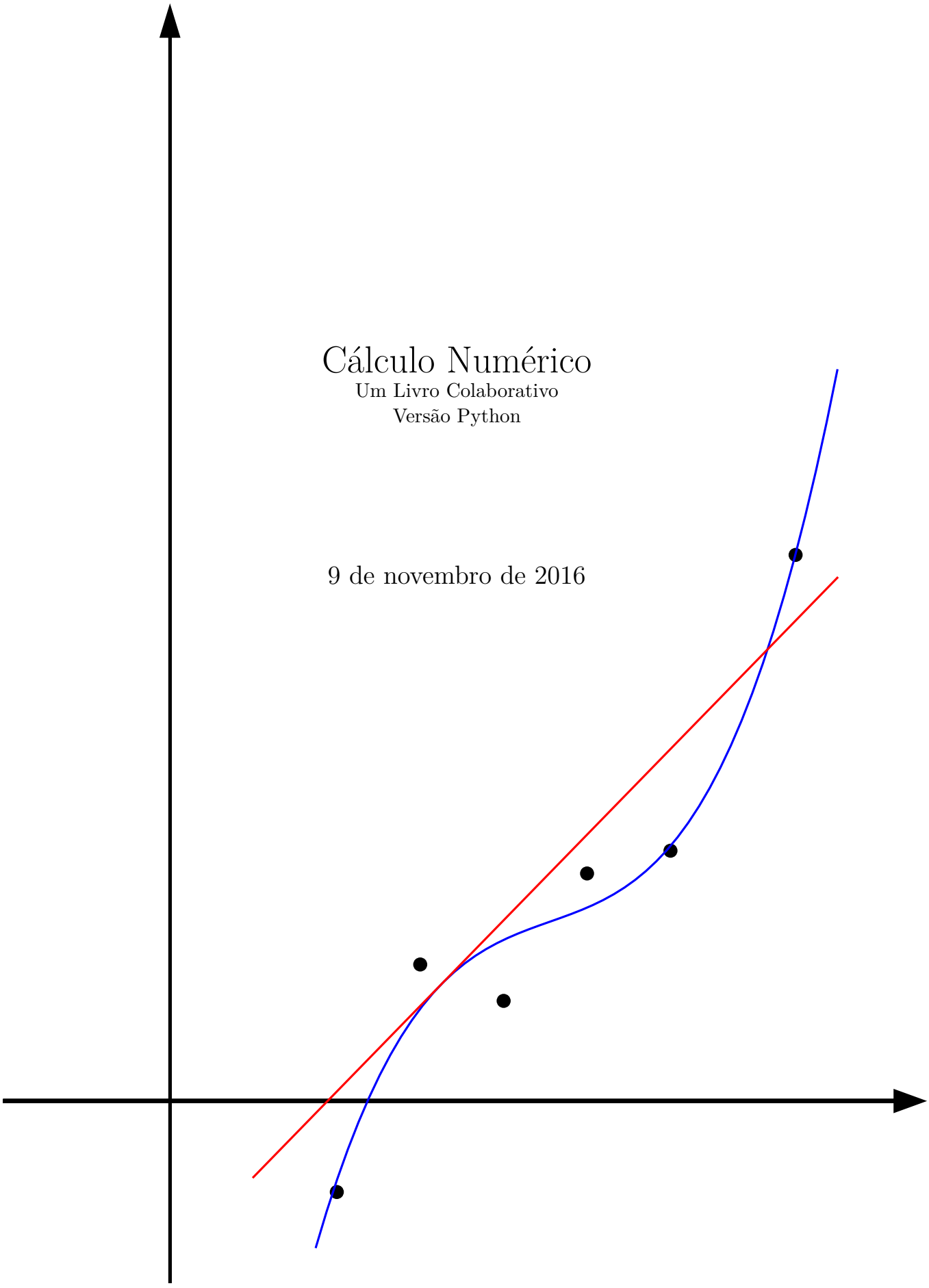
Cálculo Numérico

Um Livro Colaborativo

Versão Python

9 de novembro de 2016



Organizadores

Dagoberto Adriano Rizzotto Justo - UFRGS

Esequia Sauter - UFRGS

Fabio Souto de Azevedo - UFRGS

Leonardo Fernandes Guidi - UFRGS

Matheus Correia dos Santos - UFRGS

Pedro Henrique de Almeida Konzen - UFRGS

Licença

Nota dos organizadores

Estamos escrevendo este livro de forma colaborativa desde 2011 e, recentemente, decidimos por abrir à colaborações externas. Nosso objetivo é de produzir um material didático em nível de graduação de excelente qualidade e de acesso livre pela colaboração entre professores e alunos de universidades, institutos de educação e demais interessados na análise, estudo e aplicação de métodos numéricos nos mais diversos ramos da ciência e da tecnologia.

O sucesso do projeto depende da colaboração! Edite você mesmo o livro, dê sugestões ou nos avise de erros e imprecisões. Toda a colaboração é bem vinda. Saiba mais visitando o site oficial do projeto:

`http://www.ufrgs.br/numerico`

Nós preparamos uma série de ações para ajudá-lo a participar. Em primeiro lugar, o acesso irrestrito ao livro pode se dar através do site oficial do projeto. Disponibilizamos o livro na versão original em PDF e versões adaptadas em HTML, EPUB e Slides. Além disso, o livro está escrito em código LaTeXdisponível em repositório GitHub público.

Nada disso estaria completo sem uma licença apropriada à colaboração. Por isso, escolhemos disponibilizar o material do livro sob licença Creative Commons Atribuição-CompartilhaIgual 3.0 Não Adaptada (CC-BY-SA 3.0). Ou seja, você pode copiar, redistribuir, alterar e construir um novo material para qualquer uso, inclusive comercial. Leia a licença para maiores informações.

Lhe desejamos ótimas colaborações!

Prefácio

Este livro busca abordar os tópicos de um curso de introdução ao cálculo numérico moderno oferecido a estudantes de matemática, física, engenharias e outros. A ênfase é colocada na formulação de problemas, implementação em computador da resolução e interpretação de resultados. Pressupõe-se que o estudante domine conhecimentos e habilidades típicas desenvolvidas em cursos de graduação de cálculo, álgebra linear e equações diferenciais. Conhecimentos prévios em linguagem de computadores é fortemente recomendável, embora apenas técnicas elementares de programação sejam realmente necessárias.

Nesta versão do livro, fazemos ênfase na utilização da linguagem computacional Python para a implementação dos métodos numéricos abordados. Recomendamos ao leitor ter à sua disposição um computador com o interpretador `Python 2.7` (ou superior) e o conjunto de biblioteca SciPy instalados. Não é necessário estar familiarizado com esta linguagem, mas recomendamos a leitura do Apêndice A, no qual apresentamos uma rápida introdução a esta linguagem com ênfase naquilo que é mais excencial para a leitura do livro. Alternativamente, existem algumas soluções em nuvem que fornecem acesso a consoles *online* Python. Veja, por exemplo, o SageMathCloud.

Os métodos numéricos apresentados em código `Scilab` no livro são implementados em uma abordagem didática. Isto é, temos o objetivo de que a implementação em linguagem computacional venha a auxiliar o leitor no aprendizado das técnicas numéricas que são apresentadas no livro. Implementações computacional eficientes de técnicas de cálculo numérico podem ser obtidas na série de livros "Numerical Recipes", veja [9].

Sumário

Capítulo 1

Introdução

Cálculo numérico é a disciplina que estuda as técnicas para a solução aproximada de problemas matemáticos. Estas técnicas são de natureza analítica e computacional. As principais preocupações normalmente envolvem exatidão e perfórmance.

Aliado ao aumento contínuo da capacidade de computação disponível, o desenvolvimento de métodos numéricos tornou a simulação computacional de modelos matemáticos uma prática usual nas mais diversas áreas científicas e tecnológicas. As então chamadas simulações numéricas são constituídas de um arranjo de vários esquemas numéricos dedicados a resolver problemas específicos como, por exemplo: resolver equações algébricas, resolver sistemas lineares, interpolar e ajustar pontos, calcular derivadas e integrais, resolver equações diferenciais ordinárias, etc.. Neste livro, abordamos o desenvolvimento, a implementação, utilização e aspectos teóricos de métodos numéricos para a resolução desses problemas.

Os problemas que discutiremos não formam apenas um conjunto de métodos fundamentais, mas são, também, problemas de interesse na engenharia, na física e na matemática aplicada. A necessidade de aplicar aproximações numéricas decorre do fato de que esses problemas podem se mostrar intratáveis se dispomos apenas de meios puramente analíticos, como aqueles estudados nos cursos de cálculo e álgebra linear. Por exemplo, o teorema de Abel-Ruffini nos garante que não existe uma fórmula algébrica, isto é, envolvendo apenas operações aritméticas e radicais, para calcular as raízes de uma equação polinomial de qualquer grau, mas apenas casos particulares:

- Simplesmente isolar a incógnita para encontrar a raiz de uma equação do primeiro grau;
- Fórmula de Bhaskara para encontrar raízes de uma equação do segundo grau;
- Fórmula de Cardano para encontrar raízes de uma equação do terceiro grau;

- Existe expressão para equações de quarto grau;
- Casos simplificados de equações de grau maior que 4 onde alguns coeficientes são nulos também podem ser resolvidos.

Equações não polinomiais podem ser ainda mais complicadas de resolver exatamente, por exemplo:

$$\cos(x) = x \qquad \text{e} \qquad xe^x = 10$$

Para resolver o problema de valor inicial

$$y' + xy = x,$$
$$y(0) = 2,$$

podemos usar o método de fator integrante e obtemos $y = 1 + e^{-x^2/2}$. Já o cálculo da solução exata para o problema

$$y' + xy = e^{-y},$$
$$y(0) = 2,$$

não é possível.

Da mesma forma, resolvemos a integral

$$\int_1^2 xe^{-x^2} dx$$

pelo método da substituição e obtemos $\frac{1}{2}(e^{-1} - e^{-2})$. Porém a integral

$$\int_1^2 e^{-x^2} dx$$

não pode ser resolvida analiticamente.

A maioria dos modelos de fenômenos reais chegam em problemas matemáticos onde a solução analítica é difícil (ou impossível) de ser encontrada, mesmo quando provamos que ela existe. Nesse curso propomos calcular aproximações numéricas para esses problemas, que apesar de, em geral, serem diferentes da solução exata, mostraremos que elas podem ser bem próximas.

Para entender a construção de aproximações é necessário estudar um pouco como funciona a aritmética de computador e erros de arredondamento. Como computadores, em geral, usam uma base binária para representar números, começaremos falando em mudança de base.

Capítulo 2

Aritmética de máquina

Neste capítulo, discutiremos sobre formas de represenção de números em computadores. Iniciamos discutindo sobre representação posicional e mudança de base. Então, discutimos sobre representação com número de dígitos finitos e, mais especificamente, as representações de números inteiros, ponto fixo e ponto flutuante em computadores.

A representação de números e a aritmética em computadores levam aos chamados erros de arredondamento e de truncamento. Ao final deste capítulo, discutiremos sobre os efeitos do erro de arredondamento na computação científica.

Ao longo do capítulo, faremos alguns comentários usando códigos em `Python 2.7`. Nestes, estaremos assumindo que os seguintes módulos estão carregados:

```
>>> from __future__ import division
>>> import numpy as np
```

A primeira instrução garante que divisões de números inteiros sejam computadas em ponto flutuante (`double`) e a segunda carrega a biblioteca de computação científica numpy.

2.1 Sistema de numeração e mudança de base

Usualmente, utilizamos o sistema de numeração decimal para representar números. Esse é um sistema de numeração posicional onde a posição do dígito indica a potência de 10 que o dígito está representando.

Exemplo 2.1.1. O número 293 é decomposto como

$$\begin{aligned} 293 &= 2 \text{ centenas} + 9 \text{ dezenas} + 3 \text{ unidades} \\ &= 2 \times 10^2 + 9 \times 10^1 + 3 \times 10^0. \end{aligned}$$

O sistema de numeração posicional também pode ser usado com outras bases. Vejamos a seguinte definição.

Definição 2.1.1 (Sistema de numeração de base b)**.** *Dado um número natural $b > 1$ e o conjunto de símbolos $\{,, -, \mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots, \boldsymbol{b-1}\}$[1], a sequência de símbolos*

$$(d_n d_{n-1} \cdots d_1 d_0, d_{-1} d_{-2} \cdots)_b$$

representa o número positivo

$$d_n \cdot b^n + d_{n-1} \cdot b^{n-1} + \cdots + d_0 \cdot b^0 + d_{-1} \cdot b^{-1} + d_{-2} \cdot b^{-2} + \cdots$$

Para representar números negativos usamos o símbolo $-$ a esquerda do numeral.

Observação 2.1.1 ($b \geq 10$)**.** Para sistemas de numeração com base $b \geq 10$ é usual utilizar as seguintes notações:

- No sistema de numeração decimal ($b = 10$), costumamos representar o número sem os parênteses e o subíndice, ou seja,

$$\pm d_n d_{n-1} \ldots d_1 d_0, d_{-1} d_{-2} \ldots := \pm (d_n d_{n-1} \ldots d_1 d_0, d_{-1} d_{-2} \ldots)_{10}$$

- Se $b > 10$, usamos as letras $A, B, C, \cdots$ para completar os símbolos: $A = 10$, $B = 11$, $C = 12$, $D = 13$, $E = 14$, $F = 15$.

Exemplo 2.1.2 (Sistema binário)**.** O sistema de numeração em base dois é chamado de binário e os algarismos binários são conhecidos como *bits*, do inglês **binary digits**. Um *bit* pode assumir dois valores distintos: 0 ou 1. Por exemplo:

$$\begin{aligned} x &= (1001{,}101)_2 \\ &= 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} \\ &= 8 + 0 + 0 + 1 + 0{,}5 + 0 + 0{,}125 = 9{,}625 \end{aligned}$$

Ou seja, $(1001{,}101)_2$ é igual a 9,625 no sistema decimal.

Exemplo 2.1.3 (Sistema quaternário)**.** No sistema quaternário a base b é igual a 4. Por exemplo:

$$(301{,}2)_4 = 3 \cdot 4^2 + 0 \cdot 4^1 + 1 \cdot 4^0 + 2 \cdot 4^{-1} = 49{,}5$$

Exemplo 2.1.4 (Sistema octal)**.** No sistema octal a base é $b = 8$ e utilizamos os símbolos em $\{0, 1, 2, 3, 4, 5, 6, 7\}$. Por exemplo:

$$\begin{aligned} (1357{,}24)_8 &= 1 \cdot 8^3 + 3 \cdot 8^2 + 5 \cdot 8^1 + 7 \cdot 8^0 + 2 \cdot 8^{-1} + 4 \cdot 8^{-2} \\ &= 512 + 192 + 40 + 7 + 0{,}25 + 0{,}0625 = 751{,}3125 \end{aligned}$$

[1] Para $b > 10$, veja a Observação 2.1.1

Exemplo 2.1.5 (Sistema hexadecimal)**.** O sistema de numeração cuja a base é $b = 16$ é chamado de sistema hexadecimal. O conjunto de símbolos necessários é $S = \{“,”, -, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$. Convertendo o número $(E2AC)_{16}$ para a base 10 temos

$$\begin{aligned}(E2AC)_{16} &= 14 \cdot 16^3 + 2 \cdot 16^2 + 10 \cdot 16^1 + 12 \cdot 16^0 \\ &= 57344 + 512 + 160 + 12 = 58028\end{aligned}$$

A partir da Definição 2.1.1 acabamos de mostrar vários exemplos de conversão de números de uma sistema de numeração de base b para o sistema decimal. Agora, vamos estudar como fazer o processo inverso. Isto é, dado um número decimal $(X)_{10}$ queremos escrevê-lo em uma outra base b, i.e., queremos obter a seguinte representação:

$$\begin{aligned}(X)_{10} &= (d_n d_{n-1} \cdots d_0, d_{-1} \cdots)_b \\ &= d_n \cdot b^n + d_{n-1} \cdot b^{n-1} + \cdots + d_0 \cdot b^0 + d_{-1} \cdot b^{-1} + d_{-2} \cdot b^{-2} + \cdots\end{aligned}$$

Separando as partes inteira e fracionária de X, i.e. $X = X^{\mathrm{i}} + X^{\mathrm{f}}$, temos:

$$X^{\mathrm{i}} = d_n \cdot b^n + \cdots + d_{n-1} b^{n-1} \cdot + d_1 \cdot b^1 + d_0 \cdot b^0$$

e

$$X^{\mathrm{f}} = \frac{d_{-1}}{b^1} + \frac{d_{-2}}{b^2} + \cdots$$

Nosso objetivo é determinar os algarismos $\{d_n, d_{n-1}, ...\}$.

Primeiramente, vejamos como tratar a parte inteira X^{i}. Calculando sua divisão por b, temos:

$$\frac{X^{\mathrm{i}}}{b} = \frac{d_0}{b} + d_1 + d_2 b^1 \cdots + d_{n-1} \cdot b^{n-2} + d_n \cdot b^{n-1}.$$

Observe que d_0 é o resto da divisão de X^{i} por b, pois $d_1 + d_2 b^1 \cdots + d_{n-1} \cdot b^{n-2} + d_n \cdot b^{n-1}$ é inteiro e $\frac{d_0}{b}$ é uma fração (lembramos que $d_0 < b$). Da mesma forma, o resto da divisão de $d_1 + d_2 b^1 \cdots + d_{n-1} \cdot b^{n-2} + d_n \cdot b^{n-1}$ por b é d_1. Repetimos o processo até encontrar os símbolos $d_0, d_1, d_2, \ldots$.

Exemplo 2.1.6 (Conversão da parte inteira)**.** Vamos escrever o número 125 na base 6. Para tanto, fazemos sucessivas divisões por 6 como segue:

$$\begin{aligned}125 &= 20 \cdot 6 + 5 \quad \text{(125 dividido por 6 é igual a 20 e resta 5)} \\ &= (3 \cdot 6 + 2) \cdot 6 + 5 = 3 \cdot 6^2 + 2 \cdot 6 + 5,\end{aligned}$$

logo $125 = (325)_6$.

Estes cálculos podem ser feitos em `Python` com o auxílio do comando `%` e da função `int`. Com o primeiro calculamos o resto da divisão entre dois números, enquanto que a segunda retorna a parte inteira de um número dado. No nosso exemplo, temos:

```
>>> q = 125; d0 = (q % 6); print(q,d0)
>>> q = int(q/6); d1 = (q % 6); print(q,d1)
>>> q = int(q/6); d2 = (q % 6); print(q,d2)
```

Verifique!

Vamos converter a parte fracionária de um número decimal em uma dada base b. Usando a notação $X = X^{\mathrm{i}} + X^{\mathrm{f}}$ para as partes inteira e fracionária, respectivamente, temos:

$$bX^{\mathrm{f}} = d_{-1} + \frac{d_{-2}}{b} + \frac{d_{-3}}{b^2} + \cdots$$

Observe que a parte inteira desse produto é d_{-1} e $\frac{d_{-2}}{b} + \frac{d_{-3}}{b^2} + \cdots$ é a parte fracionária. Quando multiplicamos $\frac{d_{-2}}{b} + \frac{d_{-3}}{b^2} + \cdots$ por b novamente, encontramos d_{-2}. Repetimos o processo até encontrar todos os símbolos.

Exemplo 2.1.7 (Conversão da parte fracionária)**.** Escrever o número $125{,}58\overline{3}$ na base 6. Do exemplo anterior temos que $125 = (325)_6$. Assim, nos resta converter a parte fracionária. Para tanto, fazemos sucessivas multiplicações por 6 como segue:

$$\begin{aligned} 0{,}58\overline{3} &= 3{,}5 \cdot 6^{-1} \quad (0{,}58\overline{3} \text{ multiplicado por 6 é igual a } 3{,}5) \\ &= 3 \cdot 6^{-1} + 0{,}5 \cdot 6^{-1} \\ &= 3 \cdot 6^{-1} + (3 \cdot 6^{-1}) \cdot 6^{-1} \\ &= 3 \cdot 6^{-1} + 3 \cdot 6^{-2}, \end{aligned}$$

logo $0{,}58\overline{3} = (0{,}33)_6$. As contas feitas aqui, também podem ser computadas em `Python`. Você sabe como?

Uma maneira de converter um número dado numa base b_1 para uma base b_2 é fazer em duas partes: primeiro converter o número dado na base b_2 para base decimal e depois converter para a base b_1.

Exercícios

E 2.1.1. Converta para base decimal cada um dos seguintes números:

a) $(100)_2$ c) $(100)_b$ e) $(AA)_{16}$ g) $(3{,}12)_5$

b) $(100)_3$ d) $(12)_5$ f) $(7{,}1)_8$

E 2.1.2. Escreva os números abaixo na base decimal.

a) $(25{,}13)_8$

b) $(101{,}1)_2$

c) $(12F{,}4)_{16}$

d) $(11{,}2)_3$

E 2.1.3. Escreva cada número decimal na base b.

a) $7{,}\overline{6}$ na base $b = 5$

b) $29{,}1\overline{6}$ na base $b = 6$

E 2.1.4. Escreva cada número dado para a base b.

a) $(45{,}1)_8$ para a base $b = 2$

b) $(21{,}2)_8$ para a base $b = 16$

c) $(1001{,}101)_2$ para a base $b = 8$

d) $(1001{,}101)_2$ para a base $b = 16$

E 2.1.5. Escreva o número $x = 5{,}5$ em base binária.

E 2.1.6. Escreva o número $x = 17{,}109375$ em base hexadecimal (16).

E 2.1.7. Quantos algarismos são necessários para representar o número 937163832173947 em base binária? E em base 7? Dica: Qual é o menor e o maior inteiro que pode ser escrito em dada base com N algarismos?

E 2.1.8. Escreva $x = (12.4)_8$ em base decimal e binária.

2.2 Representação de números

Os computadores, em geral, usam a base binária para representar os números, onde as posições, chamadas de bits, assume as condições "verdadeiro" ou "falso", ou seja, 0 ou 1. Cada computador tem um número de bits fixo e, portanto, representa uma quantidade finita de números. Os demais números são tomados por proximidade àqueles conhecidos, gerando erros de arredondamento. Por exemplo,

em aritmética de computador, o número 2 tem representação exata, logo $2^2 = 4$, mas $\sqrt{3}$ não tem representação finita, logo $(\sqrt{3})^2 \neq 3$.

Veja isso em `Python`:

```
>>> 2**2 == 4
True
>>> np.sqrt(3)**2 == 3
False
```

2.2.1 Números inteiros

Tipicamente um número inteiro é armazenado num computador como uma sequência de dígitos binários de comprimento fixo denominado **registro**.

Representação sem sinal

Um registro com n bits da forma

d_{n-1}	d_{n-2}	$\cdots$	d_1	d_0

representa o número $(d_{n-1}d_{n-2}...d_1d_0)_2$.

Assim é possível representar números inteiros entre

$$(111...111)_2 = 2^{n-1} + 2^{n-2} + \cdots + 2^1 + 2^0 = 2^n - 1.$$

$$\vdots$$

$$(000...011)_2 = 3$$
$$(000...010)_2 = 2$$
$$(000...001)_2 = 1$$
$$(000...000)_2 = 0$$

Representação com bit de sinal

O bit mais significativo (o primeiro à esquerda) representa o sinal: por convenção, 0 significa positivo e 1 significa negativo. Um registro com n bits da forma

s	d_{n-2}	$\cdots$	d_1	d_0

representa o número $(-1)^s(d_{n-2}...d_1d_0)_2$. Assim é possível representar números inteiros entre -2^{n-1} e 2^{n-1}, com duas representações para o zero: $(1000...000)_2$ e $(00000...000)_2$.

Exemplo 2.2.1. Em um registro com 8 bits, teremos os números

$$\begin{aligned}
(11111111)_2 &= -(2^6 + \cdots + 2 + 1) = -127 \\
&\vdots \\
(10000001)_2 &= -1 \\
(10000000)_2 &= -0 \\
(01111111)_2 &= 2^6 + \cdots + 2 + 1 = 127 \\
&\vdots \\
(00000010)_2 &= 2 \\
(00000001)_2 &= 1 \\
(00000000)_2 &= 0
\end{aligned}$$

Representação complemento de dois

O bit mais significativo (o primeiro à esquerda) representa o coeficiente de -2^{n-1}. Um registro com n bits da forma:

d_{n-1}	d_{n-2}	$\cdots$	d_1	d_0

representa o número $-d_{n-1}2^{n-1} + (d_{n-2}...d_1d_0)_2$.

Note que todo registro começando com 1 será um número negativo.

Exemplo 2.2.2. O registro com 8 bits [01000011] representa o número:

$$-0(2^7) + (1000011)_2 = 64 + 2 + 1 = 67.$$

O registro com 8 bits [10111101] representa o número:

$$-1(2^7) + (0111101)_2 = -128 + 32 + 16 + 8 + 4 + 1 = -67.$$

Note que podemos obter a representação de -67 invertendo os dígitos de 67 em binário e somando 1.

Exemplo 2.2.3. Em um registro com 8 bits, teremos os números

$$\begin{aligned}
(11111111)_2 &= -2^7 + 2^6 + \cdots + 2 + 1 = -1 \\
&\vdots \\
(10000001)_2 &= -2^7 + 1 = -127 \\
(10000000)_2 &= -2^7 = -128 \\
(01111111)_2 &= 2^6 + \cdots + 2 + 1 = 127 \\
&\vdots \\
(00000010)_2 &= 2 \\
(00000001)_2 &= 1 \\
(00000000)_2 &= 0
\end{aligned}$$

2.2.2 Sistema de ponto fixo

O sistema de ponto fixo representa as partes inteira e fracionária do número com uma quantidade fixas de dígitos.

Exemplo 2.2.4. Em um computador de 32 bits que usa o sistema de ponto fixo, o registro

d_{31}	d_{30}	d_{29}	$\cdots$	d_1	d_0

pode representar o número

- $(-1)^{d_{31}}(d_{30}d_{29}\cdots d_{17}d_{16}, d_{15}d_{14}\cdots d_1d_0)_2$ se o sinal for representado por um dígito. Observe que nesse caso o zero possui duas representações possíveis:

  $$10000000000000000000000000000000$$

  e

  $$00000000000000000000000000000000$$

- $(d_{30}d_{29}\cdots d_{17}d_{16})_2 - d_{31}(2^{15} - 2^{-16}) + (0{,}d_{15}d_{14}\cdots d_1d_0)_2$ se o sinal do número estiver representado por uma implementação em complemento de um. Observe que o zero também possui duas representações possíveis:

  $$11111111111111111111111111111111$$

  e

  $$00000000000000000000000000000000$$

- $(d_{30}d_{29}\cdots d_{17}d_{16})_2 - d_{31}2^{15} + (0{,}d_{15}d_{14}\cdots d_1d_0)_2$ se o sinal do número estiver representado por uma implementação em complemento de dois. Nesse caso o zero é unicamente representado por

$$00000000000000000000000000000000$$

Observe que 16 dígitos são usados para representar a parte fracionária, 15 são para representar a parte inteira e um dígito, o d_{31}, está relacionado ao sinal do número.

2.2.3 Normalização

Os números $h = 6.626 \times 10^{-34}$ e $N_A = 6.0221 \times 10^{23}$ não podem ser armazenados na máquina em ponto fixo do exemplo anterior.

Entretanto, a constante

$$\begin{aligned} h &= 6626 \times 10^{-37} \\ h &= 6.626 \times 10^{-34} \\ h &= 0.6626 \times 10^{-33} \\ h &= 0.006626 \times 10^{-31} \end{aligned}$$

pode ser escrita de várias formas diferentes. Para termos uma **representação única** definimos como notação normalizada a segunda opção ($1 \leq m < 10$) que apresenta apenas um dígito diferente de zero a esquerda do ponto decimal ($m = 6.626$).

Definição 2.2.1. *Definimos que*

$$x = (-1)^s (M)_b \times b^E,$$

está na ***notação normalizada***[2] *quando* $1 \leq (M)_b < b$*, onde*

- s *é o* ***sinal*** *(*0 *para positivo e* 1 *para negativo),*
- E *é o* ***expoente****,*
- b *é a base (por ex.* 2*,* 8*,* 10 *ou* 16*),*
- $(M)_b$ *é o significando. O* ***significando*** *(também chamado de mantissa ou coeficiente) contém os dígitos significativos do número.*

[2] Em algumas referências é usado $(0.1)_b \leq (M)_b < 1$.

Exemplo 2.2.5. Os números abaixo estão em notação normalizada:

$$x_1 = (-1.011101)_2 \times 2^{(100)_2}$$

$$x_2 = (-2.325)_{10} \times 10^1$$

Exemplo 2.2.6. Represente os números $0{,}00\overline{51}$ e $1205{,}41\overline{54}$ em um sistema de ponto fixo de 4 dígitos para a parte inteira e 4 dígitos para a parte fracionária. Depois represente os mesmos números utilizando notação normalizada com 7 dígitos significativos.

Solução. As representações dos números $0{,}00\overline{51}$ e $1205{,}41\overline{54}$ no sistema de ponto fixo são 0,0051 e 1205,4154, respectivamente. Em notação normalizada, as representações são $5{,}151515 \cdot 10^{-3}$ e $1{,}205415 \cdot 10^3$, respectivamente. ◊

2.2.4 Sistema de ponto flutuante

O sistema de ponto flutuante não possui quantidade fixa de dígitos para as partes inteira e fracionária do número.

Podemos definir uma máquina F em ponto flutuante de dois modos:

$$F(\beta,|M|,|E|,BIAS) \text{ ou } F(\beta,|M|,E_{MIN},E_{MAX})$$

onde

- β é a base (em geral 2 ou 10),
- $|M|$ é o número de dígitos da mantissa,
- $|E|$ é o número de dígitos do expoente,
- $BIAS$ é um valor de deslocamento do expoente (veja a seguir),
- E_{MIN} é o menor expoente,
- E_{MAX} é o maior expoente.

Considere uma máquina com um registro de 64 bits e base $\beta = 2$. Pelo padrão IEEE754, 1 bit é usado para o sinal, 11 bits para o expoente e 52 bits são usados para o significando tal que

s	c_{10}	c_9	$\cdots$	c_0	m_1	m_2	$\cdots$	m_{51}	m_{52}

represente o número (o $BIAS = 1023$ por definição)

$$x = (-1)^s M \times 2^{c-BIAS},$$

onde a **característica** é representada por

$$c = (c_{10}c_9 \cdots c_1c_0)_2 = c_{10}2^{10} + \cdots + c_12^1 + c_02^0$$

e o significando por

$$M = (1.m_1m_2 \cdots m_{51}m_{52})_2.$$

Em base 2 não é necessário armazenar o primeiro dígito (por quê?).

Por exemplo, o registro

$$[0|100\ 0000\ 0000|1010\ 0000\ 0000...0000\ 0000]$$

representa o número

$$(-1)^0(1 + 2^{-1} + 2^{-3}) \times 2^{1024-1023} = (1 + 0.5 + 0.125)2 = 3.25.$$

O expoente deslocado

Uma maneira de representar os expoentes inteiros é deslocar todos eles uma mesma quantidade. Desta forma permitimos a representação de números negativos e a ordem deles continua crescente. O expoente é representado por um inteiro sem sinal do qual é deslocado o **BIAS**.

Tendo $|E|$ dígitos para representar o expoente, geralmente o $BIAS$ é predefinido de tal forma a dividir a tabela ao meio de tal forma que o expoente *um* seja representado pelo sequência [100...000].

Exemplo 2.2.7. Com 64 bits, pelo padrão $IEEE754$, temos que $|E| := 11$. Assim $(100\ 0000\ 0000)_2 = 2^{10} = 1024$. Como queremos que esta sequência represente o 1, definimos $BIAS := 1023$, pois

$$1024 - BIAS = 1.$$

Com 32 bits, temos $|E| := 8$ e $BIAS := 127$. E com 128 bits, temos $|E| := 15$ e $BIAS := 16383$.

Com 11 bits temos

$$\begin{aligned}
[111\ 1111\ 1111] &= reservado \\
[111\ 1111\ 1110] &= 2046 - BIAS = 1023_{10} = E_{MAX} \\
&\vdots = \\
[100\ 0000\ 0001] &= 2^{10} + 1 - BIAS = 2_{10} \\
[100\ 0000\ 0000] &= 2^{10} \quad - BIAS = 1_{10} \\
[011\ 1111\ 1111] &= 1023 \quad - BIAS = 0_{10} \\
[011\ 1111\ 1110] &= 1022 \quad - BIAS = -1_{10} \\
&\vdots = \\
[000\ 0000\ 0001] &= 1 - BIAS = -1022 = E_{MIN} \\
[000\ 0000\ 0000] &= reservado
\end{aligned}$$

O maior expoente é dado por $E_{MAX} = 1023$ e o menor expoente é dado por $E_{MIN} = -1022$.

O menor número representável positivo é dado pelo registro

$$[0|000\ 0000\ 0001|0000\ 0000\ 0000...0000\ 0000]$$

quando $s = 0$, $c = 1$ e $M = (1.000...000)_2$, ou seja,

$$MINR = (1 + 0)_2 \times 2^{1-1023} \approx 0.2225 \times 10^{-307}.$$

O maior número representável é dado por

$$[0|111\ 1111\ 1110|1111\ 1111\ ...1111\ 1111]$$

quando $s = 0$, $c = 2046$ e $M = (1.1111\ 1111....1111)_2 = 2 - 2^{-52}$, ou seja,

$$MAXR = (2 - 2^{-52}) \times 2^{2046-1023} \approx 2^{1024} \approx 0.17977 \times 10^{309}.$$

Casos especiais

O **zero** é um caso especial representado pelo registro

$$[0|000\ 0000\ 0000|0000\ 0000\ 0000...0000\ 0000]$$

Os expoentes **reservados** são usados para casos especiais:

- $c = [0000...0000]$ é usado para representar o zero (se $m = 0$) e os números subnormais (se $m \neq 0$).

- $c = [1111...1111]$ é usado para representar o infinito (se $m = 0$) e NaN (se $m \neq 0$).

Os números subnormais[3] tem a forma

$$x = (-1)^s(0.m_1m_2\cdots m_{51}m_{52})_2 \times 2^{1-BIAS}.$$

Para mais informações sobre aritmética de ponto flutuante em `Python`, veja a documentação.

2.2.5 A precisão e o epsilon de máquina

A **precisão** p de uma máquina é o número de dígitos significativos usado para representar um número. Note que $p = |M| + 1$ em binário e $p = |M|$ para outras bases.

O **epsilon de máquina**, $\epsilon_{mach} = \epsilon$, é definido de forma que $1 + \epsilon$ seja o menor número representável maior que 1, isto é, $1 + \epsilon$ é representável, mas não existem números representáveis em $(1, 1 + \epsilon)$.

Exemplo 2.2.8. Com 64 bits, temos que o epsilon será dado por

$$\begin{aligned} 1 \rightarrow \quad & (1.0000\ 0000....0000)_2 \times 2^0 \\ \epsilon \rightarrow + & (0.0000\ 0000....0001)_2 \times 2^0 = 2^{-52} \\ & (1.0000\ 0000....0001)_2 \times 2^0 \neq 1 \end{aligned}$$

Assim $\epsilon = 2^{-52}$.

2.2.6 A distribuição dos números

Utilizando uma máquina em ponto flutuante temos um número finito de números que podemos representar.

Um número muito pequeno geralmente é aproximado por zero (underflow) e um número muito grande (overflow) geralmente faz o cálculo parar. Além disso, os números não estão uniformemente espaçados no eixo real. Números pequenos estão bem próximos enquanto que números com expoentes grandes estão bem distantes.

Se tentarmos armazenar um número que não é representável, devemos utilizar o número mais próximo, gerando os erros de arredondamento.

Por simplicidade, a partir daqui nós adotaremos $b = 10$.

Exercícios

E 2.2.1. Explique a diferença entre o sistema de ponto fixo e ponto flutuante.

[3] Note que poderíamos definir números um pouco menores que o $MINR$.

2.3 Erros nas operações elementares

O erro relativo presente nas operações elementares de adição, subtração, multiplicação e divisão é da ordem do epsilon de máquina. Se estivermos usando uma máquina com 64 bits, temos que $\epsilon = 2^{-52} \approx 2{,}22E16$.

Este erro é bem pequeno! Assumindo que x e y são representados com todos dígitos corretos, temos aproximadamente 15 dígitos significativos corretos quando fazemos uma das operações $x + y$, $x - y$, $x \times y$ ou x/y.

Mesmo que fizéssemos, por exemplo, 1000 operações elementares em ponto flutuante sucessivas, teríamos, no pior dos casos, acumulado todos esses erros e perdido 3 casas decimais ($1000 \times 10^{-15} \approx 10^{-12}$).

Entretanto, quando subtraímos números muito próximos, os problemas aumentam.

2.4 Cancelamento catastrófico

Quando fazemos subtrações com números muito próximos entre si ocorre o cancelamento catastrófico, onde podemos perder vários dígitos de precisão em uma única subtração.

Exemplo 2.4.1. Efetue a operação

$$0{,}987624687925 - 0{,}987624 = 0{,}687925 \times 10^{-6}$$

usando arredondamento com seis dígitos significativos e observe a diferença se comparado com resultado sem arredondamento.

Solução. Os números arredondados com seis dígitos para a mantissa resultam na seguinte diferença

$$0{,}987625 - 0{,}987624 = 0{,}100000 \times 10^{-5}$$

Observe que os erros relativos entre os números exatos e aproximados no lado esquerdo são bem pequenos,

$$\frac{|0{,}987624687925 - 0{,}987625|}{|0{,}987624687925|} = 0{,}00003159$$

e

$$\frac{|0{,}987624 - 0{,}987624|}{|0{,}987624|} = 0\%,$$

enquanto no lado direito o erro relativo é enorme:

$$\frac{|0{,}100000 \times 10^{-5} - 0{,}687925 \times 10^{-6}|}{0{,}687925 \times 10^{-6}} = 45{,}36\%.$$

$\Diamond$

Exemplo 2.4.2. Considere o problema de encontrar as raízes da equação de segundo grau

$$x^2 + 300x - 0{,}014 = 0,$$

usando seis dígitos significativos.

Aplicando a fórmula de Bhaskara com $a = 0{,}100000 \times 10^1$, $b = 0{,}300000 \times 10^3$ e $c = 0{,}140000 \times 10^{-1}$, temos o discriminante:

$$\begin{aligned}
\Delta &= b^2 - 4 \cdot a \cdot c \\
&= 0{,}300000 \times 10^3 \times 0{,}300000 \times 10^3 \\
&+ 0{,}400000 \times 10^1 \times 0{,}100000 \times 10^1 \times 0{,}140000 \times 10^{-1} \\
&= 0{,}900000 \times 10^5 + 0{,}560000 \times 10^{-1} \\
&= 0{,}900001 \times 10^5
\end{aligned}$$

e as raízes:

$$\begin{aligned}
x_1,x_2 &= \frac{-0{,}300000 \times 10^3 \pm \sqrt{\Delta}}{0{,}200000 \times 10^1} \\
&= \frac{-0{,}300000 \times 10^3 \pm \sqrt{0{,}900001 \times 10^5}}{0{,}200000 \times 10^1} \\
&= \frac{-0{,}300000 \times 10^3 \pm 0{,}300000 \times 10^3}{0{,}200000 \times 10^1}
\end{aligned}$$

Então, as duas raízes são:

$$\begin{aligned}
\tilde{x}_1 &= \frac{-0{,}300000 \times 10^3 - 0{,}300000 \times 10^3}{0{,}200000 \times 10^1} \\
&= -\frac{0{,}600000 \times 10^3}{0{,}200000 \times 10^1} = -0{,}300000 \times 10^3
\end{aligned}$$

e

$$\tilde{x}_2 = \frac{-0{,}300000 \times 10^3 + 0{,}300000 \times 10^3}{0{,}200000 \times 10^1} = 0{,}000000 \times 10^0$$

Agora, os valores das raízes com seis dígitos significativos deveriam ser

$$x_1 = -0{,}300000 \times 10^3 \quad \text{e} \quad x_2 = 0{,}466667 \times 10^{-4}.$$

Observe que um raiz saiu com seis dígitos significativos corretos, mas a outra não possui nenhum dígito significativo correto.

Observação 2.4.1. No exemplo anterior b^2 é muito maior que $4ac$, ou seja, $b \approx \sqrt{b^2 - 4ac}$, logo a diferença

$$-b + \sqrt{b^2 - 4ac}$$

estará próxima de zero. Uma maneira padrão de evitar o cancelamento catastrófico é usar procedimentos analíticos para eliminar essa diferença. Abaixo veremos alguns exemplos.

