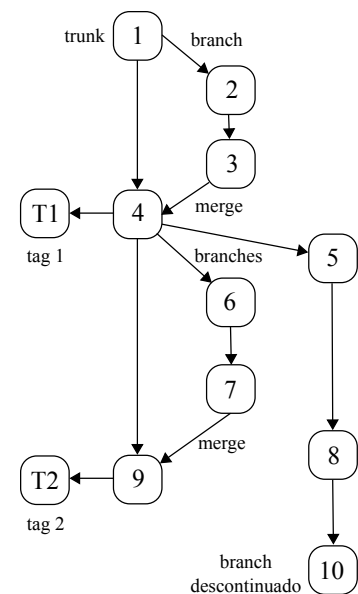


Figura 1: Exemplo da visualização do histórico de um projeto usando um sistema de controle de versões

- git branch lista branches do projeto
- git init inicializa um diretório git
- git push enviar modificações para o servidor
- git pull atualiza versão local pela versão do servidor

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 e de LaTeX.

1.8 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
- :wqa 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.

1.9 Visualização de resultados

A visualização do resultado é de extrema importância no cálculo numérico. Ela permite uma análise qualitativa dos resultados, mostrando tendências e permitindo sua compreensão global, ao invés de analisarmos uma sequência de números em uma tabela. A visualização pode ser gráfica ou através de texto, dependendo do problema a ser estudado. A plotagem de resultados em eixos bidimensionais também é fundamental para se conhecer os resultados e seu comportamento. Existem diversas ferramentas de visualização disponíveis e cabe ao estudante decidir qual ferramenta é mais apropriada para seu problema.

O **matplotlib** é uma biblioteca gráfica disponível para a linguagem Python que apresenta diversos recursos para visualização de dados. Por ser uma biblioteca, o estudante poderá aproveitar parte de sua programação numérica como entrada para os gráficos. Esta biblioteca possibilita a visualização de gráficos em diversos tipos de formato, como gráfico de barras, funções, gráficos tipo pizza, distribuições, visualização de superfícies, além de prover algumas funcionalidades relacionadas ao processamento gráfico, como um gerador de malhas triangulares usando algoritmo de Delaunay. A biblioteca pode ser encontrada no seguinte link.

<https://matplotlib.org>

O **Paraview** é uma ferramenta de código aberto e gratuita para visualização científica e análise de dados, compatíveis com diversos formatos de entrada de resultados. Diversos recursos estão disponíveis para processamento de dados e geração de gráficos auxiliares, possibilitando uma análise mais detalhada dos resultados obtidos. Gráficos bidimensionais e tridimensionais são facilmente processados pelo **Paraview** através de botões e comandos simples. Além disso, diversos recursos mais sofisticados estão disponíveis ao usuário para que se possa realizar um pós-processamento detalhado dos resultados obtidos em simulações numéricas. Módulos de Python também estão disponíveis para acessar os conteúdos do **Paraview** através de instruções em linguagem de programação. Esta ferramenta está disponível nos principais sistemas operacionais e pode ser encontrada no seguinte endereço:

<http://www.paraview.org>

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).

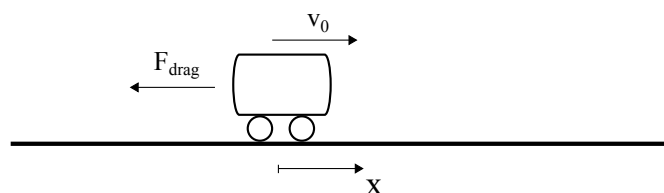


Figura 2: Desaceleração de carrinho por atrito do ar linear $F_{drag} = -bv$.

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

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

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

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 Δt . No limite onde Δt 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:

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

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 Δt .

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:

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]
$m = 1.0$	$b = 0.1$	$\Delta t = 0.1$

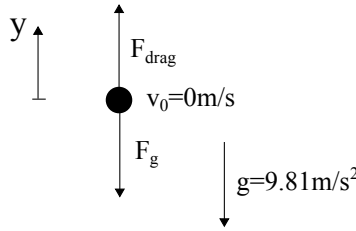
Condições iniciais:

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]
$t = 0.0$	$v_0 = 10.0$	$x = 0.0$

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.

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

$$v^{n+1} = v^n + \Delta t \frac{bv}{m} - \Delta t g \quad (13)$$

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 Δt .

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 $dx/dt = v$:

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

2.3 Lançamento de projétil

Neste exemplo deseja-se calcular como a força de atrito linear $F = bv$, ou a força de atrito quadrática $F = cv^2$ atuam em uma partícula ao ser lançada como um projétil de massa m sob efeito da força de gravidade $F = mg$.

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

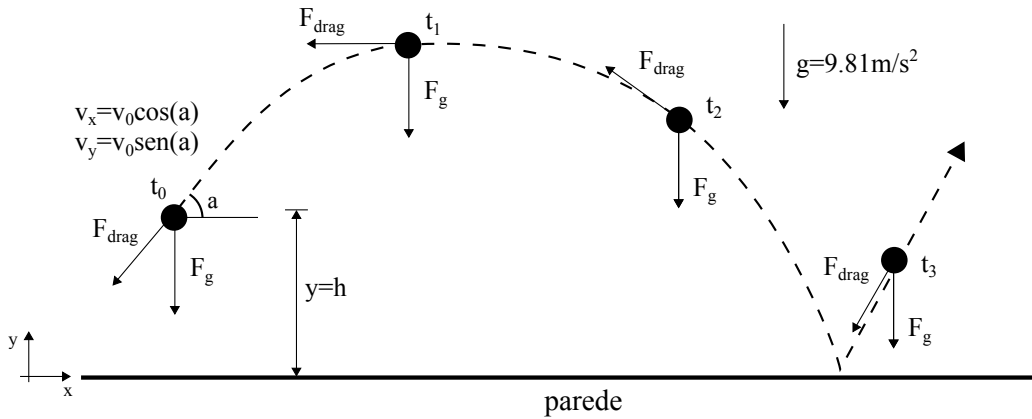
diâmetro da gota óleo	densidade do líquido	aceleração da gravidade	volume da gota
$[m]$	$[kg]/[m^3]$	$[m]/[s^2]$	$[m^3]$
$D = 1.5e - 06$	$\rho = 840.0$	$g = 9.81$	$V = \pi D^3/6.0$

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
$[kg]$	$[kg]/[m][s]$	$[kg]/[s]$	$[s]$
$m = \rho V$	$\beta = 1.6e - 04$	$b = \beta D$	$\Delta t = 1e - 7$

Como condição de contorno para este problema, adota-se condição de parede elástica em $y = 0$, ou seja, o projétil é rebatido com velocidade contrária igual à de choque, sem absorção de energia cinética pela parede.

A Fig. (5) mostra esquematicamente a trajetória do projétil ao ser lançado de uma altura $y = h$ com velocidade v_x e v_y definidas. Ao tocar na parede inferior, o projétil é rebatido e continua sua trajetória. A condição de contorno pode ser incorporada ao problema através de uma condição de monitoramento altura do projétil, ou seja, quando a altura do projétil h for menor que 0, a velocidade vertical v_y deverá mudar de sentido. Este esquema é um modelo matemático para impor condição de parede e rebatimento do projétil.


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

Para o caso de atrito linear onde $F = bv$, as equações de movimento em x e y são desacopladas, ou seja, as equações podem ser resolvidas separadamente pois uma não depende da outra. Entretanto, para o caso de atrito quadrático $F = cv^2$, as equações estão acopladas, pois $v^2 = v\sqrt{v^2} = v\sqrt{v_x^2 + v_y^2}$. Este acoplamento impede que as equações sejam resolvidas separadamente, logo tornando o problema mais sofisticado. Analiticamente é possível encontrar a solução deste problema para o caso de atrito linear. Por outro lado, a equação de movimento com atrito quadrático não tem solução analítica, fazendo com que o procedimento numérico seja a única maneira de se resolver o problema. A equação vetorial (em x e y) toma a seguinte forma:

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_g \quad \begin{cases} m \frac{dv_x}{dt} = F_{drag_x} \\ m \frac{dv_y}{dt} = F_{drag_y} + F_{g_y} \end{cases} \quad (16)$$

Dados da simulação;

- $\beta = 0.25$ $[kg/ms]$ 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

Tabela 5: Dados da simulação de lançamento de projétil.

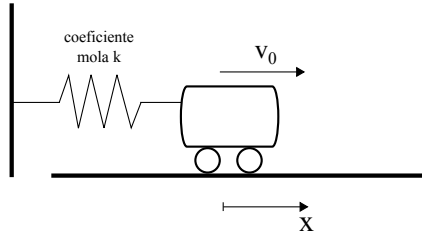
diâmetro	volume	massa	viscosidade	coeficiente	gravidade	passo de tempo
$[m]$	$[m^3]$	$[m]$	$[kg]/[m][s]$	$[kg]/[s]$	$[m]/[s^2]$	$[s]$
$D = 1.5e - 6$	$V = \pi D^3 / 6.0$	$m = 0.15$	$\beta = 0.25$	$b = \beta D$	$g = 9.81$	$\Delta t = 0.01$

Tabela 6: Condições iniciais da simulação de lançamento de projétil.

