ICMC-USP

Notas de Aula
SME0305, SME0306 & SME0602

Métodos Numéricos e Computacionais

com Aplicações para Engenharia e Progamação em `python`

Roberto F. Ausas

April 10, 2024

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo

Prefacio

Este documento está organizado por capítulos, cada um dos quais apresentando os principais métodos do cálculo numérico e computacional para os cursos de engenharia. Para cada tema, se apresentam os principais conceitos teóricos por trás dos métodos numéricos, algumas aplicações de interesse para a engenharia, a sua implementação computacional e, finalmente, problemas e atividades que o aluno precisará desenvolver, com as quais, este será avaliado ao longo do curso.

Que tópicos este curso apresenta

Nas disciplinas de cálculo numérico para engenharia são estudados os seguintes temas, que temos dividido em capítulos (a depender do curso específico, alguns tópicos poderão não estar inclusos):

Capítulo |01| Introdução à programação em `python`;
Capítulo |02| Matrizes e Redes;
Capítulo |03| Sistemas Sobredeterminados e Analise de Dados (sme0305);
Capítulo |04| Aproximação de Funções (sme0306 e sme0602);
Capítulo |05| Aproximação de Autovalores e Autovetores;
Capítulo |06| Resolução Numérica de EDOs (sme0306 e sme0602);

RFA

Contents

Prefacio ii

Contents iii

1 PROGRAMAÇÃO EM `python` 2
- 1.1 Preludio . . . 2
- 1.2 Lista 1 de exercícios . . . 3

2 CÁLCULO DE REDES EM `python` 6
- 2.1 Preludio . . . 6
  - 2.1.1 Exemplo de resolução de uma rede hidráulica . . . 6
  - 2.1.2 Conexão da rede . . . 8
- 2.2 Redes em `python` . . . 13
- 2.3 Lista 2 de exercícios . . . 16
- 2.4 Cálculos Monte Carlo em Redes . . . 17
- 2.5 Atividade 1 . . . 19
  - 2.5.1 Exercícios opcionais . . . 20

Alphabetical Index 23

setchapterpreamble[u]

PROGRAMAÇÃO EM `python` 1

1.1 Preludio

Nas primeiras aulas precisamos introduzir a principal ferramenta de trabalho que será utilizada nas disciplinas de Métodos Numéricos e Computacionais. Depois de tudo, a ideia do cálculo numérico é resolver problemas complexos de Engenharia, que não poderiam ser resolvidos manualmente, sem o auxilio de um computador.

De maneira muito geral, as linguagens de programação, podem ser divididas em duas categorias:

- **Linguagens compiladas:** Dentre as primeiras temos linguagens tais como `C`, `C++` e `Fortran`. A escrita de código neste tipo de linguagens de programação requer um domínio maior da sintaxe e das funcionalidades da linguagen. Como contrapartida, este tipo de linguagens produzem códigos que são muito rápidos pois tem sido optimizadas ao longo dos anos, e por tanto são usadas amplamente na Engenharia. De fato, a maioria dos códigos de cálculo usados na indústria (*Open Source* ou comerciais) estão feitos com elas. Ao **compilar** o código e gerar um arquivo binário optimizado, é possivel tirar o maior proveito do poder de processamento de um computador.

- **Linguagens interpretadas:** Por outra parte, as linguagens interpretadas, como `python` e `Matlab`, são relativamente simples e intuitivas, pela sua flexibilidade na sintaxe, tornando o processo de desenvolvimento mais rápido e ágil. De maneira grosseira, o código vai sendo executado a medida que se **interpreta**. Porém, se não são tomados alguns recaudos no desenho do código, a performance computacional delas pode estar bastante aquém do necessário para resolver problemas de grande porte. Um exemplo prototípico disto, é quando utilizamos uma estrutura de repetição (como um `for`) no qual realizamos um grande número de operações em cada passo. Ao longo de curso iremos chamando a atenção sobre isto, tentando introduzir boas práticas de programação.

Como comparação, vejamos o exemplo de um código simples para criar um array com números de ponto flutuante de dupla precisão, popular ele com os números $0, \ldots, N$ e imprimir o resultado na tela, usando a linguagem compilada `C` (acima) e a linguagem interpretada `python` (embaixo).

```c
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int i, N = 10;
  double *array;
  array=(double *) malloc(N*sizeof(double));
  for(i=0; i < N; i++) {
       array[i] = (double) i;
       printf("%lf\n", array[i]);
     }
  free(array);
  return 0;
}
```

```python
import numpy as np
N = 10
array = np.arange(N)
print(array)
```

Olhando para o exemplo, já vemos que uma linguagem interpretada como `python` se torna mais prática para um curso de cálculo numérico, pois o objetivo é entender os métodos numéricos, os algoritmos e como resolver problemas práticos no computador, sem gastar muito tempo no desenvolvimento de código.

Neste curso iremos adotar a linguagem `python`, a qual tem-se tornado bastante popular nos últimos anos. Para facilitar a implementação dos diferentes métodos numéricos, contamos com algumas bibliotecas específicas que facilitarão o trabalho:

- `numpy`: Para manipulação eficiente de matrizes e vetores, operações de algebra linear computacional e vários métodos numéricos.
- `scipy`: Para métodos numéricos mais avançados ou específicos, não cobertos pela anterior.
- `matplotlib`: Para plotagens e geração de gráficos em 2D e 3D.

Aviso importante

O material de estudo para este capítulo serão os `jupyter notebooks` desenvolvidos pelo professor na sala de aula, que foram disponibilizadas no repositório da disciplina. Outras fontes de informação a ser consideradas são os sites das bibliotecas que iremos utilizar, tais como `https://numpy.org` e `https://matplotlib.org/`, assim como consultas dirigidas ao professor em forma presencial ou por e-mail (`rfausas@icmc.usp.br`).