Exemplo 2.4.3. Para eliminar o cancelamento catastrófico do exemplo anterior, usamos a seguinte expansão em série de Taylor em torno da origem

$$\sqrt{1-x} = 1 - \frac{1}{2}x + O(x^2).$$

Substituindo na fórmula de Bhaskara, temos:

$$\begin{aligned} x &= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \\ &= \frac{-b \pm b\sqrt{1 - \frac{4ac}{b^2}}}{2a} \\ &\approx \frac{-b \pm b\left(1 - \frac{4ac}{2b^2}\right)}{2a} \end{aligned}$$

Observe que $\frac{4ac}{b^2}$ é um número pequeno e por isso a expansão faz sentido. Voltamos no exemplo anterior e calculamos as duas raízes com o nova expressão

$$\begin{aligned} \tilde{x}_1 &= \frac{-b - b + \frac{4ac}{2b}}{2a} = -\frac{b}{a} + \frac{c}{b} \\ &= -\frac{0{,}300000 \times 10^3}{0{,}100000 \times 10^1} - \frac{0{,}140000 \times 10^{-1}}{0{,}300000 \times 10^3} \\ &= -0{,}300000 \times 10^3 - 0{,}466667 \times 10^{-4} \\ &= -0{,}300000 \times 10^3 \end{aligned}$$

$$\begin{aligned} \tilde{x}_2 &= \frac{-b + b - \frac{4ac}{2b}}{2a} \\ &= -\frac{4ac}{4ab} \\ &= -\frac{c}{b} = -\frac{-0{,}140000 \times 10^{-1}}{0{,}300000 \times 10^3} = 0{,}466667 \times 10^{-4} \end{aligned}$$

Observe que o efeito catastrófico foi eliminado.

2.5 Condicionamento de um problema

Nesta seção, utilizaremos a seguinte descrição abstrata para o conceito de "resolver um problema": dado um conjunto de dados de entrada, encontrar os dados de saída. Se denotamos pela variável x os dados de entrada e pela variável y os dados de saída, resolver o problema significa encontrar y dado x. Em termos matemáticos, a resolução de um problema é realizada pelo mapeamento $f : x \to y$, ou simplesmente $y = f(x)$.

É certo que na maioria das aplicações, os dados de entrada do problema, isto é x, não é conhecido com total exatidão, devido a diversas fontes de erros como incertezas na coleta dos dados e erros de arredondamento. O conceito de condicionamento está relacionado com a forma como os erros nos dados de entrada influenciam os dados de saída.

Para fins de análise, denotaremos por x, os dados de entrada com precisão absoluta e por x^*, os dados com erro. Definiremos também a solução y^*, do problema com dados de entrada x^*, ou seja, $y^* = f(x^*)$.

Estamos interessados em saber se os erros cometidos na entrada $\Delta x = x - x^*$ influenciaram na saída do problema $\Delta y = y - y^*$. No caso mais simples, temos que $x \in \mathbb{R}$ e $y \in \mathbb{R}$. Assumindo que f seja diferenciável, a partir da série de Taylor

$$f(x + \Delta x) \approx f(x) + f'(x)\Delta x$$

obtemos (subtraindo $f(x)$ dos dois lados)

$$\Delta y = f(x + \Delta x) - f(x) \approx f'(x)\Delta x$$

Para relacionarmos os erros relativos, dividimos o lado esquerdo por y, o lado direito por $f(x) = y$ e obtemos

$$\frac{\Delta y}{y} \approx \frac{f'(x)}{f(x)} \frac{x\Delta x}{x}$$

sugerindo a definição de número de condicionamento de um problema.

Definição 2.5.1. *Seja f uma função diferenciável. O* ***número de condicionamento*** *de um problema é definido como*

$$\kappa_f(x) := \left| \frac{xf'(x)}{f(x)} \right|$$

e fornece uma estimativa de quanto os erros relativos na entrada $\left|\frac{\Delta x}{x}\right|$ serão amplificados na saída $\left|\frac{\Delta y}{y}\right|$.

De modo geral, quando f depende de várias variáveis, podemos obter

$$\delta_f = |f(x_1,x_2,...,x_n) - f(\bar{x}_1,\bar{x}_2,...,\bar{x}_n)| \approx \sum_{i=1}^{n} \left|\frac{\partial f}{\partial x_i}(x_1,x_2,...,x_n)\right| \delta_{x_i}$$

Uma matriz de números de condicionamento também poderia ser obtida como em [5].

Exemplo 2.5.1. Considere o problema de calcular $\sqrt{x}$ em $x = 2$. Se usarmos $x^* = 1{,}999$, quanto será o erro relativo na saída? O erro relativo na entrada é

$$\left|\frac{\Delta x}{x}\right| = \left|\frac{2 - 1{,}999}{2}\right| = 0{,}0005$$

O número de condicionamento do problema calcular a raiz é

$$\kappa_f(x) := \left|\frac{xf'(x)}{f(x)}\right| = \left|\frac{x\frac{1}{2\sqrt{x}}}{\sqrt{x}}\right| = \frac{1}{2}$$

Ou seja, os erros na entrada serão diminuídos pela metade. De fato, usando $y = \sqrt{2} = 1{,}4142136...$ e $y^* = \sqrt{1{,}999} = 1{,}41386...$, obtemos

$$\frac{\Delta y}{y} = \frac{\sqrt{2} - \sqrt{1{,}999}}{\sqrt{2}} \approx 0{,}000250031...$$

Exemplo 2.5.2. Considere a função $f(x) = \frac{10}{1-x^2}$ e $x^* = 0{,}9995$ com um erro absoluto na entrada de 0,0001.

Calculando $y^* = f(x^*)$ temos

$$y^* = \frac{10}{1 - (0{,}9995)^2} \approx 10002{,}500625157739705173$$

Mas qual é a estimativa de erro nessa resposta? Quantos dígitos significativos temos nessa resposta?

Sabendo que $f'(x) = -10/(1-x^2)^2$, o número de condicionamento é

$$\kappa_f(x) := \left|\frac{xf'(x)}{f(x)}\right| = \left|\frac{2x^2}{1-x^2}\right|$$

o que nos fornece para $x^* = 0{,}9995$,

$$\kappa_f(0{,}9995) \approx 1998{,}5$$

Como o erro relativo na entrada é

$$\left|\frac{\Delta x}{x}\right| = \left|\frac{0{,}0001}{0{,}9995}\right| \approx 0{,}00010005...$$

temos que o erro na saída será aproximadamente

$$\left|\frac{\Delta y}{y}\right| \approx \kappa_f(x)\left|\frac{\Delta x}{x}\right| \approx 1998{,}5 \times 0{,}00010005... \approx 0{,}1999$$

ou seja um erro relativo de aproximadamente 19,99%.

Note que se usarmos $x_1 = 0{,}9994$ e $x_2 = 0{,}9996$ (ambos no intervalo do erro absoluto da entrada) encontramos

$$\begin{aligned} y_1^* &\approx 8335{,}83 \\ y_2^* &\approx 12520{,}50 \end{aligned}$$

confirmando a estimativa de 19,99%.

Exemplo 2.5.3. Seja $f(x) = x\exp(x)$. Calcule o erro absoluto em se calcular $f(x)$ sabendo que $x = 2 \pm 0{,}05$.

Solução. Temos que $x \approx 2$ com erro absoluto de $\delta_x = 0{,}05$. Neste caso, calculamos δ_f, i.e. o erro absoluto em se calcular $f(x)$, por:

$$\delta_f = |f'(x)|\delta_x.$$

Como $f'(x) = (1+x)e^x$, temos:

$$\begin{aligned} \delta_f &= |(1+x)e^x| \cdot \delta_x \\ &= |3e^2| \cdot 0{,}05 = 1{,}1084. \end{aligned}$$

Portanto, o erro absoluto em se calcular $f(x)$ quando $x = 2 \pm 0{,}05$ é de 1,084. $\Diamond$

Exemplo 2.5.4. Calcule o erro relativo ao medir $f(x,y) = \frac{x^2+1}{x^2}e^{2y}$ sabendo que $x \approx 3$ é conhecido com 10% de erro e $y \approx 2$ é conhecido com 3% de erro.

Solução. Calculamos as derivadas parciais de f:

$$\frac{\partial f}{\partial x} = \frac{2x^3 - (2x^3+2x)}{x^4}e^{2y} = -\frac{2e^{2y}}{x^3}$$

e

$$\frac{\partial f}{\partial y} = 2\frac{x^2+1}{x^2}e^{2y}$$

Calculamos o erro absoluto em termos do erro relativo:

$$\frac{\delta_x}{|x|} = 0{,}1 \Rightarrow \delta_x = 3 \cdot 0{,}1 = 0{,}3$$

$$\frac{\delta_y}{|y|} = 0{,}03 \Rightarrow \delta_y = 2 \cdot 0{,}03 = 0{,}06$$

Aplicando a expressão para estimar o erro em f temos

$$\begin{aligned} \delta_f &= \left|\frac{\partial f}{\partial x}\right| \delta_x + \left|\frac{\partial f}{\partial y}\right| \delta_y \\ &= \frac{2e^4}{27} \cdot 0{,}3 + 2\frac{9+1}{9}e^4 \cdot 0{,}06 = 8{,}493045557 \end{aligned}$$

Portanto, o erro relativo ao calcular f é estimado por

$$\frac{\delta f}{|f|} = \frac{8{,}493045557}{\frac{9+1}{9}e^4} = 14\%$$

$\Diamond$

Exemplo 2.5.5. No exemplo anterior, reduza o erro relativo em x pela metade e calcule o erro relativo em f. Depois, repita o processo reduzindo o erro relativo em y pela metade.

Solução. Na primeira situação temos $x = 3$ com erro relativo de 5% e $\delta_x = 0{,}05 \cdot 3 = 0{,}15$. Calculamos $\delta_f = 7{,}886399450$ e o erro relativo em f de 13%. Na segunda situação, temos $y = 2$ com erro de 1,5% e $\delta_y = 2 \cdot 0{,}015 = 0{,}03$. Calculamos $\delta_f = 4{,}853168892$ e o erro relativo em f de 8%. Observe que mesma o erro relativo em x sendo maior, o erro em y é mais significante na função. $\Diamond$

Exemplo 2.5.6. Considere um triângulo retângulo onde a hipotenusa e um dos catetos são conhecidos a menos de um erro: hipotenusa $a = 3 \pm 0{,}01$ metros e cateto $b = 2 \pm 0{,}01$ metros. Calcule o erro absoluto ao calcular a área dessa triângulo.

Solução. Primeiro vamos encontrar a expressão para a área em função da hipotenusa a e um cateto b. A tamanho de segundo cateto c é dado pelo teorema de Pitágoras, $a^2 = b^2 + c^2$, ou seja, $c = \sqrt{a^2 - b^2}$. Portanto a área é

$$A = \frac{bc}{2} = \frac{b\sqrt{a^2 - b^2}}{2}.$$

Agora calculamos as derivadas

$$\frac{\partial A}{\partial a} = \frac{ab}{2\sqrt{a^2 - b^2}},$$

$$\frac{\partial A}{\partial b} = \frac{\sqrt{a^2 - b^2}}{2} - \frac{b^2}{2\sqrt{a^2 - b^2}},$$

e substituindo na estimativa para o erro δ_A em termos de $\delta_a = 0{,}01$ e $\delta_b = 0{,}01$:

$$\begin{aligned}\delta_A &\approx \left|\frac{\partial A}{\partial a}\right|\delta_a + \left|\frac{\partial A}{\partial b}\right|\delta_b \\ &\approx \frac{3\sqrt{5}}{5}\cdot 0{,}01 + \frac{\sqrt{5}}{10}\cdot 0{,}01 = 0{,}01565247584\end{aligned}$$

Em termos do erro relativo temos erro na hipotenusa de $\frac{0{,}01}{3} \approx 0{,}333\%$, erro no cateto de $\frac{0{,}01}{2} = 0{,}5\%$ e erro na área de

$$\frac{0{,}01565247584}{\frac{2\sqrt{3^2-2^2}}{2}} = 0{,}7\%$$

$\Diamond$

Exercícios

E 2.5.1. Considere que a variável $x \approx 2$ é conhecida com um erro relativo de 1% e a variável $y \approx 10$ com um erro relativo de 10%. Calcule o erro relativo associado a z quando:

$$z = \frac{y^4}{1+y^4}e^x.$$

Suponha que você precise conhecer o valor de z com um erro de 0,5%. Você propõe uma melhoria na medição da variável x ou y? Explique.

E 2.5.2. A corrente I em ampères e a tensão V em volts em uma lâmpada se relacionam conforme a seguinte expressão:

$$I = \left(\frac{V}{V_0}\right)^{\alpha},$$

onde α é um número entre 0 e 1 e V_0 é tensão nominal em volts. Sabendo que $V_0 = 220 \pm 3\%$ e $\alpha = -0{,}8 \pm 4\%$, calcule a corrente e o erro relativo associado quando a tensão vale $220 \pm 1\%$.
Obs:. Este problema pode ser resolvido de duas formas distintas: usando a expressão aproximada para a propagação de erro e inspecionando os valores máximos e mínimos que a expressão pode assumir. Pratique os dois métodos.

E 2.5.3. A corrente I em ampères e a tensão V em volts em uma lâmpada se relacionam conforme a seguinte expressão:

$$I = \left(\frac{V}{V_0}\right)^{\alpha}$$

Onde α é um número entre 0 e 1 e V_0 é a tensão nominal em volts. Sabendo que $V_0 = 220 \pm 3\%$ e $\alpha = 0{,}8 \pm 4\%$ Calcule a corrente e o erro relativo associado quando a tensão vale $220 \pm 1\%$. **Dica:** lembre que $x^\alpha = e^{\alpha \ln(x)}$

2.6 Mais exemplos de Cancelamento Catastrófico

Exemplo 2.6.1. Considere o seguinte processo iterativo:

$$\begin{cases} x_0 = \frac{1}{3} \\ x_{n+1} = 4x_n - 1, \qquad n \in \mathbb{N} \end{cases}.$$

Observe que $x_0 = \frac{1}{3}$, $x_1 = 4 \cdot \frac{1}{3} - 1 = \frac{1}{3}$, $x_2 = \frac{1}{3}$, ou seja, temos uma sequência constante igual a $\frac{1}{3}$. No entanto, ao calcularmos no computador, usando o sistema de numeração 'double', a sequencia obtida não é constante e, de fato, diverge. Faça o teste em `Python`, colocando:

```
>>> x = 1/3; x
```

e itere algumas vezes a linha de comando:

```
>>> x = 4*x-1; x
```

Para compreender o que acontece, devemos levar em consideração que o número $\frac{1}{3} = 0{,}\overline{3}$ possui um representação infinita tanto na base decimal quanto na base binária. Logo, sua representação de máquina incliu um erro de arredondamento. Seja ϵ a diferença entre o valor exato de $\frac{1}{3}$ e sua representação de máquina, isto é, $\tilde{x}_0 = \frac{1}{3} + \epsilon$. A sequência efetivamente calculada no computador é:

$$\begin{aligned} \tilde{x}_0 &= \frac{1}{3} + \epsilon \\ \tilde{x}_1 &= 4x_0 - 1 = 4\left(\frac{1}{3} + \epsilon\right) - 1 = \frac{1}{3} + 4\epsilon \\ \tilde{x}_2 &= 4x_1 - 1 = 4\left(\frac{1}{3} + 4\epsilon\right) - 1 = \frac{1}{3} + 4^2\epsilon \\ &\vdots \\ \tilde{x}_n &= \frac{1}{3} + 4^n\epsilon \end{aligned}$$

Portanto o limite da sequência diverge,

$$\lim_{x \to \infty} |\tilde{x}_n| = \infty$$

Qual o número de condicionamento desse problema?

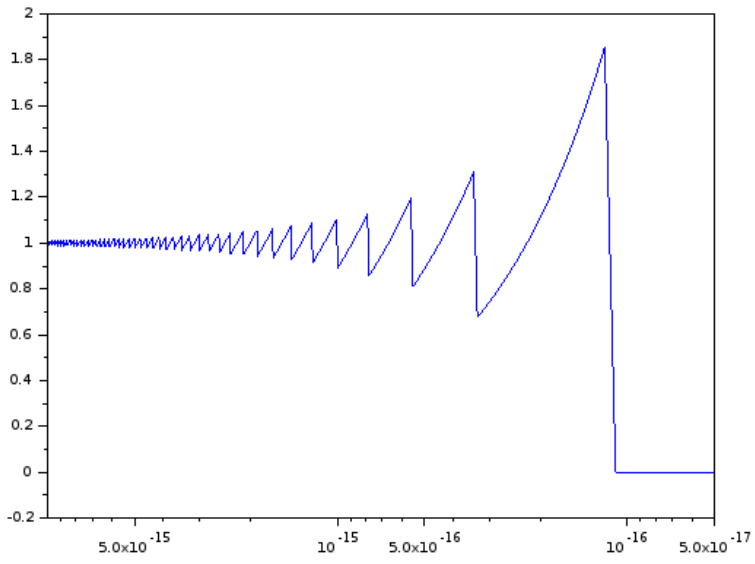


Figura 2.1: Gráfico na função do Exemplo 2.6.2.

Exemplo 2.6.2. Observe a seguinte identidade

$$f(x) = \frac{(1+x)-1}{x} = 1$$

Calcule o valor da expressão à esquerda para $x = 10^{-12}$, $x = 10^{-13}$, $x = 10^{-14}$, $x = 10^{-15}$, $x = 10^{-16}$ e $x = 10^{-17}$. Observe que quando x se aproxima do ϵ de máquina a expressão perde o significado. Veja a Figura 2.1 com o gráfico de $f(x)$ em escala logarítmica.

Exemplo 2.6.3. Neste exemplo, estamos interessados em compreender mais detalhadamente o comportamento da expressão

$$\left(1+\frac{1}{n}\right)^n$$

quando n é um número grande ao computá-la em sistemas de numeral de ponto flutuante com acurácia finita. Um resultado bem conhecido do cálculo nos diz que o limite de (2.6.3) quando n tende a infinito é o número de Euler:

$$\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n = e = 2{,}718281828459...$$

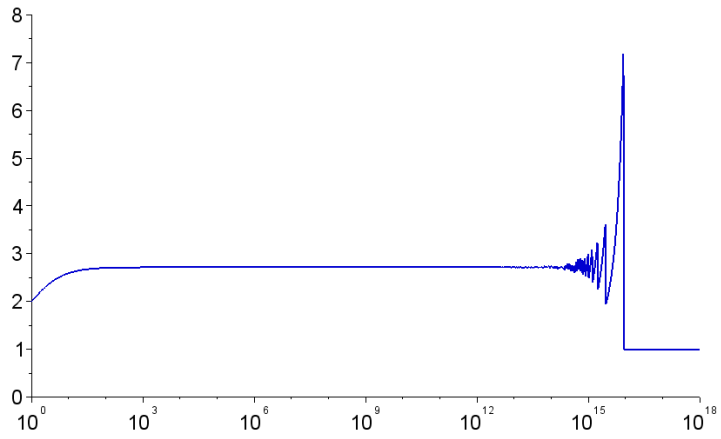


Figura 2.2: Gráfico de $\left(1+\frac{1}{n}\right)^n$ em função de n em escala linear-logarítmica variando de 10^0 até 10^{18}. Veja o Exemplo 2.6.3.

Sabemos também que a sequência produzida por (2.6.3) é crescente, isto é:

$$\left(1+\frac{1}{1}\right)^1 < \left(1+\frac{1}{2}\right)^2 < \left(1+\frac{1}{3}\right)^3 < \cdots$$

No entanto, quando calculamos essa expressão no `Scilab`, nos defrontamos

com o seguinte resultado:

n	$\left(1+\frac{1}{n}\right)^n$		n	$\left(1+\frac{1}{n}\right)^n$
1	2,0000000000000		10^2	2,7048138294215
2	2,2500000000000		10^4	2,7181459268249
3	2,3703703703704		10^6	2,7182804690957
4	2,4414062500000		10^8	2,7182817983391
5	2,4883200000000		10^{10}	2,7182820532348
6	2,5216263717421		10^{12}	2,7185234960372
7	2,5464996970407		10^{14}	2,7161100340870
8	2,5657845139503		10^{16}	1,0000000000000
9	2,5811747917132		10^{18}	1,0000000000000
10	2,5937424601000		10^{20}	1,0000000000000

Podemos resumir esses dados no gráfico de $\left(1+\frac{1}{n}\right)^n$ em função de n, veja a Figura 2.2.

Observe que quando n se torna grande, da ordem de 10^{15}, o gráfico da função deixa de se crescente e apresenta oscilações. Observe também que a expressão se torna identicamente igual a 1 depois de um certo limiar. Tais fenômenos não são intrínsecos da função $f(n) = (1 + 1/n)^n$, mas **oriundas de erros de arredondamento**, isto é, são resultados numéricos espúrios. A fim de pôr o comportamento numérico de tal expressão, apresentamos abaixo o gráfico da mesma função, porém restrito à região entre 10^{14} e 10^{16}.

Para compreendermos melhor por que existe um limiar N que, quando atingido torna a expressão do exemplo acima identicamente igual a 1, observamos a sequência de operações realizadas pelo computador:

$$n \;\to\; 1/n \;\to\; 1+1/n \;\to\; (1+1/n)^n \tag{2.1}$$

Devido ao limite de precisão da representação de números em ponto flutuante, existe um menor número representável que é maior do que 1. Este número é 1+`eps`, onde `eps` é chamado de **épsilon de máquina** e é o menor número que

somado a 1 produz um resultado superior a 1 no sistema de numeração usado. O épsilon de máquina no sistema de numeração **double** vale aproximadamente $2{,}22 \times 10^{-16}$. Em `Python` podemos obter o epsilon de máquina com o seguinte comando `numpy`:

```
>>> eps = np.finfo(float).eps
>>> print(eps)
2.22044604925e-16
>>> 1+eps == 1
False
>>> 1+eps
1.0000000000000002
```

Quando somamos a 1 um número positivo inferior ao épsilon de máquina, obtemos o número 1. Dessa forma, o resultado obtido pela operação de ponto flutuante $1 + n$ para $0 < n < 2{,}22 \times 10^{-16}$ é 1.

Portanto, quando realizamos a sequência de operações dada em (2.1), toda informação contida no número n é perdida na soma com 1 quando $1/n$ é menor que o épsilon de máquina, o que ocorre quando $n > 5 \times 10^{15}$. Assim $(1 + 1/n)$ é aproximado para 1 e a última operação se resume a 1^n, o que é igual a 1 mesmo quando n é grande.

Um erro comum é acreditar que o perda de significância se deve ao fato de $1/n$ ser muito pequeno para ser representado e é aproximando para 0. Isto é falso, o sistema de ponto de flutuante permite representar números de magnitude muito inferior ao épsilon de máquina. O problema surge da limitação no tamanho da mantissa. Observe como a seguinte sequência de operações não perde significância para números positivos x muito menores que o épsilon de máquina:

$$n \;\to\; 1/n \;\to\; 1/(1/n) \tag{2.2}$$

compare o desempenho numérico desta sequência de operações para valores pequenos de n com o da seguinte sequência:

$$n \;\to\; 1+n \;\to\; (1+n)-1. \tag{2.3}$$

Finalmente, notamos que quando tentamos calcular $\left(1+\frac{1}{n}\right)^n$ para n grande, existe perda de significância no cálculo de $1 + 1/n$.

Exemplo 2.6.4 (Analogia da balança)**.** Observe a seguinte comparação interessante que pode ser feita para ilustrar os sistemas de numeração com ponto fixo e flutuante: o sistema de ponto fixo é como uma balança cujas marcas estão igualmente espaçadas; o sistema de ponto flutuante é como uma balança cuja distância

entre as marcas é proporcional à massa medida. Assim, podemos ter uma balança de ponto fixo cujas marcas estão sempre distanciadas de 100g (100g, 200g, 300g, ..., 1Kg, 1,1Kg,...) e outra balança de ponto flutuante cujas marcas estão distanciadas sempre de aproximadamente um décimo do valor lido (100g, 110g, 121g, 133g, ..., 1Kg, 1,1Kg, 1,21Kg, ...) A balança de ponto fixo apresenta uma resolução baixa para pequenas medidas, porém uma resolução alta para grandes medidas. A balança de ponto flutuante distribui a resolução de forma proporcional ao longo da escala.

Seguindo nesta analogia, o fenômeno de perda de significância pode ser interpretado como a seguir: imagine que você deseje obter o peso de um gato (aproximadamente 4Kg). Dois processos estão disponíveis: colocar o gato diretamente na balança ou medir seu peso com o gato e, depois, sem o gato. Na balança de ponto flutuante, a incerteza associada na medida do peso do gato (sozinho) é aproximadamente 10% de 4Kg, isto é, 400g. Já a incerteza associada à medida da uma pessoa (aproximadamente 70Kg) com o gato é de 10% do peso total, isto é, aproximadamente 7Kg. Esta incerteza é da mesma ordem de grandeza da medida a ser realizada, tornado o processo impossível de ser realizado, já que teríamos uma incerteza da ordem de 14Kg (devido à dupla medição) sobre uma grandeza de 4Kg.

Exercícios

E 2.6.1. Considere as expressões:

$$\frac{\exp(1/\mu)}{1+\exp(1/\mu)}$$

e

$$\frac{1}{\exp(-1/\mu)+1}$$

com $\mu > 0$. Verifique que elas são idênticas como funções reais. Teste no computador cada uma delas para $\mu = 0{,}1$, $\mu = 0{,}01$ e $\mu = 0{,}001$. Qual dessas expressões é mais adequada quando μ é um número pequeno? Por quê?

E 2.6.2. Encontre expressões alternativas para calcular o valor das seguintes funções quando x é próximo de zero.

a) $f(x) = \frac{1-\cos(x)}{x^2}$

b) $g(x) = \sqrt{1+x} - 1$

c) $h(x) = \sqrt{x+10^6} - 10^3$

d) $i(x) = \sqrt{1+e^x} - \sqrt{2}$ Dica: Faça $y = e^x - 1$

E 2.6.3. Use uma identidade trigonométrica adequada para mostrar que:

$$\frac{1-\cos(x)}{x^2} = \frac{1}{2}\left(\frac{\sin(x/2)}{x/2}\right)^2.$$

Analise o desempenho destas duas expressões no computador quando x vale 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-200} e 0. Discuta o resultado. **Dica:** Para $|x| < 10^{-5}$, $f(x)$ pode ser aproximada por $1/2 - x^2/24$ com erro de truncamento inferior a 10^{-22}.

E 2.6.4. Reescreva as expressões:

$$\sqrt{e^{2x}+1} - e^x \qquad \text{e} \qquad \sqrt{e^{2x}+x^2} - e^x$$

de modo que seja possível calcular seus valores para $x = 100$ utilizando a aritmética de ponto flutuante ("Double") no computador.

E 2.6.5. Na teoria da relatividade restrita, a energia cinética de uma partícula e sua velocidade se relacionam pela seguinte fórmula:

$$E = mc^2\left(\frac{1}{\sqrt{1-(v/c)^2}} - 1\right),$$

onde E é a energia cinética da partícula, m é a massa de repouso, v o módulo da velocidade e c a velocidade da luz no vácuo dada por $c = 299792458m/s$. Considere que a massa de repouso $m = 9{,}10938291 \times 10^{-31}Kg$ do elétron seja conhecida com erro relativo de 10^{-9}. Qual é o valor da energia e o erro relativo associado a essa grandeza quando $v = 0{,}1c$, $v = 0{,}5c$, $v = 0{,}99c$ e $v = 0{,}999c$ sendo que a incerteza relativa na medida da velocidade é 10^{-5}?

E 2.6.6. Deseja-se medir a concentração de dois diferentes oxidantes no ar. Três sensores eletroquímicos estão disponíveis para a medida e apresentam a seguintes respostas:

$$v_1 = 270[A] + 30[B], \quad v_2 = 140[A] + 20[B] \quad \text{e} \quad v_3 = 15[A] + 200[B]$$

as tensões v_1, v_2 e v_3 são dadas em mV e as concentrações em $milimol/l$.

a) Encontre uma expressão para os valores de $[A]$ e $[B]$ em termos de v_1 e v_2 e, depois, em termos de v_1 e v_3. Dica: Se $ad \neq bc$, então a matriz A dada por

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

é inversível e sua inversa é dada por

$$A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

b) Sabendo que incerteza relativa associada às sensibilidades dos sensores 1 e 2 é de 2% e que a incerteza relativa associada às sensibilidades do sensor 3 é 10%, verifique a incerteza associada à medida feita com o par $1 - 2$ e o par $1 - 3$. Use $[A] = [B] = 10 milimol/l$. Dica: Você deve diferenciar as grandezas $[A]$ e $[B]$ em relação aos valores das tensões.

Capítulo 3

Solução de equações de uma variável

Neste capítulo buscaremos aproximações numéricas para a solução de **equações de uma variável real**. Observamos que obter uma solução para uma tal dada equação é equivalente a encontrar um **zero de uma função** apropriada. Com isso, iniciamos este capítulo discutindo sobre condições de existência e unicidade de raízes de funções de uma variável real. Então, apresentamos o **método da bisseção** como uma primeira abordagem numérica para a solução de tais equações.

Em seguida, exploramos uma outra abordagem via **iteração do ponto fixo**. Desta, obtemos o **método de Newton**[1], para o qual discutimos sua aplicação e convergência. Por fim, apresentamos o **método das secantes** como uma das possíveis variações do método de Newton.

Ao longo do capítulo, apresentamos algumas computações com `Python`. Nestas, estaremos assumindo o que os seguintes módulos estão carregados:

```
>>> from __future__ import division
>>> import numpy as np
>>> import matplotlib.pyplot as plt
```

A segunda instrução carrega a biblioteca de computação científica numpy e a terceira carrega a biblioteca gráfica matplotlib.

3.1 Existência e unicidade

O **teorema de Bolzano**[2] nos fornece condições suficientes para a existência do zero de uma função. Este é uma aplicação direta do **teorema do valor**

[1] Sir Isaac Newton, 1642 - 1727, matemático e físico inglês.

[2] Bernhard Placidus Johann Gonzal Nepomuk Bolzano, 1781 - 1848, matemático do Reino da Boêmia.

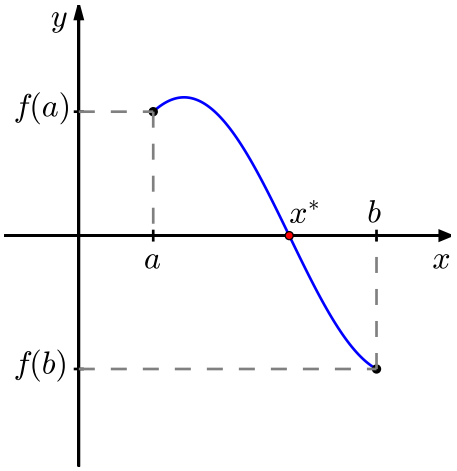


Figura 3.1: Teorema de Bolzano.

intermediário.

Teorema 3.1.1 (Teorema de Bolzano)**.** *Se $f : [a, b] \to \mathbb{R}$, $y = f(x)$, é uma função contínua tal que $f(a) \cdot f(b) < 0$, então existe $x^* \in (a, b)$ tal que $f(x^*) = 0$.*

Demonstração. O resultado é uma consequência imediata do teorema do valor intermediário que estabelece que dada uma função contínua $f : [a, b] \to \mathbb{R}$, $y = f(x)$, tal que $f(a) < f(b)$ (ou $f(b) < f(a)$), então para qualquer $d \in (f(a), f(b))$ (ou $k \in (f(b), f(a))$) existe $x^* \in (a, b)$ tal que $f(x^*) = k$. Ou seja, nestas notações, se $f(a) \cdot f(b) < 0$, então $f(a) < 0 < f(b)$ (ou $f(b) < 0 < f(a)$). Logo, tomando $k = 0$, temos que existe $x^* \in (a, b)$ tal que $f(x^*) = k = 0$. □

Em outras palavras, se $f(x)$ é uma função contínua em um dado intervalo no qual ela troca de sinal, então ela têm pelo menos um zero neste intervalo (veja a Figura 3.1).

Exemplo 3.1.1. Mostre que existe pelo menos uma solução da equação $e^x = x+2$ no intervalo $(-2,0)$.

Solução. Primeiramente, observamos que resolver a equação $e^x = x + 2$ é equivalente a resolver $f(x) = 0$ com $f(x) = e^x - x - 2$. Agora, como $f(-2) = e^{-2} > 0$ e $f(0) = -2 < 0$, temos do teorema de Bolzano que existe pelo menos um zero de $f(x)$ no intervalo $(-2, 0)$. E, portanto, existe pelo menos uma solução da equação dada no intervalo $(-2, 0)$.

Podemos usar `Python` para estudarmos esta função. Por exemplo, podemos definir a função $f(x)$ e computá-la nos extremos do intervalo dado com os seguintes comandos:

```
>>> def f(x): return np.exp(x)-x-2
...
>>> f(-2),f(0)
(0.13533528323661281, -1.0)
```

Alternativamente (e com maior precisão), podemos verificar diretamente o sinal da função nos pontos desejados com a função numpy.sign:

```
>>> np.sign(f(-2)*f(0))
-1.0
```

$\Diamond$

Quando procuramos aproximações para zeros de funções, é aconselhável isolar cada raiz em um intervalo. Desta forma, gostaríamos de poder garantir a existência e a unicidade da raiz dentro de um dado intervalo. A seguinte proposição nos fornece condições suficientes para tanto.

Proposição 3.1.1. *Se $f : [a,b] \to \mathbb{R}$ é um função diferenciável, $f(a) \cdot f(b) < 0$ e $f'(x) > 0$ (ou $f'(x) < 0$) para todo $x \in (a,b)$, então existe um único $x^* \in (a,b)$ tal que $f(x^*) = 0$.*

Em outras palavras, para garantirmos que exista um único zero de uma dada função diferenciável num intervalo, é suficiente que ela troque de sinal e seja monótona neste intervalo.

Exemplo 3.1.2. No Exemplo 3.1.1, mostramos que existe pelo menos um zero de $f(x) = e^x - x - 2$ no intervalo $(-2{,}0)$, pois $f(x)$ é contínua e $f(-2) \cdot f(0) < 0$. Agora, observamos que, além disso, $f'(x) = e^x - 1$ e, portanto, $f'(x) < 0$ para todo $x \in (-2{,}0)$. Logo, da Proposição 3.1.1, temos garantida a existência de um único zero no intervalo dado.

Podemos inspecionar o comportamento da função $f(x) = e^x - x - 2$ e de sua derivada fazendo seus gráficos no Scilab. Para tanto, podemos usar o seguinte código Python:

```
>>> def f(x): return np.exp(x)-x-2
...
>>> xx = np.linspace(-2,0)
>>> plt.plot(xx,f(xx))
>>> plt.grid(); plt.show()

>>> def fl(x): return np.exp(x)-1
...
>>> plt.plot(xx,fl(xx))
>>> plt.grid(); plt.show()
```

A discussão feita nesta seção, especialmente o teorema de Bolzano, nos fornece os fundamentos para o método da bisseção, o qual discutimos na próxima seção.

Exercícios

E 3.1.1. Mostre que $\cos x = x$ tem solução no intervalo $[0, \pi/2]$.

E 3.1.2. Mostre que $\cos x = x$ tem uma única solução no intervalo $[0, \pi/2]$.

E 3.1.3. Interprete a equação $\cos(x) = kx$ como o problema de encontrar a intersecção da curva $y = \cos(x)$ com $y = kx$. Encontre o valor positivo k para o qual essa equação admite exatamente duas raízes positivas distintas.

E 3.1.4. Mostre que a equação:

$$\ln(x) + x^3 - \frac{1}{x} = 10$$

possui uma única solução positiva.

E 3.1.5. Use o teorema de Bolzano para mostrar que o erro absoluto ao aproximar o zero da função $f(x) = e^x - x - 2$ por $\overline{x} = -1{,}841$ é menor que 10^{-3}.

E 3.1.6. Mostre que o erro absoluto associado à aproximação $\overline{x} = 1{,}962$ para a solução exata x^* de:

$$e^x + \sin(x) + x = 10$$

é menor que 10^{-4}.

E 3.1.7. Mostre que a equação

$$\ln(x) + x - \frac{1}{x} = v$$

possui uma solução para cada v real e que esta solução é única.

3.2 Método da bisseção

O **método da bisseção** explora o fato de que uma função contínua $f : [a, b] \to \mathbb{R}$ com $f(a) \cdot f(b) < 0$ tem um zero no intervalo (a, b) (veja o teorema de Bolzano 3.1.1). Assim, a ideia para aproximar o zero de uma tal função $f(x)$ é tomar, como primeira aproximação, o ponto médio do intervalo $[a, b]$, i.e.:

$$x^{(0)} = \frac{(a+b)}{2}.$$

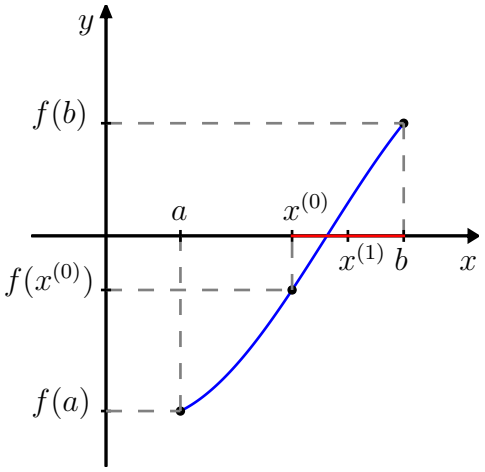


Figura 3.2: Método da bisseção.

Pode ocorrer de $f(x^{(0)}) = 0$ e, neste caso, o zero de $f(x)$ é $x^* = x^{(0)}$. Caso contrário, se $f(a) \cdot f(x^{(0)}) < 0$, então $x^* \in (a, x^{(0)})$. Neste caso, tomamos como segunda aproximação do zero de $f(x)$ o ponto médio do intervalo $[a, x^{(0)}]$, i.e. $x^{(1)} = (a + x^{(0)})/2$. Noutro caso, temos $f(x^{(0)}) \cdot f(b) < 0$ e, então, tomamos $x^{(1)} = (x^{(0)} + b)/2$. Repetimos este procedimento até obtermos a aproximação desejada (veja, Figura 3.2).

De forma mais precisa, suponha que queiramos calcular uma aproximação com uma certa precisão TOL para um zero x^* de uma dada função contínua $f : [a, b] \to \mathbb{R}$ tal que $f(a) \cdot f(b) < 0$. Iniciamos, tomando $n = 0$ e:

$$a^{(n)} = a, \quad b^{(n)} = b \quad \text{e} \quad x^{(n)} = \frac{a^{(n)} + b^{(n)}}{2}.$$

Verificamos o **critério de parada**, i.e. se $f(x^{(n)}) = 0$ ou:

$$\frac{|b^{(n)} - a^{(n)}|}{2} < TOL,$$

então $x^{(n)}$ é a aproximação desejada. Caso contrário, preparamos a próxima iteração $n+1$ da seguinte forma: se $f(a^{(n)}) \cdot f(x^{(n)}) < 0$, então setamos $a^{(n+1)} = a^{(n)}$ e $b^{(n+1)} = x^{(n)}$; noutro caso, se $f(x^{(n)}) \cdot f(b^{(n)}) < 0$, então setamos $a^{(n+1)} = x^{(n)}$ e $b^{(n+1)} = b^{(n)}$. Trocando n por $n + 1$, temos a nova aproximação do zero de $f(x)$ dada por:

$$x^{(n+1)} = \frac{a^{(n+1)} + b^{(n+1)}}{2}.$$

Tabela 3.1: Iteração do método da bisseção para o Exemplo 3.2.1.

n	$a^{(n)}$	$b^{(n)}$	$x^{(n)}$	$f(a^{(n)})f(x^{(n)})$	$\frac{\lvert b^{(n)} - a^{(n)}\rvert}{2}$
0	-2	0	-1	< 0	1
1	-2	-1	$-1{,}5$	< 0	0,5
2	-2	$-1{,}5$	$-1{,}75$	< 0	0,25
3	-2	$-1{,}75$	$-1{,}875$	> 0	0,125
4	$-1{,}875$	$-1{,}75$	$-1{,}8125$	< 0	0,0625

Voltamos a verificar o critério de parada acima e, caso não satisfeito, iteramos novamente. Iteramos até obtermos a aproximação desejada ou o número máximo de iterações ter sido atingido.

Exemplo 3.2.1. Use o método da bisseção para calcular uma solução de $e^x = x+2$ no intervalo $[-2, 0]$ com precisão $TOL = 10^{-1}$.