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

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:

$$m \frac{dv}{dt} = -kx \quad (17)$$

$$v(t) = v_0 e^{-bt/m} \quad (18)$$

$$x(t) = \int_0^t v(t) dt = x_\infty (1 - e^{-bt/m}) \quad (19)$$

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

Dados da simulação

- $m = 1.0$ massa da partícula
- $k = 0.1$ coeficiente da mola
- $\Delta t = 0.1$ passo de tempo

Condições iniciais:

- $t = 0.0$ 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 :

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{spring} + \mathbf{F}_{friction} \quad (20)$$

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:

- $t = 0.0$ 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 :

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

$\mathbf{F}_{spring} = F_{spring_y} - ky$ 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
- $t = 0.0$ 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 $F_g = mg$:

$$m \frac{d^2\theta}{dt^2} + \frac{mg}{l} \sin \theta = 0 \quad (22)$$

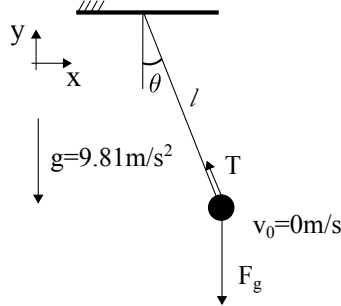


Figura 6: Pêndulo simples sob efeito de força gravitacional $F_g = mg$.

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:

$$\frac{d\theta}{dt} = v \quad (23)$$

e

$$m \frac{dv}{dt} + \frac{mg}{l} \sin \theta = 0 \quad (24)$$

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

$$\frac{\theta^{n+1} - \theta^n}{\Delta t} = v^n \longrightarrow \theta^{n+1} = \theta^n + \Delta t v^n \quad (25)$$

E depois a Eq. (24) para encontrar v no tempo $n + 1$:

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

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:

$$m \frac{d^2\theta}{dt^2} + \frac{mg}{l} \theta = 0 \quad (27)$$

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

$$\theta(t) = \theta_0 \cos(\sqrt{g/l}t) \quad (28)$$

Dados da simulação:

Condições iniciais:

Tabela 7: Dados da simulação do pêndulo submetido à força de gravidade.

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

Tabela 8: Condições iniciais da simulação do pêndulo simples.

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

3 Geração de Malha

3.1 Classificação

Malhas são estruturas geométricas usadas na discretização de equações matemáticas que serão resolvidas numericamente pelo computador. Malhas unidimensionais são definidas ao longo do eixo x com coordenadas que podem ou não variar linearmente. Para o caso de malhas bidimensionais, as coordenadas são definidas no plano $x - y$ através de pares de valores que informam a localização geométrica de cada ponto no plano. Em malhas tridimensionais, as coordenadas dos pontos são definidas no volume, ou seja, são definidas através de coordenadas x , y e z . Frequentemente em métodos numéricos a informação da interconectividade de pontos em malhas computacionais é essencial, não bastando somente a localização geométrica de cada ponto. Com isso, além das coordenadas dos pontos, uma estrutura numérica adicional se faz necessária para representação da malha computacional.

Quando se resolve uma equação através de uma aproximação numérica, o resultado é obtido nos pontos da malha computacional. (falta completar)

Quando se discretiza uma equação diferencial, uma equação é definida para cada ponto da malha. Com isso, quanto mais pontos de malha, maior é o número de equações definidas, resultando em um consumo maior de processamento computacional, já que o número de operações lógicas será proporcional ao número de pontos discretos.

- malha estruturada:
- malha não estruturada:

A geração de malhas é um problema matemático que pode ser de difícil solução. A seguir são apresentados problemas simples de geração de malha em espaços unidimensionais e bidimensionais.

3.2 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.

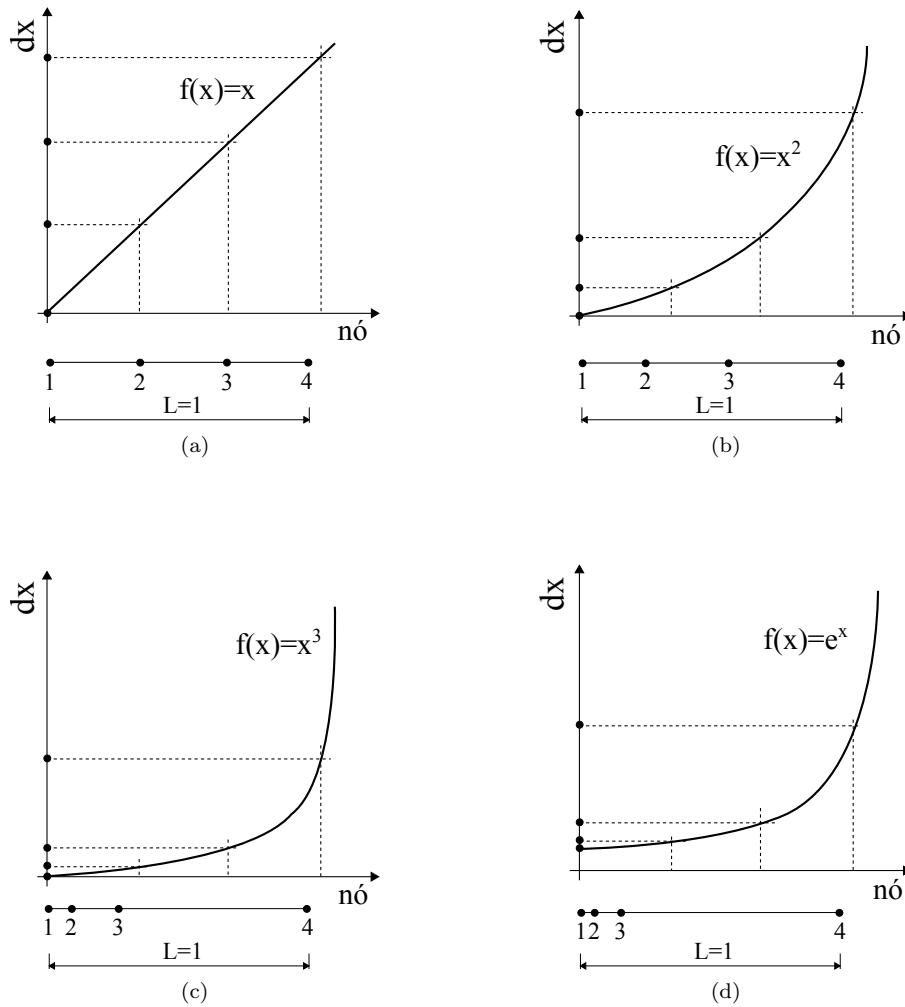


Figura 7: Distribuição de espaçamento de malha dx e representação final da malha unidimensional com dimensão $L = 1$ usando função (a) linear, (b) quadrática, (c) cúbica e (d) exponencial.

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.

3.3 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

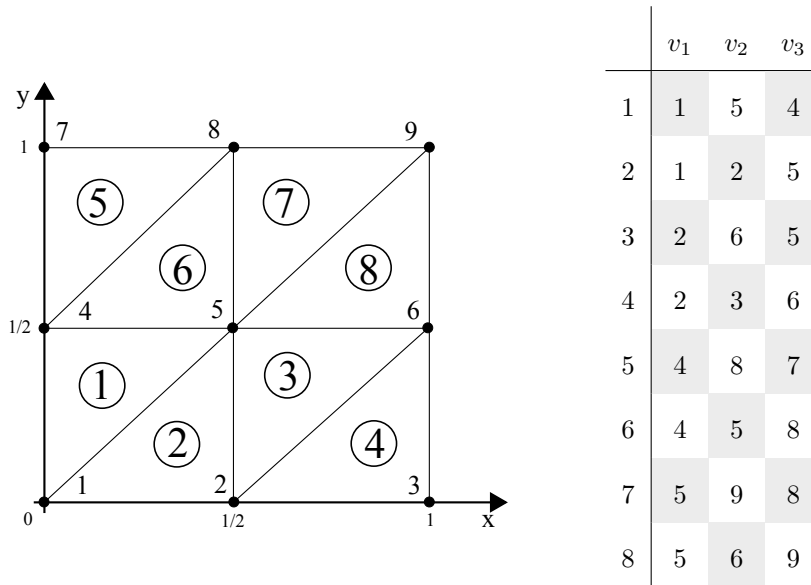


Figura 8: Distribuição de pontos e elementos em malha bidimensional. A numeração dos elementos é definida pelos números com círculos, enquanto os índices dos nós estão localizados ao lado do nó e estrutura da matriz de conectividade IEN_{ij} . Note que a numeração do primeiro vértice e do primeiro elemento começam com 1. Em algumas linguagens de programação, como o Python e C/C++, é conveniente começar a numeração com 0.

- $ne = 2(nx - 1)(ny - 1)$ número total de elementos

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:

$$y(x) = A \sin\left(\frac{2\pi}{\lambda}x - \phi\right) \quad (29)$$

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 $Y = 0$.