1.2 Lista 1 de exercícios

A lista de exercícios a ser apresentada em avaliação oral por sorteio é dada na sequência:

Exercícios

1. Fazer em python uma função que:

   - Pega dois vetores randômicos **a** e **b** de dimensão n, e dois escalares randômicos α e β e calcula um vetor **c** tal que

     $$\mathbf{c} = \alpha\mathbf{a} + \beta\mathbf{b}$$

   - Pega uma matriz randômica **A** de $n \times n$ e calcula a sua m-essima potência

     $$\mathbf{A}^m = \underbrace{\mathbf{A} \ldots \mathbf{A}}_{m \text{ vezes}}$$

     (tomar valores de $m = 2, 3, 4$).

   Em todos os casos medir o tempo necessário para realizar as operações para diferentes dimensões n. Plotar o tempo de cálculo como função da dimensão n usando a escala linear padrão e a escala `loglog`. No segundo ponto, colocar no mesmo gráfico os resultados para os diferentes valores de m. Tirar conclusões.

   Nota: Para que os resultados sejam interessantes, no primeiro ponto, tomar valores de $n = 10^5,\ 10^6, \ldots, 10^9$. Já, no segundo ponto, tomar valores de $n = 1000, 2000, 3000, 4000$.

2. Considerar uma sequência de numéros gerada da seguinte forma:

   $$x_n = a\, x_{n-1}\,(1 - x_{n-1}), \quad n = 1, 2, \ldots, N$$

   Considerar $N = 5000$, $x_0 = 0.1$ e diferentes valores de a entre 0 e 4 (p.e., $a = 1,\ 2,\ 3.8$ e 4). Calcular a média e a variança da sequência:

   $$\bar{x} = \frac{1}{N}\sum_{i=0}^{N} x_i, \quad \sigma = \frac{1}{N-1}\sum_{i=0}^{N}(x_i - \bar{x})^2$$

   Programar-lo na mão e usando funções de `numpy` já prontas Comparar os resultados.

3. No problema 2, plotar a sequéncia de valores obtida em cada caso considerando os diferentes valores de a pedidos. Fazer os gráficos usando legendas, labels, e outros atributos que achar interessante, para melhor ilustrar os resultados.

4. Continuando com a sequência do problema 3, fazer um código que gera o diagrama de bifurcações, que é um gráfico que mostra os valores que assume a sequência, como função dos valores de a. O resultado deveria ser algo do tipo mostrado na figura ao lado, que no eixo horizontal tem os valores de a usados para gerar a sequência e no eixo vertical todos os possíveis valores que toma a sequência para o correspondente valor de a. Se sugere usar pontinhos bem pequenos para gerar o gráfico. Explicar os resultados se auxiliando com os gráficos

do exercício anterior.

5. Considerar uma rede de distribuição com a mostrada na figura ao lado. Esta pode ser um exemplo de uma rede elétrica ou hidráulica e é basicamente o que se chama um *grafo*). Notar que em geral ela estará caracterizada por um certo número de *nós* (ou uniões), um certo nuúmero de *arestas* e alguma informação sobre a conectividade entre pontos.
   - Fazer uma estrutura de dados que sirva para descrever essa rede. Idealmente, a estrutura deveria incluir algum tipo de matriz ou *array* que indica como os nós e as arestas estão relacionados. Adicionalmente, a estrutura deve conter um array para descrever as coordenadas (x, y) de cada nó. Construir um exemplo e plotar a rede.
   - Fazer uma função que deleta um nó e todas as arestas que emanam dele.
   - Fazer uma outra função em `python` que permita apagar (ou *deletar*) uma aresta da rede. Notar que se a aresta possui um nó que náo pertence a nenhuma outra aresta, esse nó também precisa ser deletado.
   - Finalmente, fazer uma função que insere uma nova aresta na rede para conectar dois pontos já existentes.

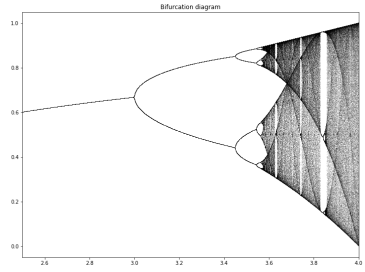

Figure 1.1: Diagrama de bifurcações.

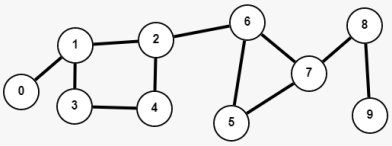

Figure 1.2: Exemplo de uma rede.

setchapterpreamble[u]

CÁLCULO DE REDES EM python 2

2.1 Preludio

No presente capítulo iremos mostrar como se formula o problema de distribuição numa rede elétrica ou hidráulica, ou em geral, qualquer quantidade ou informação que se propaga por uma rede de maneira conservativa. De maneira geral temos os seguintes:

Objetivos

- Entender a modelagem de distintos sistemas físicos tais como redes hidráulicas, elétricas ou inclusive treliças, do ponto de vista da Algebra Linear;
- Aprender como resolver tais problemas de maneira sistemática no computador;

Há muitos sistemas físicos que podem ser modelizados como uma rede na qual se propaga uma certa quantidade física. Na tabela da sequência mostramos três exemplos clássicos que aparecem na engenharia:

Sistema	Nó	Variável nodal	Aresta	Variável de aresta	Lei de conservação
Elétrico	Conexão	Voltagem	Resistencia	Corrente	Carga elétrica
Hidráulico	União	Pressão	Cano	Vazão	Massa
Estrutural	Articulação	Deslocamento	Barra	Tensão	Momento

2.1.1 Exemplo de resolução de uma rede hidráulica

Nesta parte vamos trabalhar com redes hidráulicas. Para isto, vamos considerar o exemplo de uma rede simples mostrado na figura. Posteriormente, conseguiremos generalizar isto para resolver qualquer rede.

Para o exemplo particular temos: $n_c = 7$ canos e $n_v = 5$ nós:

O objetivo é determinar:

- As pressões em cada nó: $\mathbf{p} = [p_1 \dots p_5]^\intercal$
- As vazões em cada cano: $\mathbf{Q} = [Q_1 \dots Q_7]^\intercal$.