Solução. Primeiramente, observamos que resolver a equação dada é equivalente a calcular o zero de $f(x) = e^x - x - 2$. Além disso, temos $f(-2) \cdot f(0) < 0$. Desta forma, podemos iniciar o método da bisseção tomando o intervalo inicial $[a^{(0)}, b^{(0)}] = [-2, 0]$ e:

$$x^{(0)} = \frac{a^{(0)} + b^{(0)}}{2} = -1.$$

Apresentamos as iterações na Tabela 3.1. Observamos que a precisão $TOL = 10^{-1}$ foi obtida na quarta iteração com o zero de $f(x)$ sendo aproximado por $x^{(4)} = 1{,}8125$.

Usando `Python` neste exemplos, temos:

```
>>> def f(x): return np.exp(x) - x - 2
...
>>> a=-2; b=0; x = (a+b)/2; [a,b,x]
[-2, 0, -1.0]
>>> [(b-a)/2, np.sign(f(a)*f(x))]
[1.0, -1.0]
>>> b=x; x=(a+b)/2; [a,b,x]
[-2, -1.0, -1.5]
>>> [(b-a)/2, np.sign(f(a)*f(x))]
```

e, assim, sucessivamente. Veja o código completo na Seção 3.2.1. ◊

Vamos, agora, discutir sobre a **convergência** do método da bisseção. O próximo Teorema 3.2.1 nos garante a convergência do método da bisseção.

Teorema 3.2.1 (Convergência do método da bisseção)**.** *Sejam $f : [a, b] \to \mathbb{R}$ uma função contínua tal que $f(a) \cdot f(b) < 0$ e x^* o único zero de $f(x)$ no intervalo (a, b). Então, a sequência $\{x^{(n)}\}_{n>=0}$ do método da bisseção satisfaz:*

$$|x^{(n)} - x^*| < \frac{b-a}{2^{n+1}}, \quad \forall n \geq 0,$$

i.e., $x^{(n)} \to x^*$ quando $n \to \infty$.

Demonstração. Notemos que, a cada iteração, a distância entre a aproximação $x^{(n)}$ e o zero x^* da função é menor que a metade do tamanho do intervalo $[a^{(n)}, b^{(n)}]$ (veja Figura 3.2), i.e.:

$$|x^{(n)} - x^*| < \frac{b^{(n)} - a^{(n)}}{2}.$$

Por construção do método, temos $[a^{(n)}, b^{(n)}] \subset [a^{(n-1)}, b^{(n-1)}]$ e:

$$b^{(n)} - a^{(n)} = \frac{b^{(n-1)} - a^{(n-1)}}{2}.$$

Desta forma:

$$|x^{(n)} - x^*| < \frac{b^{(n)} - a^{(n)}}{2} = \frac{b^{(n-1)} - a^{(n-1)}}{2^2} = \cdots = \frac{b^{(0)} - a^{(0)}}{2^{n+1}}, \quad \forall n \geq 1.$$

Logo, vemos que:

$$|x^{(n)} - x^*| < \frac{b-a}{2^{n+1}}, \quad \forall n \geq 0.$$

□

Observamos que a hipótese de que $f(x)$ tenha um único zero no intervalo não é necessária. Se a função tiver mais de um zero no intervalo inicial, as iterações irão convergir para um dos zeros. Veja o Exercício 3.2.3.

Observação 3.2.1. O Teorema 3.2.1 nos fornece uma estimativa para a convergência do método da bisseção. Aproximadamente, temos:

$$|x^{(n+1)} - x^*| \lesssim \frac{1}{2}|x^{(n+1)} - x^*|.$$

Isto nos leva a concluir que o método da bisseção tem **taxa de convergência** linear.

Exemplo 3.2.2. No Exemplo 3.2.1, precisamos de 4 iterações do método da bisseção para computar uma aproximação com precisão de 10^{-1} do zero de $f(x) = e^x - x - 2$ tomando como intervalo inicial $[a, b] = [-2, 0]$. Poderíamos ter estimado o número de iterações **a priori**, pois, como vimos acima:

$$|x^{(n)} - x^*| \leq \frac{b-a}{2^{n+1}}, \quad n \geq 0.$$

Logo, temos:

$$\begin{aligned} |x^{(n)} - x^*| &< \frac{b-a}{2^{n+1}} = \frac{2}{2^{n+1}} \\ &= 2^{-n} < 10^{-1} \Rightarrow n > -\log_2 10^{-1} \approx 3{,}32. \end{aligned}$$

O que está de acordo com o experimento numérico realizado naquele exemplo.

O método da bisseção tem a boa propriedade de garantia de convergência, bem como de fornecer uma simples estimativa da precisão da aproximação calculada. Entretanto, a taxa de convergência linear é superada por outros métodos. A construção de tais métodos está, normalmente, associada a iteração do ponto fixo, a qual exploramos na próxima seção.

3.2.1 Código Python: método da bisseção

O seguinte código é uma implementação em `Python` do algoritmo da bisseção. As variáveis de entrada são:

- `f` - função objetivo
- `a` - extremo esquerdo do intervalo de inspeção $[a, b]$
- `b` - extremo direito do intervalo de inspeção $[a, b]$
- `TOL` - tolerância (critério de parada)
- `N` - número máximo de iterações

A variável de saída é:

- `p` - aproximação da raiz de `f`, i.e. $f(p) \approx 0$.

```
from __future__ import division

def bissecao(f, a, b, TOL, N):
    i = 1
```

```
fa = f(a)
while (i <= N):
    #iteracao da bissecao
    p = a + (b-a)/2
    fp = f(p)
    #condicao de parada
    if ((fp == 0) or ((b-a)/2 < TOL)):
        return p
    #bissecta o intervalo
    i = i+1
    if (fa * fp > 0):
        a = p
        fa = fp
    else:
        b = p

raise NameError('Num. max. de iter. excedido!');
```

Exercícios

E 3.2.1. Considere a equação $\sqrt{x} = \cos(x)$. Use o método da bisseção com intervalo inicial $[a, b] = [0, 1]$ e $x^{(1)} = (a + b)/2$ para calcular a aproximação $x^{(4)}$ da solução desta equação.

E 3.2.2. Trace o gráfico e isole as três primeiras raízes positivas da função:

$$f(x) = 5\sin(x^2) - \exp\left(\frac{x}{10}\right)$$

em intervalos de comprimento 0,1. Então, use o método da bisseção para obter aproximações dos zeros desta função com precisão de 10^{-5}.

Exemplo 3.2.3. O polinômio $p(x) = -4 + 8x - 5x^2 + x^3$ tem raízes $x_1 = 1$ e $x_2 = x_3 = 2$ no intervalo $[1/2, 3]$.

a) Se o método da bisseção for usando com o intervalo inicial $[1/2, 3]$, para qual raiz as iterações convergem?

b) É possível usar o método da bisseção para a raiz $x = 2$? Justifique sua resposta.

E 3.2.3. Mostre que a equação do problema 3.1.7 possui uma solução no intervalo $[1, v+1]$ para todo v positivo. Dica: defina $f(x) = \ln(x) + x - \frac{1}{x} - v$ e considere a seguinte estimativa:

$$f(v+1) = f(1) + \int_1^{v+1} f'(x)dx \geq -v + \int_1^{v+1} dx = 0.$$

Use esta estimativa para iniciar o método de bisseção e obtenha o valor da raiz com pelo menos 6 algarismos significativos para $v = 1, 2, 3, 4$ e 5.

E 3.2.4. Considere o seguinte problema físico: uma plataforma está fixa a uma parede através de uma dobradiça cujo momento é dado por:

$$\tau = k\theta,$$

onde θ é angulo da plataforma com a horizontal e k é uma constante positiva. A plataforma é feita de material homogêneo, seu peso é P e sua largura é l. Modele a relação entre o ângulo θ e o peso P próprio da plataforma. Encontre o valor de θ quando $l = 1$ m, $P = 200$ N, $k = 50$ Nm/rad, sabendo que o sistema está em equilíbrio. Use o método da bisseção e expresse o resultado com 4 algarismos significativos.

E 3.2.5. Considere a equação de Lambert dada por:

$$xe^x = t,$$

onde t é um número real positivo. Mostre que esta equação possui uma única solução x^* que pertence ao intervalo $[0, t]$. Usando esta estimativa como intervalo inicial, quantos passos são necessário para obter o valor numérico de x^* com erro absoluto inferior a 10^{-6} quando $t = 1$, $t = 10$ e $t = 100$ através do método da bisseção? Obtenha esses valores.

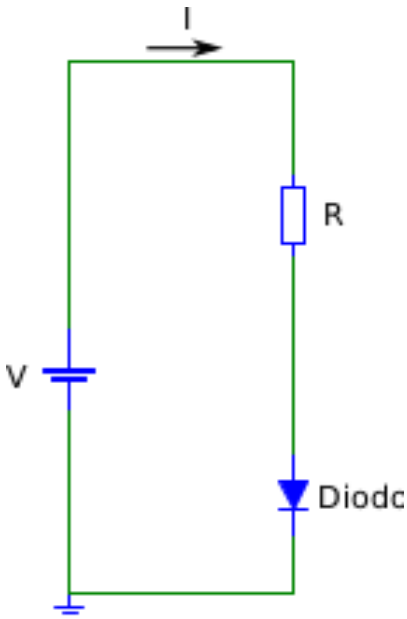
E 3.2.6. O polinômio $f(x) = x^4 - 4x^2 + 4$ possui raízes duplas em $\sqrt{2}$ e $-\sqrt{2}$. O método da bisseção pode ser aplicados a f? Explique.

E 3.2.7. O desenho abaixo mostra um circuito não linear envolvendo uma fonte de tensão constante, um diodo retificador e um resistor. Sabendo que a relação entre a corrente (I_d) e a tensão (v_d) no diodo é dada pela seguinte expressão:

$$I_d = I_R \left(\exp\left(\frac{v_d}{v_t}\right) - 1 \right),$$

onde I_R é a corrente de condução reversa e v_t, a tensão térmica dada por $v_t = \frac{kT}{q}$ com k, a constante de Boltzmann, T a temperatura de operação e q, a carga do

elétron. Aqui $I_R = 1pA = 10^{-12}$ A, $T = 300$ K. Escreva o problema como uma equação na incógnita v_d e, usando o método da bisseção, resolva este problema com 3 algarismos significativos para os seguintes casos:



a) $V = 30$ V e $R = 1$ kΩ.

b) $V = 3$ V e $R = 1$ kΩ.

c) $V = 3$ V e $R = 10$ kΩ.

d) $V = 300$ mV e $R = 1$ kΩ.

e) $V = -300$ mV e $R = 1$ kΩ.

f) $V = -30$ V e $R = 1$ kΩ.

g) $V = -30$ V e $R = 10$ kΩ.

Dica: $V = RI_d + v_d$.

E 3.2.8. Obtenha os valores de I_d no problema 3.2.7. Lembre que existem duas expressões disponíveis:

$$I_d = I_R \left(\exp\left(\frac{v_d}{v_t}\right) - 1 \right)$$

e

$$I_d = \frac{v - v_d}{R}$$

Faça o estudo da propagação do erro e decida qual a melhor expressão em cada caso.

3.3 Iteração de Ponto Fixo

Nesta seção, discutimos a abordagem da **iteração do ponto fixo** para a solução numérica de equações de uma variável real. Observamos que sempre podemos reescrever uma equação da forma $f(x) = 0$ (problema de encontrar os zeros de uma função) em uma equação equivalente na forma $g(x) = x$ (**problema de ponto fixo**). Um ponto $x = x^*$ tal que $g(x^*) = x^*$ é chamado de **ponto fixo** da função $g(x)$. Geometricamente, um ponto fixo de uma função é um ponto de interseção entre a reta $y = x$ com o gráfico da função (veja, Figura 3.3).

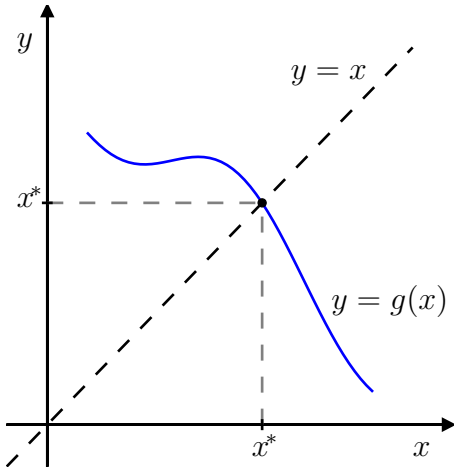


Figura 3.3: Ponto fixo $g(x^*) = x^*$.

Exemplo 3.3.1. Resolver a equação $e^x = x + 2$ é equivalente a resolver $f(x) = 0$, com $f(x) = e^x - x - 2$. Estes são equivalentes a resolver $g(x) = x$, com $g(x) = e^x - 2$. Ou seja, temos:

$$e^x = x + 2 \Leftrightarrow e^x - x - 2 = 0 \Leftrightarrow e^x - 2 = x$$

Dada uma função $g(x)$, a **iteração do ponto fixo** consiste em computar a seguinte sequência recursiva:

$$x^{(n+1)} = g(x^{(n)}), \quad n \geq 1,$$

onde $x^{(1)}$ é uma aproximação inicial do ponto fixo.

Exemplo 3.3.2 (Método babilônico)**.** O método babilônico[3] é de uma iteração de ponto fixo para extrair a raiz quadrada de um número positivo A, i.e. para resolver a equação $x^2 = A$.

Seja $r > 0$ uma aproximação para $\sqrt{A}$. Temos três possibilidades:

- $r > \sqrt{A} \Longrightarrow \frac{A}{r} < \sqrt{A} \Longrightarrow \sqrt{A} \in \left(\frac{A}{r}, r\right)$
- $r = \sqrt{A} \Longrightarrow \frac{A}{r} = \sqrt{A}$
- $r < \sqrt{A} \Longrightarrow \frac{A}{r} > \sqrt{A} \Longrightarrow \sqrt{A} \in \left(r, \frac{A}{r}\right)$

[3]Heron de Alexandria, 10 d.C. - 70 d.C., matemático grego.

Ou seja, uma aproximação melhor para $\sqrt{A}$ está no intervalo entre r e $\frac{A}{r}$ que pode ser aproximada como:

$$x = \frac{r + \frac{A}{r}}{2}$$

Aplicando esse método repetidas vezes, podemos construir a iteração (de ponto fixo):

$$\begin{aligned} x^{(1)} &= r \\ x^{(n+1)} &= \frac{x^{(n)}}{2} + \frac{A}{2x^{(n)}}, \quad n = 1,2,3,... \end{aligned}$$

Por exemplo, para obter uma aproximação para $\sqrt{5}$, podemos iniciar com a aproximação inicial $r = 2$ e $A = 5$. Então, tomamos $x^{(1)} = 2$ e daí seguem as aproximações:

$$\begin{aligned} x^{(2)} &= \frac{2}{2} + \frac{2{,}5}{2} = 2{,}25 \\ x^{(3)} &= \frac{2{,}25}{2} + \frac{2{,}5}{2{,}25} = 2{,}2361111 \\ x^{(4)} &= \frac{2{,}2361111}{2} + \frac{2{,}5}{2{,}2361111} = 2{,}236068 \\ x^{(5)} &= \frac{2{,}236068}{2} + \frac{2{,}5}{2{,}236068} = 2{,}236068 \end{aligned}$$

O método babilônico sugere que a iteração do ponto fixo pode ser uma abordagem eficiente para a solução de equações. Ficam, entretanto, as seguintes perguntas:

1. Será que a iteração do ponto fixo é convergente?

2. Caso seja convergente, será que o limite $x^* = \lim_{n\to\infty} x^{(n)}$ é um ponto fixo?

3. Caso seja convergente, qual é a taxa de convergência?

A segunda pergunta é a mais fácil de ser respondida. No caso de $g(x)$ ser contínua, se $x^{(n)} \to x^* \in \text{Dom}\,(g)$, então:

$$x^* = \lim_{n\to\infty} x^{(n)} = \lim_{n\to\infty} g(x^{(n-1)}) = g\left(\lim_{n\to\infty} x^{(n-1)}\right) = g(x^*).$$

Antes de respondermos as perguntas acima, vejamos mais um exemplo.

Tabela 3.2: Iterações do ponto fixo para o Exemplo 3.3.3.

n	$x_1^{(n)}$	$x_2^{(n)}$
1	1,700	1,700
2	2,047	1,735
3	−0,8812	1,743
4	4,3013	1,746
5	−149,4	1,746

Exemplo 3.3.3. Considere o problema de encontrar o zero da função $f(x) = x\exp(x) - 10$. Uma maneira geral de construir um problema de ponto fixo equivalente é o seguinte:

$$f(x) = 0 \Rightarrow \alpha f(x) = 0 \Rightarrow x - \alpha f(x) = x,$$

para qualquer parâmetro $\alpha \neq 0$. Consideremos, então, as seguintes duas funções:

$$g_1(x) = x - 0{,}5f(x) \quad \text{e} \quad g_2(x) = x - 0{,}05f(x).$$

Notamos que o ponto fixo destas duas funções coincide com o zero de $f(x)$. Construindo as iterações do ponto fixo:

$$x_1^{(n+1)} = g_1(x_1^{(n)}) \quad \text{e} \quad x_2^{(n+1)} = g_2(x_2^{(n)}),$$

tomando $x_1^{(1)} = x_2^{(1)} = 1{,}7$, obtemos os resultados apresentados na Tabela 3.2. Observamos que, enquanto, a iteração do ponto fixo com a função $g_1(x)$ ($\alpha = 0{,}5$) parece divergir, a iteração com a função $g_2(x)$ ($\alpha = 0{,}05$) parece convergir.

Afim de estudarmos a convergência da iteração do ponto fixo, apresentamos o Teorema do ponto fixo.

3.3.1 Teorema do ponto fixo

O Teorema do ponto fixo nos fornece condições suficientes para a existência e unicidade do ponto fixo, bem como para a convergência das iterações do método.

Definição 3.3.1. *Uma* ***contração*** *é uma função real* $g : [a, b] \to [a, b]$ *tal que:*

$$|g(x) - g(y)| \leq \beta|x - y|, \quad 0 \leq \beta < 1.$$

Observação 3.3.1. Seja $g : [a, b] \to [a, b]$, y=g(x).

- Se $g(x)$ é uma contração, então $g(x)$ função contínua.
- Se $|g'(x)| < k$, $0 < k < 1$, para todo $x \in [a, b]$, então $g(x)$ é uma contração.

Teorema 3.3.1 (Teorema do ponto fixo)**.** *Se $g : [a,b] \to [a,b]$ é uma contração, então existe um único ponto $x^* \in [a, b]$ tal que $g(x^*) = x^*$, i.e. x^* é ponto fixo de $g(x)$. Além disso, a sequência $\{x^{(n)}\}_{n\in\mathbb{N}}$ dada por:*

$$x^{(n+1)} = g(x^{(n)})$$

converge para x^* para qualquer $x^{(1)} \in [a, b]$.

Demonstração. Começamos demonstrando que existe pelo menos um ponto fixo. Para tal definimos a função $f(x) = x - g(x)$ e observamos que:

$$f(a) = a - g(a) \leq a - a = 0$$

e

$$f(b) = b - g(b) \geq b - b = 0$$

Se $f(a) = a$ ou $f(b) = b$, então o ponto fixo existe. Caso contrário, as desigualdades são estritas e a $f(x)$ muda de sinal no intervalo. Como esta função é contínua, pelo teorema de Bolzano 3.1.1, existe um ponto x^* no intervalo (a, b) tal que $f(x^*) = 0$, ou seja, $g(x^*) = x^*$. Isto mostra a existência.

Para provar que o ponto fixo é único, observamos que se x^* e x^{**} são pontos fixos, eles devem ser iguais, pois:

$$|x^* - x^{**}| = |g(x^*) - g(x^{**})| \leq \beta|x^* - x^{**}|.$$

A desigualdade $|x^* - x^{**}| \leq \beta|x^* - x^{**}|$ com $0 \leq \beta < 1$ implica $|x^* - x^{**}| = 0$.

Para demonstrar a convergência da sequência, observamos que:

$$|x^{(n+1)} - x^*| = |g(x^{(n)}) - x^*| = |g(x^{(n)}) - g(x^*)| \leq \beta|x^{(n)} - x^*|.$$

Daí, temos:

$$|x^{(n)} - x^*| \leq \beta|x^{(n-1)} - x^*| \leq \beta^2|x^{(n-2)} - x^*| \leq \cdots \leq \beta^n|x^{(0)} - x^*|.$$

Portanto, como $0 \leq \beta < 1$, temos:

$$\lim_{n\to\infty} |x^{(n)} - x^*| = 0,$$

ou seja, $x^{(n)} \to x^*$ quando $n \to \infty$. □

Exemplo 3.3.4. Mostre que o Teorema do ponto fixo se aplica a função $g(x) = \cos(x)$ no intervalo $[1/2, 1]$, i.e. que a iteração do ponto fixo converge para a solução da equação $\cos x = x$.

Solução. Basta mostrarmos que:

a) $g([1/2,1]) \subseteq [1/2,1]$;

b) $|g'(x)| < \beta, \quad 0 < \beta < 1, \quad \forall x \in [1/2,1]$.

Para provar a), observamos que $g(x)$ é decrescente no intervalo, pelo que temos:

$$0{,}54 < \cos(1) \leq \cos(x) \leq \cos(1/2) < 0{,}88$$

Como [0,54, 0,88] $\subseteq$ [0,5, 1], temos o item a).

Para provar o item b), observamos que:

$$g'(x) = -\sin(x).$$

Da mesma forma, temos a estimativa:

$$-0{,}85 < -\sin(1) \leq -\sin(x) \leq -\sin(1/2) < -0{,}47.$$

Assim, $|g'(x)| < 0{,}85$ temos a desigualdade com $\beta = 0{,}85 < 1$.

A Tabela 3.3 apresenta o comportamento numérico da iteração do ponto fixo:

$$\begin{aligned} x^{(1)} &= 0{,}7 \\ x^{(n+1)} &= \cos(x^{(n)}), \quad n \geq 1. \end{aligned}$$

$\Diamond$

3.3.2 Teste de convergência

Seja $g : [a,b]$ uma função $C^0[a,b]$ e $x^* \in (a,b)$ um ponto fixo de g. Então x^* é dito estável se existe uma região $(x^* - \delta, x^* + \delta)$ chamada bacia de atração tal que $x^{(n+1)} = g(x^{(n)})$ é convergente sempre que $x^{(0)} \in (x^* - \delta, x^* + \delta)$.

Proposição 3.3.1 (Teste de convergência)**.** *Se $g \in C^1[a,b]$ e $|g'(x^*)| < 1$, então x^* é estável. Se $|g'(x^*)| > 1$ é instável e o teste é inconclusivo quando $|g'(x^*)| = 1$.*

Exemplo 3.3.5. No Exemplo 3.3.3 observamos que a função $g_1(x)$ nos forneceu uma iteração divergente, enquanto que a função $g_2(x)$ forneceu uma iteração convergente (veja a Figura 3.4. A razão destes comportamentos é explicada pelo teste da convergência. Com efeito, sabemos que o ponto fixo destas funções está no intervalo [1,6, 1,8] e temos:

$$|g_1'(x)| = |1 - 0{,}5(x+1)e^x| > 4{,}8, \quad \forall x \in [1{,}6, 1{,}8],$$

enquanto:

$$|g_2'(x)| = |1 - 0{,}05(x+1)e^x| < 0{,}962, \quad \forall x \in [1{,}6, 1{,}8].$$

n	$x^{(n)}$
1	0,700
2	0,765
3	0,721
4	0,751
5	0,731
6	0,744
7	0,735

Tabela 3.3: Iteração do ponto fixo para o Exemplo 3.3.4.

3.3.3 Estabilidade e convergência

A fim de compreendermos melhor os conceitos de estabilidade e convergência, considere uma função $\Phi(x)$ com um ponto fixo $x^* = g(x^*)$ e analisemos o seguinte processo iterativo:

$$\begin{aligned} x^{(n+1)} &= g\left(x^{(n)}\right) \\ x^{(0)} &= x \end{aligned}$$

Vamos supor que a função $g(x)$ pode ser aproximada por seu polinômio de Taylor em torno do ponto fixo:

$$\begin{aligned} g(x) &= g(x^*) + (x - x^*)g'(x^*) + O\left((x - x^*)^2\right), n \geq 0 \\ &= x^* + (x - x^*)g'(x^*) + O\left((x - x^*)^2\right) \\ &\approx x^* + (x - x^*)g'(x^*) \end{aligned}$$

Substituindo na relação de recorrência, temos

$$x^{(n+1)} = g\left(x^{(n)}\right) \approx x^* + (x^{(n)} - x^*)g'(x^*)$$

Ou seja:

$$\left(x^{(n+1)} - x^*\right) \approx (x^{(n)} - x^*)g'(x^*)$$

Tomando módulos, temos:

$$\underbrace{\left|x^{(n+1)} - x^*\right|}_{\epsilon_{n+1}} \approx \underbrace{\left|x^{(n)} - x^*\right|}_{\epsilon_n} \left|g'(x^*)\right|,$$

onde $\epsilon_n = \left|x^{(n)} - x^*\right|$.

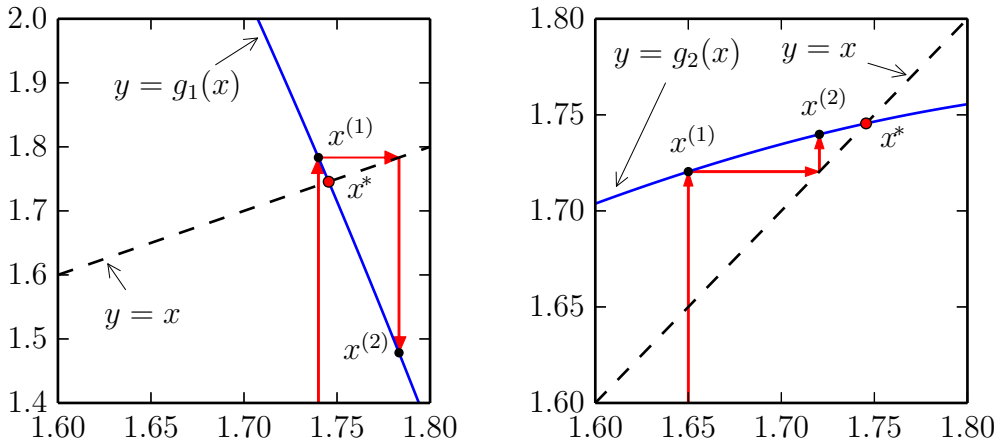


Figura 3.4: Ilustração das iterações do ponto fixo para: (esquerda) $y = g_1(x)$ e (direita) $y = g_2(x)$. Veja Exemplo 3.3.5.

Observação 3.3.2. A análise acima, concluímos:

- Se $|g'(x^*)| < 1$, então, a distância de $x^{(n)}$ até o ponto fixo x^* está diminuindo a cada passo.
- Se $|g'(x^*)| > 1$, então, a distância de $x^{(n)}$ até o ponto fixo x^* está aumentando a cada passo.
- Se $|g'(x^*)| = 1$, então, nossa aproximação de primeiro ordem não é suficiente para compreender o comportamento da sequência.

3.3.4 Erro absoluto e tolerância

Na prática, quando se aplica uma iteração como esta, não se conhece de antemão o valor do ponto fixo x^*. Assim, o erro $\epsilon_n = \left|x^{(n)} - x^*\right|$ precisa ser estimado com base nos valores calculados $x^{(n)}$. Uma abordagem frequente é analisar a evolução da diferença entre dois elementos da sequência:

$$\Delta_n = \left|x^{(n+1)} - x^{(n)}\right|$$

A pergunta natural é: Será que o erro $\epsilon_n = \left|x^{(n)} - x^*\right|$ é pequeno quando $\Delta_n = \left|x^{(n+1)} - x^{(n)}\right|$ for pequeno?

Para responder a esta pergunta, observamos que

$$x^* = \lim_{n\to\infty} x^{(n)}$$

portanto:

$$\begin{aligned} x^* - x^{(N)} &= \left(x^{(N+1)} - x^{(N)}\right) + \left(x^{(N+2)} - x^{(N+1)}\right) + \left(x^{(N+3)} - x^{(N+2)}\right) + \dots \\ &= \sum_{k=0}^{\infty} \left(x^{(N+k+1)} - x^{(N+k)}\right) \end{aligned}$$

Usamos também as expressões:

$$\begin{aligned} x^{(n+1)} &\approx x^* + (x^{(n)} - x^*)g'(x^*) \\ x^{(n)} &\approx x^* + (x^{(n-1)} - x^*)g'(x^*) \end{aligned}$$

Subtraindo uma da outra, temos:

$$x^{(n+1)} - x^{(n)} \approx (x^{(n)} - x^{(n-1)})g'(x^*)$$

Portanto:

$$x^{(N+k+1)} - x^{(N+k)} \approx (x^{(N+1)} - x^{(N)}) \left(g'(x^*)\right)^k$$

E temos:

$$\begin{aligned} x^* - x^{(N)} &= \sum_{k=0}^{\infty} \left(x^{(N+k+1)} - x^{(N+k)}\right) \\ &\approx \sum_{k=0}^{\infty} (x^{(N+1)} - x^{(N)}) \left(g'(x^*)\right)^k \\ &= (x^{(N+1)} - x^{(N)}) \frac{1}{1 - g'(x^*)}, \quad |g'(x^*)| < 1 \end{aligned}$$

Tomando módulo, temos:

$$\begin{aligned} \left|x^* - x^{(N)}\right| &\approx \left|x^{(N+1)} - x^{(N)}\right| \frac{1}{1 - g'(x^*)} \\ \epsilon_N &\approx \frac{\Delta_N}{1 - g'(x^*)} \end{aligned}$$

Observação 3.3.3. Tendo em mente a relação $x^{(n+1)} - x^{(n)} \approx (x^{(n)} - x^{(n-1)})g'(x^*)$, concluímos:

- Quando $g'(x^*) < 0$, o esquema é alternante, isto é, o sinal do erro se altera a cada passo. O erro ϵ_N pode ser estimado diretamente da diferença Δ_N, pois o denominador $1 - g'(x^*) > 1$.

- Quando $0 < g'(x^*) < 1$, o esquema é monótono e $\frac{1}{1-g'(x^*)} > 1$, pelo que o erro ϵ_N é maior que a diferença Δ_N. A relação será tão mais importante quando mais próximo da unidade for $g'(x^*)$, ou seja, quando mais lenta for a convergência. Para estimar o erro em função da diferença Δ_N, observamos que $g'(x^*) \approx \frac{x^{(n+1)}-x^{(n)}}{x^{(n)}-x^{(n-1)}}$ e

$$|g'(x^*)| \approx \frac{\Delta_n}{\Delta_{n-1}}$$

e portanto

$$\epsilon_N \approx \frac{\Delta_N}{1-\frac{\Delta_n}{\Delta_{n-1}}}.$$

Exercícios

E 3.3.1. Resolver a equação $e^x = x + 2$ é equivalente a calcular os pontos fixos da função $g(x) = e^x + 2$ (veja o Exemplo 3.3.1). Use a iteração do ponto fixo $x^{(n+1)} = g(x^n)$ com $x^{(1)} = -1{,}8$ para obter uma aproximação de uma das soluções da equação dada com 8 dígitos significativos.

E 3.3.2. Mostre que a equação:

$$\cos(x) = x$$

possui uma única solução no intervalo $[0, 1]$. Use a iteração do ponto fixo e encontre uma aproximação para esta solução com 4 dígitos significativos.

E 3.3.3. Mostre que a equação $xe^x = 10$ é equivalente às seguintes equações:

$$x = \ln\left(\frac{10}{x}\right) \quad \text{e} \quad x = 10e^{-x}.$$

Destas, considere as seguintes iterações de ponto fixo:

a) $x^{(n+1)} = \ln\left(\frac{10}{x^{(n)}}\right)$

b) $x^{(n+1)} = 10e^{-x^{(n)}}$

Tomando $x^{(1)} = 1$, verifique se estas sequências são convergentes.

E 3.3.4. Verifique (analiticamente) que a única solução real da equação:

$$xe^x = 10$$

é ponto fixo das seguintes funções:

a) $g(x) = \ln\left(\frac{10}{x}\right)$

b) $g(x) = x - \frac{xe^x - 10}{15}$

c) $g(x) = x - \frac{xe^x - 10}{10 + e^x}$

Implemente o processo iterativo $x^{(n+1)} = g(x^{(n)})$ para $n \geq 0$ e compare o comportamento. Discuta os resultados com base na teoria estudada.

E 3.3.5. Verifique (analiticamente) que a única solução real da equação:

$$\cos(x) = x$$

é ponto fixo das seguintes funções:

a) $g(x) = \cos(x)$

b) $g(x) = 0{,}4x + 0{,}6\cos(x)$

c) $g(x) = x + \frac{\cos(x) - x}{1 + \sin(x)}$

Implemente o processo iterativo $x^{(n+1)} = g(x^{(n)})$ para $n \geq 0$ e compare o comportamento.Discuta os resultados com base na teoria estudada.

E 3.3.6. Encontre a solução de cada equação com erro absoluto inferior a 10^{-6}.

a) $e^x = x + 2$ no intervalo $(-2{,}0)$.

b) $x^3 + 5x^2 - 12 = 0$ no intervalo $(1{,}2)$.

c) $\sqrt{x} = \cos(x)$ no intervalo $(0{,}1)$.

E 3.3.7. Encontre numericamente as três primeiras raízes positivas da equação dada por:

$$\cos(x) = \frac{x}{10 + x^2}$$

com erro absoluto inferior a 10^{-6}.

E 3.3.8. Calcule uma equação da reta tangente a curva $y = e^{-(x-1)^2}$ que passa pelo ponto $(3, 1/2)$.

E 3.3.9. Resolva numericamente a inequação:

$$e^{-x^2} < 2x$$

E 3.3.10. Considere os seguintes processos iterativos:

$$a\begin{cases} x^{(n+1)} &= \cos(x^{(n)}) \\ x^{(1)} &= .5 \end{cases} \quad \text{e} \quad b\begin{cases} x^{(n+1)} &= .4x^{(n)} + .6\cos(x^{(n)}) \\ x^{(1)} &= .5 \end{cases} \tag{3.1}$$

Use o teorema do ponto fixo para verificar que cada um desses processos converge para a solução da equação x^* de $\cos(x) = x$. Observe o comportamento numérico dessas sequências. Qual estabiliza mais rápido com cinco casas decimais? Discuta.

Dica: Verifique que $\cos([0.5,1]) \subseteq [0.5,1]$ e depois a mesma identidade para a função $f(x) = .4x + .6\cos(x)$.

E 3.3.11. Use o teorema do ponto fixo aplicado a um intervalo adequado para mostrar que a função $g(x) = \ln(100 - x)$ possui um ponto fixo estável.

E 3.3.12. Na hidráulica, o fator de atrito de Darcy é dado pela implicitamente pela equação de Colebrook-White:

$$\frac{1}{\sqrt{f}} = -2\log_{10}\left(\frac{\varepsilon}{14.8R_h} + \frac{2.51}{\mathrm{Re}\sqrt{f}}\right)$$

onde f é o fator de atrito, ε é a rugosidade do tubo em metros, R_h é o raio hidráulico em metros e Re é o número de Reynolds. Considere $\varepsilon = 2mm$, $R_h = 5cm$ e $Re = 10000$ e obtenha o valor de f pela iteração:

$$x^{(n+1)} = -2\log_{10}\left(\frac{\varepsilon}{14.8R_h} + \frac{2.51x^{(n)}}{\mathrm{Re}}\right)$$

E 3.3.13. Encontre uma solução aproximada para equação algébrica

$$180 - 100x = 0.052\sinh^{-1}(10^{13}x)$$

com erro absoluto inferior a 10^{-3} usando um método iterativo. Estime o erro associado ao valor de $v = 180 - 100x = 0.052\sinh^{-1}(10^{13}x)$, usando cada uma dessas expressões. Discuta sucintamente o resultado obtido. Dica: Este caso é semelhante ao problema 3.2.7.

E 3.3.14. Considere que x_n satisfaz a seguinte relação de recorrência:

$$x_{n+1} = x_n - \beta\,(x_n - x^*)$$

onde β e x^* são constantes. Prove que

$$x_n - x^* = (1-\beta)^{n-1}(x_1 - x^*).$$

Conclua que $x_n \to x^*$ quando $|1-\beta| < 1$.

E 3.3.15. Considere o seguinte esquema iterativo:

$$\begin{cases} x^{(n+1)} = x_n + q^n \\ x^{(0)} = 0 \end{cases}$$

onde $q = 1 - 10^{-6}$.

a) Calcule o limite

$$x_\infty = \lim_{n\to\infty} x^{(n)}$$

analiticamente.

b) Considere que o problema de obter o limite da sequência numericamente usando como critério de parada que $|x^{(n+1)} - x^{(n)}| < 10^{-5}$. Qual o valor é produzido pelo esquema numérico? Qual o desvio entre o valor obtido pelo esquema numérico e o valor do limite obtido no item a? Discuta. (Dica: Você não deve implementar o esquema iterativo, obtendo o valor de $x^{(n)}$ analiticamente)

c) Qual deve ser a tolerância especificada para obter o resultado com erro relativo inferior a 10^{-2}?

E 3.3.16. Considere o seguinte esquema iterativo:

$$x^{(n+1)} = x^{(n)} - [x^{(n)}]^3,\ x^{(n)} \geq 0$$

com $x^{(0)} = 10^{-2}$. Prove que $\{x^{(n)}\}$ é sequência de número reais positivos convergindo para zero. Verifique que são necessários mais de mil passos para que $x^{(n)}$ se torne menor que $0.9x^{(0)}$.

E 3.3.17.

a) Use o teorema do ponto fixo para mostrar que a função $g(x) = 1 - \sin(x)$ possui um único ponto fixo estável o intervalo $[\frac{1}{10},1]$. Construa um método iterativo $x^{(n+1)} = g(x^{(n)})$ para encontrar esse ponto fixo. Use o Scilab para encontrar o valor numérico do ponto fixo.

b) Verifique que função $\psi(x) = \frac{1}{2}[x + 1 - \sin(x)]$ possui um ponto fixo x^* que também é o ponto fixo da função g do item a. Use o Scilab para encontrar o valor numérico do ponto fixo através da iteração $x^{(n+1)} = \psi(x^{(n)})$. Qual método é mais rápido?

E 3.3.18. (*Esquemas oscilantes*)

a) Considere a função $g(x)$ e função composta $\psi(x) = g \circ g = g\,(g(x))$. Verifique todo ponto fixo de g também é ponto fixo de ψ.

b) Considere a função
$$g(x) = 10\exp(-x)$$
e função composta $\psi(x) = g \circ g = g\,(g(x))$. Mostre que ψ possui dois pontos fixos que não são pontos fixos de g.

c) No problema anterior, o que acontece quando o processo iterativo $x^{(n+1)} = g(x^{(n)})$ é inicializado com um ponto fixo de ψ que não é ponto fixo de g?

E 3.3.19. Mostre que se $f(x)$ possui uma raiz x^* então a x^* é um ponto fixo de $\phi(x) = x + \gamma(x)f(x)$. Encontre uma condição em $\gamma(x)$ para que o ponto fixo x^* de ϕ seja estável. Encontre uma condição em $\gamma(x)$ para que $\phi'(x^*) = 0$.

E 3.3.20. Considere que $x^{(n)}$ satisfaz a seguinte relação de recorrência:
$$x^{(n+1)} = x^{(n)} - \gamma f(x^{(n)})$$
onde γ é uma constante. Suponha que $f(x)$ possui um zero em x^*. Aproxime a função $f(x)$ em torno de x^* por
$$f(x) = f(x^*) + f'(x^*)(x - x^*) + O\left((x - x^*)^2\right).$$
Em vista do problema anterior, qual valor de γ você escolheria para que a sequência $x^{(n)}$ convirja rapidamente para x^*.

E 3.3.21. Considere o problema da questão 3.2.7 e dois seguintes esquemas iterativos.
$$A\begin{cases} I^{(n+1)} = \frac{1}{R}\left[V - v_t \ln\left(1 + \frac{I^{(n)}}{I_R}\right)\right], n > 0 \\ I^{(0)} = 0 \end{cases}$$
e
$$B\begin{cases} I^{(n+1)} = I_R\left[\exp\left(\frac{V - RI^{(n)}}{v_t}\right) - 1\right], n > 0 \\ I^{(0)} = 0 \end{cases}$$

Verifique numericamente que apenas o processo A é convergente para a, b e c; enquanto apenas o processo B é convergente para os outros itens.

3.4 Método de Newton-Raphson

Nesta seção, apresentamos o **método de Newton-Raphson**[4][5] para calcular o zero de funções reais de uma variável real.