3.4 Lista de nós de contorno em malha 2d

Considere uma malha arbitrária bidimensional com contornos irregulares, a matriz de conectividade e os vetores de coordenadas como mostrado na Fig. (10).

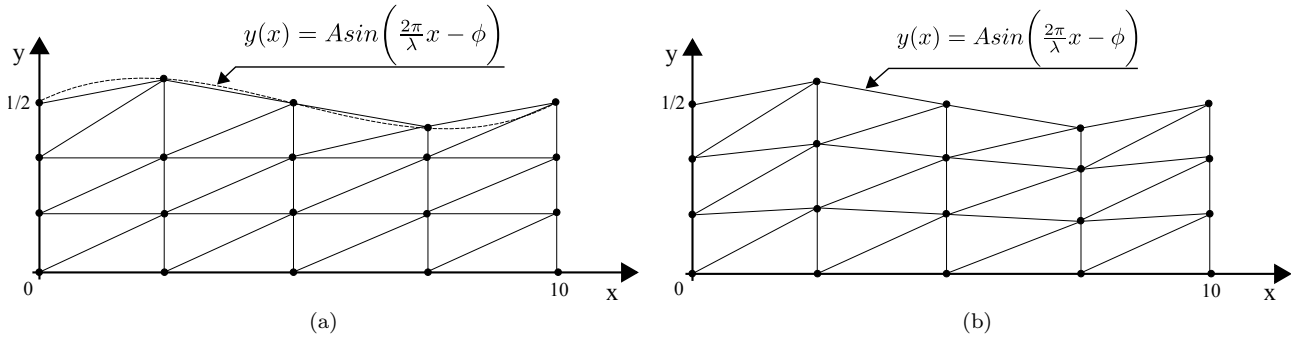
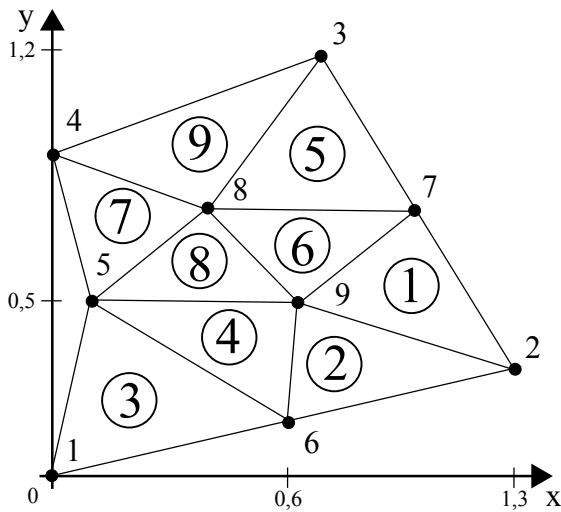


Figura 9: Malha bidimensional de triângulos com perturbação senoidal no (a) topo e (b) propagada para o interior com amortecimento da função.



IE	v_1	v_2	v_3	X	Y
1	9	2	7	0.0	0.0
2	6	2	9	1.3	0.3
3	1	6	5	0.6	0.6
4	5	6	9	0.0	0.9
5	3	8	7	0.2	0.2
6	7	8	9	0.6	0.1
7	4	5	8	0.9	0.9
8	5	9	8	0.4	0.8
9	3	4	8	0.7	0.5

Figura 10: Distribuição de pontos e elementos em malha bidimensional arbitrária.

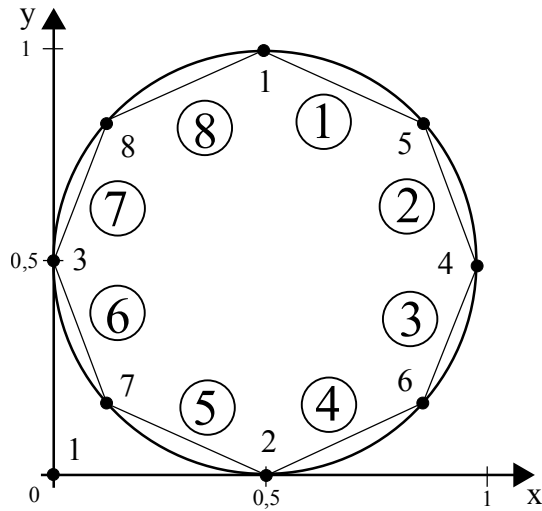
Processe duas listas contendo os nós do contorno $\Gamma = [1, 2, 3, 4, 5, 6, 7]$ e os nós do interior da malha [8, 9] em vetores diferentes, de modo que o algoritmo criado possa ser utilizado em qualquer malha bidimensional. Note que o algoritmo deve ser generalizado para qualquer malha, não devendo ser específico para a malha apresentada na Fig. (10). Com isso, é necessário que o estudante desenvolva um procedimento para classificação dos nós de interesse através de um algoritmo otimizado para processamento de malhas com grande quantidade de nós.

Dica: Crie uma lista contendo todas as arestas da malha e verifique quantos elementos compartilham cada uma das arestas. A diferença entre a informação das arestas produzirá as outras listas com nós de contorno e de interior.

3.5 Curvatura 2D

Curvatura é a quantidade na qual um objeto geométrico se desvia do plano, portanto a curvatura de um plano é zero. O cálculo da curvatura é importante em muitas aplicações de engenharia. Como exemplo importante, cita-se o cálculo da força de tensão superficial em escoamentos bifásicos que apresentam interface maleável. Este é o caso de escoamentos que envolvem líquido-líquido e líquido gás, ou ainda processo com mudança de fase, como em evaporação e condensação.

Utilizando as fórmulas de *Frenet* em 2D (ver mais detalhes em [15, 22]), a curvatura média k pode ser interpretada geometricamente como a variação do vetor tangente \mathbf{t} (ou do vetor normal \mathbf{n}) ao longo da curva s que



IEN	v_1	v_2	X	Y
1	1	5	0.0	0.0
2	5	4	0.5	0.0
3	4	6	0.0	0.0
4	6	2	1.0	0.5
5	2	7	0.853	0.853
6	7	3	0.853	0.146
7	3	8	0.146	0.146
8	8	1	0.146	0.853

Figura 11: Distribuição de pontos e elementos da malha do círculo. A matrix de conectividade apresenta elementos orientados. Note que os elementos são segmentos de retas contidos no espaço \mathbb{R}^2 , portanto apresentam coordenadas x e y .

define a interface, ou seja:

$$k\mathbf{n} = \frac{\partial \mathbf{t}}{\partial s}, \quad k\mathbf{t} = \frac{\partial \mathbf{n}}{\partial s} \quad (30)$$

Ao longo de uma curva discreta, a curvatura média discreta pode ser aproximada por:

$$k\mathbf{n} = \frac{\mathbf{t}_2 - \mathbf{t}_1}{\Delta s} \quad (31)$$

onde Δs é a soma dos comprimentos das duas arestas, ou seja, é uma aproximação do arco da circunferência que passa nos três pontos. Uma outra maneira de aproximar o arco da circunferência Δs é utilizar a distância entre os nós vizinhos ao ponto i . Uma terceira maneira de aproximação é calcular um arco de circunferência que passa nos três nós através de um ajuste de curva.

Para ilustrar simplificadamente um problema envolvendo o cálculo de curvatura média discreta k , considere uma malha de curva fechada com elementos de segmentos de reta conectados como mostrado na Fig. (11). Note que o cálculo da curvatura não requer uma curva fechada, podendo também ser aplicado em curvas abertas. Calcule a curvatura discreta em cada ponto da malha conhecendo a matriz de conectividade e os vetores de coordenadas x e y .

Considere o ponto 1 com coordenadas $x = 0.5$ e $y = 1.0$. Os vetores tangenciais unitários \mathbf{t}_{18} e \mathbf{t}_{15} podem ser obtidos conhecendo as coordenadas dos pontos 1, 5 e 8 e fazendo:

$$t_{18}x = \frac{x_8 - x_1}{\|x_8 - x_1\|}, \quad t_{18}y = \frac{y_8 - y_1}{\|y_8 - y_1\|} \quad (32)$$

$$t_{15}x = \frac{x_5 - x_1}{\|x_5 - x_1\|}, \quad t_{15}y = \frac{y_5 - y_1}{\|y_5 - y_1\|} \quad (33)$$

onde $\|x_i - x_j\|$ representa a distância entre os pontos i e j .

O vetor normal unitário associado a este ponto e apontado para fora do círculo é definido por $nx = 0.0$ e $ny = 1.0$. Note que a orientação da normal é por padrão orientada para fora, entretanto não há diferença no cálculo da curvatura média se a normal for orientada para dentro do círculo. Para um ponto genérico i da malha, o cálculo do vetor normal \mathbf{n} pode ser realizado através da rotação dos vetores tangenciais unitários e a posterior soma dos dois vetores como mostrado na Fig. (12).