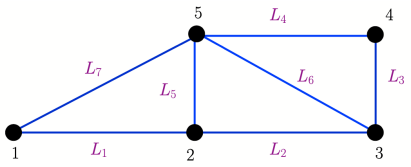

Figure 2.1: Exemplo de uma rede simples.

Conceitos necessários:

- Comportamento de um cano → **Lei constitutiva**
- Condições de acoplamento → **Conservação de massa e Continuidade**

Lei constitutiva: Comportamento de um cano

Para cada cano, $k = 1, \ldots, n_c$, que conecta os nós i e j, vamos considerar a lei de perda de pressão em função da vazão:

$$\Delta p_k = p_i^k - p_j^k = \rho_k \, L_k \, Q_k$$

$$\Rightarrow \quad Q_k = \frac{1}{\rho_k \, L_k} \Delta p_k \; = \; C_k \, (p_i^k - p_j^k)$$

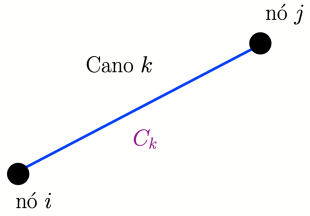

Figure 2.2: Um só cano.

- $C_k = \frac{1}{\rho_k \, L_k}$
- ρ_k é uma constante de resistência
- L_k é o comprimento de cada cano.
- A vazão é positiva quando vai de i a j.

Condições de acoplamento

- **Unicidade da pressão**: Para qualquer nó $i = 1, \ldots, n_v$,

  $$p_i^k = p_i$$

  para todo nó k que possui o nó i.

- **Conservação da massa**:

  $$\sum_{\forall \, k \text{ que possui o nó } i} Q_k{}^{(i)} \; = \; 0, \quad i = 1, 2, \ldots, n_v$$

Desta forma, teremos uma equação por cada nó da rede. Para o caso particular do exemplo:

$$\begin{aligned}
\text{Nó 1} &: \sum_k Q_k^{(1)} = C_1(p_1 - p_2) + C_7(p_1 - p_5) = 0 \\
\text{Nó 2} &: \sum_k Q_k^{(2)} = C_1(p_2 - p_1) + C_5(p_2 - p_5) + C_2(p_2 - p_3) = 0 \\
\text{Nó 3} &: \sum_k Q_k^{(3)} = C_2(p_3 - p_2) + C_6(p_3 - p_5) + C_3(p_3 - p_4) = 0 \\
\text{Nó 4} &: \sum_k Q_k^{(4)} = C_3(p_4 - p_3) + C_4(p_4 - p_5) = 0 \\
\text{Nó 5} &: \sum_k Q_k^{(5)} = C_7(p_5 - p_1) + C_5(p_5 - p_2) + C_6(p_5 - p_3) + C_4(p_5 - p_4) = 0
\end{aligned}$$

i.e., **5 equações e 5 incógnitas**. Agora vamos ordenar o sistema:

$$
\begin{aligned}
(C_1 + C_7)\,p_1 &- C_1\,p_2 &+ 0\,p_3 &+ 0\,p_4 &- C_7\,p_5 &= 0\\
-C_1\,p_1 &+ (C_1 + C_2 + C_5)\,p_2 &- C_2\,p_3 &+ 0\,p_4 &- C_5\,p_5 &= 0\\
0\,p_1 &- C_2\,p_2 &+ (C_2 + C_3 + C_6)\,p_3 &- C_3\,p_4 &- C_6\,p_5 &= 0\\
0\,p_1 &+ 0\,p_2 &- C_3\,p_3 &+ (C_3 + C_4)\,p_4 &- C_4\,p_5 &= 0\\
-C_7\,p_1 &- C_5\,p_2 &- C_6\,p_3 &- C_4\,p_4 &+ (C_4 + C_5 + C_6 + C_7)\,p_5 &= 0
\end{aligned}
$$

ou em forma matricial:

$$
\begin{pmatrix}
C_1 + C_7 & -C_1 & 0 & 0 & -C_7\\
-C_1 & C_1 + C_2 + C_5 & -C_2 & 0 & -C_5\\
0 & -C_2 & C_2 + C_3 + C_6 & -C_3 & -C_6\\
0 & 0 & -C_3 & C_3 + C_4 & -C_4\\
-C_7 & -C_5 & -C_6 & -C_4 & C_4 + C_5 + C_6 + C_7
\end{pmatrix}
\cdot
\begin{pmatrix} p_1\\ p_2\\ p_3\\ p_4\\ p_5 \end{pmatrix}
=
\begin{pmatrix} 0\\ 0\\ 0\\ 0\\ 0 \end{pmatrix}
$$

De forma geral, a matriz da rede descreve-se como

$$
A_{ij} = \begin{cases}
\sum\limits_{k \text{ conectados com } i} C_k^{(i)} & \text{se } i = j\\
-C_r & \text{se } i \neq j, \text{ com } i \text{ conectado a } j \text{ por } C_r\\
0 & \text{se não há conexão entre } i \text{ e } j
\end{cases}
$$

- A matriz é simétrica
- A soma dos elementos de qualquer linha/coluna é zero
- A matriz **não é invertível** → **A pressão está indeterminada**. Isto significa que não podemos resolver o sistema de equações até não eliminar essa indeterminação.
- Em geral, um nó estará conectado a um número relativamente pequeno de nós. Isto fará com que a matriz A possua muitos zeros

2.1.2 Conexão da rede

Agora, precisamos conectar a rede para resolver um problema que tenha algum sentido físico.

- Para remover a indeterminação se fixa o valor da pressão em algum ponto. Para o exemplo, vamos conectar o nó 3 à atmosfera, do qual podemos deduzir que:

$$p_3 = 0$$

Porém, agora teremos uma vazão Q_R para determinar.

- Também, vamos injetar uma vazão Q_B no nó 1, conectando uma bomba, e assim resolver um problema que tenha uma solução não trivial.