Assumimos que x^* é um zero de uma dada função $f(x)$ continuamente diferenciável, i.e. $f(x^*) = 0$. Afim de usar a iteração do ponto fixo, observamos que, equivalentemente, x^* é um ponto fixo da função:

$$g(x) = x + \alpha(x)f(x), \quad \alpha(x) \neq 0,$$

onde $\alpha(x)$ é uma função arbitrária que queremos escolher de forma que a iteração do ponto fixo tenha ótima taxa de convergência.

Do **Teorema do ponto fixo** temos que a taxa de convergência é dada em função do valor absoluto da derivada de $g(x)$. Calculando a derivada temos:

$$g'(x) = 1 + \alpha(x)f'(x) + \alpha'(x)f(x).$$

No ponto $x = x^*$, temos:

$$g'(x^*) = 1 + \alpha(x^*)f'(x^*) + \alpha'(x^*)f(x^*).$$

Como $f(x^*) = 0$, temos:

$$g'(x^*) = 1 + \alpha(x^*)f'(x^*).$$

Sabemos que o processo iterativo converge tão mais rápido quanto menor for $|g'(x)|$ nas vizinhanças de x^*. Isto nos leva a escolher:

$$g'(x^*) = 0,$$

e, então, temos:

$$\alpha(x^*) = -\frac{1}{f'(x^*)},$$

se $f'(x^*) \neq 0$.

A discussão acima nos motiva a introduzir o método de Newton, cujas iterações são dada por:

$$x^{(n+1)} = x^{(n)} - \frac{f\left(x^{(n)}\right)}{f'\left(x^n\right)}, \quad n \geq 1,$$

sendo $x^{(1)}$ uma aproximação inicial dada.

[4] Joseph Raphson, 1648 - 1715, matemático inglês.

[5] Também chamado apenas de método de Newton.

3.4.1 Interpretação geométrica

Seja dada uma função $f(x)$ conforme na Figura 3.5. Para tanto, escolhemos uma aproximação inicial $x^{(1)}$ e computamos:

$$x^{(2)} = x^{(1)} - \frac{f(x^{(1)})}{f'(x^{(1)})}.$$

Geometricamente, o ponto $x^{(2)}$ é a interseção da reta tangente ao gráfico da função $f(x)$ no ponto $x = x^{(1)}$ com o eixo das abscissas. Com efeito, a equação desta reta é:

$$y = f'(x^{(1)})(x - x^{(1)}) + f(x^{(1)}).$$

Assim, a interseção desta reta com o eixo das abscissas ocorre quando $(y = 0)$:

$$f'(x^{(1)})(x - x^{(1)}) + f(x^{(1)}) = 0 \Rightarrow x = x^{(1)} - \frac{f(x^{(1)})}{f'(x^{(1)})}.$$

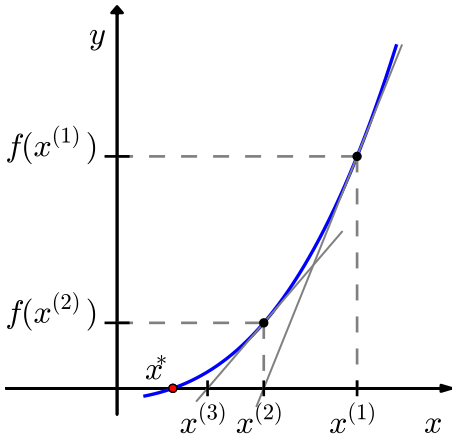


Figura 3.5: Interpretação do método de Newton.

Ou seja, dado $x^{(n)}$ a próxima aproximação $x^{(n+1)}$ é o ponto de interseção entre o eixo das abscissas e a reta tangente ao gráfico da função no ponto $x = x^{(n)}$. Observe a Figura 3.5.

3.4.2 Análise de convergência

Seja $f(x)$ um função com derivadas primeira e segunda contínuas tal que $f(x^*) = 0$ e $f'(x^*) \neq 0$. Seja também a função $g(x)$ definida como:

$$g(x) = x - \frac{f(x)}{f'(x)}.$$

Expandimos em série de Taylor em torno de $x = x^*$, obtemos:

$$g(x) = g(x^*) + g'(x^*)(x - x^*) + \frac{g''(x^*)}{2}(x - x^*)^2 + O\left((x - x^*)^3\right).$$

Observamos que:

$$\begin{aligned} g(x^*) &= x^* \\ g'(x^*) &= 1 - \frac{f'(x^*)f'(x^*) - f(x^*)f''(x^*)}{(f'(x^*))^2} = 0 \end{aligned}$$

Portanto:

$$g(x) = x^* + \frac{g''(x^*)}{2}(x - x^*)^2 + O\left((x - x^*)^3\right)$$

Com isso, temos:

$$x^{(n+1)} = g(x^{(n)}) = x^* + \frac{g''(x^*)}{2}(x^{(n)} - x^*)^2 + O\left((x - x^*)^3\right),$$

ou seja:

$$\left|x^{(n+1)} - x^*\right| \leq C\left|x^{(n)} - x^*\right|^2,$$

com constante $C = |g''(x^*)/2|$. Isto mostra que o método de Newton tem **taxa de convergência quadrática**. Mais precisamente, temos o seguinte teorema.

Teorema 3.4.1 (Método de Newton)**.** *Sejam $f \in C^2([a,b])$ com $x^* \in (a,b)$ tal que $f(x^*) = 0$ e:*

$$m := \min_{x\in[a,b]} |f'(x)| > 0 \quad e \quad M := \max_{x\in[a,b]} |f''(x)|.$$

Escolhendo $\rho > 0$ tal que:

$$q := \frac{M}{2m}\rho < 1,$$

*definimos a **bacia de atração** do método de Newton pelo conjunto:*

$$K_\rho(x^*) := \{x \in \mathbb{R};\ |x - x^*| \leq \rho\} \subset [a,b].$$

Então, para qualquer $x^{(1)} \in K_\rho(x^*)$ a iteração do método de Newton:

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})},$$

*fornece uma sequência $x^{(n)}$ que converge para x^*, i.e. $x^{(n)} \to x^*$ quando $n \to \infty$. Além disso, temos a seguinte estimativa de erro **a priori**:*

$$|x^{(n)} - x^*| \leq \frac{2m}{M} q^{(2^{n-1})}, \quad n \geq 2,$$

e a seguinte estimativa de erro ***a posteriori****:*

$$|x^{(n)} - x^*| \leq \frac{M}{2m}|x^{(n)} - x^{(n-1)}|^2, \quad n \geq 2.$$

Demonstração. Para $n \in \mathbb{N}$, $n \geq 2$, temos:

$$x^{n+1} - x^* = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})} - x^* = -\frac{1}{f(x^{(n)})}\left[f(x^{(n)}) + (x^* - x^{(n)})f'(x^{(n)}\right]. \quad (3.2)$$

Agora, para estimar o lado direito desta equação, usamos o polinômio de Taylor de grau 1 da função $f(x)$ em torno de $x = x^{(n)}$, i.e.:

$$f(x^*) = f(x^{(n)}) + (x^* - x^{(n)})f'(x^{(n)}) + \int_{x^{(n)}}^{x^*} f''(t)(x^* - t)\,dt.$$

Pela mudança de variável $t = x^{(n)} + s(x^{(n)} - x^*)$, observamos que o resto deste polinômio de Taylor na forma integral é igual a:

$$R(x^*,x^{(n)}) := (x^* - x^{(n)})^2 \int_0^1 f''\left(x^{(n)} + s(x^* - x^{(n)})\right)(1 - s)\,ds.$$

Assim, da cota da segunda derivada de $f(x)$, temos:

$$|R(x^*,x^{(n)})| \leq M|x^* - x^{(n)}|^2 \int_0^1 (1 - s)\,ds = \frac{M}{2}|x^* - x^{(n)}|^2. \quad (3.3)$$

Se $x^{(n)} \in K_\rho(x^*)$, então de (3.2) e (3.3) temos:

$$|x^{(n+1)} - x^*| \leq \frac{M}{2m}|x^{(n)} - x^*|^2 \leq \frac{M}{2m}\rho^2 < \rho. \quad (3.4)$$

Isto mostra que se $x^{(n)} \in K_\rho(x^*)$, então $x^{(n+1)} \in K_\rho(x^*)$, i.e. $x^{(n)} \in K_\rho(x^*)$ para todo $n \in \mathbb{R}$.

Agora, obtemos a estimativa **a priori** de (3.4.2), pois:

$$|x^{(n)} - x^*| \leq \frac{2m}{M}\left(\frac{M}{2m}|x^{(n-1)} - x^*|\right)^2 \leq \cdots \leq \frac{2m}{M}\left(\frac{M}{2m}|x^{(1)} - x^*|\right)^{2^{n-1}}.$$

Logo:

$$|x^{(n)} - x^*| \leq \frac{2m}{M}q^{2^{n-1}},$$

donde também vemos que $x^{(n)} \to x^*$ quando $n \to \infty$, pois $q < 1$.

Por fim, para provarmos a estimativa **a posteriori** tomamos a seguinte expansão em polinômio de Taylor:

$$f(x^{(n)}) = f(x^{(n-1)}) + (x^{(n)} - x^{(n-1)})f'(x^{(n-1)}) + R(x^{(n)},x^{(n-1)}).$$

Aqui, temos:

$$f(x^{(n-1)}) + (x^{(n)} - x^{(n-1)})f'(x^{(n-1)}) = 0$$

e, então, conforme acima:

$$|f(x^{(n)})| = |R(x^{(n)}),x^{(n-1)}| \leq \frac{M}{2}|x^{(n)} - x^{(n-1)}|^2.$$

Com isso e do Teorema do valor médio, concluímos:

$$|x^{(n)} - x^*| \leq \frac{1}{m}|f(x^{(n)}) - f(x^*)| \leq \frac{M}{2m}|x^{(n)} - x^{(n-1)}|^2.$$

□

Exemplo 3.4.1. Estime o raio ρ da bacia de atração $K_\rho(x^*)$ para a função $f(x) = \cos(x) - x$ restrita ao intervalo $[0, \pi/2]$.

Solução. O raio da bacia de atração é tal que:

$$\rho < \frac{2m}{M}$$

onde $m := \min |f'(x)|$ e $M := \max |f''(x)|$ com o mínimo e o máximo tomados em um intervalo $[a, b]$ que contenha o zero da função $f(x)$. Aqui, por exemplo, podemos tomar $[a, b] = [0, \pi/2]$. Como, neste caso, $f'(x) = -\sin(x) - 1$, temos que $m = 1$. Também, como $f''(x) = -\cos x$, temos $M = 1$. Assim, concluímos que $\rho < 2$ (lembrando que $K_\rho(x^*) \subset [0, \pi/2]$). Ou seja, neste caso as iterações de Newton convergem para o zero de $f(x)$ para qualquer escolha da aproximação inicial $x^{(1)} \in [0, \pi/2]$. ◊

Exercícios

E 3.4.1. Considere o problema de calcular as soluções positivas da equação:

$$\text{tg}\,(x) = 2x^2.$$

a) Use o método gráfico para isolar as duas primeiras raízes positivas em pequenos intervalos. Use a teoria estudada em aula para argumentar quanto à existência e unicidade das raízes dentro intervalos escolhidos.

b) Calcule o número de iterações necessárias para que o método da bisseção aproxime cada uma das raízes com erro absoluto inferior a 10^{-8}. Calcule as raízes por este método usando este número de passos.

c) Calcule cada uma das raízes pelo método de Newton com oito dígitos significativos e discuta a convergência comparando com o item b).

Obs: Alguns alunos encontraram como solução $x_1 \approx 1{,}5707963$ e $x_2 \approx 4{,}7123890$. O que eles fizeram de errado?

E 3.4.2. Isole e encontre as cinco primeiras raízes positivas da equação com 6 dígitos corretos através de traçado de gráfico e do método de Newton.

$$\cos(10x) = e^{-x}.$$

Dica: a primeira raiz positiva está no intervalo (0,0.02). Fique atento.

E 3.4.3. Encontre as raízes do polinômio $f(x) = x^4 - 4x^2 + 4$ através do método de Newton. O que você observa em relação ao erro obtido? Compare com a situação do problema 3.2.6.

E 3.4.4. Encontre as raízes reais do polinômio $f(x) = \frac{x^5}{100} + x^4 + 3x + 1$ isolando-as pelo método do gráfico e depois usando o método de Newton. Expresse a solução com 7 dígitos significativos.

E 3.4.5. Considere o método de Newton aplicado para encontrar a raiz de $f(x) = x^3 - 2x + 2$. O que acontece quando $x^{(0)} = 0$? Escolha um valor adequado para inicializar o método e obter a única raiz real desta equação.

E 3.4.6. Justifique a construção do processo iterativo do método de Newton através do conceito de estabilidade de ponto fixo e convergência do método da iteração. Dica: Considere os problemas 3.3.19 e 3.3.20.

E 3.4.7. Entenda a interpretação geométrica ao método de Newton. Encontre uma valor para iniciar o método de Newton aplicado ao problema $f(x) = xe^{-x} = 0$ tal que o esquema iterativo divirja.

E 3.4.8. Aplique o método de Newton à função $f(x) = \frac{1}{x} - u$ e construa um esquema computacional para calcular a inversa de u com base em operações de multiplicação e soma/subtração.

E 3.4.9. Aplique o método de Newton à função $f(x) = x^n - A$ e construa um esquema computacional para calcular $\sqrt[n]{A}$ para $A > 0$ com base em operações de multiplicação e soma/subtração.

E 3.4.10. Considere a função dada por

$$\psi(x) \quad = \quad \ln\left(15 - \ln(x)\right)$$

definida para $x > 0$

a) (1.5) Use o teorema do ponto fixo para provar que se x_0 pertence ao intervalo [1,3], então a sequência dada iterativamente por

$$x^{(n+1)} = \psi(x^{(n)}), n \geq 0$$

converge para o único ponto fixo, x^*, de ψ. Construa a iteração $x^{(n+1)} = \psi(x^{(n)})$ e obtenha numericamente o valor do ponto fixo x^*. Expresse a resposta com 5 algarismos significativos corretos.

b) (1.0) Construa a iteração do método de Newton para encontrar x^*, explicitando a relação de recorrência e iniciando com $x_0 = 2$. Use o Scilab para obter a raiz e expresse a resposta com oito dígitos significativos corretos.

3.5 Método das secantes

O **método das secantes** é uma variação do método de Newton, evitando a necessidade de conhecer-se a derivada analítica de $f(x)$. Dada uma função $f(x)$, a ideia é aproximar sua derivada pela razão fundamental:

$$f'(x) \approx \frac{f(x) - f(x_0)}{x - x_0}, \quad x \approx x_0.$$

Mais precisamente, o método de Newton é uma iteração de ponto fixo da forma:

$$x^{(n+1)} = x^{(n)} - \alpha(x^{(n)}) f(x^{(n)}), \quad n \geq 1,$$

onde $x^{(1)}$ é uma aproximação inicial dada e $\alpha(x^{(n)}) = 1/f'(x^{(n)})$. Usando a aproximação da derivada acima, com $x = x^{(n)}$ e $x_0 = x^{(n-1)}$, temos:

$$\alpha(x^{(n)}) = \frac{1}{f'(x^{(n)})} \approx \frac{x^{(n)} - x^{(n-1)}}{f(x^{(n)}) - f(x^{(n-1)})}.$$

Isto nos motiva a introduzir a **iteração do método das secantes** dada por:

$$x^{(n+1)} = x^{(n)} - f(x^{(n)}) \frac{x^{(n)} - x^{(n-1)}}{f(x^{(n)}) - f(x^{(n-1)})}, \quad n \geq 2.$$

Observe que para inicializarmos a iteração acima precisamos de duas aproximações iniciais, a saber, $x^{(1)}$ e $x^{(2)}$. Maneiras apropriadas de escolher estas aproximações podem ser inferidas da interpretação geométrica do método.

Exemplo 3.5.1. Encontre as raízes de $f(x) = \cos(x) - x$.

Solução. Da inspeção do gráfico das funções $y = \cos(x)$ e $y = x$, sabemos que esta equação possui uma raiz em torno de $x = 0{,}8$. Iniciamos o método com $x_0 = 0{,}7$ e $x_1 = 0{,}8$.

$x^{(n-1)}$	$x^{(n)}$	m	$x^{(n+1)}$
0,7	0,8	$\frac{f(0,8)-f(0,7)}{0,8-0,7} =$ $-1{,}6813548$	$0{,}8 - \frac{f(0,8)}{-1,6813548} =$ 0,7385654
0,8	0,7385654	$-1{,}6955107$	0,7390784
0,7385654	0,7390784	$-1{,}6734174$	0,7390851
0,7390784	0,7390851	$-1{,}6736095$	0,7390851

◊

3.5.1 Interpretação geométrica

Enquanto, o método de Newton está relacionado às retas tangentes ao gráfico da função objetivo $f(x)$, o método das secantes, como o próprio nome indica, está relacionado às retas secantes.

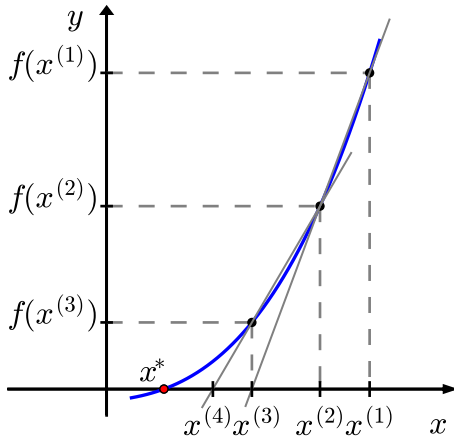


Figura 3.6: Método das secantes.

Sejam $f(x)$ e as aproximações $x^{(1)}$ e $x^{(2)}$ do zero x^* desta função (veja Fi-

gura 3.6). A iteração do método das secantes fornece:

$$x^{(3)} = x^{(2)} - f(x^{(2)})\frac{x^{(2)} - x^{(1)}}{f(x^{(2)}) - f(x^{(1)})}.$$

De fato, $x^{(3)}$ é o ponto de interseção da reta secante ao gráfico de $f(x)$ pelos pontos $x^{(1)}$ e $x^{(2)}$ com o eixo das abscissas. Com efeito, a equação desta reta secante é:

$$y = \frac{f(x^{(2)}) - f(x^{(1)})}{x^{(2)} - x^{(1)}}(x - x^{(2)}) + f(x^{(2)}).$$

Esta reta intercepta o eixo das abscissas no ponto x tal que $y = 0$, i.e.:

$$\frac{f(x^{(2)}) - f(x^{(1)})}{x^{(2)} - x^{(1)}}(x - x^{(2)}) + f(x^{(2)}) \Rightarrow x = x^{(2)} - f(x^{(2)})\frac{x^{(2)} - x^{(1)}}{f(x^{(2)}) - f(x^{(1)})}.$$

3.5.2 Análise de convergência

Uma análise assintótica semelhante aquela feita para o método de Newton nos indica que, para uma função $f(x)$ duas vezes diferenciável, as iterações do método da secante satisfazem:

$$|x^{(n+1)} - x^*| \approx C|x^{(n)} - x^*||x^{(n-1)} - x^*|,$$

para aproximações iniciais suficientemente próximas de x^*, onde $f(x^*) = 0$. Além disso, veremos que:

$$|x^{(n+1)} - x^*| \leq C|x^{(n)} - x^*|^{1,6}$$

sob certas condições. Ou seja, o método das secantes tem **taxa de convergência superlinear**.

Teorema 3.5.1 (Método das secantes)**.** *Seja $f \in C^2([a,b])$ uma função com $x^* \in (a,b)$ tal que $f(x^*) = 0$. Sejam, também:*

$$m := \min_{x\in[a,b]} |f'(x)| > 0 \quad e \quad M := \max_{x\in[a,b]} |f''(x)| < \infty.$$

Além disso, seja $\rho > 0$ tal que:

$$q := \frac{M}{2m}\rho < 1, \quad K_\rho(x^*) := \{x \in \mathbb{R};\ |x - x^*| \leq \rho\} \subset [a,b].$$

*Então, para aproximações iniciais $x^{(1)}, x^{(2)} \in K_\rho(x^*)$, com $x^{(1)} \neq x^{(2)}$, temos que as iterações do método das secantes $x^{(n)} \in K_\rho(x^*)$, $n \geq 1$, e $x^{(n)} \to x^*$, quando $n \to \infty$. Além disso, vale a seguinte estimativa de convergência **a priori**:*

$$|x^{(n)} - x^*| \leq \frac{2m}{M}q^{\gamma_{n-1}}, \quad n \geq 1,$$

onde $\{\gamma_n\}_{n\in\mathbb{N}}$ *é a sequência de Fibonacci*[6][7]*, bem como vale a estimativa* ***a posteriori****:*

$$|x^{(n)} - x^*| \leq \frac{M}{2m}|x^{(n)} - x^{(n-1)}||x^{(n-1)} - x^{(n-2)}|, \quad n \geq 3.$$

Demonstração. Sejam $n \in \mathbb{N}$, $n \geq 2$, e $x^{(n)}, x^{(n-1)} \in K_\rho(x^*)$, tal que $x^{(n)} \neq x^{(n-1)}$, $x^{(n)} \neq x^*$ e $x^{(n-1)} \neq x^*$. Seja, também:

$$g(x^{(n)},x^{(n-1)}) := x^{(n)} - f(x^{(n)})\frac{x^{(n)} - x^{(n-1)}}{f(x^{(n)}) - f(x^{(n-1)})}.$$

Com isso, temos:

$$\begin{aligned} g(x^{(n)},x^{(n-1)}) - x^* &= x^{(n)} - f(x^{(n)})\frac{x^{(n)} - x^{(n-1)}}{f(x^{(n)}) - f(x^{(n-1)})} - x^* \\ = \quad & \frac{x^{(n)} - x^{(n-1)}}{f(x^{(n)}) - f(x^{(n-1)})}\left\{(x^{(n)} - x^*)\frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} - f(x^{(n)}) + f(x^*)\right\}. \end{aligned}$$

Então, da cota assumida para primeira derivada de $f(x)$ e do Teorema do valor médio, temos:

$$|g(x^{(n)},x^{(n-1)}) - x^*| \leq \frac{|x^{(n)} - x^*|}{m}\left|\frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} - \frac{f(x^{(n)}) - f(x^*)}{x^{(n)} - x^*}\right|. \quad (3.5)$$

Agora, iremos estimar este último termo a direita. Para tanto, começamos observando que da expansão em polinômio de Taylor de ordem 0 da função $f(x)$ com resto na forma integral, temos:

$$\begin{aligned} \frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} &= -\int_0^1 \tfrac{d}{dr} f(x^{(n)} + r(x^{(n-1)} - x^{(n)}))\tfrac{dr}{x^{(n)} - x^{(n-1)}} \\ &= \int_0^1 f'(x^{(n)} + r(x^{(n-1)} - x^{(n)}))\, dr \end{aligned}$$

De forma análogo, temos:

$$\frac{f(x^{(n)}) - f(x^*)}{x^{(n)} - x^*} = \int_0^1 f'(x^{(n)} + r(x^* - x^{(n)}))\, dr$$

Logo, temos:

$$\begin{aligned} &\frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} - \frac{f(x^{(n)}) - f(x^*)}{x^{(n)} - x^*} = \\ &\int_0^1 \left[f'(x^{(n)} + r(x^{(n-1)} - x^{(n)})) - f'(x^{(n)} + r(x^* - x^{(n)}))\right] dr. \end{aligned} \quad (3.6)$$

[6]Leonardo Fibonacci, c. 1170 - c. 1250, matemático italiano.

[7]A sequência de Fibonacci $\{\gamma_n\}_{n\in\mathbb{N}}$ é definida por $\gamma_0 = \gamma_1 = 1$ e $\gamma_{n+1} = \gamma_n - \gamma_{n-1}$, $n \geq 1$.

Agora, novamente temos:

$$\begin{aligned}
&f'(x^{(n)} + r(x^{(n-1)} - x^{(n)})) - f'(x^{(n)} + r(x^* - x^{(n)})) \\
&= \int_0^r \frac{d}{ds} f'(x^{(n)} + r(x^{(n-1)} - x^{(n)}) + s(x^* - x^{(n-1)}))\, ds \\
&= \int_0^r f''(x^{(n)} + r(x^{(n-1)} - x^{(n)}) + s(x^* - x^{(n-1)}))\, ds(x^* - x^{(n-1)}).
\end{aligned}$$

Então, retornando à Equação (3.6) e usando a assumida cota para a segunda derivada, obtemos:

$$\left| \frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} - \frac{f(x^{(n)}) - f(x^*)}{x^{(n)} - x^*} \right| \leq \frac{M}{2} |x^{(n-1)} - x^*|.$$

Agora, retornando à Equação (3.5), obtemos:

$$|g(x^{(n)}, x^{(n-1)}) - x^*| \leq \frac{M}{2m} |x^{(n)} - x^*||x^{(n-1)} - x^*| \leq \frac{M}{2m} \rho^2 < \rho.$$

Portanto, concluímos que as iterações do método da secantes $x^{(n)}$ permanecem no conjunto $K_\rho(x^*)$, se começarem nele. Além disso, temos demonstrado que:

$$|x^{(n+1)} - x^*| \leq \frac{M}{2m} |x^{(n)} - x^*||x^{(n-1)} - x^*|.$$

Com isso, temos:

$$\rho_n := \frac{M}{2m} |x^{(n)} - x^*| \Rightarrow \rho_{n+1} \leq \rho_n \rho_{n-1}, \quad n \geq 2.$$

Como $\rho_1 \leq q$ e $\rho_2 \leq q$, temos $\rho_n \leq q^{\gamma_{n-1}}$, $n \geq 1$. Isto mostra a estimativa de convergência **a priori**:

$$|x^n - x^*| \leq \frac{2m}{M} q^{\gamma_{n-1}}.$$

Além disso, como $\gamma_n \to \infty$ quando $n \to \infty$ e $q < 1$, temos que as iterações do método das secantes $x^{(n)} \to x^*$ quando $n \to \infty$.

Por fim, mostramos a estimativa de convergência **a posteriori**. Para tanto, da cota assumida para a primeira derivada e do Teorema do valor médio, temos, para $n \geq 3$:

$$\begin{aligned}
|x^{(n)} - x^*| &\leq \frac{1}{m} |f(x^{(n)} - f(x^*)| \\
&= \frac{1}{m} \left| f(x^{(n-1)}) + (x^{(n)} - x^{(n-1)}) \frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} \right| \\
&= \frac{1}{m} \left| x^{(n)} - x^{(n-1)} \right| \left| \frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} + \frac{f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} \right|.
\end{aligned}$$

Agora, da iteração do método das secantes:

$$x^{(n)} = x^{(n-1)} - f(x^{(n-1)})\frac{x^{(n-1)} - x^{(n-2)}}{f(x^{(n-1)}) - f(x^{(n-2)})},$$

temos:

$$\frac{f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}} = -\frac{f(x^{(n-1)}) - f(x^{(n-2)})}{x^{(n-1)} - x^{(n-2)}}.$$

Logo:

$$|x^{(n)} - x^*| \leq \frac{1}{m}|x^{(n)} - x^{(n-1)}|\left|\frac{f(x^{(n-1)}) - f(x^{(n)})}{x^{(n-1)} - x^{(n)}} - \frac{f(x^{(n-1)}) - f(x^{(n-2)})}{x^{(n-1)} - x^{(n-2)}}\right|$$

Observamos que o último termo pode ser estimado como feito acima para o termo análogo na Inequação (3.5). Com isso, obtemos a estimativa desejada:

$$|x^{(n)} - x^*| \leq \frac{M}{2m}|x^{(n)} - x^{(n-1)}||x^{(n)} - x^{(n-2)}|.$$

□

Proposição 3.5.1 (Sequência de Fibonacci)**.** *A sequência de Fibonacci* $\{\gamma_n\}_{n\in\mathbb{N}}$ *é assintótica a* $\gamma_n \sim \lambda_1^{n+1}/\sqrt{5}$ *e:*

$$\lim_{n\to\infty}\frac{\gamma_{n+1}}{\gamma_n} = \lambda_1,$$

onde $\lambda_1 = (1+\sqrt{5})/2 \approx 1{,}618$ *é a porção áurea.*

Demonstração. A sequência de Fibonacci $\{\gamma_n\}_{n\in\mathbb{N}}$ é definida por $\gamma_0 = \gamma_1 = 1$ e $\gamma_{n+1} = \gamma_n + \gamma_{n-1}$, $n \geq 1$. Logo, satisfaz a seguinte equação de diferenças:

$$\gamma_{n+2} - \gamma_{n+1} - \gamma_n = 0, \quad n \in \mathbb{N}.$$

Tomando $\gamma_n = \lambda^n$, $\lambda \neq 0$ temos:

$$\lambda^n\left(\lambda^2 - \lambda - 1\right) = 0 \Rightarrow \lambda^2 - \lambda - 1 = 0 \Rightarrow \lambda_{1,2} = \frac{1 \pm \sqrt{5}}{2}.$$

Portanto, $\gamma_n = c_1\lambda_1^n + c_2\lambda_2^n$. Como $\gamma_0 = \gamma_1 = 1$, as constantes satisfazem:

$$\begin{aligned} c_1 + c_2 &= 1 \\ c_1\lambda_1 + c_2\lambda_2 &= 1 \end{aligned} \Rightarrow c_1 = \frac{1+\sqrt{5}}{2\sqrt{5}}, \quad c_2 = -\frac{1-\sqrt{5}}{2\sqrt{5}}.$$

Ou seja, obtemos a seguinte forma explícita para os números de Fibonacci:

$$\gamma_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right].$$

Daí, segue imediatamente o enunciado. □

Observação 3.5.1. Sob as hipóteses do Teorema 3.5.1 e da Proposição 3.5.1, temos:

$$\begin{aligned}\lim_{n\to\infty}\frac{|x^{(n+1)}-x^*|}{|x^{(n)}-x^*|^{\lambda_1}} &\leq \lim_{n\to\infty}\frac{M}{2m}|x^{(n)}-x^*|^{1-\lambda_1}|x^{(n-1)}-x^*| \\ &\leq \lim_{n\to\infty}\left(\frac{2m}{M}\right)^{1-\lambda_1} q^{(2-\lambda_1)\lambda_1^n/\sqrt{5}} = 0.\end{aligned}$$

Isto mostra que o método das secantes (nestas hipóteses) tem taxa de convergência superlinear ($\lambda_1 \approx 1{,}6$).

3.6 Critérios de parada

Quando usamos métodos iterativos precisamos determinar um critério de parada. A Tabela 3.4 indica critérios de parada usuais para os métodos que estudamos neste capítulo.

Tabela 3.4: Quadro comparativo.

Método	Convergência	Erro	Critério de parada
Bisseção	Linear $(p=1)$	$\epsilon_{n+1} = \frac{1}{2}\epsilon$	$\frac{b_n - a_n}{2} <$ erro
Iteração linear	Linear $(p=1)$	$\epsilon_{n+1} \approx \lvert\phi'(x^*)\rvert\varepsilon_n$	$\frac{\lvert\Delta_n\rvert}{1-\frac{\Delta_n}{\Delta_{n-1}}} <$ erro, $\Delta_n < \Delta_{n-1}$
Newton	Quadrática $(p=2)$	$\epsilon_{n+1} \approx \frac{1}{2}\left\lvert\frac{f''(x^*)}{f'(x^*)}\right\rvert\varepsilon_n^2$	$\lvert\Delta_n\rvert <$ erro
Secante	$p = \frac{\sqrt{5}+1}{2} \approx 1{,}618$	$\varepsilon_{n+1} \approx \left\lvert\frac{f''(x^*)}{f'(x^*)}\right\rvert\varepsilon_n\varepsilon_{n-1} \approx M\varepsilon_n^{\phi}$	$\lvert\Delta_n\rvert <$ erro

Observação 3.6.1. O erro na tabela sempre se refere ao erro absoluto esperado. Nos três últimos métodos, é comum que se exija como critério de parada que a

condição seja satisfeita por alguns poucos passos consecutivos. Outros critérios podem ser usados. No métodos das secantes, deve-se ter o cuidado de evitar divisões por zero quando $x_{n+1} - x_n$ muito pequeno em relação à resolução do sistema de numeração.

Exercícios

E 3.6.1. Refaça as questões **??**, 3.4.2, 3.4.3 e 3.4.4, usando o método das secantes.

E 3.6.2. Dê uma interpretação geométrica ao método das secantes. Qual a vantagem do método das secantes sobre o método de Newton?

E 3.6.3. Aplique o método das secantes para resolver a equação

$$e^{-x^2} = 2x$$

E 3.6.4. Refaça o problema 3.2.7 usando o método de Newton e das secantes.

E 3.6.5. Seja dada uma função $f(x)$ duas vezes continuamente diferenciável. Faça uma análise assintótica para mostrar que as iterações do método das secantes satisfazem:

$$|x^{(n+1)} - x^*| \approx C|x^{(n)} - x^*||x^{(n-1)} - x^*|,$$

para aproximações iniciais $x^{(1)}$ e $x^{(2)}$ suficientemente próximas de x^*, onde $f(x^*) = 0$.

3.7 Exercícios finais

E 3.7.1. A equação

$$\cos(\pi x) = e^{-2x}$$

tem infinitas raízes. Usando métodos numéricos encontre as primeiras raízes dessa equação. Verifique a j-ésima raiz (z_j) pode ser aproximada por $j - 1/2$ para j grande. Use o método de Newton para encontrar uma aproximação melhor para z_j.

E 3.7.2. A corrente elétrica, I, em Ampères em uma lâmpada em função da tensão elétrica, V, é dada por

$$I = \left(\frac{V}{150}\right)^{0.8}$$

Qual a potência da lâmpada quando ligada em série com uma resistência de valor R a uma fonte de 150V quando. (procure erro inferior a 1%)

a) $R = 0\Omega$

b) $R = 10\Omega$

c) $R = 50\Omega$

d) $R = 100\Omega$

E) $R = 500\Omega$

E 3.7.3. (Bioquímica) A concentração sanguínea de um medicamente é modelado pela seguinte expressão

$$c(t) = Ate^{-\lambda t}$$

onde $t > 0$ é o tempo em minutos decorrido desde a administração da droga. A é a quantidade administrada em mg/ml e λ é a constante de tempo em $\min^{-1}$. Responda:

a) Sendo $\lambda = 1/3$, em que instantes de tempo a concentração é metade do valor máximo. Calcule com precisão de segundos.

b) Sendo $\lambda = 1/3$ e $A = 100mg/ml$, durante quanto tempo a concentração permanece maior que $10mg/ml$.

E 3.7.4. Considere o seguinte modelo para crescimento populacional em um país:

$$P(t) = A + Be^{\lambda t}.$$

onde t é dado em anos. Use t em anos e $t = 0$ para 1960. Encontre os parâmetros A, B e λ com base nos anos de 1960, 1970 e 1991 conforme tabela:

Ano	população
1960	70992343
1970	94508583
1980	121150573
1991	146917459

Use esses parâmetros para calcular a população em 1980 e compare com o valor do censo.

E 3.7.5. Uma boia esférica flutua na água. Sabendo que a boia tem 10ℓ de volume e 2Kg de massa. Calcule a altura da porção molhada da boia.

E 3.7.6. Uma boia cilíndrica tem secção transversal circular de raio 10cm e comprimento 2m e pesa 10Kg. Sabendo que a boia flutua sobre água com o eixo do cilindro na posição horizontal, calcule a altura da parte molhada da boia.

E 3.7.7. Encontre com 6 casas decimais o ponto da curva $y = \ln x$ mais próximo da origem.

E 3.7.8. Um computador é vendido pelo valor a vista de R$2.000,00 ou em 1+15 prestações de R$200,00. Calcule a taxa de juros associada à venda a prazo.

E 3.7.9. O valor de R$110.000,00 é financiado conforme a seguinte programa de pagamentos:

Mês	pagamento
1	20.000,00
2	20.000,00
3	20.000,00
4	19.000,00
5	18.000,00
6	17.000,00
7	16.000,00

Calcule a taxa de juros envolvida. A data do empréstimo é o mês zero.

E 3.7.10. Depois de acionado um sistema de aquecedores, a temperatura em um forno evolui conforme a seguinte equação

$$T(t) = 500 - 800e^{-t} + 600e^{-t/3}.$$

onde T é a temperatura em Kelvin e t é tempo em horas.

a) Obtenha analiticamente o valor de $\lim_{t\to\infty} T(t)$.

b) Obtenha analiticamente o valor máximo de $T(t)$ e o instante de tempo quando o máximo acontece

c) Obtenha numericamente com precisão de minutos o tempo decorrido até que a temperatura passe pela primeira vez pelo valor de equilíbrio obtido no item a.

c) Obtenha numericamente com precisão de minutos a duração do período durante o qual a temperatura permanece pelo menos 20% superior ao valor de equilíbrio.

E 3.7.11. Encontre os pontos onde a elipse que satisfaz $\frac{x^2}{3} + y^2 = 1$ intersepta a parábola $y = x^2 - 2$.

E 3.7.12. Encontre a área do maior retângulo que é possível inscrever entre a curva $e^{-x^2}(1 + \cos(x))$ e o eixo $y = 0$.

E 3.7.13. Uma indústria consome energia elétrica de duas usinas fornecedoras. O custo de fornecimento em reais por hora como função da potência consumida em kW é dada pelas seguintes funções

$$\begin{aligned} C_1(x) &= 500 + .27x + 4.1 \cdot 10^{-5}x^2 + 2.1 \cdot 10^{-7}x^3 + 4.2 \cdot 10^{-10}x^4 \\ C_2(x) &= 1000 + .22x + 6.3 \cdot 10^{-5}x^2 + 8.5 \cdot 10^{-7}x^3 \end{aligned}$$

Onde $C_1(x)$ e $C_2(x)$ são os custos de fornecimento das usinas 1 e 2, respectivamente. Calcule o custo mínimo da energia elétrica quando a potência total consumida é $1500kW$.

E 3.7.14. A pressão de saturação (em bar) de um dado hidrocarboneto pelo ser modelada pela equação de Antoine:

$$\ln\left(P^{sat}\right) = A - \frac{B}{T + C}$$

onde T é a temperatura e A, B e C são constantes dadas conforme a seguir:

Hidrocarboneto	A	B	C
N-pentano	9.2131	2477.07	-39.94
N-heptano	9.2535	2911.32	-56.51

a) Calcule a temperatura de bolha de uma mistura de N-pentano e N-heptano à pressão de 1.2bar quando as frações molares dos gases são $z_1 = z_2 = 0.5$. Para tal utilize a seguinte equação:

$$P = \sum_i z_i P_i^{sat}$$

b) Calcule a temperatura de orvalho de uma mistura de N-pentano e N-heptano à pressão de 1.2bar quando as frações molares dos gases são $z_1 = z_2 = 0.5$. Para tal utilize a seguinte equação:

$$\frac{1}{P} = \sum_i \frac{z_i}{P_i^{sat}}$$

E 3.7.15. Encontre os três primeiros pontos de mínimo da função

$$f(x) = e^{-x/11} + x\cos(2x)$$

para $x > 0$ com erro inferior a 10^{-7}.

Capítulo 4

Solução de sistemas lineares

Muitos problemas da engenharia, física e matemática estão associados à solução de sistemas de equações lineares. Nesse capítulo, tratamos de técnicas numéricas empregadas para obter a solução desses sistemas. Iniciamos por uma rápida revisão do Método de eliminação gaussiana do ponto de vista computacional. No contexto de análise da propagação dos erros de arredondamento, introduzimos o método de eliminação gaussiana com pivotamento parcial, bem como, apresentamos o conceito de condicionamento de um sistema linear. Então, passamos a discutir sobre técnicas iterativos, mais especificamente, sobre os métodos de Jacobi e Gauss-Seidel.

Considere o sistema de equações lineares:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned}$$

onde m é o número de equações e n é o número de incógnitas. Este sistema pode ser escrito na forma matricial:

$$Ax = b$$

onde:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \text{ e } b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Definimos também a matriz completa de uma sistema como $Ax = b$ como $[A|b]$, isto é:

$$[A|b] = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array}\right]$$

Salvo especificado ao contrário, assumiremos ao longo deste capítulo que a matriz dos coeficientes A é uma matriz real não-singular.