Para o cálculo dos vetores tangentes \mathbf{t}_1 e \mathbf{t}_2 é necessário o processamento da malha para que se construa uma estrutura computacional (lista ou sequência) com a informação dos vizinhos de cada nó, com isso sugere-se construir uma lista associando ao nó 1 os nós 5 e 8. Ao nó 2 associa-se os nós vizinhos 7 e 6, e assim por diante.

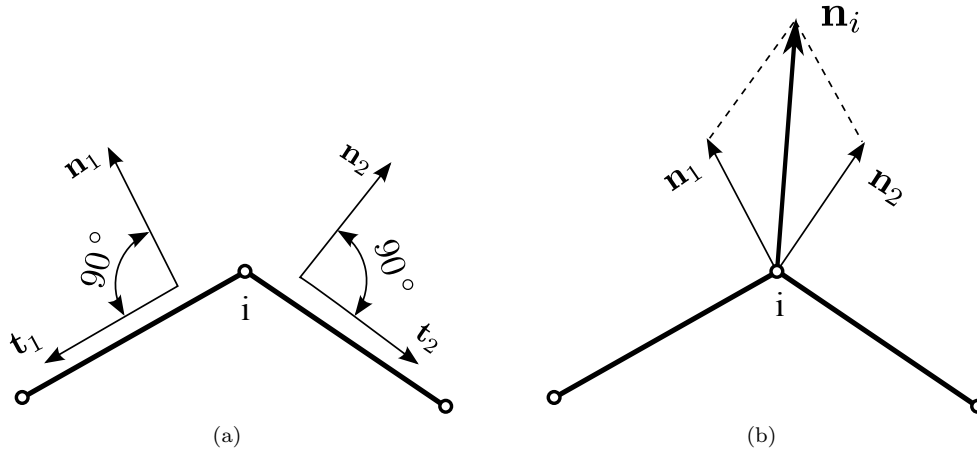


Figura 12: Cálculo do vetor normal em espaço bidimensional. (a) O vetor normal de cada aresta pode ser encontrado pela rotação do vetor tangente calculado anteriormente por 90° . (b) O vetor normal \mathbf{n}_i é encontrado através da soma dos dois vetores normais unitários \mathbf{n}_1 and \mathbf{n}_2 .

Com esse procedimento torna-se fácil o cálculo dos vetores normais \mathbf{n} e tangenciais \mathbf{t} , bem como o cálculo do comprimento do arco de circunferência s .

Calcule a curvatura de todos os pontos da malha apresenta na Fig. (11) e compare com a solução analítica da curvatura em um círculo através da seguinte equação:

$$k = \frac{1}{r} \quad (34)$$

onde r representa o raio da circunferência, que no caso do exercício em questão é igual a 0,5.

Crie outras malhas de curvas e segmentos de reta fechados e verifique se a curvatura calculada está com resultado próximo ou igual à solução analítica. Note que a curvatura de um segmento de reta é igual a zero.

3.6 Curvatura 3D

O cálculo de curvatura média k em três dimensões segue a mesma estratégia do cálculo de curvatura média bidimensional. Entretanto, o domínio do problema é tridimensional envolvendo uma superfície, cuja malha computacional pode ser representada por triângulos conectados. Uma extensão das fórmulas de *Frenet* para o caso tridimensional pode ser encontrada em detalhes em [1] e [4]. Nesta seção trataremos do caso tridimensional envolvendo uma superfície fechada contendo triângulos como elementos discretos. A Fig. (13) mostra uma representação esquemática de uma parte de um domínio computacional Ω_s de uma malha de triângulos, ilustrando o esquema adotado para o cálculo da curvatura média k no nó i . Verifica-se que o cálculo da curvatura média k_i é realizada através de operações geométricas envolvendo a 1a. família de nós vizinhos ao nó i . Entende-se por nós vizinhos, os nós que estão conectados diretamente ao nó de interesse.

A Fig. (14) mostra a avaliação da curvatura em cada elemento pertencente à família de vizinhos ao nó i , sendo que a curvatura k_i é obtida pela soma de todos os elementos triangulares vizinhos e ponderada pela soma das áreas baricêntricas através da Eq. (35). Na Fig. (14a), os pontos médios das arestas são conectados ao baricentro do triângulo para obtenção da área baricêntrica do triângulo. Na Fig. (14b) é ilustrada a metodologia para o cálculo de \mathbf{t}_n , que representa o vetor unitário da soma dos dois vetores unitários tangentes \mathbf{t}_1 e \mathbf{t}_2 . O vetor resultante \mathbf{t}_n deve então ser integrado no segmento de reta que liga os pontos médios das arestas do triângulo como descrito na Fig. (14c), transformando a carga concentrada em carga distribuída. Para o cálculo de k_i , deve-se somar o módulo da carga concentrada de todos os triângulos conectados ao nó i e dividir pelo somatório das áreas baricêntricas dos mesmos elementos triangulares considerados.

$$\kappa_i = \frac{\left| \sum_{j=1}^m (\mathbf{t}_n d)_j \right|}{\sum_{j=1}^m A_j^i} \quad (35)$$

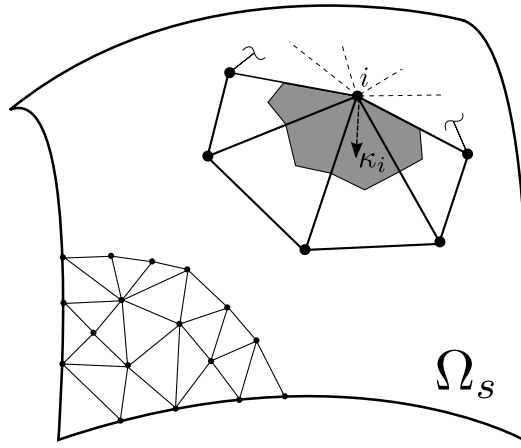


Figura 13: Representação esquemática do cálculo da curvatura média κ_i em um nó arbitrário de uma superfície, que pode ser realizado através de operações geométricas nos nós e triângulos vizinhos ao nó i and ponderados pela área baricêntrica (colorida de cinza).

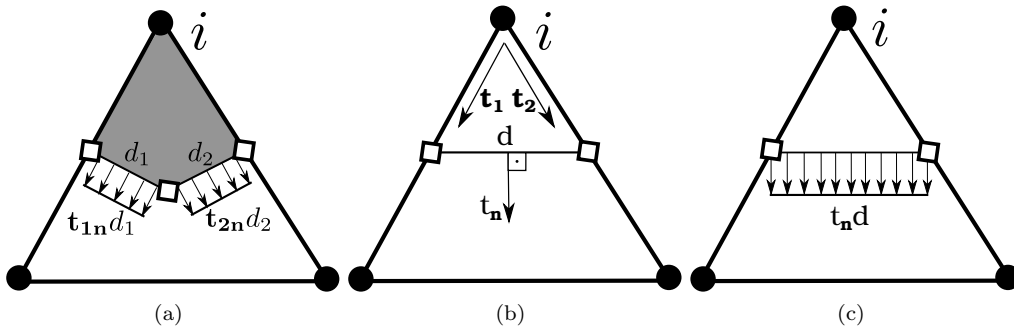


Figura 14: (a) A avaliação da força no elemento é feita através da soma das forças distribuídas $\mathbf{t}_{1n}d_1$ e $\mathbf{t}_{2n}d_2$. (b) Usando o teorema de Stokes, a força no elemento distribuída pode ser calculada ortogonalizando um dos dois vetores linearmente independentes \mathbf{t}_1 e \mathbf{t}_2 no segmento de reta d que conecta os dois nós médios da aresta. (c) Um cálculo da curvatura média κ_i é encontrado pela divisão do módulo do cálculo das forças distribuídas ($|\mathbf{t}_n d|$) pela soma das áreas baricênticas (Eq. 35.).

Para o cálculo discreto dos vetores normais associados ao nós da malha de superfície, sugere-se uma estratégia semelhante utilizada ao caso bidimensional. A normal n_i pode ser calculada através da soma dos vetores normais associados a cada triângulo vizinho ao nó i . Tais vetores normais podem ser obtidos através de produto vetorial dos vetores tangentes \mathbf{t}_1 e \mathbf{t}_2 . Com isso, soma-se os vetores normais unitários de todos os triângulos vizinhos ao nó i e obtém-se o vetor normal n_i .

Em [4], um método para o cálculo da área baricêntrica dos triângulos A_j é sugerida para melhorar a aproximação da curvatura k_i através da avaliação de qualidade dos triângulos vizinhos ao nó i .

Usando um gerador de malha como o GMSH (ver [8], construa uma malha de superfície esférica com raio $r = 0.4$ formada por triângulos e calcule o valor da curvatura de todos os pontos da malha. Refine a malha e recalcule os valores de k_i para todos os pontos. Verifique se o cálculo da curvatura melhora com o refinamento da malha. Em uma superfície esférica, a curvatura média é constante e apresenta solução analítica dada pela seguinte equação:

$$k = \frac{2}{r} \quad (36)$$

Faça uma estimativa do erro da curvatura numérica calculada usando a Eq. (35) e a solução analítica através da seguinte relação:

$$\text{Error}_\kappa = \sqrt{\frac{\sum (\kappa_i - \kappa_A)^2}{\sum (\kappa_i)^2}} \quad (37)$$

onde k_i representa a curvatura numérica calculada e k_A a curvatura analítica dada pela Eq. (36).