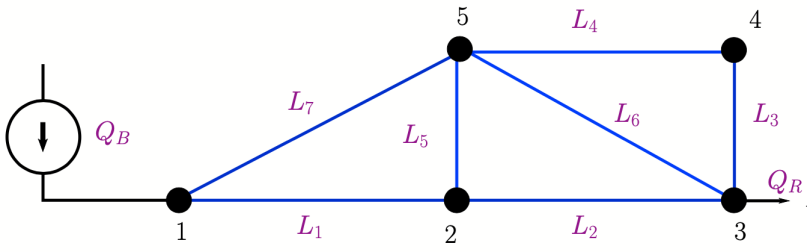

Figure 2.3: Rede conectada a uma bomba e descarregando na atmosfera.

Conexão da rede no nível algébrico

Precisamos incorporar no nível do sistema de equações a informação sobre a conexão da rede que acabamos de mencionar:

- Se estamos injetando no nó 1, então, a sua equação agora é:

$$\sum Q_r^{(1)} = (C_1 + C_7)\, p_1 - C_1\, p_2 + 0\, p_3 + 0\, p_4 - C_7\, p_5 = Q_B$$

- Similarmente, a equação para o nó 3 agora é:

$$\sum Q_r^{(3)} = 0\, p_1 - C_2\, p_2 + (C_2 + C_3 + C_6)\, p_3 - C_3\, p_4 - C_6\, p_5 = Q_R$$

em que Q_R é a **nova** incógnita. i.e., seria melhor colocar-la no lado esquerdo:

$$0\, p_1 - C_2\, p_2 + (C_2 + C_3 + C_6)\, p_3 - C_3\, p_4 - C_6\, p_5 - 1\, Q_R = 0$$

- mas, se já sabemos que $p_3 = 0$, então, temos uma equação adicional:

$$0\, p_1 + 0\, p_2 + 1\, p_3 + 0\, p_4 + 0\, p_5 = 0$$

No entanto, estamos interessados em achar as pressões, então, podemos **ignorar** a equação de balanço do nó 3 e ficar apenas com essa última equação, resultando o novo sistema de equações (ainda de 5×5):

$$\begin{pmatrix} C_1+C_7 & -C_1 & 0 & 0 & -C_7 \\ -C_1 & C_1+C_2+C_5 & -C_2 & 0 & -C_5 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -C_3 & C_3+C_4 & -C_4 \\ -C_7 & -C_5 & -C_6 & -C_4 & C_4+C_5+C_6+C_7 \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{pmatrix} = \begin{pmatrix} Q_B \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Esse sistema matriz vetor, sim pode ser resolvido.

De forma geral podemos descrever a matriz $\widetilde{A}$ e o vetor **b** do sistema como: Seja n_{atm} é o índice do nó em que a pressão é fixada ($p_{n_{atm}} = 0$) e n_B o índice do nó em que a bomba é conectada.

$$\widetilde{A}_{ij} = \begin{cases} A_{ij} & \text{se } i \neq n_{atm} \\ 0 & \text{se } i = n_{atm} \ e \ j \neq n_{atm} \\ 1 & \text{se } i = j = n_{atm} \end{cases}$$

$$b_i = \begin{cases} 0 & \text{se } i \neq n_{atm} \\ Q_B & \text{se } i = n_B \\ 0 & \text{se } i = n_{atm} \end{cases}$$

Ficando um sistema matriz-vetor do seguinte tipo:

$$\begin{pmatrix} A_{11} & A_{12} & \dots & A_{1n_{atm}} & \dots & A_{1n_B} & \dots & A_{1n_p} \\ A_{21} & A_{22} & \dots & A_{1n_{atm}} & \dots & A_{1n_B} & \dots & A_{2n_p} \\ . & . & \dots & . & \dots & . & \dots & . \\ . & . & \dots & . & \dots & . & \dots & . \\ . & . & \dots & . & \dots & . & \dots & . \\ 0 & 0 & \dots & 1 & \dots & 0 & \dots & 0 \\ . & . & \dots & . & \dots & . & \dots & . \\ . & . & \dots & . & \dots & . & \dots & . \\ . & . & \dots & . & \dots & . & \dots & . \\ A_{n_B1} & A_{n_B2} & \dots & A_{n_Bn_{atm}} & \dots & A_{n_Bn_B} & \dots & A_{n_Bn_p} \\ . & . & \dots & . & \dots & . & \dots & . \\ . & . & \dots & . & \dots & . & \dots & . \\ . & . & \dots & . & \dots & . & \dots & . \\ A_{n_p1} & A_{n_p2} & \dots & A_{n_pn_{atm}} & \dots & A_{n_pn_B} & \dots & A_{n_pn_p} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ . \\ . \\ . \\ p_{n_{atm}} \\ . \\ . \\ . \\ p_{n_B} \\ . \\ . \\ . \\ p_{n_p} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ . \\ . \\ . \\ 0 \\ . \\ . \\ . \\ Q_B \\ . \\ . \\ . \\ 0 \end{pmatrix}$$

Exemplo numérico

Para colocar números concretos, consideremos uma vazão $Q_B = 3\,m^3/s$ e as seguintes propriedades para os canos da rede: Inserindo estes valores

Cano	ρ_k [bar s / m^4]	L_k [m]	C_k [bar s / m^3]$^{-1}$
1	0.1	5	2
2	0.1	5	2
3	0.1	10	1
4	0.1	5	2
5	0.1	10	1
6	0.1	5	2
7	0.1	5	2

chegamos no sistema matriz-vetor (conferir):

$$\begin{pmatrix} 4 & -2 & 0 & 0 & -2 \\ -2 & 5 & -2 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -2 \\ -2 & -1 & -2 & -2 & 7 \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{pmatrix} = \begin{pmatrix} 3 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