Ao longo do capítulo, apresentamos algumas computações com `Python`. Nestas, estaremos assumindo que a biblioteca numpy e seu módulo numpy.linalg estão carregados:

```
>>> from __future__ import division
>>> import numpy as np
>>> from numpy import linalg
```

4.1 Eliminação gaussiana

A **eliminação gaussiana**, também conhecida como **escalonamento**, é um método para resolver sistemas lineares. Este método consiste em manipular o sistema através de determinadas operações elementares, transformanda a matriz que representa o sistema em uma matriz triangular. Uma vez, triangularizado o sistema, a solução pode ser obtida via substituição regressiva. Naturalmente estas operações elementares devem preservar a solução do sistema e consistem em

1. Multiplicação de um linha por uma constante não nula.

2. Substituição de uma linha por ela mesma somada a um múltiplo de outra linha.

3. Permutação de duas linhas.

Exemplo 4.1.1 (Eliminação gaussiana sem pivotamento)**.** Resolva o sistema

$$\begin{aligned} x + y + z &= 1 \\ 2x + y - z &= 0 \\ 4x + 4y + 2z &= 2 \end{aligned}$$

Solução. A matriz completa do sistema é escrita como

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 4 & 4 & 2 & 2 \\ 2 & 1 & -1 & 0 \end{array}\right]$$

No primeiro passo, subtraímos da segunda linha o quádruplo da primeira e subtraímos da terceira linha o dobro da primeira linha:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 0 & -2 & -2 \\ 0 & -1 & -3 & -1 \end{array}\right].$$

No segundo passo, permutamos a segunda linha com a terceira:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -1 & -3 & -1 \\ 0 & 0 & -2 & -2 \end{array}\right].$$

Neste momento, a matriz já se encontra na forma triangular. Da terceira linha, encontramos $-2z = -21$, ou seja, $z = 1$. Substituindo na segunda equação, temos $-y - 3z = -2$, ou seja, $y = -1$ e finalmente, da primeira linha, $x + y + z = 1$, resultando em $x = 1$. ◊

4.1.1 Eliminação gaussiana com pivotamento parcial

A eliminação gaussiana com **pivotamento parcial** consiste em fazer uma permutação de linhas de forma a escolher o maior pivô (em módulo) a cada passo.

Exemplo 4.1.2 (Eliminação gaussiana com pivotamento parcial)**.** Resolva o sistema

$$\begin{aligned} x + y + z &= 1 \\ 2x + y - z &= 0 \\ 2x + 2z + z &= 1 \end{aligned}$$

Solução. A matriz completa do sistema é

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & 0 \\ 2 & 2 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 2 & 1 & -1 & 0 \\ 1 & 1 & 1 & 1 \\ 2 & 2 & 1 & 1 \end{bmatrix}$$
$$\sim \begin{bmatrix} 2 & 1 & -1 & 0 \\ 0 & 1/2 & 3/2 & 1 \\ 0 & 1 & 2 & 1 \end{bmatrix}$$
$$\sim \begin{bmatrix} 2 & 1 & -1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 1/2 & 3/2 & 1 \end{bmatrix}$$
$$\sim \begin{bmatrix} 2 & 1 & -1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1/2 & 1/2 \end{bmatrix}$$

Encontramos $1/2z = 1/2$, ou seja, $z = 1$. Substituímos na segunda equação e temos $y + 2z = 1$, ou seja, $y = -1$ e, finalmente $2x + y - z = 0$, resultando em $x = 1$. ◊

Exemplo 4.1.3. Resolva o sistema por eliminação gaussiana com pivotamento parcial.

$$\begin{bmatrix} 0 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 8 \\ 9 \\ 6 \end{bmatrix}$$

Solução. Construímos a matriz completa:

$$\left[\begin{array}{ccc|c} 0 & 2 & 2 & 8 \\ 1 & 2 & 1 & 9 \\ 1 & 1 & 1 & 6 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 2 & 1 & 9 \\ 0 & 2 & 2 & 8 \\ 1 & 1 & 1 & 6 \end{array}\right]$$
$$\sim \left[\begin{array}{ccc|c} 1 & 2 & 1 & 9 \\ 0 & 2 & 2 & 8 \\ 0 & -1 & 0 & -3 \end{array}\right]$$
$$\sim \left[\begin{array}{ccc|c} 1 & 2 & 1 & 9 \\ 0 & 2 & 2 & 8 \\ 0 & 0 & 1 & 1 \end{array}\right]$$
$$\sim \left[\begin{array}{ccc|c} 1 & 2 & 0 & 8 \\ 0 & 2 & 0 & 6 \\ 0 & 0 & 1 & 1 \end{array}\right]$$
$$\sim \left[\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 0 & 2 & 0 & 6 \\ 0 & 0 & 1 & 1 \end{array}\right]$$

Portanto $x = 2$, $y = 3$ e $z = 1$. ◊

Exemplo 4.1.4 (Problema com elementos com grande diferença de escala)**.**

$$\left[\begin{array}{cc} \varepsilon & 2 \\ 1 & \varepsilon \end{array}\right] \left[\begin{array}{c} x \\ y \end{array}\right] = \left[\begin{array}{c} 4 \\ 3 \end{array}\right]$$

Executamos a eliminação gaussiana sem pivotamento parcial para $\varepsilon \neq 0$ e $|\varepsilon| << 1$:

$$\left[\begin{array}{cc|c} \varepsilon & 2 & 4 \\ 1 & \varepsilon & 3 \end{array}\right] \sim \left[\begin{array}{cc|c} \varepsilon & 2 & 4 \\ 0 & \varepsilon - \frac{2}{\varepsilon} & 3 - \frac{4}{\varepsilon} \end{array}\right]$$

Temos

$$y = \frac{3 - 4/\varepsilon}{\varepsilon - 2/\varepsilon}$$

e

$$x = \frac{4 - 2y}{\varepsilon}$$

Observe que a expressão obtida para y se aproximada de 2 quando ε é pequeno:

$$y = \frac{3 - 4/\varepsilon}{\varepsilon - 2/\varepsilon} = \frac{3\varepsilon - 4}{\varepsilon^2 - 2} \longrightarrow \frac{-4}{-2} = 2, \quad \text{quando } \varepsilon \to 0.$$

Já expressão obtida para x depende justamente da diferença $2 - y$:

$$x = \frac{4 - 2y}{\varepsilon} = \frac{2}{\varepsilon}(2 - y)$$

Assim, quando ε é pequeno, a primeira expressão, implementado em um sistema de ponto flutuante de acurácia finita, produz $y = 2$ e, consequentemente, a expressão para x produz $x = 0$. Isto é, estamos diante um problema de cancelamento catastrófico.

Agora, quando usamos a eliminação gaussiana com pivotamento parcial, fazemos uma permutação de linhas de forma a escolher o maior pivô a cada passo:

$$\left[\begin{array}{cc|c} \varepsilon & 2 & 4 \\ 1 & \varepsilon & 3 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & \varepsilon & 3 \\ \varepsilon & 2 & 4 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & \varepsilon & 3 \\ 0 & 2 - \varepsilon^2 & 4 - 3\varepsilon \end{array}\right]$$

Continuando o procedimento, temos:

$$y = \frac{4 - 4\varepsilon}{2 - \varepsilon^2}$$

e

$$x = 3 - \varepsilon y$$

Observe que tais expressões são analiticamente idênticas às anteriores, no entanto, são mais estáveis numericamente. Quando ε converge a zero, y converge a 2, como no caso anterior. No entanto, mesmo que $y = 2$, a segunda expressão produz $x = 3 - \varepsilon y$, isto é, a aproximação $x \approx 3$ não depende mais de obter $2 - y$ com precisão.

Exercícios

E 4.1.1. Resolva o seguinte sistema de equações lineares

$$\begin{aligned} x + y + z &= 0 \\ x + 10z &= -48 \\ 10y + z &= 25 \end{aligned}$$

Usando eliminação gaussiana com pivotamento parcial (não use o computador para resolver essa questão).

E 4.1.2. Resolva o seguinte sistema de equações lineares

$$\begin{aligned} x + y + z &= 0 \\ x + 10z &= -48 \\ 10y + z &= 25 \end{aligned}$$

Usando eliminação gaussiana com pivotamento parcial (não use o computador para resolver essa questão).

E 4.1.3. Calcule a inversa da matriz

$$A = \begin{bmatrix} 1 & 2 & -1 \\ -1 & 2 & 0 \\ 2 & 1 & -1 \end{bmatrix}$$

usando eliminação gaussiana com pivotamento parcial.

E 4.1.4. Demonstre que se $ad \neq bc$, então a matriz A dada por:

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

é inversível e sua inversa é dada por:

$$A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

4.2 Complexidade de Algoritmos em Álgebra Linear

Dados dois algoritmos diferentes para resolver o mesmo problema, como podemos escolher qual desses algoritmos é o melhor? Se pensarmos em termos de **eficiência** (ou custo computacional), queremos saber qual desses algoritmos consome menos recursos para realizar a mesma tarefa.

Em geral podemos responder essa pergunta de duas formas: em termos de tempo ou de espaço.

Quando tratamos de **eficiência espacial**, queremos saber quanta memória (em geral RAM) é utilizada pelo algoritmo para armazenar os dados, sejam matrizes, vetores ou escalares.

Quando tratamos de **eficiência temporal**, queremos saber quanto tempo um algoritmo leva para realizar determinada tarefa. Vamos nos concentrar nessa segunda opção, que em geral é a mais difícil de ser respondida.

Obviamente o tempo vai depender do tipo de computador utilizado. É razoável de se pensar que o tempo vai ser proporcional ao número de operações de ponto flutuante (flops) feito pelo algoritmo (observe que o tempo total não depende apenas disso, mas também de outros fatores como memória, taxas de transferências de dados da memória para o cpu, redes,...). Entretanto vamos nos concentrar na contagem do número de operações (flops) para realizar determinada tarefa.

No passado (antes dos anos 80), os computadores demoravam mais tempo para realizar operações como multiplicação e divisão, se comparados a adição ou subtração. Assim, em livros clássicos eram contados apenas o custo das operações $\times$ e $/$. Nos computadores atuais as quatro operações básicas levam o mesmo tempo. Entretanto, na maioria dos algoritmos de álgebra linear o custo associado as multiplicações e divisões é proporcional ao custo das somas e subtrações (pois a maioria dessas operações podem ser escritas como a combinação de produtos internos). Dessa forma, na maior parte deste material levaremos em conta somente multiplicações e divisões, a não ser que mencionado o contrário.

Tenha em mente que a ideia é estimar o custo a medida que o tamanho dos vetores e matrizes cresce muito (para n grande).

Exemplo 4.2.1 (Produto escalar-vetor)**.** Qual o custo para multiplicar um escalar por um vetor?

Solução. Seja $a \in \mathbf{R}$ e $\vec{x} \in \mathbf{R}^n$, temos que

$$a\vec{x} = [a \times x_1, a \times x_2, ...,a \times x_n] \tag{4.1}$$

usando n multiplicações, ou seja, um custo computacional, C, de

$$C = n \text{ flops.} \tag{4.2}$$

$\Diamond$

Exemplo 4.2.2 (Produto vetor-vetor)**.** Qual o custo para calcular o produto interno $\vec{x} \cdot \vec{y}$?

Solução. Sejam $\vec{x}, \vec{y} \in \mathbf{R}^n$, temos que

$$\vec{x} \cdot \vec{y} = x_1 \times y_1 + x_2 \times y_2 + ... + x_n \times y_n \tag{4.3}$$

São realizadas n multiplicações (cada produto x_i por y_i) e $n-1$ somas, ou seja, o custo total de operações é de

$$C := (n) + (n-1) = 2n - 1 \text{ flops} \tag{4.4}$$

$\Diamond$

Exemplo 4.2.3 (Produto matriz-vetor)**.** Qual o custo para calcular o produto de matriz por vetor $A\vec{x}$?

Solução. Sejam $A \in \mathbf{R}^{n \times n}$ e $\vec{x} \in \mathbf{R}^n$, temos que

$$\begin{bmatrix} a_{11} & a_{12} & \cdots a_{1n} \\ \vdots & & \vdots \\ a_{n1} & & \cdots a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11} \times x_1 + a_{12}x_2 + ... + a_{1n} \times x_n \\ \vdots \\ a_{n1} \times x_1 + a_{n2}x_2 + ... + a_{nn} \times x_n \end{bmatrix} \tag{4.5}$$

Para obter o primeiro elemento do vetor do lado direito devemos multiplicar a primeira linha de A pelo vetor coluna $\vec{x}$. Note que esse é exatamente o custo do produto vetor-vetor do exemplo anterior. Como o custo para cada elemento do vetor do lado direito é o mesmo e temos n elementos, teremos que o custo para multiplicar matriz-vetor é[1]

$$C := n \cdot (2n - 1) = 2n^2 - n \text{ flops.} \tag{4.7}$$

A medida que $n \to \infty$, temos

$$\mathcal{O}(2n^2 - n) = \mathcal{O}(2n^2) = \mathcal{O}(n^2) \text{ flops.} \tag{4.8}$$

$\Diamond$

Exemplo 4.2.4 (Produto matriz-matriz)**.** Qual o custo para calcular o produto de duas matrizes AB?

[1] Contando apenas multiplicações/divisões obtemos

$$n \cdot \mathcal{O}(n) = \mathcal{O}(n^2) \text{ flops.} \tag{4.6}$$

Solução. Sejam $A, B \in \mathbf{R}^{n\times n}$ temos que

$$\begin{bmatrix} a_{11} & a_{12} & \cdots a_{1n} \\ \vdots & & \vdots \\ a_{n1} & & \cdots a_{nn} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots a_{1n} \\ \vdots & & \vdots \\ b_{n1} & & \cdots b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots c_{1n} \\ \vdots & & \vdots \\ c_{n1} & & \cdots c_{nn} \end{bmatrix} \tag{4.9}$$

onde o elemento d_{ij} é o produto da linha i de A pela coluna j de B,

$$d_{ij} = a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + ... + a_{i2} \times b_{2j} \tag{4.10}$$

Note que esse produto tem o custo do produto vetor-vetor, ou seja, $2n-1$. Como temos $n \times n$ elementos em D, o custo total para multiplicar duas matrizes é[2]

$$C = n \times n \times (2n-1) = 2n^3 - n^2 \text{ flops.} \tag{4.12}$$

$\Diamond$

4.3 Sistemas triangulares

Considere um sistema linear onde a matriz é triangular superior, ou seja,

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

tal que todos elementos abaixo da diagonal são iguais a zero.

Podemos resolver esse sistema iniciando pela última equação e isolando x_n obtemos

$$x_n = b_n/a_{nn} \tag{4.13}$$

Substituindo x_n na penúltima equação

$$a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n = b_{n-1} \tag{4.14}$$

[2]Contando apenas $\times$ e / obtemos

$$n \times n \times (n) = n^3 \text{ flops.} \tag{4.11}$$

e isolando x_{n-1} obtemos

$$x_{n-1} = (b_{n-1} - a_{n-1,n}x_n)/a_{n-1,n-1} \tag{4.15}$$

e continuando desta forma até a primeira equação obteremos

$$x_1 = (b_1 - a_{12}x_2 \cdots - a_{1n}x_n)/a_{11}. \tag{4.16}$$

De forma geral, temos que

$$x_i = (b_i - a_{i,i+1}x_{i+1} \cdots - a_{i,n}x_n)/a_{i,i}, \quad i = 2, \ldots, n. \tag{4.17}$$

4.3.1 Algoritmo para resolução de um sistema triangular superior

Para resolver um sistema triangular superior iniciamos da última linha em direção a primeira.

```
1.  function [x]=solveU(U,b)  // U:= matriz triangular superior
2.      n=size(U,1)           // b:= vetor
3.      x(n)=b(n)/U(n,n)
4.      for i=n-1:-1:1
5.          x(i)=(b(i)-U(i,i+1:n)*x(i+1:n)  )/U(i,i)
6.      end
7.  endfunction
```

4.3.2 Algoritmo para resolução de um sistema triangular inferior

Para resolver um sistema triangular inferior podemos fazer o processo inverso iniciando da primeira equação.

```
1. function [x]=solveL(L,b)   // L: matriz triangular inferior
2.    n=size(L,1)             // b: vetor
3.    x(1)=b(1)/L(1,1)
4.    for i=2:n
5.        x(i)=(b(i)-L(i,1:i-1)*x(1:i-1)  )/L(i,i)
6.    end
7.endfunction
```

Custo computacional

Vamos contar o número total de flops para resolver um sistema triangular inferior. Note que o custo para um sistema triangular superior será o mesmo.

Na linha 3, temos uma divisão, portanto 1 flop.

Na linha 5 quando $i = 2$, temos

`x(2)=(b(2)-L(2,1:1)*x(1:1))/L(2,2)`,

ou seja, 1 subtração+1 multiplicação + 1 divisão = 3 flops.

Quando $i = 3$,

`x(3)=(b(3)-L(3,1:2)*x(1:2))/L(3,3)`

temos 1 subtração+(2 multiplicações + 1 soma) +1 divisão = 5 flops.

Quando $i = 4$, temos 1 subtração+(3 multiplicações + 2 somas) +1 divisão = 7 flops.

Até que para $i = n$, temos

`x(n)=(b(n)-L(n,1:n-1)*x(1:n-1))/L(n,n)`,

com 1 subtração+($n-1$ multiplicações + $n-2$ somas) + 1 divisão, ou seja, $1 + (n - 1 + n - 2) + 1 = 2n - 1$ flops.

Somando todos esses custos[3] temos que o custo para resolver um sistema triangular inferior é

$$1 + 3 + 5 + 7 + ... + 2n - 1 = \sum_{k=1}^{n}(2k - 1) = 2\sum_{k=1}^{n} k - \sum_{k=1}^{n} 1 \tag{4.19}$$

e utilizando que a soma dos k inteiros é uma progressão aritmética[4]

$$2(n(n + 1)/2) - n = n^2 \text{ flops.} \tag{4.20}$$

4.4 Fatoração LU

Considere um sistema linear onde a matriz A é densa[5]. Para resolver o sistema, podemos transformar a matriz A nas matrizes L, triangular inferior, e U, triangular superior de tal forma que $A = LU$.

[3] Contando apenas multiplicações/divisões obtemos

$$(n^2 + n)/2 \text{ flops.} \tag{4.18}$$

[4] Temos que $\sum_{k=1}^{n} k = n(n + 1)/2, \qquad \sum_{k=1}^{n} 1 = n$

[5] Diferentemente de uma matriz esparsa, uma matriz densa possui a maioria dos elementos diferentes de zero.

Sendo assim o sistema pode ser reescrito tal que

$$\begin{aligned} Ax &= b \\ (LU)x &= b \\ L(Ux) &= b \\ Ly = b \quad &\text{e} \quad Ux = y \end{aligned}$$

Assim ao invés de resolvermos o sistema original, devemos resolver um sistema triangular inferior e um sistema triangular superior.

A matriz U da fatoração[6] LU é a matriz obtida ao final do escalonamento da matriz A.

A matriz L inicia igual a identidade I. Os elementos da matriz L são os múltiplos do primeiro elemento da linha de A a ser zerado dividido pelo pivô acima na mesma coluna.

Por exemplo, para zerar o primeiro elemento da segunda linha de A, calculamos

$$L_{21} = A_{21}/A_{11}$$

e fazemos

$$A_{2,:} \Leftarrow A_{2,:} - L_{21}A_{1,:}$$

Note que usaremos $A_{i,:}$ para nos referenciarmos a linha i de A. Da mesma forma, se necessário usaremos $A_{:,j}$ para nos referenciarmos a linha j de A.

Para zerar o primeiro elemento da terceira linha de A, temos

$$L_{31} = A_{31}/A_{11}$$

e fazemos

$$A_{3,:} \Leftarrow A_{3,:} - L_{31}A_{1,:}$$

até chegarmos ao último elemento da primeira coluna de A.

Repetimos o processo para as próximas colunas, escalonando a matriz A e coletando os elementos L_{ij} abaixo da diagonal[7].

4.4.1 Algoritmo para fatoração LU

O algoritmo para fatoração LU pode ser escrito como

[6] Não vamos usar pivotamento nesse primeiro exemplo.

[7] Perceba que a partir da segunda coluna para calcular L_{ij} não usamos os elementos de A, mas os elementos da matriz A em processo de escalonamento

```
 1. function [L,A]=fatoraLU(A)
 2.     n=size(A,1)
 3.     L=eye(n,n)
 4.     for j=1:n-1
 5.         for i=j+1:n
 6.             L(i,j    )=A(i,j)/A(j,j)
 7.             A(i,j+1:n)=A(i,j+1:n)-L(i,j)*A(j,j+1:n)
 8.             A(i,j    )=0
 9.         end
10.     end
11. endfunction
```

Custo computacional

Podemos analisar o custo computacional reduzindo o problema em problemas menores.

Na linha 4, iniciamos com $j = 1$. Desta forma i varia de 2 até n na linha 5.

A linha 6 terá sempre 1 flop.

A linha 7, com $j = 1$ tem um bloco de tamanho `2:n` contabilizando $n - 1$ flops do produto e $n - 1$ flops da subtração.

Nas linhas 6-8 são feitas $(2(n-1)+1) = 2n-1$ flops independente do valor de i. Como i varia de 2 até n, teremos que o bloco é repetido $n - 1$ vezes, ou seja, o custo das linhas 5-9 é

$$(n-1) \times (2(n-1)+1) = 2(n-1)^2 + (n-1) \tag{4.21}$$

Voltamos a linha 4 quando $j = 2$. Das linhas 6-8 teremos $n - 2$ flops (o bloco terá um elemento a menos) que será repetido $n - 2$ vezes, pois `i=3:n`, ou seja,

$$(n-2) \times (2(n-2)+1) = 2(n-2)^2 + (n-2) \tag{4.22}$$

Para $j = 3$, temos $2(n-3)^2 + (n-3)$.

Para $j = n - 2$, temos $2(2)^2 + 2$.

Finalmente, para $j = n - 1$, temos $2 \cdot 1^2 + 1$.

Somando todos esses custos, temos

$$\begin{aligned}(n-1)+2(n-1)^2 &+ (n-2)+2(n-2)^2+...+(2)+2(2)^2+1+2\cdot 1\\ &= \sum_{k=1}^{n-1} 2k^2+k\\ &= 2\sum_{k=1}^{n-1} k^2+\sum_{k=1}^{n-1} k\\ &= 2\frac{(n-1)n(2n-1)}{6}+\frac{n(n-1)}{2}\\ &= \frac{2n^3}{3}-\frac{n^2}{2}-\frac{n}{6} \text{ flops.}\end{aligned}$$

4.4.2 Custo computacional para resolver um sistema linear usando fatoração LU

Para calcularmos o custo computacional de um algoritmo completo, uma estratégia é separar o algoritmo em partes menores mais fáceis de calcular.

Para resolver o sistema, devemos primeiro fatorar a matriz A nas matrizes L e U. Vimos que o custo é

$$\frac{2n^3}{3}-\frac{n^2}{2}-\frac{n}{6} \text{ flops.}$$

Depois devemos resolver os sistemas $Ly=b$ e $Ux=y$. O custo de resolver os dois sistemas é (devemos contar duas vezes)

$$2n^2 \text{ flops.}$$

Somando esses 3 custos, temos que o custo para resolver um sistema linear usando fatoração LU é

$$\frac{2n^3}{3}+\frac{3n^2}{2}-\frac{n}{6} \text{ flops.}$$

Quando n cresce, prevalessem os termos de mais alta ordem, ou seja,

$$\mathcal{O}(\frac{2n^3}{3}+\frac{3n^2}{2}-\frac{n}{6})=\mathcal{O}(\frac{2n^3}{3}+\frac{3n^2}{2})=\mathcal{O}(\frac{2n^3}{3})$$

4.4.3 Custo para resolver m sistemas lineares

Devemos apenas multiplicar m pelo custo de resolver um sistema linear usando fatoração LU, ou seja, o custo será

$$m(\frac{2n^3}{3}+\frac{3n^2}{2}-\frac{n}{6})=\frac{2mn^3}{3}+\frac{3mn^2}{2}-\frac{mn}{6}$$

e com $m = n$ temos

$$\frac{2n^4}{3} + \frac{3n^3}{2} - \frac{n^2}{6}.$$

Porém, se estivermos resolvendo n sistemas com *a mesma matriz* A (e diferente lado direito $\vec{b}$ para cada sistema) podemos fazer a fatoração LU uma única vez e contar apenas o custo de resolver os sistemas triangulares obtidos.

Custo para fatoração LU de A: $\frac{2n^3}{3} - \frac{n^2}{2} - \frac{n}{6}$.

Custo para resolver m sistemas triangulares inferiores: mn^2.

Custo para resolver m sistemas triangulares superiores: mn^2.

Somando esses custos obtemos

$$\frac{2n^3}{3} - \frac{n^2}{2} - \frac{n}{6} + 2mn^2$$

que quando $m = n$ obtemos

$$\frac{8n^3}{3} - \frac{n^2}{2} - \frac{n}{6} \text{ flops.}$$

4.4.4 Custo para calcular a matriz inversa de A

Como vemos em Álgebra Linear, um método para obter a matriz A^{-1} é realizar o escalonamento da matriz $[A|I]$ onde I é a matriz identidade. Ao terminar o escalonamento, o bloco do lado direito conterá A^{-1}.

Isto é equivalente a resolver n sistemas lineares com a mesma matriz A e os vetores da base canônica $\vec{e}_i = [0,...,0,1,0,....0]^T$ tal que

$$A\vec{x}_i = \vec{e}_i, \qquad i = 1 : n$$

onde $\vec{x}_i$ serão as colunas da matriz A inversa, já que $AX = I$.

O custo para resolver esses n sistemas lineares foi calculado na seção anterior como

$$\frac{8n^3}{3} - \frac{n^2}{2} - \frac{n}{6}.$$

Exemplo 4.4.1. Qual o melhor método para resolver um sistema linear: via fatoração LU ou calculando a inversa de A e obtendo $x = A^{-1}b$?

4.5 Condicionamento de sistemas lineares

Quando lidamos com matrizes no corpo do números reais (ou complexos), existem apenas duas alternativas: i) a matriz é inversível; ii) a matriz não é inversível e, neste caso, é chamada de matriz singular. Ao lidarmos em aritmética de

precisão finita, encontramos uma situação mais sutil: alguns problema lineares são mais difíceis de serem resolvidos, pois os erros de arredondamento se propagam de forma mais significativa que em outros problemas. Neste caso falamos de problemas bem-condicionados e mal-condicionados. Intuitivamente falando, um problema bem-condicionado é um problema em que os erros de arredondamento se propagam de forma menos importante; enquanto problemas mal-condicionados são problemas em que os erros se propagam de forma mais relevante.

Um caso típico de sistema mal-condicionado é aquele cujos coeficiente estão muito próximos ao de um problema singular. Considere o seguinte exemplo:

Exemplo 4.5.1. Observe que o sistema

$$\begin{bmatrix} 71 & 41 \\ \lambda & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 100 \\ 70 \end{bmatrix} \tag{4.23}$$

é impossível quando $\lambda = \frac{71\times 30}{41} \approx 51{,}95122$.

Considere os próximos três sistemas:

a) $\begin{bmatrix} 71 & 41 \\ 51 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 100 \\ 70 \end{bmatrix}$, com solução $\begin{bmatrix} 10/3 \\ -10/3 \end{bmatrix}$,

b) $\begin{bmatrix} 71 & 41 \\ 52 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 100 \\ 70 \end{bmatrix}$, com solução $\begin{bmatrix} -65 \\ 115 \end{bmatrix}$,

c) $\begin{bmatrix} 71 & 41 \\ 52 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 100{,}4 \\ 69{,}3 \end{bmatrix}$, com solução $\begin{bmatrix} -85{,}35 \\ 150{,}25 \end{bmatrix}$.

Pequenas variações nos coeficientes das matrizes fazem as soluções ficarem bem distintas, isto é, pequenas variações nos dados de entrada acarretaram em grandes variações na solução do sistema. Quando isso acontece, dizemos que o problema é mal-condicionado.

Precisamos uma maneira de medir essas variações. Como os dados de entrada e os dados de saída são vetores (ou matrizes), precisamos introduzir as definições de norma de vetores e matrizes.

4.5.1 Norma de vetores

Definimos a **norma** L^p, $1 \leq p \leq \infty$, de um vetor em $v = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$ por:

$$\|v\|_p := \left(\sum_{i=1}^{n} |v_i|^p \right)^{1/p} = \left(|v_1|^p + |v_2|^p + \cdots + |v_n|^p \right)^{1/p}, \quad 1 \leq p < \infty.$$

Para $p = \infty$, definimos a norma L^∞ (**norma do máximo**) por:

$$\|v\|_\infty = \max_{1 \leq j \leq n} \{|v_j|\}.$$

Proposição 4.5.1 (Propriedades de normas)**.** *Sejam dados $\alpha \in \mathbb{R}$ um escalar e os vetores u,$v \in \mathbb{R}^n$. Então, para cada $1 \leq p \leq \infty$, valem as seguintes propriedades:*

a) $\|u\|_p = 0 \Leftrightarrow u = 0$.

b) $\|\alpha u\|_p = |\alpha| \|u\|_p$.

c) $\|u + v\|_p \leq \|u\|_p + \|v\|_p$ *(**desigualdade triangular**).*

d) $\|u\|_p \to \|u\|_\infty$ *quando* $p \to \infty$.

Demonstração. Demonstramos cada item em separado.

a) Se $u = 0$, então segue imediatamente da definição da norma L^p, $1 \leq p \leq \infty$, que $\|u\|_p = 0$. Reciprocamente, se $\|u\|_\infty = 0$, então, para cada $i = 1, 2, \ldots, n$, temos:

$$|u_i| \leq \max_{1 \leq j \leq n} \{|u_j|\} = \|u\|_\infty = 0 \Rightarrow u_i = 0.$$

Isto é, $u = 0$. Agora, se $\|u\|_p = 0$, $1 \leq p < \infty$, então:

$$0 = \|u\|_p^p := \sum_{i=1}^{n} |u_i|^p \leq n \|u\|_\infty \Rightarrow \|u\|_\infty = 0.$$

Logo, pelo resultado para a norma do máximo, concluímos que $u = 0$.

b) Segue imediatamente da definição da norma L^p, $1 \leq p \leq \infty$.

c) Em construção ...

d) Em construção ...

□

Exemplo 4.5.2. Calcule a norma L^1, L^2 e L^∞ do vetor coluna $v = (1, 2, -3, 0)$.

Solução.

$$\begin{aligned} \|v\|_1 &= 1+2+3+0=6 \\ \|v\|_2 &= \sqrt{1+2^2+3^2+0^2}=\sqrt{14} \\ \|v\|_\infty &= \max\{1,2,3,0\}=3 \end{aligned}$$

◊

4.5.2 Norma de matrizes

Definimos a norma induzida L^p de uma matriz $A=[a_{i,j}]_{i,j=1}^{n,n}$ da seguinte forma:

$$\|A\|_p = \sup_{\|v\|_p=1} \|Av\|_p,$$

ou seja, a norma p de uma matriz é o máximo valor assumido pela norma de Av entre todos os vetores de norma unitária.

Temos as seguintes propriedades, se A e B são matrizes, I é a matriz identidade, v é um vetor e λ é um real (ou complexo):

$$\begin{aligned} \|A\|_p &= 0 \iff A=0 \\ \|\lambda A\|_p &= |\lambda|\|A\|_p \\ \|A+B\|_p &\leq \|A\|_p+\|B\|_p \quad \text{(desigualdade do triângulo)} \\ \|Av\|_p &\leq \|A\|_p\|v\|_p \\ \|AB\|_p &\leq \|A\|_p\|B\|_p \\ \|I\|_p &= 1 \\ 1 &= \|I\|_p=\|AA^{-1}\|_p \leq \|A\|_p\|A^{-1}\|_p \quad \text{(se A é inversível)} \end{aligned}$$

Casos especiais:

$$\begin{aligned} \|A\|_1 &= \max_{j=1}^{n} \sum_{i=1}^{n} |A_{ij}| \\ \|A\|_2 &= \sqrt{\max\{|\lambda| : \lambda \in \sigma(AA^*)\}} \\ \|A\|_\infty &= \max_{i=1}^{n} \sum_{j=1}^{n} |A_{ij}| \end{aligned}$$

onde $\sigma(M)$ é o conjunto de autovalores da matriz M.

Exemplo 4.5.3. Calcule as normas 1, 2 e ∞ da seguinte matriz:

$$A=\begin{bmatrix} 3 & -5 & 7 \\ 1 & -2 & 4 \\ -8 & 1 & -7 \end{bmatrix}$$

Solução.

$$\begin{aligned} \|A\|_1 &= \max\{12, 8, 18\} = 18 \\ \|A\|_\infty &= \max\{15, 7, 16\} = 16 \\ \|A\|_2 &= \sqrt{\max\{0{,}5865124, 21{,}789128, 195{,}62436\}} = 13{,}98657 \end{aligned}$$

Em `Python` podemos computar normas L^p's de matrizes usando a função numpy.linalg.norm. Neste exemplo, temos:

```
>>> A = np.array([[3,-5,7],
...               [1,-2,4],
...               [-8,1,-7]], dtype='double')
>>> np.linalg.norm(A,1)
18
>>> np.linalg.norm(A,np.inf)
16
>>> np.linalg.norm(A,2)
13.986577820518308
```

$\Diamond$

4.5.3 Número de condicionamento

O condicionamento de um sistema linear é um conceito relacionado à forma como os erros se propagam dos dados de entrada para os dados de saída, ou seja, se o sistema

$$Ax = y$$

possui uma solução x para o vetor y, quando varia a solução x quando o dado de entrado y varia. Consideramos, então, o problema

$$A(x + \delta_x) = y + \delta_y$$

Aqui δ_x representa a variação em x e δ_y representa a respectiva variação em y. Temos:

$$Ax + A\delta_x = y + \delta_y$$

e, portanto,

$$A\delta_x = \delta_y.$$

Queremos avaliar a magnitude do erro relativo em y, representado por $\|\delta_y\|/\|y\|$ em função da magnitude do erro relativo $\|\delta_x\|/\|x\|$.

$$\begin{aligned}\frac{\|\delta_x\|/\|x\|}{\|\delta_y\|/\|y\|} &= \frac{\|\delta_x\|}{\|x\|}\frac{\|y\|}{\|\delta_y\|} \\ &= \frac{\|A^{-1}\delta_y\|}{\|x\|}\frac{\|Ax\|}{\|\delta_y\|} \\ &\leq \frac{\|A^{-1}\|\|\delta_y\|}{\|x\|}\frac{\|A\|\|x\|}{\|\delta_y\|} \\ &= \|A\|\|A^{-1}\|\end{aligned}$$

Assim, definimos o número de condicionamento de uma matriz inversível A como

$$k_p(A) = \|A\|_p\|A^{-1}\|_p$$

O número de condicionamento, então, mede o quão instável é resolver o problema $Ax = y$ frente a erros no vetor de entrada x.

Obs: O número de condicionamento depende da norma escolhida.

Obs: O número de condicionamento da matriz identidade é 1.

Obs: O número de condicionamento de qualquer matriz inversível é igual ou maior que 1.

Exercícios

E 4.5.1. Calcule o valor de λ para o qual o problema

$$\begin{cases} 71x + 41y = 10 \\ \lambda x + 30y = 4 \end{cases}$$

é impossível, depois calcule os números de condicionamento com norma 1,2 e ∞ quando $\lambda = 51$ e $\lambda = 52$.

E 4.5.2. Calcule o número de condicionamento da matriz

$$A = \begin{bmatrix} 3 & -5 & 7 \\ 1 & -2 & 4 \\ -8 & 1 & -7 \end{bmatrix}$$

nas normas 1, 2 e ∞.

E 4.5.3. Calcule o número de condicionamento das matrizes

$$\begin{bmatrix} 71 & 41 \\ 52 & 30 \end{bmatrix}$$

e

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 4 & 5 & 5 \end{bmatrix}$$

usando as normas 1,2 e ∞.

E 4.5.4. Usando a norma 1, calcule o número de condicionamento da matriz

$$A = \begin{bmatrix} 1 & 2 \\ 2+\varepsilon & 4 \end{bmatrix}$$

em função de ε quando $0 < \varepsilon < 1$. Interprete o limite $\varepsilon \to 0$.

E 4.5.5. Considere os sistemas:

$$\begin{cases} 100000x - 9999.99y = -10 \\ -9999.99x + 1000.1y = 1 \end{cases} \quad \text{e} \quad \begin{cases} 100000x - 9999.99y = -9.999 \\ -9999.99x + 1000.1y = 1.01 \end{cases}$$

Encontre a solução de cada um e discuta.

E 4.5.6. Considere os vetores de 10 entradas dados por

$$x_j = \sin(j/10), \quad y_j = j/10 \quad z_j = j/10 - \frac{(j/10)^3}{6}, \quad j = 1, \ldots, 10$$

Use o `Scilab` para construir os seguintes vetores de erro:

$$e_j = \frac{|x_j - y_j|}{|x_j|} \quad f_j = \frac{|x_j - z_j|}{x_j}$$

Calcule as normas 1, 2 e ∞ de e e f

4.6 Métodos iterativos para sistemas lineares

Na seção anterior tratamos de métodos diretos para a resolução de sistemas lineares. Em um **método direto** (por exemplo, solução via fatoração LU) obtemos

uma aproximação da solução depois de realizarmos um número finito de operações (só teremos a solução ao final do processo).

Veremos nessa seção dois **métodos iterativos** básicos para obter uma aproximação para a solução de um sistema linear. Geralmente em um método iterativo iniciamos com uma aproximação para a solução (que pode ser ruim) e vamos melhorando essa aproximação através de sucessivas iterações.

4.6.1 Método de Jacobi

O método de Jacobi pode ser obtido a partir do sistema linear

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= y_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= y_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= y_n \end{aligned}$$

Isolando o elemento x_1 da primeira equação temos

$$x_1^{(k+1)} = \frac{y_1 - \left(a_{12}x_2^{(k)} + \cdots + a_{1n}x_n^{(k)}\right)}{a_{11}} \tag{4.24}$$

Note que utilizaremos os elementos $x_i^{(k)}$ da iteração k (a direita da equação) para estimar o elemento x_1 da próxima iteração.