Repita o exercício para outras superfícies conhecidas, como o plano (curvatura média analítica nula), o cilindro e o torus.

3.7 Qualidade de malha

O monitoramento da qualidade de malha é um recurso que deve ser utilizado para o controle da distribuição de pontos e formação de elementos de malha, resultando diretamente na diminuição de erros associados às discretizações espaciais.

Em malhas triangulares, diversos parâmetros de medição de qualidade de malha estão disponíveis. Pode-se citar o controle de ângulos internos como parâmetro de qualidade de malha. Quanto mais elementos triangulares equiláteros, melhor é a aproximação espacial dos operadores.

Considere três malhas triangulares arbitrárias com diferentes graus de refinamento como mostradas na Fig. (15). Calcule a distribuição de qualidade de malha e plote em um gráfico de barras contendo a informação da quantidade de elementos por nível de classificação. Considere que um triângulo equilátero tenha medida de qualidade 1 e um triângulo com um dos ângulos internos tendendo a 180° tenha medida de qualidade 0.

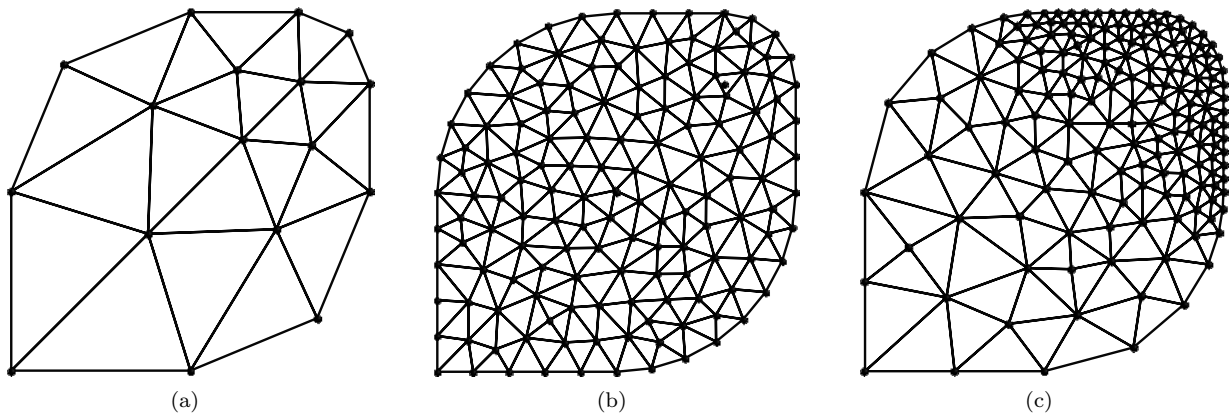


Figura 15: Malha arbitrária de triângulos criada pelo gerador de malha GMSH, disponível gratuitamente pela internet: (a) malha grossa e (b) malha uniformemente refinada e (c) malha refinada por região.

4 Mecânica do contínuo

4.1 Método de Diferenças Finitas

4.1.1 Introdução

O Método de Diferenças Finitas (MDF) é um método usado para análise numérica, através da discretização de equações diferenciais em malhas computacionais, através da aproximação de derivadas por diferenças finitas. O MDF utiliza as expansões em séries de Taylor para descrever as derivadas de uma variável utilizando as diferenças dos valores da variável em vários pontos no espaço ou no tempo. Exemplificando, toma-se f como contínua no intervalo $[a, b]$ no domínio unidimensional de interesse e de classe C^N , ou seja, possui derivadas até a ordem N e elas são contínuas. Aplicando-se o teorema de Taylor, obtem-se:

$$f(x) = f(x_0) + (\Delta x) \left. \frac{df}{dx} \right|_{x_0} + \frac{\Delta x^2}{2!} \left. \frac{d^2 f}{dx^2} \right|_{x_0} + \cdots + R_N \quad (38)$$

em que $\Delta x = x - x_0$ e R_N são os termos de ordem superior a 2 e sua derivada representada por:

$$\left. \frac{df}{dx} \right|_i = \frac{f(x_i + \Delta x) - f(x_i)}{\Delta x} - \left. \frac{(\Delta x)}{2!} \frac{d^2 f}{dx^2} \right|_i - R_d^N \quad (39)$$

onde i representa os pontos de malha e R_d^N os termos de ordem superior a 2.

A série de Taylor é infinita, sendo normalmente truncada a partir dos termos de ordem 2 inclusive. O truncamento introduz um erro que é inerente ao processo de discretização das equações e que, juntamente com o arredondamento nos cálculos feitos pelo computador e a aproximação numérica das condições iniciais e de contorno representa a principal fonte de erros deste método.

Nas seções seguintes, propõem-se utilizar o MDF para solução de problemas térmicos unidimensionais e bi-dimensionais e problemas de transporte, oferecendo ao estudante um panorama geral do método aplicado à problemas de engenharia térmica.

4.1.2 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:

$$\frac{d^2T}{dx^2} = 0 \quad (40)$$

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 [7]). 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. (40) em sua forma discreta utilizando diferenças finitas centradas, é reescrita para o nó da malha i como:

$$\frac{T_{i-1} - 2T_i + T_{i+1}}{dx^2} = 0 \quad (41)$$

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:

$$\frac{T_1 - 2T_2 + T_3}{dx^2} = 0 \quad (42)$$

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,

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

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.

$$\frac{T_1 - 2T_2 + T_3}{dx^2} = 0 \quad (43)$$

$$\frac{T_4 - 2T_3 + T_2}{dx^2} = 0 \quad (44)$$

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](#)).

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:

$$\frac{T_1 - 2T_2 + T_3}{dx^2} = 0 \longrightarrow 1T_1 = 0.0 \quad (45)$$

e

$$\frac{T_4 - 2T_3 + T_2}{dx^2} = 0 \longrightarrow 1T_4 = 1.0 \quad (46)$$

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 [6]). 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.

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

4.1.3 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:

$$\alpha \frac{d^2 T}{dx^2} = Q \quad (47)$$

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

4.1.4 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:

$$\frac{dT}{dt} = \alpha \frac{d^2 T}{dx^2} \quad (48)$$

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

A solução da equação em estado permanente se escreve:

$$T(x) = c_1 x + c_2 \quad (49)$$

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:

$$c_1 = \frac{T_{s2} - T_{s1}}{L} \quad c_2 = T_{s1} \quad (50)$$

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:

$$T(x) = \frac{T_{s2} - T_{s1}}{L}x + T_{s1} \quad (51)$$

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

$$T(x) = \frac{x}{L} \quad (52)$$

4.1.5 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:

$$\frac{dT}{dt} = \alpha \frac{d^2T}{dx^2} + Q \quad (53)$$

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

A solução da equação em estado permanente se escreve:

$$T(x) = -\frac{QL}{2k} \left(x - \frac{x^2}{L} \right) + \frac{T_{s2}}{L}x + T_{s1} \quad (54)$$

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:

$$T(x) = -\frac{QL}{2k} \left(x - \frac{x^2}{L} \right) \quad (55)$$

4.1.6 Solução de equação de transporte transiente

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 $T(t, x)$:

$$\frac{dT}{dt} + v \frac{dT}{dx} = 0 \quad (56)$$

O número de Courant (CFL) é definido como:

$$CFL = v \frac{dt}{dx} \quad (57)$$

Tabela 11: Condições iniciais da simulação para equação de transporte.

tempo inicial	velocidade inicial	CFL	perfil inicial
[s]	[m]/[s]	—	[m]
$t = 0.0$	$a = 1.0$	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:

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{T_{i+1}^n - T_{i-1}^n}{2dx} = 0 \quad (58)$$

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:

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{T_{i+1}^n - T_i^n}{dx} = 0 \quad (59)$$

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{T_i^n - T_{i-1}^n}{dx} = 0 \quad (60)$$

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

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{-1T_{i+2}^n + 4T_{i+1}^n - 3T_i^n}{2dx} = 0 \quad (61)$$

$$\frac{T_i^{n+1} - T_i^n}{dt} - v^n \frac{3T_i^n - 4T_{i-1}^n + T_{i-2}^n}{2dx} = 0 \quad (62)$$

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} :

$$\frac{DT}{Dt} \approx \frac{T^{n+1}(x^n) - T^n(x_d^{n-1})}{\Delta t} = 0 \quad (63)$$

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

$$\frac{x^n - x_d^{n-1}}{\Delta t} = v^n \quad (64)$$

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

$$x_d^{n-1} = x^n - \Delta t v^n \quad (65)$$

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 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} :

$$\frac{T^{n+1}(x^n) - T^n(x_d^{n-1})}{\Delta t} = 0 \quad (66)$$

$$T^{n+1}(x^n) = T^n(x_d^{n-1}) \quad (67)$$

4.2 Método de Elementos Finitos

4.2.1 Introdução

O Método de Elementos Finitos é largamente utilizado para encontrar a solução de problemas diferenciais. Nos anos 50, o método de elementos finitos teve grande utilização na mecânica de sólidos. Apenas a partir da década de 70, após a consolidação do método de Galerkin para equações de difusão, pesquisadores começaram a investir no campo da dinâmica dos fluidos, pode-se citar [24], [18], [20], [3], [13], [21] e tantos outros. Em seguida, vários autores contribuíram para o desenvolvimento de metodologias específicas como métodos de Petrov-Galerkin generalizados [10], [11], [14], métodos adaptativos [19], métodos de Taylor-Galerkin [5], [17], método de Galerkin descontínuo [20] etc.