E agora o que precisamos fazer é a resolução do sistema, o qual pode ser feito pelo chamado escalonamento da matriz:

$$\left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ -2 & 5 & -2 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -2 & 0 \\ -2 & -1 & -2 & -2 & 7 & 0 \end{array}\right) \xrightarrow{\ell_2 \leftarrow \ell_2 + \frac{1}{2}\ell_1} \left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ 0 & 4 & -2 & 0 & -2 & \frac{3}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -2 & 0 \\ -2 & -1 & -2 & -2 & 7 & 0 \end{array}\right) \xrightarrow{\ell_5 \leftarrow \ell_5 + \frac{1}{2}\ell_1}$$

$$\left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ 0 & 4 & -2 & 0 & -2 & \frac{3}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -2 & 0 \\ 0 & -2 & -2 & -2 & 6 & \frac{3}{2} \end{array}\right) \xrightarrow{\ell_5 \leftarrow \ell_5 + \frac{1}{2}\ell_2} \left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ 0 & 4 & -2 & 0 & -2 & \frac{3}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -2 & 0 \\ 0 & 0 & -3 & -2 & 5 & \frac{9}{4} \end{array}\right) \xrightarrow{\ell_4 \leftarrow \ell_4 + \ell_3}$$

$$\left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ 0 & 4 & -2 & 0 & -2 & \frac{3}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & -2 & 0 \\ 0 & 0 & -3 & -2 & 5 & \frac{9}{4} \end{array}\right) \xrightarrow{\ell_5 \leftarrow \ell_5 + 3\ell_3} \left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ 0 & 4 & -2 & 0 & -2 & \frac{3}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & -2 & 0 \\ 0 & 0 & 0 & -2 & 5 & \frac{9}{4} \end{array}\right) \xrightarrow{\ell_5 \leftarrow \ell_5 + \frac{2}{3}\ell_4}$$

$$\left(\begin{array}{ccccc|c} 4 & -2 & 0 & 0 & -2 & 3 \\ 0 & 4 & -2 & 0 & -2 & \frac{3}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & -2 & 0 \\ 0 & 0 & 0 & 0 & \frac{11}{3} & \frac{9}{4} \end{array}\right) \xrightarrow{\ell_i \leftarrow \ell_i / a_{ii}} \left(\begin{array}{ccccc|c} 1 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & \frac{3}{4} \\ 0 & 1 & -\frac{1}{2} & 0 & -\frac{1}{2} & \frac{3}{8} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -\frac{2}{3} & 0 \\ 0 & 0 & 0 & 0 & 1 & \frac{27}{44} \end{array}\right)$$

$$\begin{pmatrix} 1 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} \\ 0 & 1 & -\frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -\frac{2}{3} \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{pmatrix} = \begin{pmatrix} \frac{3}{4} \\ \frac{3}{8} \\ 0 \\ 0 \\ \frac{27}{44} \end{pmatrix}$$

o que finalmente resulta:

$$\begin{aligned} 1\,p_1 - \tfrac{1}{2}\,p_2 + 0\,p_3 + 0\,p_4 - \tfrac{1}{2}\,p_5 &= \tfrac{3}{4} \rightarrow p_1 = \tfrac{123}{88} \\ 1\,p_2 - \tfrac{1}{2}\,p_3 + 0\,p_4 - \tfrac{1}{2}\,p_5 &= \tfrac{3}{8} \rightarrow p_2 = \tfrac{60}{88} \\ + 1\,p_3 + 0\,p_4 + 0\,p_5 &= 0 \rightarrow p_3 = 0 \\ + 1\,p_4 - \tfrac{2}{3}\,p_5 &= 0 \rightarrow p_4 = \tfrac{54}{132} \\ 1\,p_5 &= \tfrac{27}{44} \rightarrow p_5 = \tfrac{27}{44} \end{aligned}$$

Verificação dos resultados

Lembremos que a equação para o nó 3 era:

$$\sum Q_r^{(3)} = 0\,p_1 - C_2\,p_2 + (C_2 + C_3 + C_6)\,p_3 - C_3\,p_4 - C_6\,p_5 = Q_R$$

ou, colocando os valores da tabela

$$\sum Q_r^{(3)} = 0 - 2\,p_2 + 5\,p_3 - 1\,p_4 - 2\,p_5 = Q_R$$

e inserindo as pressões que acabamos de achar:

$$Q_R = \sum Q_r^{(3)} = 0 - 2\,\tfrac{60}{88} + 0 - 1\,\tfrac{54}{132} - 2\,\tfrac{27}{44} = -3$$

ou seja, pelo nó 3 está saindo exatamente o que injectamos no nó 1.

2.2 Redes em python

A matriz de conectividades e montagem da matriz A

Uma forma "prática" de montar a matriz A, utiliza a chamada matriz de conectividades $\texttt{conec} \in \mathbb{N}^{n_c \times 2}$. Esta é uma estrutura muito conveniente para descrever a rede e é um ingrediente chave no método dos Elementos Finitos:

$$\texttt{conec} = \begin{pmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 4 \\ 4 & 5 \\ 5 & 2 \\ 5 & 3 \\ 5 & 1 \end{pmatrix}$$

Olhando para a matriz **global** da rede, a ideia é ir acumulando nela, a matriz **local** associada a cada cano. Para o cano k, de conductividade C_k, teremos uma matriz de 2×2, que chamaremos a matriz local do cano.

$$\mathbf{C}_{loc_k} = \begin{pmatrix} C_k & -C_k \\ -C_k & C_k \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\text{Cano 1}} \begin{pmatrix} C_1 & -C_1 & 0 & 0 & 0 \\ -C_1 & C_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\text{Cano 2}}$$

$$\begin{pmatrix} C_1 & -C_1 & 0 & 0 & 0 \\ -C_1 & C_1 + C_2 & -C_2 & 0 & 0 \\ 0 & -C_2 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\text{Cano 3}} \begin{pmatrix} C_1 & -C_1 & 0 & 0 & 0 \\ -C_1 & C_1 + C_2 & -C_2 & 0 & 0 \\ 0 & -C_2 & C_2 + C_3 & -C_3 & 0 \\ 0 & 0 & -C_3 & C_3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \ldots$$

$$\ldots \xrightarrow{\text{Cano 7}} \begin{pmatrix} C_1 + C_7 & -C_1 & 0 & 0 & -C_7 \\ -C_1 & C_1 + C_2 + C_5 & -C_2 & 0 & -C_5 \\ 0 & -C_2 & C_2 + C_3 + C_6 & -C_3 & -C_6 \\ 0 & 0 & -C_3 & C_3 + C_4 & -C_4 \\ -C_7 & -C_5 & -C_6 & -C_4 & C_4 + C_5 + C_6 + C_7 \end{pmatrix}$$