Da mesma forma, isolando o elemento x_i de cada equação i, para todo $i = 2,...,n$ podemos construir a iteração

$$\begin{aligned} x_1^{(k+1)} &= \frac{y_1 - \left(a_{12}x_2^{(k)} + \cdots + a_{1n}x_n^{(k)}\right)}{a_{11}} \\ x_2^{(k+1)} &= \frac{y_2 - \left(a_{21}x_1^{(k)} + a_{23}x_3^{(k)} + \cdots + a_{2n}x_n^{(k)}\right)}{a_{22}} \\ &\vdots \\ x_n^{(k+1)} &= \frac{y_2 - \left(a_{n1}x_1^{(k)} + \cdots + a_{n,n-2}x_{n-2}^{(k)} + a_{n,n-1}x_{n-1}^{(k)}\right)}{a_{nn}} \end{aligned}$$

Em notação mais compacta, o método de Jacobi consiste na iteração

$$\begin{aligned} x^{(1)} &= \text{aproximação inicial} \\ x_i^{(k)} &= \left(y_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}x_j^{(k)}\right)/a_{ii} \end{aligned}$$

Exemplo 4.6.1. Resolva o sistema

$$\begin{aligned} 10x + y &= 23 \\ x + 8y &= 26 \end{aligned}$$

usando o método de Jacobi iniciando com $x^{(1)} = y^{(1)} = 0$.

$$\begin{aligned} x^{(k+1)} &= \frac{23 - y^{(k)}}{10} \\ y^{(k+1)} &= \frac{26 - x^{(k)}}{8} \\ x^{(2)} &= \frac{23 - y^{(1)}}{10} = 2{,}3 \\ y^{(2)} &= \frac{26 - x^{(1)}}{8} = 3{,}25 \\ x^{(3)} &= \frac{23 - y^{(2)}}{10} = 1{,}975 \\ y^{(3)} &= \frac{26 - x^{(2)}}{8} = 2{,}9625 \end{aligned}$$

Código Python: Jacobi

```
from __future__ import division
import numpy as np
from numpy import linalg

def jacobi(A,b,x0,tol,N):
    #preliminares
    A = A.astype('double')
    b = b.astype('double')
    x0 = x0.astype('double')

    n=np.shape(A)[0]
    x = np.zeros(n)
    it = 0
    #iteracoes
    while (it < N):
        it = it+1
        #iteracao de Jacobi
        for i in np.arange(n):
            x[i] = b[i]
```

```
            for j in np.concatenate((np.arange(0,i),np.arange(i+1,n))):
                x[i] -= A[i,j]*x0[j]
            x[i] /= A[i,i]
        #tolerancia
        if (np.linalg.norm(x-x0,np.inf) < tol):
            return x
        #prepara nova iteracao
        x0 = np.copy(x)
    raise NameError('num. max. de iteracoes excedido.')
```

4.6.2 Método de Gauss-Seidel

Assim como no método de Jacobi, no método de Gauss-Seidel também isolamos o elemento x_i da equação i. Porém perceba que a equação para $x_2^{(k+1)}$ depende de $x_1^{(k)}$ na iteração k. Intuitivamente podemos pensar em usar $x_1^{(k+1)}$ que acabou de ser calculado e temos

$$x_2^{(k+1)} = \frac{y_2 - \left(a_{21}x_1^{(k+1)} + a_{23}x_3^{(k)} + \cdots + a_{2n}x_n^{(k)}\right)}{a_{22}}$$

Aplicando esse raciocínio podemos construir o método de Gauss-Seidel como

$$\begin{aligned}
x_1^{(k+1)} &= \frac{y_1 - \left(a_{12}x_2^{(k)} + \cdots + a_{1n}x_n^{(k)}\right)}{a_{11}} \\
x_2^{(k+1)} &= \frac{y_2 - \left(a_{21}x_1^{(k+1)} + a_{23}x_3^{(k)} + \cdots + a_{2n}x_n^{(k)}\right)}{a_{22}} \\
&\vdots \\
x_n^{(k+1)} &= \frac{y_2 - \left(a_{n1}x_1^{(k+1)} + \cdots + a_{n(n-1)}x_{n-1}^{(k+1)}\right)}{a_{nn}}
\end{aligned}$$

Em notação mais compacta, o método de Gauss-Seidel consiste na iteração:

$$\begin{aligned}
x^{(1)} &= \text{aproximação inicial} \\
x_i^{(k)} &= \frac{y_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}}{a_{ii}}
\end{aligned}$$

Exemplo 4.6.2. Resolva o sistema

$$\begin{aligned}
10x + y &= 23 \\
x + 8y &= 26
\end{aligned}$$

usando o método de Guass-Seidel iniciando com $x^{(1)} = y^{(1)} = 0$.

$$\begin{aligned} x^{(k+1)} &= \frac{23 - y^{(k)}}{10} \\ y^{(k+1)} &= \frac{26 - x^{(k+1)}}{8} \\ x^{(2)} &= \frac{23 - y^{(1)}}{10} = 2{,}3 \\ y^{(2)} &= \frac{26 - x^{(2)}}{8} = 2{,}9625 \\ x^{(3)} &= \frac{23 - y^{(2)}}{10} = 2{,}00375 \\ y^{(3)} &= \frac{26 - x^{(3)}}{8} = 2{,}9995312 \end{aligned}$$

Código Python: Gauss-Seidel

```
from __future__ import division
import numpy as np
from numpy import linalg

def gauss_seidel(A,b,x0,tol,N):
    #preliminares
    A = A.astype('double')
    b = b.astype('double')
    x0 = x0.astype('double')

    n=np.shape(A)[0]
    x = np.copy(x0)
    it = 0
    #iteracoes
    while (it < N):
        it = it+1
        #iteracao de Jacobi
        for i in np.arange(n):
            x[i] = b[i]
            for j in np.concatenate((np.arange(0,i),np.arange(i+1,n))):
                x[i] -= A[i,j]*x[j]
            x[i] /= A[i,i]
            print(x[i],A[i,i])
        #tolerancia
```

```
        if (np.linalg.norm(x-x0,np.inf) < tol):
            return x
        #prepara nova iteracao
        x0 = np.copy(x)
    raise NameError('num. max. de iteracoes excedido.')
```

4.6.3 Análise de convergência

Nesta seção, discutimos sobre a análise de convergência de métodos iterativos para solução de sistema lineares. Para tanto, consideramos um sistema linear $Ax = b$, onde $A = [a_{i,j}]_{i,j=1}^{n,n}$ é a matriz (real) dos coeficientes, $b = (a_j)_{j=1}^{n}$ é um vetor dos termos constantes e $x = (x_j)_{j=1}^{n}$ é o vetor incógnita. No decorrer, assumimos que A é uma matriz não-singular.

Geralmente, métodos iterativos são construídos como uma iteração de ponto fixo. No caso de um sistema linear, reescreve-se a equação matricial em um problema de ponto fixo equivalente, i.e.:

$$Ax = b \Leftrightarrow x = Tx + c,$$

onde $T = [t_{i,j}]_{i,j=1}^{n,n}$ é chamada de **matriz da iteração** e $c = (c_j)_{j=1}^{n}$ de **vetor da iteração**. Construídos a matriz T e o vetor c, o método iterativo consiste em computar a iteração:

$$x^{(k+1)} = Tx^{(k)} + c, \quad n \geq 1,$$

onde $x^{(1)}$ é uma aproximação inicial dada.

Afim de construirmos as matrizes e os vetores de iteração do método de Jacobi e de Gauss-Seidel, decompomos a matriz A da seguinte forma:

$$A = L + D + U,$$

onde D é a matriz diagonal $D = \operatorname{diag}(a_{11}, a_{22}, \ldots, a_{nn})$, i.e.:

$$D := \begin{bmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{bmatrix},$$

e, respectivamente, L e U são as seguintes matrizes triangular inferior e superior:

$$L := \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21} & 0 & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & 0 \end{bmatrix}, \quad U := \begin{bmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ 0 & 0 & 0 & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Exemplo 4.6.3. Considere o seguinte sistema linear:

$$\begin{aligned} 3x_1 + x_2 - x_3 &= 2 \\ -x_1 - 4x_2 + x_3 &= -10 \\ x_1 - 2x_2 - 5x_3 &= 10 \end{aligned}$$

Escreva o sistema na sua forma matricial $Ax = b$ identificando a matriz dos coeficientes A, o vetor incógnita x e o vetor dos termos constantes b. Em seguida, faça a decomposição $A = L + D + U$.

Solução. A forma matricial deste sistema é $Ax = b$, onde:

$$A = \begin{bmatrix} 3 & 1 & -1 \\ -1 & -4 & 1 \\ 1 & -2 & -5 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \text{e} \quad b = \begin{bmatrix} 2 \\ -10 \\ 10 \end{bmatrix}.$$

A decomposição da matriz A nas matrizes L triangular inferior, D diagonal e U triangular superior é:

$$\underbrace{\begin{bmatrix} 3 & 1 & -1 \\ -1 & -4 & 1 \\ 1 & -2 & -5 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & -2 & 0 \end{bmatrix}}_{L} + \underbrace{\begin{bmatrix} 3 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & -5 \end{bmatrix}}_{D} + \underbrace{\begin{bmatrix} 0 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{U}.$$

Em `Python`, podemos construir as matrizes L, D e U, da seguinte forma:

```
>>> A = np.array([[3,1,-1],
...               [-1,-4,1],
...               [1,-2,5]],
...               dtype='double')
```

```
>>> D = np.diag(np.diag(A))
>>> L = np.tril(A)-D
>>> U=np.triu(A)-D
```

◊

Iteração de Jacobi

Vamos, agora, usar a decomposição discutida acima para construir a matriz de iteração T_J e o vetor de iteração c_J associado ao método de Jacobi. Neste caso, temos:

$$\begin{aligned} Ax = b &\Leftrightarrow (L+D+U)x = b \\ &\Leftrightarrow Dx = -(L+U)x + b \\ &\Leftrightarrow x = \underbrace{-D^{-1}(L+U)}_{=:T_J}x + \underbrace{D^{-1}b}_{=:c_J}. \end{aligned}$$

Ou seja, a iteração do método de Jacobi escrita na forma matricial é:

$$x^{(k+1)} = T_J x^{(k)} + c_J, \quad k \geq 1,$$

com $x^{(1)}$ uma aproximação inicial dada, sendo $T_J := -D^{-1}(L+U)$ a matriz de iteração e $c_J = D^{-1}b$ o vetor da iteração.

Exemplo 4.6.4. Construa a matriz de iteração T_J e o vetor de iteração c_J do método de Jacobi para o sistema dado no Exemplo 4.6.3.

Solução. A matriz de iteração é dada por:

$$T_J := -D^{-1}(L+U) = -\underbrace{\begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{5} \end{bmatrix}}_{D^{-1}} \underbrace{\begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & 2 & 0 \end{bmatrix}}_{(L+U)} = \begin{bmatrix} 0 & -\frac{1}{3} & \frac{1}{3} \\ -\frac{1}{4} & 0 & \frac{1}{4} \\ \frac{1}{5} & \frac{2}{5} & 0 \end{bmatrix}.$$

O vetor da iteração de Jacobi é:

$$c_J := D^{-1}b = \underbrace{\begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{5} \end{bmatrix}}_{D^{-1}} \underbrace{\begin{bmatrix} 2 \\ -10 \\ 10 \end{bmatrix}}_{b} = \begin{bmatrix} \frac{2}{3} \\ \frac{5}{2} \\ -2 \end{bmatrix}.$$

Em `python`, podemos computar T_J e c_J da seguinte forma:

```
>>> TJ = -np.linalg.inv(D).dot(L+U);
>>> cJ = np.linalg.inv(D).dot(b);
```

◊

Iteração de Gauss-Seidel

A forma matricial da iteração do método de Gauss-Seidel também pode ser construída com base na decomposição $A = L + D + U$. Para tando, fazemos:

$$\begin{aligned} Ax = b &\Leftrightarrow (L+D+U)x = b \\ &\Leftrightarrow (L+D)x = -Ux + b \\ &\Leftrightarrow x = \underbrace{-(L+D)^{-1}U}_{=:T_G} x + \underbrace{(L+D)^{-1}b}_{=:c_G} \end{aligned}$$

Ou seja, a iteração do método de Gauss-Seidel escrita na forma matricial é:

$$x^{(k+1)} = T_G x^{(k)} + c_G, \quad k \geq 1,$$

com $x^{(1)}$ uma aproximação inicial dada, sendo $T_G := -(L+D)^{-1}U$ a matriz de iteração e $c_J = (L+D)^{-1}b$ o vetor da iteração.

Exemplo 4.6.5. Construa a matriz de iteração T_G e o vetor de iteração c_G do método de Gauss-Seidel para o sistema dado no Exemplo 4.6.3.

Solução. A matriz de iteração é dada por:

$$T_G := -(L+D)^{-1}U = --\underbrace{\begin{bmatrix} 3 & 0 & 0 \\ -1 & -4 & 0 \\ 1 & -2 & -5 \end{bmatrix}^{-1}}_{(L+D)^{-1}} \underbrace{\begin{bmatrix} 0 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{U} = \begin{bmatrix} 0 & -\frac{1}{3} & \frac{1}{3} \\ 0 & \frac{1}{12} & \frac{1}{6} \\ 0 & -\frac{1}{10} & 0 \end{bmatrix}.$$

O vetor da iteração de Gauss-Seidel é:

$$c_G := (L+D)^{-1}b = \underbrace{\begin{bmatrix} 3 & 0 & 0 \\ -1 & -4 & 0 \\ 1 & -2 & -5 \end{bmatrix}^{-1}}_{(L+D)^{-1}} \underbrace{\begin{bmatrix} 2 \\ -10 \\ 10 \end{bmatrix}}_{b} = \begin{bmatrix} \frac{2}{3} \\ \frac{7}{3} \\ -\frac{28}{10} \end{bmatrix}.$$

Em `Python`, podemos computar T_G e c_G da seguinte forma:

```
-->TG = -np.linalg.inv(L+D).dot(U);
-->cG = np.linalg.inv(L+D).dot(b);
```

◊

Condições de convergência

Aqui, vamos discutir condições necessárias e suficientes para a convergência de métodos iterativos. Isto é, dado um sistema $Ax = b$ e uma iteração:

$$x^{(k+1)} = Tx^{(k)} + c, \quad k \geq 1,$$

$x^{(1)}$ dado, estabelecemos condições nas quais $x^{(k)} \to x^*$, onde x^* é a solução do sistema dado, i.e. $x^* = Tx^* + c$ ou, equivalentemente, $Ax^* = b$.

Lema 4.6.1. *Seja T uma matriz real $n \times n$. O limite $\lim_{k\to\infty} \|T^k\|_p = 0$, $1 \leq p \leq \infty$, se, e somente se, $\rho(T) < 1$.*

Demonstração. Aqui, fazemos apenas um esboço da demonstração. Para mais detalhes, veja [8], Teorema 4, pág. 14.

Primeiramente, suponhamos que $\|T\|_p < 1$, $1 \leq p \leq \infty$. Como (veja [8], Lema 2, pág. 12):

$$\rho(T) \leq \|T\|_p,$$

temos $\rho(T) < 1$, o que mostra a implicação.

Agora, suponhamos que $\rho(T) < 1$ e seja $0 < \epsilon < 1 - \rho(T)$. Então, existe $1 \leq p \leq \infty$ tal que (veja [8], Teorema 3, página 12):

$$\|T\|_p \leq \rho(T) + \epsilon < 1.$$

Assim, temos:

$$\lim_{k\to\infty} \|T^k\|_p \leq \lim_{k\to\infty} \|T\|_p^m = 0.$$

Da equivalência entre as normas segue a recíproca. □

Observação 4.6.1. Observamos que:

$$\lim_{k\to\infty} \|T^k\|_p = 0, \quad ,1 \leq p \leq \infty, \Leftrightarrow \lim_{k\to\infty} t_{ij}^k = 0, \quad 1 \leq i,j \leq n.$$

Lema 4.6.2. *Se $\rho(T) < 1$, então existe $(I - T)^{-1}$ e:*

$$(I - T)^{-1} = \sum_{k=0}^{\infty} T^k.$$

Demonstração. Primeiramente, provamos a existência de $(I - T)^{-1}$. Seja λ um autovalor de T e x um autovetor associado, i.e. $Tx = \lambda x$. Então, $(I - T)x = (1 - \lambda)x$. Além disso, temos $|\lambda| < \rho(T) < 1$, logo $(1 - \lambda) \neq 0$, o que garante que $(I - T)$ é não singular. Agora, mostramos que $(I - T)^{-1}$ admite a expansão acima. Do Lema 4.6.1 e da Observação 4.6.1 temos:

$$(I - T)\sum_{k=0}^{\infty} T^k = \lim_{m\to\infty} (I - T)\sum_{k=0}^{m} T^k = \lim_{m\to\infty} (I - T^{m+1}) = I,$$

o que mostra que $(I-T)^{-1} = \sum_{k=0}^{\infty} T^k$. □

Teorema 4.6.1. *A sequência recursiva* $\{x^{(k)}\}_{k\in\mathbb{N}}$ *dada por:*

$$x^{(k+1)} = Tx^{(k)} + c$$

converge para solução de $x = Tx + c$ *para qualquer escolha de* $x^{(1)}$ *se, e somente se,* $\rho(T) < 1$.

Demonstração. Primeiramente, assumimos que $\rho(T) < 1$. Observamos que:

$$\begin{aligned} x^{(k+1)} &= Tx^{(k)} + c = T(Tx^{(k-1)} + c) + c \\ &= T^2x^{(k-1)} + (I+T)c \\ &\vdots \\ &= T^{(k)}x^{(1)} + \left(\textstyle\sum_{k=0}^{k-1} T^k\right) c. \end{aligned}$$

Daí, do Lema 4.6.1 e do Lema 4.6.2 temos:

$$\lim_{k\to\infty} x^{(k)} = (I-T)^{(-1)}c.$$

Ora, se x^* é a solução de $x = Tx + c$, então $(I-T)x^* = c$, i.e. $x^* = (I-T)^{-1}c$. Logo, temos demonstrado que $x^{(k)}$ converge para a solução de $x = Tx + c$, para qualquer escolha de $x^{(1)}$.

Agora, suponhamos que $x^{(k)}$ converge para x^* solução de $x = Tx + c$, para qualquer escolha de $x^{(1)}$. Seja, então, y um vetor arbitrário e $x^{(1)} = x^* - y$. Observamos que:

$$\begin{aligned} x^* - x^{(k+1)} &= (Tx^* + c) - (Tx^{(k)} + c) \\ &= T(x^* - x^{(k)}) \\ &\vdots \\ &= T^{(k)}(x^* - x^{(1)}) = T^{(k)}y. \end{aligned}$$

Logo, para qualquer $1 \le p \le \infty$, temos, :

$$0 = \lim_{k\to\infty} x^* - x^{(k+1)} = \lim_{k\to\infty} T^{(k)}y.$$

Como y é arbitrário, da Observação 4.6.1 temos $\lim_{k\to\infty} \|T^{(k)}\|_p = 0$, $1 \le p \le \infty$. Então, o Lema 4.6.1 garante que $\rho(T) < 1$. □

Observação 4.6.2. Pode-se mostrar que tais métodos iterativos tem taxa de convergência super linear com:

$$\|x^{(k+1)} - x^*\| \approx \rho(T)^k \|x^{(1)} - x^*\|.$$

Para mais detalhes, veja [8], pág. 61-64.

Exemplo 4.6.6. Mostre que, para qualquer escolha da aproximação inicial, ambos os métodos de Jacobi e Gauss-Seidel são convergentes quando aplicados ao sistema linear dado no Exemplo 4.6.3.

Solução. Do Teorema 4.6.1, vemos que é necessário e suficiente que $\rho(T_J) < 1$ e $\rho(T_G) < 1$. Computando estes raios espectrais, obtemos $\rho(T_J) \approx 0{,}32$ e $\rho(T_G) \approx 0{,}13$. Isto mostra que ambos os métodos serão convergentes. $\Diamond$

Condição suficiente

Uma condição suficiente porém não necessária para que os métodos de Gauss-Seidel e Jacobi convirjam é a que a matriz seja **estritamente diagonal dominante**.

Definição 4.6.1. *Uma matriz A é **estritamente diagonal dominante** quando:*

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, i = 1,...,n$$

Definição 4.6.2. *Uma matriz A é **diagonal dominante** quando*

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, i = 1,...,n$$

e para ao menos um i, a_{ii} é estritamente maior que a soma dos elementos fora da diagonal.

Teorema 4.6.2. *Se a matriz A for diagonal dominante[8], então os métodos de Jacobi e Gauss-Seidel serão convergentes independente da escolha inicial $x^{(1)}$.*

Se conhecermos a solução exata x do problema, podemos calcular o erro relativo em cada iteração como:

$$\frac{\|x - x^{(k)}\|}{\|x\|}.$$

Em geral não temos x, entretanto podemos estimar o vetor **resíduo** $r^{(k)} = b - A\tilde{x^{(k)}}$. Note que quando o erro tende a zero, o resíduo também tende a zero.

[8] E consequentemente estritamente diagonal dominante.

Teorema 4.6.3. *O erro relativo e o resíduo estão relacionados como (veja [3])*

$$\frac{\|x - x^{(k)}\|}{\|x\|} \leq \kappa(A)\frac{\|r\|}{\|b\|}$$

onde $k(A)$ é o número de condicionamento.

Exemplo 4.6.7. Ambos os métodos de Jacobi e Gauss-Seidel são convergentes para o sistema dado no Exemplo 4.6.3, pois a matriz dos coeficientes deste é uma matriz estritamente diagonal dominante.

Exercícios

E 4.6.1. Considere o problema de 5 incógnitas e cinco equações dado por

$$\begin{aligned} x_1 - x_2 &= 1 \\ -x_1 + 2x_2 - x_3 &= 1 \\ -x_2 + (2+\varepsilon)x_3 - x_4 &= 1 \\ -x_3 + 2x_4 - x_5 &= 1 \\ x_4 - x_5 &= 1 \end{aligned}$$

a) Escreva na forma $Ax = b$ e resolva usando eliminação gaussiana para $\varepsilon = 10^{-3}$ no `Scilab`.

b) Obtenha o vetor incógnita x com $\varepsilon = 10^{-3}$ usando o comando $A\backslash b$.

c) Obtenha o vetor incógnita x com $\varepsilon = 10^{-3}$ usando Jacobi com tolerância 10^{-2}. Compare o resultado com o resultado obtido no item d.

d) Obtenha o vetor incógnita x com $\varepsilon = 10^{-3}$ usando Gauss-Seidel com tolerância 10^{-2}. Compare o resultado com o resultado obtido no item d.

e) Discuta com base na relação esperada entre tolerância e exatidão conforme estudado na primeira área para problemas de uma variável.

E 4.6.2. Resolva o seguinte sistema pelo método de Jacobi e Gauss-Seidel:

$$\begin{cases} 5x_1 + x_2 + x_3 &= 50 \\ -x_1 + 3x_2 - x_3 &= 10 \\ x_1 + 2x_2 + 10x_3 &= -30 \end{cases}$$

Use como critério de paragem tolerância inferior a 10^{-3} e inicialize com $x^0 = y^0 = z^0 = 0$.

E 4.6.3. Refaça a questão **??** construindo um algoritmo que implemente os métodos de Jacobi e Gauss-Seidel.

E 4.6.4. Considere o seguinte sistema de equações lineares:

$$\begin{aligned} x_1 - x_2 &= 0 \\ -x_{j-1} + 5x_j - x_{j+1} &= \cos(j/10), \quad 2 \leq j \leq 10 \\ x_{11} &= x_{10}/2 \end{aligned} \tag{4.25}$$

Construa a iteração para encontrar a solução deste problema pelos métodos de Gauss-Seidel e Jacobi. Usando esses métodos, encontre uma solução aproximada com erro absoluto inferior a 10^{-5}.

E 4.6.5. Resolva o problema 4.8.1 pelos métodos de Jacobi e Gauss-Seidel.

E 4.6.6. Faça uma permutação de linhas no sistema abaixo e resolva pelos métodos de Jacobi e Gauss-Seidel:

$$\begin{aligned} x_1 + 10x_2 + 3x_3 &= 27 \\ 4x_1 + x_3 &= 6 \\ 2x_1 + x_2 + 4x_3 &= 12 \end{aligned}$$

4.7 Método da potência para cálculo de autovalores

Consideremos uma matriz $A \in \mathbb{R}^{n,n}$ diagonalizável, isto é, existe um conjunto $\{v_j\}_{j=1}^n$ de autovetores de A tais que qualquer elemento $x \in \mathbb{R}^n$ pode ser escrito como uma combinação linear dos v_j. Sejam $\{\lambda_j\}_{j=1}^n$ o conjunto de autovalores associados aos autovetores tal que um deles seja dominante, ou seja,

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots |\lambda_n| > 0$$

Como os autovetores são l.i., todo vetor $x \in \mathbb{R}^n$, $x = (x_1,x_2,...,x_n)$, pode ser escrito com combinação linear dos autovetores da seguinte forma:

$$x = \sum_{j=1}^{n} \beta_j v_j. \tag{4.26}$$

O método da potência permite o cálculo do autovetor dominante com base no comportamento assintótico (i.e. "no infinito") da sequência

$$x, Ax, A^2x, A^3x, \ldots$$

.

Por questões de convergência, consideramos a seguinte sequência semelhante à anterior, porém normalizada:

$$\frac{x}{\|x\|}, \frac{Ax}{\|Ax\|}, \frac{A^2x}{\|A^2x\|}, \frac{A^3x}{\|A^3x\|}, \ldots,$$

que pode ser obtida pelo seguinte processo iterativo:

$$x^{(k+1)} = \frac{A^kx}{\|A^kx\|}$$

Observamos que se x está na forma (4.26), então A^kx pode ser escrito como

$$A^kx = \sum_{j=1}^{n} \beta_j A^k v_j = \sum_{j=1}^{n} \beta_j \lambda_j^k v_j = \beta_1 \lambda_1^k \left(v_1 + \sum_{j=2}^{n} \frac{\beta_j}{\beta_1} \left(\frac{\lambda_j}{\lambda_1} \right)^k v_j \right)$$

Como $\left|\frac{\lambda_j}{\lambda_1}\right| < 1$ para todo $j \geq 2$, temos

$$\sum_{j=2}^{n} \frac{\beta_j}{\beta_1} \left(\frac{\lambda_j}{\lambda_1} \right)^k v_j \to 0.$$

Assim

$$\frac{A^kx}{\|A^kx\|} = \frac{\beta_1 \lambda_1^k}{\|A^kx\|} \left(v_1 + O\left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right)) \right) \tag{4.27}$$

Como a norma de $\frac{A^kx}{\|A^kx\|}$ é igual a um, temos

$$\left\| \frac{\beta_1 \lambda_1^k}{\|A^kx\|} v_1 \right\| \to 1$$

e, portanto,

$$\left| \frac{\beta_1 \lambda_1^k}{\|A^kx\|} \right| \to \frac{1}{\|v_1\|}$$

Ou seja, se definimos $\alpha^{(k)} = \frac{\beta_1 \lambda_1^k}{\|A^kx\|}$, então

$$|\alpha^{(k)}| \to 1$$

Retornando a (4.27), temos:

$$\frac{A^k x}{\|A^k x\|} - \alpha^{(k)} v_1 \to 0$$

Observe que um múltiplo de autovetor também é um autovetor e, portanto,

$$\frac{A^k x}{\|A^k x\|}$$

é um esquema que oscila entre os autovetores ou converge para o autovetor v_1.

Uma vez que temos o autovetor v_1 de A, podemos calcular λ_1 da seguinte forma:

$$Av_1 = \lambda_1 v_1 \implies v_1^T A v_1 = v_1^T \lambda_1 v_1 \implies \lambda_1 = \frac{v_1^T A v_1}{v_1^T v_1}$$

Observe que a última identidade é válida, pois $\|v_1\| = 1$ por construção.

Exercícios

E 4.7.1. Calcule o autovalor dominante e o autovetor associado da matriz

$$\begin{bmatrix} 4 & 41 & 78 \\ 48 & 28 & 21 \\ 26 & 13 & 11 \end{bmatrix}$$

Expresse sua resposta com seis dígitos significativos

E 4.7.2. Calcule o autovalor dominante e o autovetor associado da matriz

$$\begin{bmatrix} 3 & 4 \\ 2 & -1 \end{bmatrix}$$

usando o método da potência inciando com o vetor $x = [1 \quad 1]^T$

E 4.7.3. A norma L_2 de um matriz A é dada pela raiz quadrada do autovalor dominante da matriz A^*A, isto é:

$$\|A\|_2 = \sqrt{\max\{|\lambda| : \lambda \in \sigma(A^*A)\}} :$$

Use o método da potência para obter a norma L_2 da seguinte matriz:

$$A = \begin{bmatrix} 69 & 84 & 88 \\ 15 & -40 & 11 \\ 70 & 41 & 20 \end{bmatrix}$$

Expresse sua resposta com seis dígitos significativos

E 4.7.4. Os autovalores de uma matriz triangular são os elementos da diagonal principal. Verifique o método da potência aplicada à seguinte matriz:

$$\begin{bmatrix} 2 & 3 & 1 \\ 0 & 3 & -1 \\ 0 & 0 & 1 \end{bmatrix}.$$

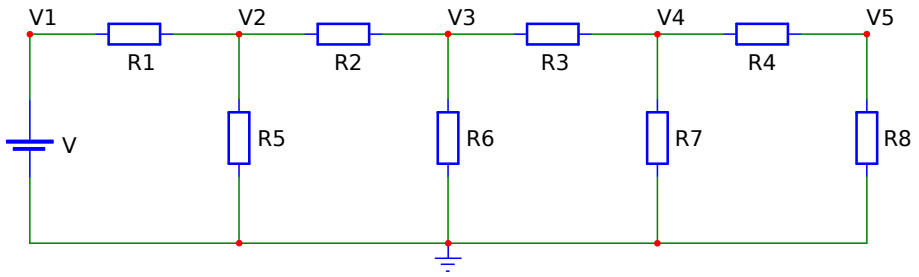
4.8 Exercícios finais

E 4.8.1. O circuito linear da figura 4.8.1 pode ser modelado pelo sistema (**??**). Escreva esse sistema na forma matricial sendo as tensões V_1, V_2, V_3, V_4 e V_5 as cinco incógnitas. Resolva esse problema quando $V = 127$ e

a) $R_1 = R_2 = R_3 = R_4 = 2$ e $R_5 = R_6 = R_7 = 100$ e $R_8 = 50$

b) $R_1 = R_2 = R_3 = R_4 = 2$ e $R_5 = 50$ e $R_6 = R_7 = R_8 = 100$

$$\begin{aligned} V_1 &= V \\ \frac{V_1 - V_2}{R_1} + \frac{V_3 - V_2}{R_2} - \frac{V_2}{R_5} &= 0 \\ \frac{V_2 - V_3}{R_2} + \frac{V_4 - V_3}{R_3} - \frac{V_3}{R_6} &= 0 \\ \frac{V_3 - V_4}{R_3} + \frac{V_5 - V_4}{R_4} - \frac{V_4}{R_7} &= 0 \\ \frac{V_4 - V_5}{R_4} - \frac{V_5}{R_8} &= 0 \end{aligned}$$



Complete a tabela abaixo representado a solução com 4 algarismos significativos:

Caso	V_1	V_2	V_3	V_4	V_5
a					
b					

Então, refaça este problema reduzindo o sistema para apenas 4 incógnitas (V_2, V_3, V_4 e V_5).

E 4.8.2. Resolva os seguintes problemas:

a) Encontre o polinômio $P(x) = ax^2 + bx + c$ que passa pelos pontos $(-1, -3)$, $(1, -1)$ e (2,9).

b) Encontre os coeficientes A e B da função $f(x) = A\sin(x) + B\cos(x)$ tais que $f(1) = 1.4$ e $f(2) = 2.8$.

c) Encontre a função $g(x) = A_1\sin(x) + B_1\cos(x) + A_2\sin(2x) + B_2\cos(2x)$ tais que $f(1) = 1$, $f(2) = 2$, $f(3) = 3$ e $f(4) = 4$.

Capítulo 5

Solução de sistemas de equações não lineares

O método de Newton aplicado a encontrar a raiz x^* da função $y = f(x)$ estudado na primeira área de nossa disciplina consiste em um processo iterativo. Em cada passo deste processo, dispomos de uma aproximação $x^{(k)}$ para x^* e construímos uma aproximação $x^{(k+1)}$. Cada passo do método de Newton envolve os seguintes procedimentos:

- Linearização da função $f(x)$ no ponto $x^{(k)}$:

$$f(x) = f(x^{(k)}) + (x - x^{(k)})f'(x^{(k)}) + O\left(|x - x^{(k)}|^2\right)$$

- A aproximação $x^{(k+1)}$ é definida como o valor de x em que a linearização $f(x^{(k)}) + (x - x^{(k)})f'(x^{(k)})$ passa por zero.

Observação: $y = f(x^{(k)}) + (x - x^{(k)})f'(x^{(k)})$ é a equação da reta que tangencia a curva $y = f(x)$ no ponto $(x^{(k)}, f(x^{(k)}))$.

Queremos, agora, generalizar o método de Newton a fim de resolver problemas de várias equações e várias incógnitas, ou seja, encontrar $x_1, x_2, \ldots x_n$ que satisfazem as seguinte equações:

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0 \\ f_2(x_1, x_2, \ldots, x_n) &= 0 \\ &\vdots \\ f_n(x_1, x_2, \ldots, x_n) &= 0 \end{aligned}$$

Podemos escrever este problema na forma vetorial definindo o vetor $x = [x_1,x_2,\ldots,x_n]^T$ e a função vetorial

$$F(x) = \begin{bmatrix} f_1(x_1,x_2,\ldots,x_n) \\ f_2(x_1,x_2,\ldots,x_n) \\ \vdots \\ f_n(x_1,x_2,\ldots,x_n) \end{bmatrix}$$

Exemplo 5.0.1. Suponha que queiramos resolver numericamente os seguinte sistema de duas equações e duas incógnitas:

$$\begin{aligned} \frac{x_1^2}{3} + x_2^2 &= 1 \\ x_1^2 + \frac{x_2^2}{4} &= 1 \end{aligned}$$

Então definimos

$$F(x) = \begin{bmatrix} \frac{x_1^2}{3} + x_2^2 - 1 \\ \\ x_1^2 + \frac{x_2^2}{4} - 1 \end{bmatrix}$$

Neste momento, dispomos de um problema na forma $F(x) = 0$ e precisamos desenvolver uma técnica para linearizar a função $F(x)$. Para tal, precisamos de alguns conceitos do Cálculo II.

Observe que $F(x) - F(x^{(0)})$ pode ser escrito como

$$F(x) - F(x^{(0)}) = \begin{bmatrix} f_1(x_1,x_2,\ldots,x_n) - f_1(x_1^{(0)},x_2^{(0)},\ldots,x_n^{(0)}) \\ f_2(x_1,x_2,\ldots,x_n) - f_2(x_1^{(0)},x_2^{(0)},\ldots,x_n^{(0)}) \\ \vdots \\ f_n(x_1,x_2,\ldots,x_n) - f_n(x_1^{(0)},x_2^{(0)},\ldots,x_n^{(0)}) \end{bmatrix}$$

Usamos a regra da cadeia

$$df_i = \frac{\partial f_i}{\partial x_1}dx_1 + \frac{\partial f_i}{\partial x_2}dx_2 + \cdots + \frac{\partial f_i}{\partial x_n}dx_n = \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j}dx_j$$

e aproximamos as diferenças por derivadas parciais:

$$f_i(x_1,x_2,\ldots,x_n) - f_i(x_1^{(0)},x_2^{(0)},\ldots,x_n^{(0)}) \approx \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j}\left(x_j - x_j^{(0)}\right)$$

Portanto,

$$F(x) - F(x^{(0)}) \approx \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \begin{bmatrix} x_1 - x_1^{(0)} \\ x_2 - x_2^{(0)} \\ \vdots \\ x_n - x_n^{(0)} \end{bmatrix} \tag{5.1}$$

Definimos então a matriz jacobiana por

$$J_F = \frac{\partial(f_1, f_2, \ldots, f_n)}{\partial(x_1, x_2, \ldots, x_n)} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

A matriz jacobiana de uma função ou simplesmente, o Jacobiano de uma função $F(x)$ é a matriz formada pelas suas derivadas parciais:

$$(J_F)_{ij} = \frac{\partial f_i}{\partial x_j}$$

Nestes termos podemos reescrever (5.1) como

$$F(x) \approx F(x^{(0)}) + J_F(x^{(0)})(x - x^{(0)})$$

Esta expressão é chama de linearização de $F(x)$ no ponto $x^{(0)}$ e generaliza a linearização em uma dimensão dada por $f(x) \approx f(x^{(0)}) + f'(x^{(0)})(x - x^{(0)})$

5.1 O método de Newton para sistemas

Vamos agora construir o método de Newton-Raphson, ou seja, o método de Newton generalizado para sistemas. Assumimos, portanto, que a função $F(x)$ é diferenciável e que existe um ponto x^* tal que $F(x^*) = 0$. Seja $x^{(k)}$ uma aproximação para x^*, queremos construir uma nova aproximação $x^{(k+1)}$ através da linearização de $F(x)$ no ponto $x^{(k)}$.

- Linearização da função $F(x)$ no ponto $x^{(k)}$:

$$F(x) = F(x^{(k)}) + J_F\left(x^{(k)}\right)\left(x - x^{(k)}\right) + O\left(\|x - x^{(k)}\|^2\right)$$

- A aproximação $x^{(k)}$ é definida como o ponto x em que a linearização $F(x^{(k)}) + J_F\left(x^{(k)}\right)\left(x - x^{(k)}\right)$ é nula, ou seja:

$$F(x^{(k)}) + J_F\left(x^{(k)}\right)\left(x^{(k+1)} - x^{(k)}\right) = 0$$

Supondo que a matriz jacobina seja inversível no ponto $x^{(k)}$, temos:

$$\begin{aligned} J_F\left(x^{(k)}\right)\left(x^{(k+1)} - x^{(k)}\right) &= -F(x^{(k)}) \\ x^{(k+1)} - x^{(k)} &= -J_F^{-1}\left(x^{(k)}\right) F(x^{(k)}) \\ x^{(k+1)} &= x^{(k)} - J_F^{-1}\left(x^{(k)}\right) F(x^{(k)}) \end{aligned}$$

Desta forma, o método iterativo de Newton-Raphson para encontrar as raízes de $F(x) = 0$ é dado por:

$$\begin{cases} x^{(k+1)} = x^{(k)} - J_F^{-1}\left(x^{(k)}\right) F(x^{(k)}), \ n \geq 0 \\ x^{(0)} = \text{dado inicial} \end{cases}$$

Observação 5.1.1. Usamos subíndices para indicar o elemento de um vetor e super-índices para indicar o passo da iteração. Assim $x^{(k)}$ se refere à iteração k e $x_i^{(k)}$ se refere à componente i no vetor $x^{(k)}$.

Observação 5.1.2. A notação $J_F^{-1}\left(x^{(k)}\right)$ enfatiza que a jacobiana deve ser calculada a cada passo.

Observação 5.1.3. Podemos definir o passo $\Delta^{(k)}$ como

$$\Delta^{(k)} = x^{(k+1)} - x^{(k)}$$

Assim, $\Delta^{(k)} = -J_F^{-1}\left(x^{(k)}\right) F(x^{(k)})$, ou seja, $\Delta^{(k)}$ resolve o problema linear:

$$J_F\left(x^{(k)}\right)\Delta^{(k)} = -F(x^{(k)})$$

Em geral, é menos custoso resolver o sistema acima do que calcular o inverso da jacobiana e multiplicar pelo vetor $F(x^{(k)})$.

Exemplo 5.1.1. Retornamos ao nosso exemplo inicial, isto é, resolver numericamente os seguinte sistema não-linear:

$$\begin{aligned} \frac{x_1^2}{3} + x_2^2 &= 1 \\ x_1^2 + \frac{x_2^2}{4} &= 1 \end{aligned}$$

Para tal, definimos a função $F(x)$:

$$F(x) = \begin{bmatrix} \frac{x_1^2}{3} + x_2^2 - 1 \\ x_1^2 + \frac{x_2^2}{4} - 1 \end{bmatrix}$$

cuja jacobiana é:

$$J_F = \begin{bmatrix} \frac{2x_1}{3} & 2x_2 \\ 2x_1 & \frac{x_2}{2} \end{bmatrix}$$

Faremos a implementação numérica em `Python`. Para tal definimos as funções que implementarão $F(x)$ e a $J_F(x)$

```
>>> def F(x):
...     y = np.zeros(2)
...     y[0] = x[0]**2/3 + x[1]**2 - 1
...     y[1] = x[0]**2 + x[1]**2/4 - 1
...     return y
...
>>> def JF(x):
...     y = np.zeros((2,2))
...     y[0,0] = 2*x[0]/3
...     y[0,1] = 2*x[1]
...     y[1,0] = 2*x[0]
...     y[1,1] = x[1]/2
...     return y
...
```

Desta forma, se x é uma aproximação para a raiz, pode-se calcular a próxima aproximação através dos comandos:

```
>>> delta = -np.linalg.inv(JF(x)).dot(F(x))
>>> x = x + delta
```

Ou simplesmente

```
>>> x = x - np.linalg.inv(JF(x)).dot(F(x))
```

Observe que as soluções exatas desse sistema são $\left(\pm\sqrt{\frac{9}{11}}, \pm\sqrt{\frac{8}{11}}\right)$.

Exemplo 5.1.2. Encontre uma aproximação para a solução do sistema

$$\begin{aligned} x_1^2 &= \cos(x_1 x_2) + 1 \\ \sin(x_2) &= 2\cos(x_1) \end{aligned}$$

que fica próxima ao ponto $x_1 = 1{,}5$ e $x_2 = 0{,}5$.

Solução. Vamos, aqui, dar as principais ideias para se obter a solução usando o método de Newton. Começamos definindo nossa aproximação inicial por $x^{(1)} = (1{,}5, 0{,}5)$. Então iteramos:

$$x^{(n+1)} = x^{(n)} - J_F^{-1}(x)F(x), \quad n \geq 1.$$

onde

$$F(x) = \begin{bmatrix} x_1^2 - \cos(x_1 x_2) - 1 \\ \sin(x_2) - 2\cos(x_1) \end{bmatrix}$$

e sua jacobiana é

$$J_F(x) = \begin{bmatrix} 2x_1 + x_2\sin(x_1 x_2) & x_1\sin(x_1 x_2) \\ 2\sin(x_1) & \cos(x_2) \end{bmatrix}$$

As iterações convergem para $x = (1{,}3468109,\ 0{,}4603195)$.