O começo relativamente tardio no campo da Física de fluidos se deve, principalmente, à presença do termo convectivo e ao forte acoplamento entre velocidade e pressão, presentes nas equações de conservação. O termo convectivo apresenta produto de incógnitas, caracterizando a não-linearidade do problema e gerando operadores não simétricos, de difícil solução. Com o aumento do número de *Reynolds*, o termo convectivo exerce maior influência no escoamento, aumentando ainda mais a dificuldade de solução das equações.

No caso de transferência de calor, o MEF pode ser utilizado para solucionar problemas de condução em barras e chapas, bem como em problemas que envolvam convecção e acoplamentos não-lineares. Além disto, o MEF é um método poderoso para discretização em malhas com geometrias complexas.

No método de elementos finitos, o domínio é dividido em um número de sub-domínios finitos conhecidos como elementos. Uma simples função é utilizada para caracterizar a variação de cada variável dentro do elemento. Esta função recebe o nome *função de forma*. A justaposição da variação das variáveis em cada elemento é usada para descrever todo o campo da variável de interesse.

Podem-se considerar como características principais do método dois conceitos:

- a utilização da forma fraca, ou variacional, do problema;
- solução aproximada da equação variacional através do uso de funções dos elementos finitos.

A forma fraca é resultado da ponderação da equação original em sua forma forte (forma diferencial inicial, não alterada) em um domínio qualquer. É necessário encontrar uma solução tentativa u que satisfaça a seguinte expressão:

$$\int_{\Omega} \left(\frac{du}{dx} \right)^2 dx < \infty \quad (68)$$

Uma função que satisfaça a Eq. (68) é chamada de **função** H^1 . Uma outra classe de funções é necessária, são as chamadas **funções peso**, que são funções do tipo H^1 com uma característica a mais: para o caso onde a condição de contorno de Dirichlet, a função peso é nula. É importante notar que a forma forte e a forma fraca das equações de governo são equivalentes (ver [12]). Considere agora a forma forte da equação de calor em 1D:

$$\frac{d}{dx} \left(k \frac{du}{dx} \right) - f = 0 \quad \text{em} \quad \Omega \quad (69)$$

$$u = u_1 \quad \text{em} \Gamma_1 \quad (70)$$

$$u = u_2 \quad \text{em} \Gamma_2 \quad (71)$$

onde u é o campo de interesse do problema, x é a variável independente espacial, Γ_1 e Γ_2 representam os dois pontos extremos do domínio Ω , que para este estudo é uma barra unidimensional, ou seja, as variações de temperatura T nas direções y e z podem ser desprezadas. As condições de contorno são do tipo Dirichlet, ou seja, u é definido nas extremidades do domínio: $u = u_1$ em Γ_1 e $u = u_2$ em Γ_2 .

O termo de derivada mais alta deve passar pelo processo de integração por partes para separação das parcelas do domínio e do contorno e para redução da ordem do operador. Este procedimento de redução de ordem do

operador possibilita a escolha posterior de funções de forma de ordem mais baixa, reduzindo o custo computacional:

Achar $u \in H_1^1$ tal que, para qualquer $W \in H_0^1$

$$\int_{\Omega} w \left[k \frac{d}{dx} \left(\frac{du}{dx} \right) - f \right] = 0 \quad (72)$$

onde w representa a função peso e Ω representa o domínio do problema, que para esse caso-exemplo está contido no intervalo fechado, $\Omega \in [0, 1]$.

A integração por partes resulta em duas parcelas como mostrado na Eq. (73), a primeira é avaliada no contorno Γ e a segunda no domínio Ω . Note que até este momento, nenhuma aproximação foi realizada e o problema ainda não foi discretizado.

$$w \frac{du}{dx} \Big|_{\Gamma} - \int_{\Omega} k \frac{dw}{dx} \frac{du}{dx} d\Omega - \int_{\Omega} w f d\Omega = 0 \quad (73)$$

A seguir serão obtidas soluções aproximadas para a equação diferencial em questão através da substituição de aproximações para as variáveis $w(x)$ e $u(x)$ que chamaremos aqui de $w^e(x)$ e $u^e(x)$ respectivamente. Considerando Ω o domínio do problema e Ω^e o domínio do elemento, então:

$$u^e(x) = \sum_{n=1}^{np} u_j N_j^e(x) \quad (74)$$

$$w^e(x) = \sum_{n=1}^{np} w_i N_i^e(x) \quad (75)$$

sendo a malha formada por ne elementos que podem ter dimensões de comprimento diferentes e np pontos. u_j e w_i são constantes a serem determinadas e $N_j^e(x)$ e $N_i^e(x)$ são as funções de forma (ou de interpolação) que podem ser diferentes ou iguais. No método de Galerkin, as funções de forma para u e w são iguais, ou seja, $N_j^e(x) = N_i^e(x)$. Na Fig. (16), dois tipos de funções de forma são apresentados para discretização das equações. A escolha destas funções resultará em diferentes aproximações. O estudante poderá achar que a escolha de uma função de mais alta ordem sempre será melhor do ponto de vista de aproximação. No entanto, o uso de elementos de alta ordem podem trazer instabilidade numérica e aumento significativo do uso de recursos computacionais.

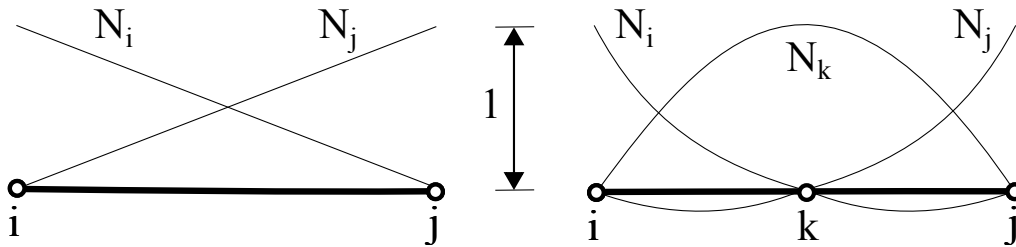


Figura 16: Representação de elemento linear e elemento quadrático

A diferença entre os dois elementos está no grau do polinômio interpolador. Para o elemento linear, a função interpoladora apresenta grau 1 enquanto que, no elemento quadrático, a função interpoladora é de grau 2. Elementos de baixa ordem apresentam custo computacional baixo comparado aos de mais alta ordem e são utilizados sempre que possível, pois garantem boas aproximações dependendo do tamanho do domínio do elemento. Com isso, uma boa escolha de elementos deve ser realizada para solução eficiente do problema. Aplicando-se o método de Galerkin na forma fraca da equação de calor unidimensional, chega-se a:

$$\sum_e \int_{\Omega^e} \sum_{i,j \in e} \alpha \frac{dN_i^e}{dx} \frac{dN_j^e}{dx} u_j d\Omega = \sum_e \int_{\Omega^e} \sum_{i \in e} N_i^e f d\Omega + c.c. \quad (76)$$

arrumando termos com somatório dentro da integral, a equação define um sistema linear do tipo

$$\mathbf{K}u = \mathbf{f} + c.c. \quad (77)$$

onde u tem como componentes os valores nodais u_n , a matriz \mathbf{K} e o vetor \mathbf{f} representados por uma montagem especial conhecida com *Assembling*:

$$\mathbf{K} = \sum_{e=1}^{nele} \mathcal{A} k^e; \quad \mathbf{f} = \sum_{e=1}^{nele} \mathcal{A} f^e \quad (78)$$

e \mathcal{A} é o operador montagem. A matriz k^e e o vetor f^e são definidos por

$$k_{i,j} = \int_{\Omega^e} \alpha \frac{dN_i^e}{dx} \frac{dN_j^e}{dx} d\Omega \quad i, j = 2 \quad (79)$$

$$f_i = \int_{\Omega^e} N_i^e f d\Omega \quad i = 2 \quad (80)$$

As integrais acima podem ser resolvidas analiticamente ou através de métodos numéricos. Em elementos finitos, o método numérico mais popular é a quadratura gaussiana, porém, para o caso unidimensional, o resultado analítico dessas integrais é simples. Dependendo do tipo de função de forma escolhido, o erro de aproximação diminui. Para este caso, considera-se o elemento linear representado por:

$$N_i = 1 - \frac{x - x_i}{h} \quad (81)$$

$$N_j = \frac{x - x_j}{h} \quad (82)$$

onde h representa o tamanho característico do elemento, obtem-se:

$$k^e = \frac{\alpha}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \quad \text{e} \quad f^e = h \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Realizando o procedimento de *Assembling* chega-se ao sistema linear (Eq. 77):