Para fazer tal montagem precisaremos programar uma função de `python` que automatize esse processo de montagem de matrizes locais, o que deveria ser algo do tipo:

```
def Assembly(conec, C):
    nv = ...  # numero de nos
    nc = ...  # numero de canos
    A = np.zeros(shape=(nv,nv))
    for k in range(nc):
        n1 = conec[k,0]
        n2 = conec[k,1]
        .
        .

    return A
```

em que `C` é o vetor que possui os valores das condutâncias dos canos da rede. Os detalhes ficarão como exercício para a lista 2.

Montagem do sistema final e Resolução

Uma vez que temos a matriz A do sistema, precisamos montar a matriz $\tilde{A}$ e o vetor de lado direito que é forma o sistema de equações definitivo que precisa ser resolvido para determinar as pressões nos nós da rede, que é o que nos interessa em concreto. Para isto, precisaremos programar uma função de `python` que modifique a linha da matriz na posição `natm` daquele nó que foi conectado à atmosfera e também cria um vetor de lado direito que incorpora a informação da vazão injetada pela bomba `QB` na posição `nB`. A função deveria ser algo do tipo[1]

1: A função `np.linalg.solve` essencialmente faz o escalonamento da matriz para encontrar a solução.

```
def SolveNetwork(conec, C, natm, nB, QB):
    Atilde = Assembly( ... )
    Atilde[natm, :] = ...
    .
    .
    b = np.zeros( ... )
    .
    pressure = np.linalg.solve(Atilde, ... )
    return pressure
```

Os detalhes ficarão como exercício para a lista 2.

Post-processo da solução

Uma vez resolvido o problema, estamos interessados em analizar a solução, para extrair informações de interesse do cálculo:

- Cálculo das vazões nos canos;
- Cálculo da potência consumida pela bomba;
- Visualização e plotagem da solução.

Para o primeiro ponto, uma forma elegante de fazer o cálculo é usar a seguinte fórmula:

$$\mathbf{Q} = \mathbf{K}\,\mathbf{D}\,\mathbf{p}$$

em que $\mathbf{K} \in \mathbb{R}^{n_c \times n_c}$ é a matriz diagonal com as conductâncias dos canos definida por

$$\mathbf{K}_{ij} = \begin{cases} C_i & \text{se } i = j \\ 0 & \text{no resto} \end{cases}$$

e a matriz $\mathbf{D} \in \mathbb{R}^{n_c \times n_v}$ é a matriz definida por:

$$\mathbf{D}_{kj} = \begin{cases} 1 & \text{se } j = \texttt{conec[k,0]} \\ -1 & \text{se } j = \texttt{conec[k,1]} \\ 0 & \text{no resto} \end{cases}$$

Para o segundo ponto, pode-se provar que a potência consumida pela bomba para fazer circular o fluido é dada por:

$$W = \mathbf{p}^\intercal \ (\mathbf{D}^\intercal \, \mathbf{K} \, \mathbf{D}) \ \mathbf{p}$$

isto surge do fato de que a potência consumida pela bomba tem que ser igual a energia dissipada nos canos da rede, i.e.,

$$W = \sum_{k=1}^{n_c} Q_k \Delta p_k$$

em que Q_k é a vazão pelo cano k e Δp_k é a diferença de pressão nos extremos do cano k. Mas, se colocarmos as vazões e as diferenças de pressão em vetores, podemos escrever

$$W = \mathbf{Q} \cdot \mathbf{\Delta} p = \mathbf{Q}^\intercal \, \mathbf{\Delta} p$$

i.e., W é o igual ao produto escalar desses vetores. Agora, notemos que $\mathbf{\Delta} p = \mathbf{D}\, p$ e $\mathbf{Q} = \mathbf{K}\,\mathbf{D}\,\mathbf{p}$, então

$$W = \mathbf{Q}^\intercal \, \mathbf{\Delta} p = (\mathbf{K}\,\mathbf{D}\,\mathbf{p})^\intercal \, \mathbf{\Delta} p = (\mathbf{p}^\intercal \mathbf{D}^\intercal \mathbf{K}^\intercal)(\mathbf{D}\,\mathbf{p}) = \mathbf{p}^\intercal (\mathbf{D}^\intercal \mathbf{K} \mathbf{D})\, \mathbf{p}$$

pois o produto de matrizes é **associativo** e $\mathbf{K} = \mathbf{K}^\intercal$, por ser ela uma matriz diagonal[2].

Finalmente, para visualizar a solução, na lista de exercícios iremos disponibilizar algumas funções de `python` que permitirão mostrar a distribuição de pressões em certo tipo de redes como a mostrada na figura.

2: Notar que o resultado é um número, i.e., uma matriz de 1×1, pois a potência que tem unidades de Energia por unidade de tempo (p.e., [J/s]) é de fato uma quantidade escalar.

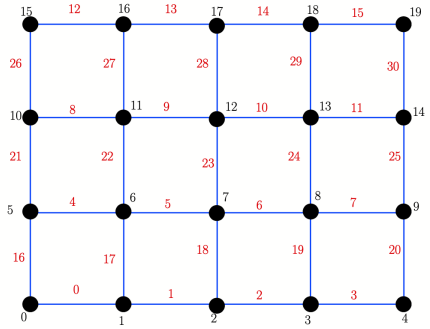

Figure 2.4: Exemplo de uma rede de tipo grade possuido $m = 5$ nós na horizontal e $n = 4$ nós na vertical (i.e., $n_v = m\,n = 20$ nós). Notar que a numeração de nós e arestas já é feita começando em 0.