Em `Python`, podemos implementá-las com o seguinte código:

```
def F(x):
    y = np.zeros(2)

    y[0] = x[0]**2 - np.cos(x[0]*x[1]) - 1
    y[1] = np.sin(x[1]) - 2*np.cos(x[0])
```

```
    return y

def JF(x):
    y = np.zeros((2,2))

    y[0,0] = 2*x[0] + x[1]*np.sin(x[0]*x[1])
    y[0,1] = x[0]*np.sin(x[0]*x[1])

    y[1,0] =  2*np.sin(x[0])
    y[1,1] = np.cos(x[1])

    return y
```

E agora, basta iterar:

```
>>> x = np.array([1.5,0.5])
>>> x=x-np.linalg.inv(JF(x)).dot(F(x))
```

◊

5.1.1 Código Python: Newton para Sistemas

```
from __future__ import division
import numpy as np
from numpy import linalg

def newton(F,JF,x0,TOL,N):
    #preliminares
    x = np.copy(x0).astype('double')
    k=0
    #iteracoes
    while (k < N):
       k += 1
       #iteracao Newton
       delta = -np.linalg.inv(JF(x)).dot(F(x))
       x = x + delta
       #criterio de parada
       if (np.linalg.norm(delta,np.inf) < TOL):
           return x

    raise NameError('num. max. iter. excedido.')
```

Exercícios

E 5.1.1. Encontre uma aproximação numérica para o seguinte problema não-linear de três equações e três incógnitas:

$$\begin{aligned} 2x_1 - x_2 &= \cos(x_1) \\ -x_1 + 2x_2 - x_3 &= \cos(x_2) \\ -x_2 + x_3 &= \cos(x_3) \end{aligned}$$

Partindo das seguintes aproximações iniciais:

a) $x^{(0)} = [1,\ 1,\ 1]^T$

b) $x^{(0)} = [-0{,}5,\ -2,\ -3]^T$

c) $x^{(0)} = [-2,\ -3,\ -4]^T$

d) $x^{(0)} = [0,\ 0,\ 0]^T$

5.2 Linearização de uma função de várias variáveis

5.2.1 O gradiente

Considere primeiramente uma função $f : \mathbb{R}^n \to \mathbb{R}$, ou seja, uma função que mapeia n variáveis reais em um único real, por exemplo:

$$f(x) = x_1^2 + x_2^2/4$$

Para construirmos a linearização, fixemos uma direção no espaço $\mathbb{R}^n$, ou seja um vetor v:

$$v = [v_1, v_2, \cdots, v_n]^T$$

Queremos estudar como a função $f(x)$ varia quando "andamos" na direção v a partir do ponto $x^{(0)}$. Para tal, inserimos um parâmetro real pequeno h, dizemos que

$$x = x^{(0)} + hv$$

e definimos a função auxiliar

$$g(h) = f(x^0 + hv).$$

Observamos que a função $g(h)$ é uma função de $\mathbb{R}$ em $\mathbb{R}$.

A linearização de $g(h)$ em torno de $h = 0$ é dada por

$$g(h) = g(0) + hg'(0) + O(h^2)$$

Observamos que $g(h) = f(x^{(0)} + hv)$ e $g(0) = f(x^{(0)})$. Precisamos calcular $g'(0)$:

$$g'(h) = \frac{d}{dh}g(h) = \frac{d}{dh}f(x^{(0)} + hv)$$

Pela regra da cadeia temos:

$$\frac{d}{dh}f(x^{(0)} + hv) = \sum_{j=1}^{n} \frac{\partial f}{\partial x_j}\frac{dx_j}{dh}$$

Observamos que $x_j = x_j^{(0)} + hv_j$, portanto

$$\frac{dx_j}{dh} = v_j$$

Assim:

$$\frac{d}{dh}f(x^{(0)} + hv) = \sum_{j=1}^{n} \frac{\partial f}{\partial x_j}v_j$$

Observamos que esta expressão pode ser vista como o produto interno entre o gradiente de f e o vetor v:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

Na notação cálculo vetorial escrevemos este produto interno como $\nabla f \cdot v = v \cdot \nabla f$ na notação de produto matricial, escrevemos $(\nabla f)^T v = v^T \nabla f$. Esta quantidade é conhecida como **derivada direcional** de f no ponto $x^{(0)}$ na direção v, sobretudo quando $\|v\| = 1$.

Podemos escrever a linearização $g(h) = g(0) + hg'(0) + O(h^2)$ como

$$f(x^{(0)} + hv) = f(x^{(0)}) + h\nabla^T f(x^{(0)})\, v + O(h^2)$$

Finalmente, escrevemos $x = x^{(0)} + hv$, ou seja, $hv = x - x^{(0)}$

$$f(x) = f(x^{(0)}) + \nabla^T f(x^{(0)})\,(x - x^{(0)}) + O(\|x - x^{(0)}\|^2)$$

Observação 5.2.1. Observe a semelhança com a linearização no caso em uma dimensão. A notação $\nabla^T f(x^{(0)})$ é o transposto do vetor gradiente associado à função $f(x)$ no ponto $x^{(0)}$:

$$\nabla^T f(x^{(0)}) = \left[\frac{\partial f\left(x^{(0)}\right)}{\partial x_1}, \ \frac{\partial f\left(x^{(0)}\right)}{\partial x_2}, \ \cdots, \frac{\partial f\left(x^{(0)}\right)}{\partial x_n} \right]$$

5.2.2 A matriz jacobiana

Interessamo-nos, agora, pela linearização da função $F : \mathbb{R}^n \to \mathbb{R}^n$. Lembramos que $F(x)$ pode ser escrita como um vetor de funções $f_j : \mathbb{R}^\times \to \mathbb{R}$:

$$F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{bmatrix}$$

Linearizando cada uma das funções f_j, temos:

$$F(x) = \underbrace{\begin{bmatrix} f_1\left(x^{(0)}\right) + \nabla^T f_1(x^{(0)}) \left(x - x^{(0)}\right) + O(\|x - x^{(0)}\|^2) \\ f_2\left(x^{(0)}\right) + \nabla^T f_2(x^{(0)}) \left(x - x^{(0)}\right) + O(\|x - x^{(0)}\|^2) \\ \vdots \\ f_n\left(x^{(0)}\right) + \nabla^T f_n(x^{(0)}) \left(x - x^{(0)}\right) + O(\|x - x^{(0)}\|^2) \end{bmatrix}}_{\text{Vetor coluna}}$$

ou, equivalentemente:

$$F(x) = \underbrace{\begin{bmatrix} f_1\left(x^{(0)}\right) \\ f_2\left(x^{(0)}\right) \\ \vdots \\ f_n\left(x^{(0)}\right) \end{bmatrix}}_{\text{Vetor coluna}} + \underbrace{\begin{bmatrix} \nabla^T f_1(x^{(0)}) \\ \nabla^T f_2(x^{(0)}) \\ \vdots \\ \nabla^T f_n(x^{(0)}) \end{bmatrix}}_{\text{Matriz jacobiana}} \underbrace{\left(x - x^{(0)}\right)}_{\text{Vetor coluna}} + O(\|x - x^{(0)}\|^2)$$

Podemos escrever a linearização de $F(x)$ na seguinte forma mais enxuta:

$$F(x) = F\left(x^{(0)}\right) + J_F(x^{(0)})\left(x - x^{(0)}\right) + O\left(\left\|x - x^{(0)}\right\|^2\right)$$

A matriz jacobiana J_F é matriz cujas linhas são os gradientes transpostos de f_j, ou seja:

$$J_F = \frac{\partial(f_1, f_2, \ldots, f_n)}{\partial(x_1, x_2, \ldots, x_n)} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

A matriz jacobiana de uma função ou simplesmente, o Jacobiano de uma função $F(x)$ é a matriz formada pelas suas derivadas parciais:

$$(J_F)_{ij} = \frac{\partial f_i}{\partial x_j}$$

Exemplo 5.2.1. Calcule a matriz jacobiana da função

$$F(x) = \begin{bmatrix} \frac{x_1^2}{3} + x_2^2 - 1 \\ \\ x_1^2 + \frac{x_2^2}{4} - 1 \end{bmatrix}$$

$$J_F = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} \frac{2x_1}{3} & 2x_2 \\ \\ 2x_1 & \frac{x_2}{2} \end{bmatrix}$$

Capítulo 6

Interpolação

Neste capítulo, discutimos sobre problemas de **interpolação**. Mais precisamente, dado um conjunto com n pontos $\{(x_i, y_i) \in \mathbb{R}^2\}_{i=1}^n$ e uma família de funções $\mathcal{F} = \{f : \mathbb{R} \to \mathbb{R}; y = f(x)\}$, o problema de interpolação consiste em encontrar uma função $f \in \mathcal{F}$ tal que:

$$f(x_i) = y_i, \quad i = 1, 2, \ldots, n.$$

Chamamos uma tal $f(x)$ de **função interpoladora** dos pontos dados. Ou ainda, dizemos que $f(x)$ interpola os pontos dados.

Exemplo 6.0.2. Um dos problemas de interpolação mais simples é o de entrar a equação da reta que passa por dois pontos dados. Por exemplo, sejam dados o conjunto de pontos $\{(1, 1), (2, 2)\}$ e a família de funções $\mathcal{F} = \{f(x) = a + bx; a,b \in \mathbb{R}\}$. Para que uma $f(x)$ na família seja a função interpoladora do conjunto de pontos dados, precisamos que

$$\begin{array}{l} a + bx_1 = y_1 \\ a + bx_2 = y_2 \end{array} \quad \text{i.e.} \quad \begin{array}{l} a + b = 1 \\ a + 2b = 2 \end{array}$$

o que nos fornece $a = 0$ e $b = 1$. Então, a função interpoladora é $f(x) = x$. Os pontos e a reta interpolada estão esboçados na Figura 6.1.

Um problema de interpolação cuja a família de funções constitui-se de polinômios é chamado de problema de interpolação polinomial.

6.1 Interpolação polinomial

Interpolação polinomial é um caso particular do problema geral de interpolação, no qual a família de funções é constituída de polinômios. Para o problema ser bem determinado é necessário restringirmos o grau dos polinômios.

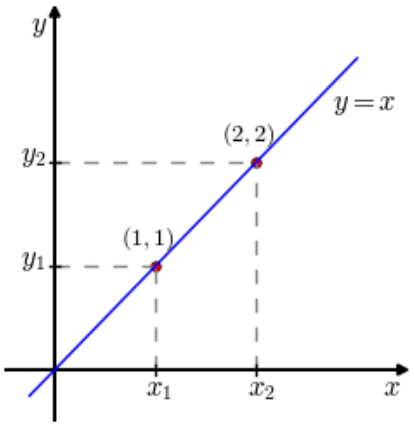


Figura 6.1: Exemplo de interpolação de dois pontos por uma reta, veja o Exemplo 6.0.2.

Observe que para o problema ser bem determinado, é necessário restringirmos o grau dos polinômios. Dado um conjunto de n pontos a serem interpolados $\{(x_i,y_i)\}_{i=1}^n$, $x_i \neq x_j$ para $i \neq j$, a família de polinômios $\mathcal{F} = \mathbb{P}_{n-1}$ deve ser escolhida, onde:

$$\mathbb{P}_{n-1} := \{p(x) = a_1 + a_2x + \cdots + a_nx^{n-1}; a_i \in \mathbb{R}, i = 1,2,\ldots,n\},$$

i.e., a família dos polinômios reais de grau menor ou igual a $n-1$.

O Exemplo 6.0.2 discute um dos casos mais simples de interpolação polinomial, o qual consiste em interpolar uma reta por dois pontos. Neste caso, a família de funções consiste de polinômios de grau 1. Se buscarmos interpolar uma parábola pelos dois pontos dados, o problema fica subdeterminado, pois existem infinitas parábolas que passam por dois pontos dados. Além disso, se buscarmos interpolar uma reta por três pontos dados, o problema estaria sobredeterminado e poderia não ter solução se os pontos não fossem colineares. Veja o Exercício 6.1.3.

Assim, dado um conjunto com n pontos $\{(x_i,y_i)\}_{i=1}^n$, chamamos de **polinômio interpolador** o polinômio de grau menor ou igual a $n-1$ que interpola os pontos dados.

Exemplo 6.1.1. Encontre o polinômio interpolador do conjunto de pontos $\{(0,1), (1,2), (2,4), (3,8)\}$.

Solução. Como o conjunto consiste de 4 pontos, o polinômio interpolador deve ser da forma:

$$p(x) = a_1 + a_2x + a_3x^2 + a_4x^3.$$

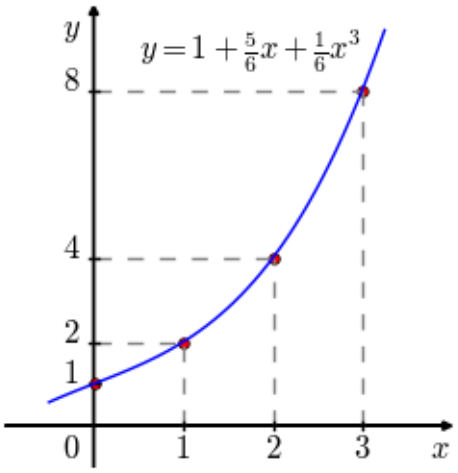


Figura 6.2: Polinômio interpolador do conjunto de pontos $\{(0,1),\ (1,2),\ (2,4),\ (3,8)\}$. Veja o Exemplo 6.1.1.

As condições de interpolação são $p(x_i) = y_i$, $i = 1, 2, 3, 4$, o que nos leva ao sistema linear:

$$\begin{aligned} a_1 & & & & & & & = & 1 \\ a_1 & + & a_2 & + & a_3 & + & a_4 & = & 2 \\ a_1 & + & 2a_2 & + & 4a_3 & + & 8a_4 & = & 4 \\ a_1 & + & 3a_2 & + & 9a_3 & + & 27a_4 & = & 8 \end{aligned}$$

cuja solução é $a_1 = 1$, $a_2 = \frac{5}{6}$, $a_3 = 0$ e $a_4 = \frac{1}{6}$. Portanto, o polinômio interpolador é $p(x) = 1 + \frac{5}{6}x + \frac{1}{6}x^3$. Veja Figura 6.2.

Em `Python`, podemos encontrar o polinômio interpolador e esboçar seu gráfico com os seguintes comandos:

```
>>> xi = np.array([0,1,2,3], dtype='double')
>>> yi = np.array([1,2,4,8], dtype='double')
>>> A = np.array([xi**3,xi**2,xi**1,xi**0]).transpose()
>>> a = np.linalg.inv(A).dot(yi);a
array([ 0.16666667,  0.        ,  0.83333333,  1.        ])
>>> xx = np.linspace(-0.5,3.25);
>>> plt.plot(xi,yi,'ro',xx,np.polyval(a,xx),'b-')
>>> plt.grid();plt.show()
```

◊

Teorema 6.1.1. *Seja* $\{(x_i,y_i)\}_{i=1}^n$ *um conjunto de* n *pares ordenados de números reais tais que* $x_i \neq x_j$ *se* $i \neq j$*, então existe um único polinômio* $p(x)$ *de grau* $n-1$ *ou inferior que passa por todos os pontos dados, isto é,* $p(x_i) = y_i, i = 1, \ldots, n$.

Demonstração. Observe que o problema de encontrar os coeficientes a_1, a_2,..., a_n do polinômio

$$p(x) = a_1 + a_2x + a_3x^2 + \cdots a_nx^{n-1} = \sum_{k=1}^{n} a_kx^{k-1}$$

tal que $p(x_i) = y_i$ é equivalente a resolver o sistema linear com n equações e n incógnitas dado por

$$\begin{aligned} a_1 + a_2x_1 + a_3x_1^2 + \cdots + a_nx_1^{n-1} &= y_1, \\ a_1 + a_2x_2 + a_3x_2^2 + \cdots + a_nx_2^{n-1} &= y_2, \\ &\vdots \\ a_1 + a_2x_n + a_3x_n^2 + \cdots + a_nx_n^{n-1} &= y_n. \end{aligned}$$

O qual pode ser escrito na forma matricial como

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ 1 & x_3 & x_3^2 & \cdots & x_3^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}$$

A matriz envolvida é uma **matriz de Vandermonde** de ordem n cujo determinante é dado por

$$\prod_{1 \leq i < j \leq n} (x_j - x_i)$$

É fácil ver que se as abscissas são diferentes dois a dois, então o determinante é não nulo. Disto decorre que a matriz envolvida é inversível e,portanto, o sistema possui uma solução e esta solução é única. □

Esta abordagem direta que usamos no Exemplo 6.1.1 e na demonstração do Teorema 6.1.1 se mostra ineficiente quando o número de pontos é grande e quando existe grande variação nas abscissas. Neste caso a matriz de Vandermonde é mal condicionada (ver [6]), acarretando um aumento dos erros de arredondamento na solução do sistema.

Uma maneira de resolver este problema é escrever o polinômio em uma base que produza um sistema bem condicionado.

Exercícios

E 6.1.1. Encontre o polinômio interpolador para o conjunto de pontos $\{(-2,-47), (0,-3), (1,4), (2,41)\}$. Então, faça um gráfico com os pontos e o polinômio interpolador encontrado.

E 6.1.2. Encontre o polinômio interpolador para o conjunto de pontos $\{(-1,1{,}25), (0{,}5,0{,}5), (1,1{,}25), (1{,}25,1{,}8125)\}$.

E 6.1.3. Mostre que:

a) Existem infinitas parábolas que interpolam dois pontos dados $\{(x_1,y_1),(x_2,y_2)\}$, com $x_1 \neq x_2$.

b) Não existe reta que interpola os pontos $\{(1,1),(2,2{,}1),(3,3)\}$.

c) Não existe parábola de equação $y = a_1 + a_2x + a_3x^2$ que interpola dois pontos dados $\{(x_1,y_1),(x_1,y_2)\}$, com $y_1 \neq y_2$. Mas, existem infinitas parábolas de equação $x = a_1 + a_2y + a_3y^2$ que interpolam estes pontos.

6.2 Diferenças divididas de Newton

Dado um conjunto com n pontos $\{(x_i,y_i)\}_{i=1}^n$, o **método das diferenças divididas de Newton** consiste em construir o polinômio interpolador da forma

$$\begin{aligned} p(x) &= a_1 + a_2(x-x_1) + a_3(x-x_1)(x-x_2) + \cdots \\ &+ a_n(x-x_1)(x-x_2)\cdots(x-x_{n-1}). \end{aligned}$$

Como $p(x_i) = y_i$, $i = 1,2,\ldots,n$, os coeficientes a_i satisfazem o seguinte sistema triangular inferior:

$$\begin{aligned} a_1 &= y_1 \\ a_1 + a_2(x_2-x_1) &= y_2 \\ a_1 + a_2(x_3-x_1) + a_3(x_3-x_1)(x_3-x_2) &= y_3 \\ &\vdots \\ a_1 + a_2(x_n-x_1) + \cdots + a_n(x_n-x_1)\cdots(x_n-x_{n-1}) &= y_n \end{aligned}$$

Resolvendo de cima para baixo, obtemos

$$\begin{aligned} a_1 &= y_1 \\ a_2 &= \frac{y_2 - a_1}{x_2 - x_1} = \frac{y_2 - y_1}{x_2 - x_1} \\ a_3 &= \frac{y_3 - a_2(x_3 - x_1) - a_1}{(x_3 - x_1)(x_3 - x_2)} = \frac{\frac{y_3 - y_2}{(x_3 - x_2)} - \frac{y_2 - y_1}{(x_2 - x_1)}}{(x_3 - x_1)} \\ &\dots \end{aligned}$$

Note que os coeficientes são obtidos por diferenças das ordenadas divididas por diferenças das abscissas dos pontos dados. Para vermos isso mais claramente, introduzimos a seguinte notação:

$$\begin{aligned} f[x_j] &:= y_j \\ f[x_j, x_{j+1}] &:= \frac{f[x_{j+1}] - f[x_j]}{x_{j+1} - x_j} \\ f[x_j, x_{j+1}, x_{j+2}] &:= \frac{f[x_{j+1}, x_{j+2}] - f[x_j, x_{j+1}]}{x_{j+2} - x_j} \\ &\vdots \\ f[x_j, x_{j+1}, \dots, x_{j+k}] &:= \frac{f[x_{j+1}, x_{j+2}, \dots, x_{j+k}] - f[x_j, x_{j+1}, \dots, x_{j+k-1}]}{x_{j+k} - x_j} \end{aligned}$$

Chamamos $f[x_j]$ de diferença dividida de ordem zero (ou primeira diferença dividida), $f[x_i,x_j+1]$ de diferença dividida de ordem 1 (ou segunda diferença dividida) e assim por diante.

Uma inspeção cuidadosa dos coeficientes obtidos em (6.2) nos mostra que

$$a_k = f[x_1,x_2,\dots,x_k]$$

Isto nos permite esquematizar o método conforme apresentado na Tabela 6.1.

Exemplo 6.2.1. Use o método de diferenças divididas para encontrar o polinômio que passe pelos pontos $(-1,3),(0,1),(1,3),(3,43)$.

j	x_j	$f[x_j]$	$f[x_{j-1},x_j]$	$f[x_{j-2},x_{j-1},x_j]$
1	x_1	$\boldsymbol{f[x_1]}$		
			$\boldsymbol{f[x_1,x_2]} = \frac{f[x_2]-f[x_1]}{x_2-x_1}$	
2	x_2	$f[x_2]$		$\boldsymbol{f[x_1,x_2,x_3]} = \frac{f[x_2,x_3]-f[x_1,x_2]}{x_3-x_1}$
			$f[x_2,x_3] = \frac{f[x_3]-f[x_2]}{x_3-x_2}$	
3	x_2	$f[x_2]$		

Tabela 6.1: Esquema de diferenças divididas para um conjunto com três pontos $\{(x_i, y_i)\}_{i=1}^3$.

Solução. Usando o esquema apresentado na Tabela 6.1, obtemos

j	x_j	$f[x_j]$	$f[x_{j-1},x_j]$	$f[x_{j-2},x_{j-1},x_j]$	$f[x_{j-3},x_{j-2},x_{j-1},x_j]$
1	-1	**3**			
			$\frac{1-3}{0-(-1)} = \mathbf{-2}$		
2	0	1		$\frac{2-(-2)}{1-(-1)} = \mathbf{2}$	
			$\frac{3-1}{1-0} = 2$		$\frac{6-2}{3-(-1)} = \mathbf{1}$
3	1	3		$\frac{20-2}{3-0} = 6$	
			$\frac{43-3}{3-1} = 20$		
4	3	43			

Portanto, o polinômio interpolador do conjunto de pontos dados é

$$p(x) = 3 - 2(x+1) + 2(x+1)x + (x+1)x(x-1)$$

ou, equivalentemente, $p(x) = x^3 + 2x^2 - x + 1$. ◊

6.3 Polinômios de Lagrange

Outra maneira clássica de resolver o problema da interpolação polinomial é através dos polinômios de Lagrange. Dado um conjunto de pontos $\{x_j\}_{j=1}^n$ distintos

dois a dois, definimos os polinômios de Lagrange como os polinômios de grau $n-1$ que satisfazem

$$L_k(x_j) = \begin{cases} 1, & \text{se } k = j \\ 0, & \text{se } k \neq j \end{cases}$$

Assim, o polinômio de grau $n-1$ que interpola os pontos dados, tais $p(x_j) = y_j, j = 1, \ldots, n$ é dado por

$$p(x) = y_1 L_1(x) + y_2 L_2(x) + \cdots + y_n L_n(x) = \sum_{k=1}^{n} y_k L_k(x)$$

Para construir os polinômios de Lagrange, podemos analisar a sua forma fatorada, ou seja:

$$L_k(x) = c_k \prod_{\substack{j=1 \\ j \neq i}}^{n} (x - x_j)$$

onde o coeficiente c_k é obtido da condição $L_k(x_k) = 1$:

$$L_k(x_k) = c_k \prod_{\substack{j=1 \\ j \neq i}}^{n} (x_k - x_j) \Longrightarrow c_k = \frac{1}{\prod_{\substack{j=1 \\ j \neq i}}^{n} (x_k - x_j)}$$

Portanto,

$$L_k(x) = \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_k - x_j)}$$

Observação 6.3.1. O problema de interpolação quando escrito usando como base os polinômios de Lagrange produz um sistema linear diagonal.

Exemplo 6.3.1. Encontre o polinômio da forma $p(x) = a_1 + a_2 x + a_3 x^2 + a_4 x^3$ que passa pelos pontos $(0,0)$, $(1,1)$, $(2,4)$, $(3,9)$. Escrevemos:

$$\begin{aligned}
L_1(x) &= \frac{(x-1)(x-2)(x-3)}{(0-1)(0-2)(0-3)} = -\frac{1}{6}x^3 + x^2 - \frac{11}{6}x + 1 \\
L_2(x) &= \frac{x(x-2)(x-3)}{1(1-2)(1-3)} = \frac{1}{2}x^3 - \frac{5}{2}x^2 + 3x \\
L_3(x) &= \frac{x(x-1)(x-3)}{2(2-1)(2-3)} = -\frac{1}{2}x^3 + 2x^2 - \frac{3}{2}x \\
L_4(x) &= \frac{x(x-1)(x-2)}{3(3-1)(3-2)} = \frac{1}{6}x^3 - \frac{1}{2}x^2 + \frac{1}{3}x
\end{aligned}$$

Assim temos:

$$P(x) = 0 \cdot L_1(x) + 1 \cdot L_2(x) + 4 \cdot L_3(x) + 9 \cdot L_4(x) = x^2$$

Em `Python`, podemos usar a função numpy.poly, por exemplo:

```
>>> xi = np.array([0,1,2,3], dtype='double')
>>> yi = np.array([0,1,4,9], dtype='double')
>>> L1 = np.poly([xi[1],xi[2],xi[3]]);L1=L1/np.polyval(L1,xi[0]);L1
array([-0.16666667,  1.        , -1.83333333,  1.        ])
>>> L2 = np.poly([xi[0],xi[2],xi[3]]);L2=L2/np.polyval(L2,xi[1]);L2
array([ 0.5, -2.5,  3. ,  0. ])
>>> L3 = np.poly([xi[0],xi[1],xi[3]]);L3=L3/np.polyval(L3,xi[2]);L3
array([-0.5,  2. , -1.5, -0. ])
>>> L4 = np.poly([xi[0],xi[1],xi[2]]);L4=L4/np.polyval(L4,xi[3]);L4
array([ 0.16666667, -0.5       ,  0.33333333,  0.        ])
>>> a = yi[0]*L1 + yi[1]*L2 + yi[2]*L3 + yi[3]*L4;a
array([ 0.,  1.,  0.,  0.])
```

Para plotar os pontos e o polinômio interpolador, podemos usar:

```
>>> xx = np.linspace(-0.25,3.25)
>>> plt.plot(xi,yi,'ro',xx,np.polyval(a,xx),'b-')
>>> plt.grid();plt.show()
```

Exemplo 6.3.2. Encontre o polinômio da forma $p(x) = a_1 + a_2x + a_3x^2 + a_4x^3$ que passa pelos pontos $(0,0)$, $(1,1)$, $(2,0)$, $(3,1)$.

Solução. Como as abscissas são as mesmas do exemplo anterior, podemos utilizar os mesmos polinômios de Lagrange, assim temos:

$$p(x) = 0 \cdot L_1(x) + 1 \cdot L_2(x) + 0 \cdot L_3(x) + 1 \cdot L_4(x) = \frac{2}{3}x^3 - 3x^2 + \frac{10}{3}x.$$

$\Diamond$

6.4 Aproximação de funções reais por polinômios interpoladores

Teorema 6.4.1. *Dados $n+1$ pontos distintos, x_0, x_1, $\cdots$, x_n, dentro de um intervalo $[a,b]$ e uma função f com $n+1$ derivadas contínuas nesse intervalo*

($f \in C^{n+1}[a,b]$), então para cada x em $[a,b]$, existe um número $\xi(x)$ em (a,b) tal que

$$f(x) = P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)(x-x_1)\cdots(x-x_n),$$

onde $P(x)$ é o polinômio interpolador. Em especial, pode-se dizer que

$$|f(x) - P(x)| \leq \frac{M}{(n+1)!}\left|(x-x_0)(x-x_1)\cdots(x-x_n)\right|,$$

onde

$$M = \max_{x\in[a,b]} |f^{(n+1)}(\xi(x))|$$

Exemplo 6.4.1. Considere a função $f(x) = \cos(x)$ e o polinômio $P(x)$ de grau 2 tal que $P(0) = \cos(0) = 1$, $P(\frac{1}{2}) = \cos(\frac{1}{2})$ e $P(1) = \cos(1)$. Use a fórmula de Lagrange para encontrar $P(x)$. Encontre o erro máximo que se assume ao aproximar o valor de $\cos(x)$ pelo de $P(x)$ no intervalo [0,1]. Trace os gráficos de $f(x)$ e $P(x)$ no intervalo [0,1] no mesmo plano cartesiano e, depois, trace o gráfico da diferença $\cos(x) - P(x)$. Encontre o erro efetivo máximo $|\cos(x) - P(x)|$.

$$\begin{aligned} P(x) &= 1\frac{(x-\frac{1}{2})(x-1)}{(0-\frac{1}{2})(0-1)} + \cos\left(\frac{1}{2}\right)\frac{(x-0)(x-1)}{(\frac{1}{2}-0)(\frac{1}{2}-1)} + \cos(1)\frac{(x-0)(x-\frac{1}{2})}{(1-0)(1-\frac{1}{2})} \\ &\approx 1 - 0{,}0299720583066x - 0{,}4297256358252x^2 \end{aligned}$$

Para encontrar o erro máximo, precisamos estimar $|f'''(x)| = |\sin(x)| \leq \sin(1) < 0{,}85$ e

$$\max_{x\in[0,1]} \left|x\left(x-\frac{1}{2}\right)(x-1)\right|$$

O polinômio de grau três $Q(x) = x\left(x-\frac{1}{2}\right)(x-1)$ tem um mínimo (negativo) em $x_1 = \frac{3+\sqrt{3}}{6}$ e um máximo (positivo) em $x_2 = \frac{3-\sqrt{3}}{6}$. Logo:

$$\max_{x\in[0,1]} \left|x\left(x-\frac{1}{2}\right)(x-1)\right| \leq \max\{|Q(x_1)|, |Q(x_2)|\} \approx 0{,}0481125.$$

Portanto:

$$|f(x) - P(x)| < \frac{0{,}85}{3!}0{,}0481125 \approx 0{,}0068159 < 7\cdot 10^{-3}$$

Para encontrar o erro efetivo máximo, basta encontrar o máximo de $|P(x) - cos(x)|$. O mínimo (negativo) de $P(x) - cos(x)$ acontece em $x_1 = 4{,}29\cdot 10^{-3}$ e o máximo (positivo) acontece em $x_2 = 3{,}29\cdot 10^{-3}$. Portanto, o erro máximo efetivo é $4{,}29\cdot 10^{-3}$.

Exemplo 6.4.2. Considere o problema de aproximar o valor da integral $\int_0^1 f(x)dx$ pelo valor da integral do polinômio $P(x)$ que coincide com $f(x)$ nos pontos $x_0 = 0$, $x_1 = \frac{1}{2}$ e $x_2 = 1$. Use a fórmula de Lagrange para encontrar $P(x)$. Obtenha o valor de $\int_0^1 f(x)dx$ e encontre uma expressão para o erro de truncamento.

O polinômio interpolador de $f(x)$ é

$$\begin{aligned} P(x) &= f(0)\frac{(x-\frac{1}{2})(x-1)}{(0-\frac{1}{2})(0-1)} + f\left(\frac{1}{2}\right)\frac{(x-0)(x-1)}{(\frac{1}{2}-0)(\frac{1}{2}-1)} + f(1)\frac{(x-0)(x-\frac{1}{2})}{(1-0)(1-\frac{1}{2})} \\ &= f(0)(2x^2-3x+1) + f\left(\frac{1}{2}\right)(-4x^2+4x) + f(1)(2x^2-x) \end{aligned}$$

e a integral de $P(x)$ é:

$$\begin{aligned} \int_0^1 P(x)dx &= \left[f(0)\left(\frac{2}{3}x^3 - \frac{3}{2}x^2 + x\right)\right]_0^1 + \left[f\left(\frac{1}{2}\right)\left(-\frac{4}{3}x^3 + 2x^2\right)\right]_0^1 \\ &+ \left[f(1)\left(\frac{2}{3}x^3 - \frac{1}{2}x^2\right)\right]_0^1 \\ &= f(0)\left(\frac{2}{3} - \frac{3}{2} + 1\right) + f\left(\frac{1}{2}\right)\left(-\frac{4}{3} + 2\right) + f(1)\left(\frac{2}{3} - \frac{1}{2}\right) \\ &= \frac{1}{6}f(0) + \frac{2}{3}f\left(\frac{1}{2}\right) + \frac{1}{6}f(1) \end{aligned}$$

Para fazer a estimativa de erro usando o Teorema 6.4.1, e temos

$$\begin{aligned} \left|\int_0^1 f(x)dx - \int_0^1 P(x)dx\right| &= \left|\int_0^1 f(x) - P(x)dx\right| \\ &\leq \int_0^1 |f(x) - P(x)|dx \\ &\leq \frac{M}{6}\int_0^1 \left|x\left(x-\frac{1}{2}\right)(x-1)\right|dx \\ &= \frac{M}{6}\left[\int_0^{1/2} x\left(x-\frac{1}{2}\right)(x-1)dx\right. \\ &- \left.\int_{1/2}^1 x\left(x-\frac{1}{2}\right)(x-1)dx\right] \\ &= \frac{M}{6}\left[\frac{1}{64} - \left(-\frac{1}{64}\right)\right] = \frac{M}{192}. \end{aligned}$$

Lembramos que $M = \max_{x\in[0,1]} |f'''(x)|$.

Observação 6.4.1. Existem estimativas melhores para o erro de truncamento para este esquema de integração numérica. Veremos com mais detalhes tais esquemas na teoria de integração numérica.

Exemplo 6.4.3. Use o resultado do exemplo anterior para aproximar o valor das seguintes integrais:

a) $\int_0^1 \ln(x+1)dx$

b) $\int_0^1 e^{-x^2}dx$

Solução. Usando a fórmula obtida, temos que

$$\int_0^1 \ln(x+1)dx \approx 0{,}39 \pm \frac{1}{96}$$

$$\int_0^1 e^{-x^2}dx \approx 0{,}75 \pm \frac{3{,}87}{192}$$

$\Diamond$

Exercícios

E 6.4.1. Use as mesmas técnicas usadas o resultado do Exemplo 6.4.2 para obter uma aproximação do valor de:

$$\int_0^1 f(x)dx$$

através do polinômio interpolador que coincide com $f(x)$ nos pontos $x = 0$ e $x = 1$.

6.5 Interpolação linear segmentada

Considere o conjunto $(x_i,y_i)_{j=1}^n$ de n pontos. Assumiremos que $x_{i+1} > x_i$, ou seja, as abscissas são distintas e estão em ordem crescente. A função linear que interpola os pontos x_i e x_{i+1} no intervalo i é dada por

$$P_i(x) = y_i\frac{(x_{i+1}-x)}{(x_{i+1}-x_i)} + y_{i+1}\frac{(x-x_i)}{(x_{i+1}-x_i)}$$

O resultado da interpolação linear segmentada é a seguinte função contínua definida por partes no intervalo $[x_1,x_n]$:

$$f(x) = P_i(x), \qquad x \in [x_i,x_{i+1}]$$

Exemplo 6.5.1. Construa uma função linear por partes que interpola os pontos (0,0), (1,4), (2,3), (3,0), (4,2), (5,0).

A função procurada pode ser construída da seguinte forma:

$$f(x) = \begin{cases} 0\frac{x-1}{0-1} + 1\frac{x-0}{1-0} & , 0 \leq x < 1 \\ 4\frac{x-2}{1-2} + 3\frac{x-1}{2-1} & , 1 \leq x < 2 \\ 3\frac{x-3}{2-3} + 0\frac{x-2}{3-2} & , 2 \leq x \leq 3 \end{cases}$$

Simplificando, obtemos:

$$f(x) = \begin{cases} x & , 0 \leq x < 1 \\ -x + 5 & , 1 \leq x < 2 \\ -3x + 9 & , 2 \leq x \leq 3 \end{cases}$$

A Figura 6.3 é um esboço da função $f(x)$ obtida.

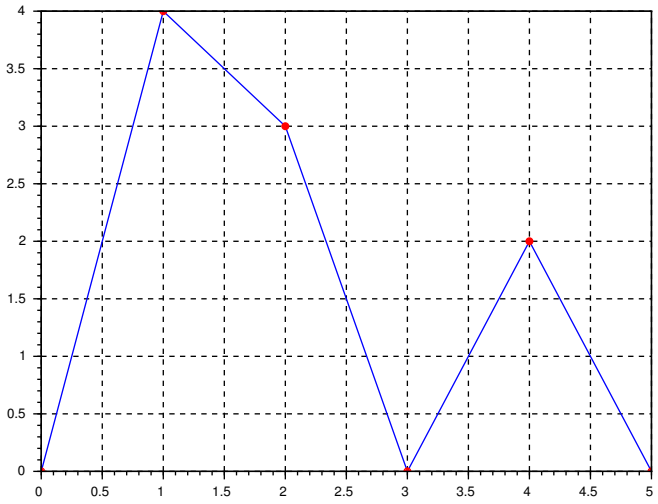


Figura 6.3: Interpolação linear segmentada.

6.6 Interpolação cúbica segmentada - spline

Dado um conjunto de n pontos $(x_j,y_j)_{j=1}^{n}$ tais que $x_{j+1} > x_j$, ou seja, as abscissas são distintas e estão em ordem crescente; um spline cúbico que interpola

estes pontos é uma função $s(x)$ com as seguintes propriedades:

i Em cada segmento $[x_j,x_{j+1}]$, $j = 1,2,\ldots n-1$ $s(x)$ é um polinômio cúbico.

ii para cada ponto, $s(x_j) = y_j$, i.e., o spline interpola os pontos dados.

iii $s(x) \in C^2$, i.e., é função duas vezes continuamente diferenciável.

Da primeira hipótese, escrevemos

$$s(x) = s_j(x), x \in [x_j,x_{j+1}], \quad j = 1,\ldots,n-1$$

com

$$s_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$$

O problema agora consiste em obter os 4 coeficientes de cada um desses $n-1$ polinômios cúbicos.

Veremos que a simples definição de spline produz $4n-6$ equações linearmente independentes:

$$\begin{aligned}
s_j(x_j) &= y_j, & j &= 1,\ldots,n-1\\
s_j(x_{j+1}) &= y_{j+1}, & j &= 1,\ldots,n-1\\
s'_j(x_{j+1}) &= s'_{j+1}(x_{j+1}), & j &= 1,\ldots,n-2\\
s''_j(x_{j+1}) &= s''_{j+1}(x_{j+1}), & j &= 1,\ldots,n-2
\end{aligned}$$

Como

$$s'_j(x) = b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2 \tag{6.1}$$

e

$$s''_j(x) = 2c_j + 6d_j(x - x_j), \tag{6.2}$$

temos, para $j = 1,\ldots,n-1$, as seguintes equações

$$\begin{aligned}
a_j &= y_j,\\
a_j + b_j(x_{j+1} - x_j) + c_j(x_{j+1} - x_j)^2 + d_j(x_{j+1} - x_j)^3 &= y_{j+1},\\
b_j + 2c_j(x_{j+1} - x_j) + 3d_j(x_{j+1} - x_j)^2 &= b_{j+1},\\
c_j + 3d_j(x_{j+1} - x_j) &= c_{j+1},
\end{aligned}$$

Por simplicidade, definimos

$$h_j = x_{j+1} - x_j$$

e temos

$$\begin{aligned} a_j &= y_j, \\ a_j + b_j h_j + c_j h_j^2 + d_j h_j^3 &= y_{j+1}, \\ b_j + 2c_j h_j + 3d_j h_j^2 &= b_{j+1}, \\ c_j + 3d_j h_j &= c_{j+1}, \end{aligned} \tag{6.3}$$

que podem ser escrita da seguinte maneira

$$a_j = y_j, \tag{6.4}$$

$$d_j = \frac{c_{j+1} - c_j}{3h_j}, \tag{6.5}$$

$$\begin{aligned} b_j &= \frac{y_{j+1} - y_j - c_j h_j^2 - \frac{c_{j+1}-c_j}{3h_j} h_j^3}{h_j}, \\ &= \frac{3y_{j+1} - 3y_j - 3c_j h_j^2 - c_{j+1} h_j^2 + c_j h_j^2}{3h_j} \\ &= \frac{3y_{j+1} - 3y_j - 2c_j h_j^2 - c_{j+1} h_j^2}{3h_j} \end{aligned} \tag{6.6}$$

Trocando o índice j por $j-1$ na terceira equação (6.3), $j = 2, \ldots, n-1$

$$b_{j-1} + 2c_{j-1}h_{j-1} + 3d_{j-1}h_{j-1}^2 = b_j \tag{6.7}$$

e, portanto,

$$\begin{aligned} \frac{3y_j - 3y_{j-1} - 2c_{j-1}h_{j-1}^2 - c_j h_{j-1}^2}{3h_{j-1}} + 2c_{j-1}h_{j-1} + c_j h_{j-1} - c_{j-1}h_{j-1} \\ = \frac{3y_{j+1} - 3y_j - 2c_j h_j^2 - c_{j+1} h_j^2}{3h_j}. \end{aligned} \tag{6.8}$$

Fazendo as simplificações, obtemos:

$$c_{j-1}h_{j-1} + c_j(2h_j + 2h_{j-1}) + c_{j+1}h_j = 3\frac{y_{j+1} - y_j}{h_j} - 3\frac{y_j - y_{j-1}}{h_{j-1}}. \tag{6.9}$$

É costumeiro acrescentar a incógnita c_n ao sistema. A incógnita c_n não está relacionada a nenhum dos polinômios interpoladores. Ela é uma construção artificial que facilita o cálculo dos coeficientes do spline. Portanto, a equação acima pode ser resolvida para $j = 2, \ldots, n-1$.