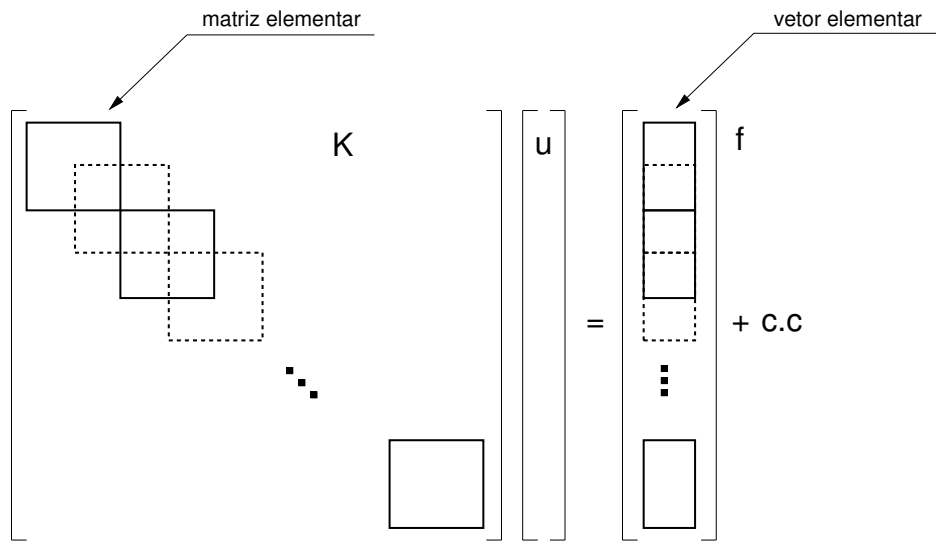


Figura 17: Sistema linear resultante da discretização pelo método de elementos finitos

e a solução do sistema linear fornece os valores da incógnita u nos nós dos elementos da malha.

A utilização do método de elementos finitos vem se tornando constante. A reutilização de código na forma de módulos e bibliotecas é uma grande aliada no desenvolvimento de grandes projetos, uma vez definida a base do

código (montagem de matrizes e vetores, solução do sistema linear), uma mudança na geometria e nas condições de contorno do problema altera o tipo de problema físico estudado, representando aproximadamente 10% de modificações no código numérico. Uma melhor discussão do método de elementos finitos e a utilização de alguns métodos poderão ser encontrados nas seguintes referências ([12], [16] e [25]).

O procedimento para o uso do MEF em equações diferenciais é:

1. encontrar forma fraca do problema através da ponderação da equação diferencial pela função peso w ;
2. reduzir derivadas de alta ordem através da integração por partes;
3. substituir as variáveis contínuas $u(x)$ e $w(x)$ pelas variáveis discretas $u^e(x)$ e $w^e(x)$ respectivamente;
4. arrumar os termos com somatórios e reescrevendo a equação na forma diferencial;
5. resolver o sistema linear do tipo $Ax = b$, encontrando a solução x nos nós da malha.

4.2.2 Solução de problema térmico permanente 1D

Neste problema deseja-se calcular a distribuição de temperatura unidimensional em regime permanente em uma barra com dimensão $L = 1$, dividida por 4 elementos lineares uniformemente espaçados e temperaturas constantes $T(x = 0) = 0$ e $T(x = L) = 1$ nas extremidades da barra através do método de elementos finitos.

A equação de modela este fenômeno físico é descrita por:

$$\frac{d^2 T}{dx^2} = 0 \quad \text{em} \quad \Omega \quad (83)$$

$$T = 0 \quad \text{em} \quad x = 0 \quad (84)$$

$$T = 1 \quad \text{em} \quad x = L \quad (85)$$

onde T é o campo de interesse do problema, x é a variável independente espacial. As condições de contorno são do tipo Dirichlet, ou seja, as temperaturas estão definidas nas extremidades da barra:

A forma fraca é resultado da ponderação da equação original em sua forma forte (forma diferencial inicial, Eq. (85)) no domínio do problema $\Omega = [0, 1]$:

$$\int_{\Omega} w \left(\frac{d^2 T}{dx^2} \right) d\Omega = 0 \quad (86)$$

O termo de derivada mais alta deve passar pelo processo de integração por partes para separação das parcelas do domínio e do contorno e para redução da ordem do operador:

Achar $T \in H_{\Gamma}^1$ tal que, para qualquer $W \in H_0^1$

$$w \frac{dT}{dx} \Big|_{\Gamma} - \int_{\Omega} \frac{dw}{dx} \frac{du}{dx} d\Omega = 0 \quad (87)$$

Usando o método de Galerkin serão obtidas soluções aproximadas para a equação diferencial em questão. Considerando Ω o domínio do problema e Ω^e o domínio do elemento, então:

$$T^e(x) = \sum_{n=1}^5 T_j N_j^e(x) \quad (88)$$

$$w^e(x) = \sum_{n=1}^5 w_i N_i^e(x) \quad (89)$$

sendo a malha formada por $ne = 4$ elementos e $np = 5$ pontos.

$$w \frac{dT}{dx} \Big|_0^1 - \sum_e \int_0^1 \sum_{i,j \in e} \frac{dN_i^e}{dx} \frac{dN_j^e}{dx} T_j d\Omega = 0 \quad (90)$$

O primeiro termo da equação representa o termo de contorno, que pode ser expandido e reescrito da seguinte forma:

$$w \frac{dT(1)}{dx} - w \frac{dT(0)}{dx} - \sum_0^1 \int_{\Omega^e} \sum_{i,j \in e} \frac{dN_i^e}{dx} \frac{dN_j^e}{dx} T_j d\Omega = 0 \quad (91)$$

Como as duas condições de contorno são do tipo *Dirichlet*, a escolha das funções de forma impõe que w seja nulo, com isso os dois termos de contorno se anulam. Reescrevendo a equação:

$$\sum_e \int_0^1 \sum_{i,j \in e} \frac{dN_i^e}{dx} \frac{dN_j^e}{dx} T_j d\Omega = 0 \quad (92)$$

Esta equação pode ainda ser escrita na forma matricial:

$$K_{i,j} T_j = 0 \quad (93)$$

5 Escoamento multifásico

5.1 Introdução

Escoamentos multifásicos são encontrados abundantemente na natureza. Ascensão de bolhas de ar em um copo de água gaseificada, o movimento das ondas do mar e o deslocamento de dunas de areia são alguns exemplos típicos deste complexo e intrigante fenômeno. Em dutos de petróleo também encontra-se escoamentos com mais de uma fase. Nestes dutos verificamos a presença de um líquido que se movimenta continuamente e carrega consigo partículas de sólidos dispersas, além de bolhas que se quebram e se aglomeram ao serem transportadas. Notamos ainda que o sangue contém partículas que são transportadas para promoção da nutrição de órgãos do nosso corpo.

Escoamentos com partículas são caracterizados por uma fase contínua, que pode ser líquida ou gasosa, e uma fase dispersa. A fase contínua é representada pela descrição do campo de velocidades e pressão através das equações de quantidade de movimento e continuidade utilizando-se a hipótese do contínuo ([referência para a hipótese - livro](#)). A fase dispersa, por sua vez, pode ser descrita através das equações da mecânica de massas pontuais, caracterizada pela quantidade de partículas presentes no escoamento, bem como pela dimensão e massa das mesmas. Dependendo destes fatores, e de alguns outros, o padrão do escoamento muda.

Nas seções seguintes sugerem-se exercícios para a modelagem simplificada de escoamentos multifásicos envolvendo uma fase contínua (água, ar e azeite) e uma fase dispersa, representado por uma partícula solitária ou por múltiplas partículas.

5.2 Escoamento com partícula - 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} . O perfil de velocidades é uma função unidimensional, pois só varia na direção y , entretanto as partículas estão localizadas no plano $x - y$. Diversas condições de contorno podem ser adotadas para as partículas, como condições de contorno de não deslizamento, elásticas e periódicas. Neste problema, deseja-se modelar o efeito de paredes elásticas para $y = 0$ e $y = y_{max}$, onde o efeito de elasticidade é completo, ou seja, a partícula rebate na parede com a mesma velocidade de choque. Para os contornos em $x = 0$ e $x = x_{max}$, modelaremos um canal infinito, onde a condição de periodicidade deve ser imposta em $x = x_{max}$ para repetir a trajetória da partícula no início do canal quando a coordenada x da partícula apresentar valor $x > x_{max}$. Note que a fase contínua (fluido) poderá ser modelada conforme Tabela (13).