Aviso importante

O material de estudo para este capítulo serão as notas apresentadas no presente capítulo, os `jupyter notebooks` disponibilizados pelo professor, assim como consultas dirigidas ao professor em forma presencial ou por e-mail (`rfausas@icmc.usp.br`).

2.3 Lista 2 de exercícios

A lista de exercícios a ser apresentada em avaliação oral por sorteio é dada na sequência:

Exercícios

1. Completar a função de `python` que faz a montagem da matriz A. Testar na rede que tem sido usada como exemplo.
2. Completar a função de `python` que cria a matriz $\tilde{A}$ e vetor de lado direito do sistema b de equações definitivo a ser resolvido, resolve o sistema e retorna o vetor de pressões com a solução. Testar na rede que tem sido usada como exemplo. Conferir a resposta com a obtida anteriormente.

3. Fazer uma função de `python` que calcula o vetor de vazões na rede segundo o explicado anteriormente. Explicar como é que o cálculo das matrizes **K** e **D** leva a determinar as vazões.
4. Incluir o cálculo da potência na função que foi programada no exercício anterior.
5. Considerar a rede hidráulica de tipo grade gerada com a função que foi disponibilizada no `jupyter notebook`. Montar um exemplo de resolução no qual a rede possui 10×12 nós, uma vazão $Q_B = 3\,m^3/s$ é injetada no nó do canto inferior esquerdo e a descarga à atmosfera é feita pelo nó do canto superior direito. Aplique todas as funções desenvolvidas até agora e visualize a solução usando a função `PlotPressure` disponibilizada no `jupyter notebook`. Mudar o ponto de descarga e comparar com a solução anterior.
6. Considerar redes hidráulicas de tipo grade de tamanho variável $m \times n$ (p.e., 10×10, 20×20, 40×40, 80×80, etc.) Fazer um código que resolve o sistema para encontrar a solução e mede o tempo de cálculo envolvido. Plotar em escala loglog o tempo como função do número de incógnitas $n_v = m\,n$.

2.4 Cálculos Monte Carlo em Redes

Objetivos

- Realizar cálculos estocásticos para estimar a probabilidade de certos eventos em sistemas complexos, tais como uma rede hidráulica composta de centenas o milhares de arestas.
- Estes cálculos estocásticos são a base dos chamados métodos Monte Carlo, que são uma classe de métodos computacionais baseados em amostragens aleatórias.
- Estes métodos são muitos poderosos e flexíveis para resolver problemas complexos. Muitas vezes são a única forma de resolver-los.

Vejamos um exemplo de um problema complexo que poderiamos resolver usando estes métodos:

Análise de risco

- Um bairro conta com uma rede hidráulica conhecida (ver figura Figure 2.5), alimentada por uma bomba que injeta uma vazão $Q_B = 10\ \mathrm{m}^3/\mathrm{s}$ no nó do canto inferior esquerdo, i.e., o nó `nB = 0` das redes que geramos usando a função `GeraRede(...)`
- No nó `natm = nv - 1` é fixada uma pressão nula.
- A rede possui dois tipos de canos: **grossos**, com "condutância" $C = 20$, e os **finos**, com $C = 2$.
- Anualmente, os canos **finos** (apenas eles) ficam obstruidos com probabilidade p_O. Quando obstruidos, seu valor de "condutância" se reduz a `Centup=0.2`, ou seja, 10 vezes menor do que o valor

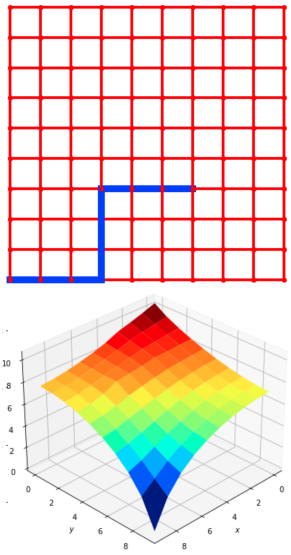

Figure 2.5: Rede nominal que será usada para realizar a analise de risco.

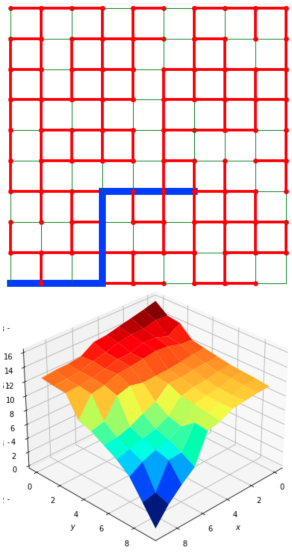

Figure 2.6: Exemplo de uma rede com canos entupido que será usada para realizar a analise de risco.

nominal (ver exemplo de uma rede com obtruções na figura Figure 2.6).

Pergunta

Qual é a probabilidade de que, passado um ano, a pressão máxima na rede seja maior do que 12?

Mas, vejamos que acontece neste problema:

- Seja M o número de canos finos, todos distintos por ocuparem lugares distintos na rede.
- Existem 2^M casos diferentes. Então, supondo, por exemplo, $M = 100$ teremos $2^{100} = 1.27 \times 10^{30}$ casos.
- Cada caso que deve ser analisado corresponde a um vetor de condutâncias da rede diferente.
- Para cada caso, podemos calcular deterministicamente todas as pressões, em particular a máxima:

```
Cnew = RandomFailFinos(C, p0, Centup)
pressure = SolveNetwork(conec, Cnew, nB, QB, natm)
pmax = np.max(pressure)
```

  em que a função `RandomFailFinos` gera uma instancia de possíveis canos entupidos na rede.
- Então, podemos avaliar a probabilidade como:

$$\text{Prob} = \frac{\#\text{ num. de casos que aconteceu o evento}}{\#\text{ num. total de possíveis casos}}$$

- Se formos calcular todos os possíveis casos surge uma **dificuldade:**

Aviso importante

Se o cálculo de um caso particular demora **1 nanosegundo**, para analisar todos demoramos $\mathbf{4 \times 10^{13}}$ **anos**.

Para entender a ideia por trás dos métodos Monte Carlo (MC), e como que eles podem nos auxiliar para responder a pergunta anterior, vamos fazer um exemplo simples:

Cálculo do número π

Vamos estimar el valor do número π da seguinte forma: Se consideramos a região quadrada $[-1, 1] \times [-1, 1]$ e um círculo inscrito, a razão das áreas entre estes é $\pi/4$ (ver figura).

Então, podemos propor um cálculo estocástico que consistirá em jogar pontos dentro dessa região quadrada. Alguns desses pontos irão cair dentro do circulo e outros fora. Como é de se esperar, a razão entre a quantidade de pontos que cai dentro e a quantidade total de pontos jogados, deveria tender justamente a razão entre as áreas de círculo e do quadrado ($\pi/4$).

Se denotarmos por N_c e N ao número de pontos dentro do círculo e ao número total de pontos, respectivamente:

$$\text{Prob} = \frac{\pi}{4} = \lim_{N\to\infty} \frac{\#\text{num. pontos no círculo}}{\#\text{num. total de pontos}} = \lim_{N\to\infty} \frac{N_c}{N}$$

Isto leva ao seguinte roteiro que precisamos executar:

1. Gerar (x_i, y_i), formado por dois números x_i e y_i entre -1 e 1, independentes e com probabilidade uniforme.
2. Calcular θ_i, definida como igual a 1 se $x_i^2 + y_i^2 < 1$ e igual a 0 se não. Seja $\Theta = \{\theta_i\}$ a sequência gerada.
3. Tirar a média

$$\hat{\theta} = \frac{1}{N}\sum_{i=1}^{N} \theta_i$$

o qual se materializa no seguinte código de python:

```
# Computation of pi by stocastic method

N = 10000 # Number of realizations
Nc = 0
for i in range(N):
 x = -1.0 + 2.0*np.random.rand()
 y = -1.0 + 2.0*np.random.rand()
 if (x**2 + y**2 < 1.0):
  Nc = Nc + 1;

Prob = Nc / N
print('The estimate for pi is:', 4*Prob)
```

Os resultados para diferente número de realizações se mostram na figura ao lado e na tabela seguinte para $4 \times p$ (o que deveria tender a π)[3]:

#	$N = 10^3$	$N = 10^4$	$N = 10^5$	$N = 10^6$	$N = 10^7$
1	3.0800	3.1492	3.1450	3.1418	3.1405
2	3.2320	3.1340	3.1396	3.1400	−−
3	3.1240	3.1412	3.1469	3.1376	−−
4	3.0800	3.1660	3.1488	3.1406	−−

Agora que entedemos como funciona um método MC vamos aplicar ele para fazer a analise de risco em redes hidráulicas, que é o tema desenvolvido na Atividade 1.

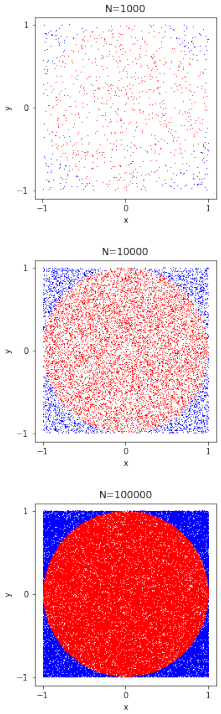

Figure 2.7: Exemplo do cálculo MC.

3: Um dos pontos a destacar é que estamos usando uma distribuição uniforme de probabilidades (`np.random.rand()`) para gerar os pontos randômicos, i.e., a probabilidade é a mesma para qualquer número em $(0, 1)$ e os pontos gerados são "independentes" um dos outros. Isto é um ingrediente essencial do método MC.

2.5 Atividade 1

A primeira atividade a ser desenvolvida em grupo e com relatório que será entregue em data a ser definida, segue na sequência:

Exercícios

1. Fazer uma função que pega as condutâncias da rede e gera uma lista com as novas condutâncias dos valores individuais dos canos que foram entupidos numa determinada realização.

Considerar uma estrutura do tipo:

```
def RandomFailFinos(C, pO, Centup):
    Cnew = np.copy(C)
    nc = ...
    for k in range(nc):
        x = np.random.rand()
        if( ..... ):
            Cnew[k] = ...

    return Cnew
```

2. Realizar o cálculo de probabilidade Prob de que, passado um ano, a pressão máxima na rede seja maior do que 12?
   Para isto:
   - Seguir um roteiro de cálculo similar ao que foi feito no exemplo de cálculo do número π;
   - Realizar entre 5000 e 10000 realizações para cada caso;
   - Considerar diferentes valores de p_O e fazer uma tabela mostrando Prob como função de p_O;
   - Plotar a evolução dessa probabilidade como função do número de realizações;
   - Tirar conclusões.

2.5.1 Exercícios opcionais

Na sequência tem um exercício extra que é opcional. Quem quiser, pode apresentar ele para o professor de maneira individual, valendo como uma das avaliações orais.

BONUS EXTRA

1. Considerar uma rede com os seguintes parâmetros:
   - `n = 8`
   - `m = 9`
   - `QB = 3`
   - `natm = n*m - 1`
   - `nB = 0`
   - As condutâncias dependem de um parâmetro x. Considerar os casos:

$$\begin{aligned} CH &= 2.3 + 0.1\,(x-1)^2, \\ CV &= 1.8 + 0.2\,(x-1)^2 \end{aligned}$$

   e

$$\begin{aligned} CH &= 2.3 + 10\,e^{-(x-5)^2}, \\ CV &= 1.8 + 10\,e^{-(x-5)^2} \end{aligned}$$

   A ideia é plotar em cada caso, a potência dissipada como função do parâmetro x e determinar visualmente para que valor de x a potencia é igual a 6. Barrer valores de x num intervalo

interessante. Aplicar as funções desenvolvidas nos exercícios anteriores.

Alphabetical Index

preface, ii