A equação vetorial de movimento da partícula no plano $x - y$ se escreve:

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{lift} + \mathbf{F}_g \quad (94)$$

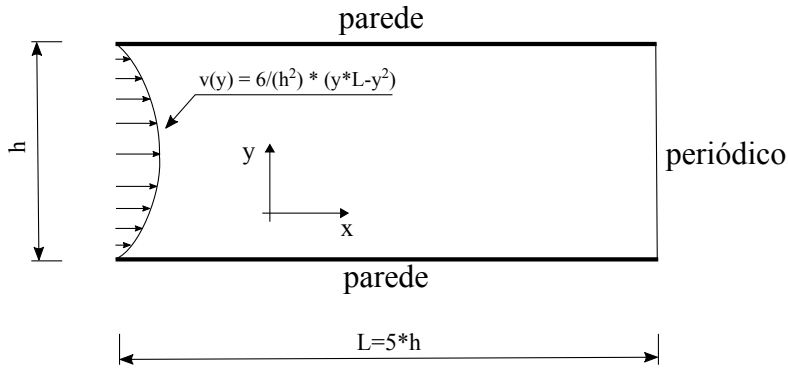


Figura 18: Domínio numérico e condições de contorno para partículas e perfil de velocidade.

onde:

$$\mathbf{F}_{\text{drag}} = 3\pi\mu d_p v \quad (95)$$

$$\mathbf{F}_{\text{lift}} = 1.61\sqrt{\mu\rho_d} dp^2 |u - v| \frac{du}{dy} \sqrt{\left| \frac{du}{dy} \right|} \quad (96)$$

$$\mathbf{F}_g = mg \quad (97)$$

Note que esta equação vetorial tem solução analítica, não havendo necessidade de solução numérica, 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 não deve ser calculado neste exemplo, tomando a forma:

$$v(y) = \frac{6}{L^2}(yL - y^2) \quad (98)$$

As condições de contorno deste problema pode ser visualizadas na Fig. (18):

Dados da simulação:

fluido (fase contínua)

Tabela 12: 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)

condição inicial

- $v_x = 0.3 [m]/[s]$ velocidade
- $v_y = -0.1 [m]/[s]$ velocidade
- $x = 1.7 [m]$ posição
- $y = 0.3 [m]$ posição
- $t = 0.0 [s]$ tempo

Tabela 13: 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	$d_p = 1.0e - 3$	$\rho = 2700.0$

- $g = 9.81$ $[m]/[s^2]$ gravidade

passo de tempo

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

5.3 Escoamento com múltiplas partículas

5.3.1 Acoplamento *One-way* sem choque entre partículas

Neste exemplo, deseja-se conhecer a trajetória de múltiplas partículas dispersas em um meio contínuo como no exercício anterior. Estas partículas estão em um canal sob efeito de um perfil de velocidades não conhecido e submetida às forças de gravidade F_g , de arrasto F_{drag} e de sustentação F_{lift} .

O perfil de velocidades é uma função unidimensional, pois só varia na direção y , entretanto as partículas estão localizadas no plano $x - y$. As condições de contorno deste problema são idênticas às condições de contorno do problema anterior, ou seja, de paredes elásticas para $y = 0$ e $y = y_{max}$ e condição de contorno periódica para $x = 0$ e $x = x_{max}$.

A equação vetorial de movimento da uma partícula no plano $x - y$ se escreve:

$$m \frac{d\mathbf{v}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{lift} + \mathbf{F}_g \quad (99)$$

onde:

$$\mathbf{F}_{drag} = 3\pi\mu d_p v \quad (100)$$

$$\mathbf{F}_{lift} = 1.61\sqrt{\mu\rho_d} dp^2 |u - v| \frac{du}{dy} \sqrt{\left| \frac{du}{dy} \right|} \quad (101)$$

$$\mathbf{F}_g = mg \quad (102)$$

Para a representação de múltiplas partículas, pode-se usar estruturas numéricas que possibilitem a descrição de partículas simultaneamente. O uso de vetores numéricos é uma opção para a realização de operações básicas como soma e multiplicação. Com isso, deve-se alocar espaço para um vetor com um número n de itens, de modo que n represente o número total de partículas do problema. Tendo em vista as equações do movimento da partícula e das forças, vetores de posição x e y e de velocidade v_x e v_y devem ser criados. Na tabela abaixo são sugeridos alguns valores iniciais para o problema de múltiplas partículas. Note que os vetores de posição x e y são iniciados através de uma função aleatória de posição, onde os valores entre parênteses representam os limites da aleatoriedade. O mesmo se aplica para os vetores de velocidade v_x e v_y , garantindo assim que as partículas não sejam inicializadas com os mesmos valores de posição e velocidade.

condição inicial

- $numParticles = 1$ número total de partículas
- $v_x = np.random.uniform(-0.1, 0.1, (numParticles, 1))$

- $v_y = \text{np.random.uniform}(-0.1, 0.1, (\text{numParticles}, 1))$
- $x = \text{np.random.uniform}(2.0, 1.0, (\text{numParticles}, 1))$
- $y = \text{np.random.uniform}(0.8h, 0.2h, (\text{numParticles}, 1))$
- $time = 0.0$ [s] tempo
- $g = 9.81$ [m]/[s²] gravidade

passo de tempo

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

Referências

- [1] G.R. Anjos. *A 3D ALE Finite Element Method for Two-Phase Flows with Phase Change*. PhD thesis, École Polytechnique Fédérale de Lausanne, July 2012.
- [2] G. Booch. *Object Oriented Analysis and Design with Applications*. Addison-Wesley, 2a. edition, 1993.
- [3] T.J. Chung. *Finite Element Analysis in Fluid Dynamics*. McGraw-Hill, 1st. edition, 1978.
- [4] M. Desbrun, M. Meyer, P. Schröder, and A. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of Siggraph*, pages 317–324, 1999.
- [5] J. Donea. A taylor-galerkin method for convective transport problems. *Int. J. Num. Methods Eng.*, 20: 101–19, 1984.
- [6] J. Fish and T. Belytschko. *A First Course in Finite Elements*. John Wiley & Sons, Ltd, 1 edition, 2007.
- [7] A.O. Fortuna. *Técnicas Computacionais para Dinâmica dos Fluidos*. edUSP, 1a. edition, 2000.
- [8] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11): 1309–1331, 2009.
- [9] M.T Heath. *Scientific Computing - An Introductory Survey*. McGraw Hill, 2a. edition, 2002.
- [10] J.C. Heinrich, P.S. Huyakorn, O.C. Zienkiewicz, and A.R. Mitchell. An upwind finite element scheme for two-dimensional convective transport equation. *Int. J. Num. Meth. Eng.*, 1:131–44, 1977.
- [11] T. Hughes, L. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: I. symmetric forms of the compressible euler and navier-stokes equation and the second law of thermodynamics. *Computer Methodos in Applied Mechanics and Engineering*, 54:223–34, 1986.
- [12] T.J.R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Civil and Mechanical Engineering. Dover Publications, New York, 1987. ISBN 9780486135021.
- [13] T.J.R. Hughes and A.N. Brooks. *Finite Elements in Fluids - A Theoretical Framework for Petrov-Galerkin Methods with Discontinuous Weighting Functions: Application to the Streamline Upwind Procedure*, volume 4. John Wiley & Sons, New York, 1967. Selected papers from the Third International Conference on Finite Elements in Flow Problems, Canada, June 10-13, 1980.
- [14] C. Johnson. *Numerical Solution of Partial Differential Equations on the Finite Element Method*. Lund - Sweden, 1st. edition, 1987.
- [15] E. Kreyszig. *Differential Geometry*. Dover, New York, 1a. edition, 1991.
- [16] R. W. Lewis, P. Nithiarasu, and K. N. Seetharamu. *Fundamentals of the Finite Element Method for Heat and Fluid Flow*. Wiley John and Sons, 2004.
- [17] R. Löhner, K. Morgan, and O.C. Zienkiewicz. An adaptive finite element procedure for compressible high speed flows. *Computer Methodos in Applied Mechanics and Engineering*, 51:441–65, 1985.

- [18] J.T. Oden and L.C.Jr. Wellford. Analysis of viscous flow by the finite element method. *AIAA J*, 10:1590–9, 1972.
- [19] J.T. Oden, L. Demkowicz, W. Rachowicz, and T.A. Westerman. Toward a universal h-p adaptive finite element strategy, part 2: A posterior error estimation. *Computer Methods in Applied Mechanics and Engineering*, 77:113–80, 1989.
- [20] J.T. Oden, I. Babuska, and C.E. Baumann. A discontinuous hp finite element method for diffusion problems. *Journal of Computational Physics.*, 146:491–519, 1998.
- [21] O. Pironneau. *Finite Element Methods for Flows*. Wiley, 1st. edition, 1989.
- [22] F.S Souza and N. Mangiavacchi. A lagrangian level-set approach for the simulation of incompressible two-fluid flows. *International Journal for Numerical Methods in Fluids*, 47:1393–1401, 2004.
- [23] John R. Taylor. *Mecânica Clássica*. University Science Book, 2013.
- [24] O.C. Zienkiewicz and Y.K. Cheung. *Finite elements in the solution of field problems*. The Engineer, 507-10 edition, 1965.
- [25] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method: Its Basis and Fundamentals*. Butterworth-Heinemann, Oxford, 5th. edition, 2000. ISBN 9780080472775.