Para determinar unicamente os n coeficientes c_n precisamos acrescentar duas equações linearmente independentes às $n-2$ equações dadas por (6.9). Essas duas equações adicionais definem o tipo de spline usado.

6.6.1 Spline natural

Uma forma de definir as duas equações adicionais para completar o sistema (6.9) é impor condições de fronteira livres (ou naturais), ou seja,

$$S''(x_1) = S''(x_n) = 0. \tag{6.10}$$

Substituindo na equação (6.2)

$$s_1''(x_1) = 2c_1 + 6d_1(x_1 - x_1) = 0 \Longrightarrow c_1 = 0.$$

e

$$s_{n-1}''(x_n) = 2c_{n-1} + 6d_{n-1}(x_n - x_{n-1}) = 0.$$

Usando o fato que

$$c_{n-1} + 3d_{n-1}h_{n-1} = c_n$$

temos que

$$c_n = -3d_{n-1}(x_n - x_{n-1}) + 3d_{n-1}h_{n-1} = 0.$$

Essa duas equações para c_1 e c_n juntamente com as equações (6.9) formam um sistema de n equações $Ac = z$, onde

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ h_1 & 2h_2 + 2h_1 & h_2 & 0 & \cdots & 0 & 0 \\ 0 & h_2 & 2h_3 + 2h_2 & h_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & h_{n-2} & 2h_{n-2} + 2h_{n-1} & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \tag{6.11}$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} \quad \text{e} \quad z = \begin{bmatrix} 0 \\ 3\frac{y_3 - y_2}{h_2} - 3\frac{y_2 - y_1}{h_1} \\ 3\frac{y_4 - y_3}{h_3} - 3\frac{y_3 - y_2}{h_2} \\ \vdots \\ 3\frac{y_{n-1} - y_{n-2}}{h_{n-2}} - 3\frac{y_{n-2} - y_{n-3}}{h_{n-3}} \\ 0 \end{bmatrix} \tag{6.12}$$

Observe que a matriz A é diagonal dominante estrita e, portanto, o sistema $Ac = z$ possui solução única. Calculado c, os valores dos a_n, b_n e d_n são obtidos diretamente pelas expressões (6.4), (6.6) e (6.5), respectivamente.

Exemplo 6.6.1. Construa um spline cúbico natural que passe pelos pontos (2, 4,5), (5, − 1,9), (9, 0,5) e (12, − 0,5).

Solução. O spline desejado é uma função definida por partes da forma:

$$S(x) = \begin{cases} a_1 + b_1(x-2) + c_1(x-2)^2 + d_1(x-2)^3 & , 2 \leq x < 5 \\ a_2 + b_2(x-5) + c_2(x-5)^2 + d_2(x-5)^3 & , 5 \leq x < 9 \\ a_3 + b_3(x-9) + c_3(x-9)^2 + d_3(x-9)^3 & , 9 \leq x \leq 12 \end{cases} . \tag{6.13}$$

Os coeficientes c_1, c_2 e c_3 resolvem o sistema $Ac = z$, onde

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & 2\cdot 3 + 2\cdot 4 & 4 & 0 \\ 0 & 4 & 2\cdot 4 + 2\cdot 3 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & 14 & 4 & 0 \\ 0 & 4 & 14 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} \quad \text{e} \quad z = \begin{bmatrix} 0 \\ 3\frac{0,5-(-1,9)}{4} - 3\frac{(-1,9)-4,5}{3} \\ 3\frac{-0,5-0,5}{3} - 3\frac{0,5-(-1,9)}{4} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 8,2 \\ -2,8 \\ 0 \end{bmatrix}$$

Observe que c_4 é um coeficiente artificial para o problema. A solução é $c_1 = 0$, $c_2 = 0{,}7$, $c_3 = -0{,}4$ e $c_4 = 0$. Calculamos os demais coeficientes usando as expressões (6.4), (6.6) e (6.5):

$$\begin{aligned} a_1 &= y_1 = 4{,}5 \\ a_2 &= y_2 = -1{,}9 \\ a_3 &= y_3 = 0{,}5 \end{aligned}$$

$$\begin{aligned} d_1 &= \frac{c_2 - c_1}{3h_1} = \frac{0{,}7 - 0}{3\cdot 3} = 0{,}0777778 \\ d_2 &= \frac{c_3 - c_2}{3h_2} = \frac{-0{,}4 - 0{,}7}{3\cdot 4} = -0{,}0916667 \\ d_3 &= \frac{c_4 - c_3}{3h_3} = \frac{0 + 0{,}4}{3\cdot 3} = 0{,}0444444 \end{aligned}$$

$$\begin{aligned}
b_1 &= \frac{y_2 - y_1}{h_1} - \frac{h_1}{3}(2c_1 + c_2) \\
&= \frac{-1{,}9 - 4{,}5}{3} - \frac{3}{3}(2 \cdot 0 - 0{,}7) = -2{,}8333333 \\
b_2 &= \frac{y_3 - y_2}{h_2} - \frac{h_2}{3}(2c_2 + c_3) \\
&= \frac{0{,}5 - (-1{,}9)}{4} - \frac{4}{3}(2 \cdot 0{,}7 + 0{,}4) = -0{,}7333333 \\
b_3 &= \frac{y_4 - y_3}{h_3} - \frac{h_3}{3}(2c_3 + c_4) \\
&= \frac{-0{,}5 - 0{,}5}{3} - \frac{3}{3}(2 \cdot (-0{,}4) + 0) = 0{,}4666667
\end{aligned}$$

Portanto:

$$S(x) = \begin{cases} 4{,}5 - 2{,}833(x-2) + 0{,}078(x-2)^3 & , 2 \leq x < 5 \\ -1{,}9 - 0{,}733(x-5) + 0{,}7(x-5)^2 - 0{,}092(x-5)^3 & , 5 \leq x < 9 \\ 0{,}5 + 0{,}467(x-9) - 0{,}4(x-9)^2 + 0{,}044(x-9)^3 & , 9 \leq x \leq 12 \end{cases}$$

◊

6.6.2 Spline fixado

Alternativamente, para completar o sistema (6.9), podemos impor condições de contorno fixadas, ou seja,

$$\begin{aligned}
S'(x_1) &= f'(x_1) \\
S'(x_n) &= f'(x_n).
\end{aligned}$$

Substituindo na equação (6.1)

$$s_1'(x_1) = b_1 + 2c_1(x_1 - x_1) + 3d_j(x_1 - x_1)^2 = f'(x_1) \Longrightarrow b_1 = f'(x_1) \tag{6.14}$$

e

$$\begin{aligned}
s_{n-1}'(x_n) &= b_{n-1} + 2c_{n-1}(x_n - x_{n-1}) + 3d_j(x_n - x_{n-1})^2 \\
&= b_{n-1} + 2c_{n-1}h_{n-1} + 3d_{n-1}h_{n-1}^2 = f'(x_n)
\end{aligned} \tag{6.15}$$

Usando as equações (6.5) e (6.6) para $j = 1$ e $j = n - 1$, temos:

$$2c_1h_1 + c_2h_1 = 3\frac{y_2 - y_1}{h_1} - 3f'(x_1) \tag{6.16}$$

e

$$c_{n-1}h_{n-1} + c_n h_{n-1} = 3f'(x_n) - 3\frac{y_n - y_{n-1}}{h_{n-1}} \tag{6.17}$$

Essas duas equações juntamente com as equações (6.9) formam um sistema de n equações $Ac = z$, onde

$$A = \begin{bmatrix} 2h_1 & h_1 & 0 & 0 & \cdots & 0 & 0 \\ h_1 & 2h_2 + 2h_1 & h_2 & 0 & \cdots & 0 & 0 \\ 0 & h_2 & 2h_3 + 2h_2 & h_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & h_{n-2} & 2h_{n-2} + 2h_{n-1} & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & h_{n-1} & 2h_{n-1} \end{bmatrix}$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} \quad \text{e} \quad z = \begin{bmatrix} 3\frac{y_2 - y_1}{h_1} - 3f'(x_1) \\ 3\frac{y_3 - y_2}{h_2} - 3\frac{y_2 - y_1}{h_1} \\ 3\frac{y_4 - y_3}{h_3} - 3\frac{y_3 - y_2}{h_2} \\ \vdots \\ 3\frac{y_{n-1} - y_{n-2}}{h_{n-2}} - 3\frac{y_{n-2} - y_{n-3}}{h_{n-3}} \\ 3f'(x_n) - 3\frac{y_n - y_{n-1}}{h_{n-1}} \end{bmatrix}$$

Observe que a matriz A é diagonal dominante estrita e, portanto, o sistema $Ac = z$ possui solução única. Calculado c, os valores dos a_n, b_n e d_n são obtidos diretamente pelas expressões (6.4), (6.6) e (6.5), respectivamente.

Exemplo 6.6.2. Construa um spline cúbico com fronteira fixada que interpola a função $y = \sin(x)$ nos pontos $x = 0$, $x = \frac{\pi}{2}$, $x = \pi$, $x = \frac{3\pi}{2}$ e $x = 2\pi$.

O spline desejado passa pelos pontos (0,0), (π/2,1), (π,0), ($3\pi/2, -1$) e (2π,0) e tem a forma:

$$S(x) = \begin{cases} a_1 + b_1 x + c_1 x^2 + d_1 x^3 & , 0 \leq x < \frac{\pi}{2} \\ a_2 + b_2(x - \frac{\pi}{2}) + c_2(x - \frac{\pi}{2})^2 + d_2(x - \frac{\pi}{2})^3 & , \frac{\pi}{2} \leq x < \pi \\ a_3 + b_3(x - \pi) + c_3(x - \pi)^2 + d_3(x - \pi)^3 & , \pi \leq x < \frac{3\pi}{2} \\ a_4 + b_4(x - \frac{3\pi}{2}) + c_4(x - \frac{3\pi}{2})^2 + d_4(x - \frac{3\pi}{2})^3 & , \frac{3\pi}{2} \leq x \leq 2\pi \end{cases}.$$

Observe que ele satisfaz as condição de contorno $f'(0) = cos(0) = 1$ e $f'(2\pi) = cos(2\pi) = 1$.

Os coeficientes c_1, c_2, c_3 e c_4 resolvem o sistema $Ac = z$, onde:

$$A = \begin{bmatrix} \pi & \pi/2 & 0 & 0 & 0 \\ \pi/2 & 2\pi & \pi/2 & 0 & 0 \\ 0 & \pi/2 & 2\pi & \pi/2 & 0 \\ 0 & 0 & \pi/2 & 2\pi & \pi/2 \\ 0 & 0 & 0 & \pi/2 & \pi \end{bmatrix}$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} \quad \text{e} \quad z = \begin{bmatrix} 3\frac{1-0}{\pi/2} - 3 \cdot 1 \\ 3\frac{0-1}{\pi/2} - 3\frac{1-0}{\pi/2} \\ 3\frac{-1-0}{\pi/2} - 3\frac{0-1}{\pi/2} \\ 3\frac{0-(-1)}{\pi/2} - 3\frac{(-1)-0}{\pi/2} \\ 3 \cdot 1 - 3\frac{0-(-1)}{\pi/2} \end{bmatrix} = \begin{bmatrix} 6/\pi - 3 \\ -12/\pi \\ 0 \\ 12/\pi \\ 3 - 6/\pi \end{bmatrix}$$

Aqui c_5 é um coeficiente artificial para o problema. A solução é $c_1 = -0{,}0491874$, $c_2 = -0{,}5956302$, $c_3 = 0$, $c_4 = 0{,}5956302$ e $c_5 = 0{,}0491874$. Calculamos os demais coeficientes usando as expressões (6.4), (6.6) e (6.5):

$$\begin{aligned} a_1 &= y_1 = 0 \\ a_2 &= y_2 = 1 \\ a_3 &= y_3 = 0 \\ a_4 &= y_3 = -1 \end{aligned}$$

$$\begin{aligned} d_1 &= \frac{c_2 - c_1}{3h_1} = \frac{-0{,}5956302 - (-0{,}0491874)}{3 \cdot \pi/2} = -0{,}1159588 \\ d_2 &= \frac{c_3 - c_2}{3h_2} = \frac{0 - (-0{,}5956302)}{3 \cdot \pi/2} = 0{,}1263967 \\ d_3 &= \frac{c_4 - c_3}{3h_3} = \frac{0{,}5956302 - 0}{3 \cdot \pi/2} = 0{,}1263967 \\ d_4 &= \frac{c_5 - c_4}{3h_4} = \frac{0{,}0491874 - 0{,}5956302}{3 \cdot \pi/2} = -0{,}1159588 \end{aligned}$$

$$
\begin{aligned}
b_1 &= \frac{y_2 - y_1}{h_1} - \frac{h_1}{3}(2c_1 + c_2) \\
&= \frac{1-0}{\pi/2} - \frac{\pi/2}{3}(2 \cdot (-0{,}0491874) - 0{,}5956302) = 1 \\
b_2 &= \frac{y_3 - y_2}{h_2} - \frac{h_2}{3}(2c_2 + c_3) \\
&= \frac{0-1}{\pi/2} - \frac{\pi/2}{3}(2 \cdot (-0{,}5956302) + 0) = -0{,}0128772 \\
b_3 &= \frac{y_4 - y_3}{h_3} - \frac{h_3}{3}(2c_3 + c_4) \\
&= \frac{-1-0}{\pi/2} - \frac{\pi/2}{3}(2 \cdot 0 + 0{,}5956302) = -0{,}9484910 \\
b_4 &= \frac{y_5 - y_4}{h_4} - \frac{h_4}{3}(2c_4 + c_5) \\
&= \frac{0-(-1)}{\pi/2} - \frac{\pi/2}{3}(2 \cdot 0{,}5956302 + 0{,}0491874) = -0{,}0128772
\end{aligned}
$$

Portanto,

$$
S(x) = \begin{cases}
x - 0{,}049x^2 - 0{,}12x^3 & , 0 \leq x < \frac{\pi}{2} \\
1 + -0{,}01(x - \frac{\pi}{2}) - 0{,}6(x - \frac{\pi}{2})^2 + 0{,}13(x - \frac{\pi}{2})^3 & , \frac{\pi}{2} \leq x < \pi \\
-0{,}95(x - \pi) + 0{,}13(x - \pi)^3 & , \pi \leq x < \frac{3\pi}{2} \\
-1 - 0{,}01(x - \frac{3\pi}{2}) + 0{,}6(x - \frac{3\pi}{2})^2 - 0{,}12(x - \frac{3\pi}{2})^3 & , \frac{3\pi}{2} \leq x \leq 2\pi
\end{cases}
$$

6.6.3 Resumo sobre Splines

Dado um conjunto de pontos (x_i,y_i), $i = 1,2,\ldots,n$, um spline cúbico é a seguinte função interpoladora definida por partes:

$$
S(x) = \begin{cases}
a_1 + b_1(x - x_1) + c_1(x - x_1)^2 + d_1(x - x_1)^3 & , x_1 \leq x < x_2 \\
a_2 + b_2(x - x_2) + c_2(x - x_2)^2 + d_2(x - x_2)^3 & , x_2 \leq x < x_3 \\
\vdots & \vdots \\
a_{n-1} + b_{n-1}(x - x_{n-1}) + c_{n-1}(x - x_{n-1})^2 + d_{n-1}(x - x_{n-1})^3 & , x_{n-1} \leq x \leq x_n
\end{cases}
$$

Definindo-se $h_j = x_{j+1} - x_j$, os coeficientes c_j, $j = 1,2,\ldots,n$, são solução do

sistema linear $Ac = z$, onde:

Spline Natural $s_1''(x_1) = 0$ e $s_{n-1}''(x_n) = 0$	Spline Fixado $s_1'(x_1) = f'(x_1)$ e $s_{n-1}'(x_n) = f'(x_n)$
$a_{i,j} = \begin{cases} 1 & ,j = i = 1 \\ h_{i-1} & ,j = i-1, i < n \\ 2(h_i + h_{i-1}) & ,j = i, 1 < i < n \\ h_i & ,j = i+1, i > 1 \\ 1 & ,j = i = n \\ 0 & ,\text{caso contrário.} \end{cases}$	$a_{i,j} = \begin{cases} 2h_1 & ,j = i = 1 \\ h_{i-1} & ,j = i-1 \\ 2(h_i + h_{i-1}) & ,j = i, 1 < i < n \\ h_i & ,j = i+1 \\ 2h_{n-1} & ,j = i = n \\ 0 & ,\text{caso contrário.} \end{cases}$
$z_i = \begin{cases} 0 & ,i = 1 \\ 3\frac{y_{i+1}-y_i}{h_i} - 3\frac{y_i-y_{i-1}}{h_{i-1}} & ,1 < i < n \\ 0 & ,i = n \end{cases}$	$z_i = \begin{cases} 3\frac{y_2-y_1}{h_1} - 3f'(x_1) & ,i = 1 \\ 3\frac{y_{i+1}-y_i}{h_i} - 3\frac{y_i-y_{i-1}}{h_{i-1}} & ,1 < i < n \\ 3f'(x_n) - 3\frac{y_n-y_{n-1}}{h_{n-1}} & ,i = n \end{cases}$

os coeficientes a_j, b_j e d_j, $j = 1,2,\ldots,n-1$, são calculados conforme segue:

$$\begin{aligned} a_j &= y_j \\ b_j &= \frac{3y_{j+1} - 3y_j - 2c_j h_j^2 - c_{j+1} h_j^2}{3h_j} \\ d_j &= \frac{c_{j+1} - c_j}{3h_j} \end{aligned}$$

Capítulo 7

Ajuste de curvas

Neste capítulo, discutimos sobre problemas de **ajuste de curvas** pelo **método dos mínimos quadrados**. Mais precisamente, dado um conjunto de N pontos $\{(x_j, y_j) \in \mathbb{R}^2\}_{j=1}^{N}$ e uma família de funções $\mathcal{F} = \{f : \mathbb{R} \to \mathbb{R}; y = f(x)\}$, o problema de ajuste de curvas consiste em encontrar uma função da família $\mathcal{F}$ que melhor se ajusta aos pontos dados, não necessariamente que os interpola.

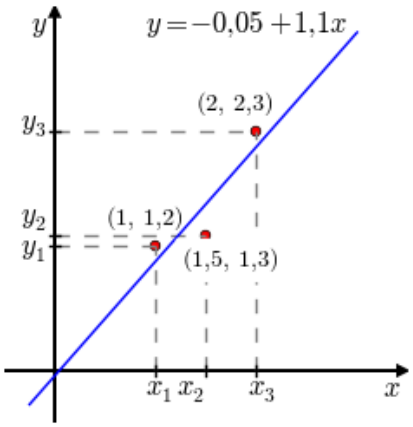


Figura 7.1: Exemplo de um problema de ajuste de uma reta entre três pontos, veja o Exemplo 7.0.3.

Aqui, o termo "melhor se ajusta" é entendido no sentido de mínimos quadrados, i.e. buscamos encontrar uma função $f \in \mathcal{F}$ tal que $f(x)$ resolve o seguinte problema de minimização

$$\min_{f \in \mathcal{F}} \sum_{j=1}^{N} \left(f(x_j) - y_j\right)^2,$$

ou seja, $f(x)$ é a função da família $\mathcal{F}$ cujo erro quadrático entre y_j e $f(x_j)$, $j =$

$1, 2, \ldots, N$, é mínimo. A expressão

$$\begin{aligned} R &:= \sum_{j=1}^{N} \left(f(x_j) - y_j\right)^2 \\ &= \left(f(x_1) - y_1\right)^2 + \left(f(x_2) - y_2\right)^2 + \cdots + \left(f(x_N) - y_N\right)^2 \end{aligned}$$

é chamada de **resíduo** e consiste na soma dos quadrados das diferenças entre a ordenadas y_j e o valor da função procurada $f(x_j)$.

Exemplo 7.0.3. Dado o conjunto de pontos $\{(1, 1{,}2), (1{,}5, 1{,}3), (2, 2{,}3)\}$ e a família de retas $f(x) = a + bx$, podemos mostrar que $f(x) = -0{,}05 + 1{,}1x$ é a reta que melhor aproxima os pontos dados no sentido de mínimos quadrados. Os pontos e a reta ajustada e são esboçados na Figura 7.1.

Na sequência, discutimos o procedimento de ajuste de uma reta, então, mostramos a generalização da técnica para problemas lineares de ajuste e, por fim, discutimos alguns problemas de ajuste não lineares.

Ao longo deste capítulo, estaremos assumindo que as seguintes bibliotecas e módulos `Python` estão carregadas:

```
>>> from __future__ import division
>>> import numpy as np
>>> from numpy import linalg
>>> import matplotlib.pyplot as plt
```

7.1 Ajuste de uma reta

Nesta seção, discutiremos o procedimento de ajuste de uma reta a um conjunto de pontos dados. Em outras palavras, discutiremos o método de solução para o problema de encontrar o polinômio do primeiro grau que melhor se aproxima a um dado conjunto de pontos pelo método dos mínimos quadrados.

Seja, então, $\{(x_1,y_1), (x_2,y_2), \ldots, (x_N,y_N)\}$ um conjunto de N pontosdados. Buscamos encontrar a função $f(x) = a_1 + a_2x$ tal que o resíduo

$$R = \sum_{j=1}^{N} (f(x_j) - y_j)^2$$

seja mínimo.

Para tal, primeiro observamos que $f(x_j) = a_1 + a_2x_j$ e, portanto, o resíduo pode ser escrito explicitamente como uma função de a_1 e a_2 conforme a seguinte expressão:

$$R(a_1,a_2) = \sum_{j=1}^{N} (a_1 + a_2x_j - y_j)^2.$$

Observamos que $R(a_1,a_2)$ é uma forma quadrática e que seu mínimo ocorre quando suas derivadas parciais primeiras são iguais a zero, isto é

$$\begin{aligned}\frac{\partial R}{\partial a_1} &= \frac{\partial}{\partial a_1}\sum_{j=1}^{N}(a_1+a_2x_j-y_j)^2=0\\ \frac{\partial R}{\partial a_2} &= \frac{\partial}{\partial a_2}\sum_{j=1}^{N}(a_1+a_2x_j-y_j)^2=0\end{aligned}$$

ou seja,

$$\begin{aligned}2\sum_{j=1}^{N}(a_1+a_2x_j-y_j)\cdot 1 &= 0\\ 2\sum_{j=1}^{N}(a_1+a_2x_j-y_j)\cdot x_j &= 0\end{aligned}$$

e isolando as incógnitas temos

$$\begin{aligned}a_1\sum_{j=1}^{N}1+a_2\sum_{j=1}^{N}x_j &= \sum_{j=1}^{N}y_j\\ a_1\sum_{j=1}^{N}x_j+a_2\sum_{j=1}^{N}x_j^2 &= \sum_{j=1}^{N}y_jx_j.\end{aligned}$$

Observando que $\sum_{j=1}^{N}1=N$, o sistema linear acima pode ser escrito na forma matricial $Ma=w$, i.e.,

$$\underbrace{\begin{bmatrix}N & \sum_{j=1}^{N}x_j\\ \sum_{j=1}^{N}x_j & \sum_{j=1}^{N}x_j^2\end{bmatrix}}_{M}\underbrace{\begin{bmatrix}a_1\\ a_2\end{bmatrix}}_{a}=\underbrace{\begin{bmatrix}\sum_{j=1}^{N}y_j\\ \sum_{j=1}^{N}x_jy_j\end{bmatrix}}_{w}. \tag{7.1}$$

Este sistema linear de duas equações e duas incógnitas admite uma única solução quando o determinante da matriz dos coeficientes for não nulo, isto é

$$N\sum_{j=1}^{N}x_j^2-\left(\sum_{j=1}^{N}x_j\right)^2\neq 0$$

Pode-se mostrar usando a **desigualdade de Cauchy–Schwarz** que isto acontece quando existem pelo menos duas abscissas diferentes envolvidas no ajuste.

Usando a fórmula da inversa de uma matriz dois-por-dois, chegamos às seguintes fórmulas para os coeficientes a_1 e a_2:

$$\begin{aligned} a_1 &= \frac{\sum_{j=1}^N x_j^2 \cdot \sum_{j=1}^N y_j - \sum_{j=1}^N x_j \cdot \sum_{j=1}^N x_j y_j}{N \sum_{j=1}^N x_j^2 - \left(\sum_{j=1}^N x_j\right)^2} \\ a_2 &= \frac{N \sum_{j=1}^N x_j y_j - \sum_{j=1}^N x_j \cdot \sum_{j=1}^N y_j}{N \sum_{j=1}^N x_j^2 - \left(\sum_{j=1}^N x_j\right)^2} \end{aligned} \tag{7.2}$$

Por fim, observamos que o sistema $Ma = w$ descrito na equação (7.1) pode ser reescrito na forma $V^T V a = V^T y$, onde $V := [1\ x]$ é a matriz dos coeficientes do seguinte sistema linear sobre determinado:

$$\begin{aligned} a_1 + a_2 x_1 &= y_1 \\ a_1 + a_2 x_2 &= y_2 \\ &\vdots \\ a_1 + a_2 x_N &= y_N \end{aligned} \tag{7.3}$$

Se os pontos dados não são colineares, este sistema não têm solução. Mas, sempre que pelo menos duas abscissas foram diferentes, $M = V^T V$ é uma matriz invertível e (veja o Exercício 7.1.4), então

$$a = \left(V^T V\right)^{-1} V^T y, \tag{7.4}$$

nos fornece a chamada solução por mínimos quadrados do sistema (7.3). Note que esta é uma forma de se obter os coeficientes $a = (a_1, a_2)$ equivalente àquela dada em (7.2).

Exemplo 7.1.1. Retornemos ao exemplo 7.0.3. Isto é, dado o conjunto de pontos $\{(1,1{,}2), (1{,}5, 1{,}3), (2, 2{,}3)\}$, encontrar a função do tipo $f(x) = a_1 + a_2 x$ que melhor se ajusta os pontos dados no sentido de mínimos quadrados.

Solução. Usando as fórmulas em (7.2), obtemos

$$\begin{aligned} a_1 &= \frac{7{,}25 \cdot 4{,}8 - 4{,}5 \cdot 7{,}75}{3 \cdot 7{,}25 - 20{,}25} = -0{,}05, \\ a_2 &= \frac{3 \cdot 7{,}75 - 4{,}5 \cdot 4{,}8}{3 \cdot 7{,}25 - 20{,}25} = 1{,}1. \end{aligned}$$

Ou seja, verificamos que, de fato, a função $f(x) = -0{,}05 + 1{,}1x$ corresponde à reta que melhor ajusta os pontos dados no sentido de mínimos quadrados. Os pontos e a reta ajustada estão esboçados na Figura 7.1.

Deixamos ao leitor a verificação de que os coeficientes a_1 e a_2 também podem ser obtidos pela expressão (7.4).

Em `Python`, podemos computar os coeficientes a_1 e a_2 da seguinte forma:

```
>>> xi = np.array([1, 1.5, 2])
>>> yi = np.array([1.2,1.3,2.3])
>>> V = np.array([xi**1,xi**0]).transpose();V
array([[ 1. ,  1. ],
       [ 1.5,  1. ],
       [ 2. ,  1. ]])
>>> a = ((np.linalg.inv((V.transpose()).dot(V))).dot(V.transpose())).dot(yi);a
array([ 1.1 , -0.05])
```

Então, o gráfico da função ajustada e dos pontos pode ser obtido com os comandos:

```
>>> xx = np.linspace(0.5,2.5)
>>> plt.plot(xi,yi,'ro',xx,np.polyval(a,xx),'b-')
>>> plt.grid();plt.show()
```

$\Diamond$

O procedimento apresentado de ajuste de uma reta por mínimos quadrados pode ser generalizado para qualquer família de funções que seja um espaço vetorial de dimensão finita. Problemas de ajuste com tais famílias de funções é o que chamamos de problemas de ajuste linear, os quais exploramos em detalhe na próxima seção.

Exercícios

E 7.1.1. Sejam dados o conjunto de pontos $\{(0{,}23, -0{,}54),\ (-0{,}30, -0{,}54),\ (0{,}04, -0{,}57)\}$. Encontre a função $f(x) = a_1 + a_2 x$ que melhor se ajusta no sentido de mínimos quadrados aos pontos dados. Faça, então, um gráfico com os pontos e o esboço da função ajustada.

E 7.1.2. Seja dado o conjunto de pontos $\{(-0{,}35, 0{,}2),\ (0{,}15, -0{,}5),\ (0{,}23, 0{,}54),\ (0{,}35, 0{,}7)\}$. Encontre a função $f(x) = a_1 + a_2 x$ que melhor se ajusta no sentido de mínimos quadrados aos pontos dados. Faça, então, um gráfico com os pontos e o esboço da função ajustada.

E 7.1.3. Seja dado o conjunto de pontos $\{(-1{,}94, 1{,}02),\ (-1{,}44, 0{,}59),\ (0{,}93, -0{,}28),\ (1{,}39, -1{,}04)\}$. Encontre a função $f(x) = a_1 + a_2 x$ que melhor se ajusta no sentido de mínimos quadrados aos pontos dados. Então, responda cada item:

a) Encontre o valor de $f(1)$.

b) Encontre o valor de $f(0{,}93)$.

c) Encontre o valor de $|f(0{,}93) - (-0{,}28)|$.

d) Encontre o valor do resíduo $R = \sum_{j=1}^{N}(f(x_j) - y_j)^2$.

Forneça os valores calculados com 7 dígitos significativo por arredondamento.

E 7.1.4.

a) Mostre que o sistema linear $Ma = w$ descrito na equação 7.1 pode ser reescrito na forma $V^T Va = V^T y$, onde $V = [1\ x]$.

b) Mostre que V, como definido no item a), tem posto igual a 2 quando pelo menos duas abscissas do conjunto de pontos $\{(x_j, y_j)\}_{j=1}^{N}$ são diferentes. E, portanto, $M = V^T V$ é uma matriz invertível.

7.2 Ajuste linear geral

O problem geral de ajuste linear consiste em dada uma família $\mathcal{F}$ gerada pelo conjunto de m funções $\{f_1(x), f_2(x), \ldots, f_m(x)\}$ e um conjunto de n pontos $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, calcular os coeficientes a_1, a_2, ..., a_m tais que a função dada por

$$f(x) = \sum_{j=1}^{m} a_j f_j(x) = a_1 f_1(x) + a_2 f_2(x) + \ldots + a_m f_m(x)$$

minimiza o resíduo

$$R = \sum_{i=1}^{n} \left[f(x_i) - y_i\right]^2 .$$

Aqui, a minimização é feita por todas as possíveis escolhas dos coeficientes a_1, a_2, ..., a_m.

Com o objetivo de tornar a desenvolvimento mais claro, vamos escrever R como a soma dos resíduos parciais:

$$R = \sum_{i=1}^{n} R_i, \quad \text{onde} \quad R_i := \left[f(x_i) - y_i\right]^2 .$$

Do fato que $f(x_i) = \sum_{j=1}^{m} a_j f_j(x_i)$, temos que cada resíduo pode ser escrito como

$$R_i = \left[\sum_{j=1}^{m} a_j f_j(x_i) - y_i\right]^2 .$$

A fim de encontrar o ponto de mínimo, resolvemos o sistema oriundo de igualar a zero cada uma das derivadas parciais de R em relação aos m coeficientes a_j, i.e. devemos resolver:

$$\begin{aligned}
\frac{\partial R}{\partial a_1} &= 2\sum_{i=1}^{n}\frac{\partial R_i}{\partial a_1} = 2\sum_{i=1}^{n}\left[\sum_{j=1}^{m} a_j f_j(x_i) - y_i\right] f_1(x_i) = 0,\\
\frac{\partial R}{\partial a_2} &= 2\sum_{i=1}^{n}\frac{\partial R_i}{\partial a_2} = 2\sum_{i=1}^{n}\left[\sum_{j=1}^{m} a_j f_j(x_i) - y_i\right] f_2(x_i) = 0,\\
&\vdots\\
\frac{\partial R}{\partial a_m} &= 2\sum_{i=1}^{n}\frac{\partial R_i}{\partial a_m} = 2\sum_{i=1}^{n}\left[\sum_{j=1}^{m} a_j f_j(x_i) - y_i\right] f_m(x_i) = 0.
\end{aligned}$$

Dividindo cada equação por 2 e escrevendo na forma matricial, obtemos $Ma = w$, onde a matriz M é dada por:

$$M = \begin{bmatrix}
\sum_{i=1}^{n} f_1(x_i)^2 & \sum_{i=1}^{n} f_2(x_i)f_1(x_i) & \cdots & \sum_{i=1}^{n} f_m(x_i)f_1(x_i)\\
\sum_{i=1}^{n} f_1(x_i)f_2(x_i) & \sum_{i=1}^{n} f_2(x_i)^2 & \cdots & \sum_{i=1}^{n} f_m(x_i)f_2(x_i)\\
\sum_{i=1}^{n} f_1(x_i)f_3(x_i) & \sum_{i=1}^{n} f_2(x_i)f_3(x_i) & \cdots & \sum_{i=1}^{n} f_m(x_i)f_3(x_i)\\
\vdots & \vdots & \ddots & \vdots\\
\sum_{i=1}^{n} f_1(x_i)f_m(x_i) & \sum_{i=1}^{n} f_2(x_i)f_m(x_i) & \cdots & \sum_{i=1}^{n} f_m(x_i)^2
\end{bmatrix}.$$

E os vetores a e w, por:

$$a = \begin{bmatrix} a_1\\ a_2\\ \vdots\\ a_m \end{bmatrix} \qquad \text{e} \qquad w = \begin{bmatrix}
\sum_{i=1}^{n} f_1(x_i)y_i\\
\sum_{i=1}^{n} f_2(x_i)y_i\\
\sum_{i=1}^{n} f_3(x_i)y_i\\
\vdots\\
\sum_{i=1}^{n} f_m(x_i)y_i
\end{bmatrix}$$

Agora, observamos que $M = V^T V$ e $w = V^T y$, onde a matriz V é dada por:

$$V = \begin{bmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_m(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_m(x_2) \\ f_1(x_3) & f_2(x_3) & \cdots & f_m(x_3) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_n) & f_2(x_n) & \cdots & f_m(x_n) \end{bmatrix}$$

e é o vetor coluna $y = (y_1, y_2, \ldots, y_N)$,

Então, o problema de ajuste se reduz a resolver o sistema linear $Ma = w$, ou $V^T V a = V^T y$. Este sistema linear tem solução única se a matriz M for inversível. O teorema a seguir mostra que isto acontece sempre a matriz V possui posto m, ou seja, o número de linhas linearmente independentes for igual ao número de colunas.[1]

Teorema 7.2.1. *A matriz $M = V^T V$ é quadrada de ordem m e é inversível sempre que o posto da matriz V é igual a número de colunas m.*

Demonstração. Para provar que M é inversível, precisamos mostrar que se v é um vetor de ordem m e $Mv = 0$, então $v = 0$. Suponha, então, que $Mv = 0$, isto é, $V^T V v = 0$. Tomando o produto interno da expressão $V^T V v = 0$ com v, temos:

$$0 = \left\langle V^T V v, v \right\rangle = \langle Vv, Vv \rangle = \|Vv\|^2$$

Portato $Mv = 0$ implica obrigatoriamente $Vv = 0$. Como o posto de V é igual ao número de colunas, v precisar ser o vetor nulo. □

Observação 7.2.1. Este problema é equivalente a resolver pelo métodos dos mínimos quadrados o seguinte sistema linear:

$$\begin{bmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_m(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_m(x_2) \\ f_1(x_3) & f_2(x_3) & \cdots & f_m(x_3) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_n) & f_2(x_n) & \cdots & f_m(x_n) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}$$

O caso de ajuste de um reta para um conjunto de pontos é um caso particular de ajuste linear.

[1] Nota-se que o posto não pode ultrapassar o número de colunas.

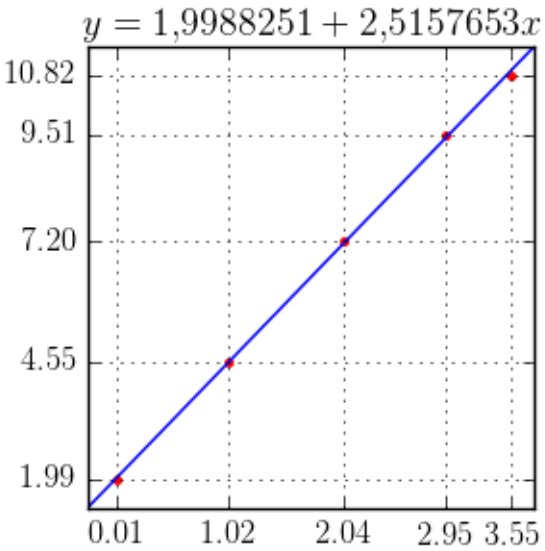


Figura 7.2: Gráfico da solução do problema apresentado no Exemplo 7.2.1.

Exemplo 7.2.1. Encontre a reta que melhor se ajusta aos pontos dados na seguinte tabela:

i	1	2	3	4	5
x_i	0,01	1,02	2,04	2,95	3,55
y_i	1,99	4,55	7,20	9,51	10,82

Solução. O problema consiste em ajustar uma função da forma $f(x) = a_1 + a_2 x$ no conjunto de pontos dados. Notamos que $f(x)$ é uma função da família gerada pelo conjunto de funções $\{f_1(x) = 1, f_2(x) = x\}$. Então, aplicando o procedimento acima, temos que o vetor dos coeficientes $a = (a_1, a_2)$ é solução por mínimos quadrados do sistema linear $Va = y$, onde:

$$V = \begin{bmatrix} f_1(x_1) & f_2(x_1) \\ f_1(x_2) & f_2(x_2) \\ f_1(x_3) & f_2(x_3) \\ f_1(x_4) & f_2(x_4) \\ f_1(x_5) & f_2(x_5) \end{bmatrix} = \begin{bmatrix} 1 & 0{,}01 \\ 1 & 1{,}02 \\ 1 & 2{,}04 \\ 1 & 2{,}95 \\ 1 & 3{,}55 \end{bmatrix}.$$

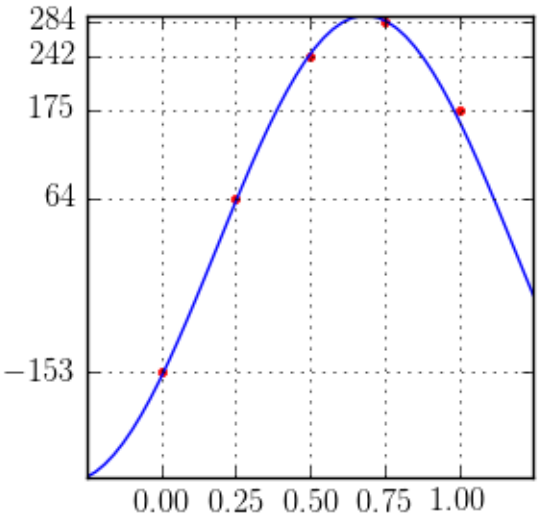


Figura 7.3: Gráfico da solução do problema apresentado no Exemplo 7.2.2.

Ou seja, é a solução do sistema $V^TVa = V^Ty$ dado por

$$\begin{bmatrix} 5 & 9{,}57 \\ 9{,}57 & 26{,}5071 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 34{,}07 \\ 85{,}8144 \end{bmatrix}$$

A solução desse sistema é $a_1 = 1{,}9988251$ e $a_2 = 2{,}5157653$. A Figura 7.2, apresenta um gráfico dos pontos e da reta ajustada.

$\Diamond$

Exemplo 7.2.2. Encontre a função $f(x) = a_1 \operatorname{sen}(\pi x) + a_2 \cos(\pi x)$ que melhor se ajusta pelo critérios dos mínimos quadrados aos seguintes pontos dados

i	1	2	3	4	5
x_i	0,00	0,25	0,50	0,75	1,00
y_i	−153	64	242	284	175

Solução. Pelo procedimento visto nesta seção, temos que os coeficientes a_1 e a_2 são dados pela solução por mínimos quadrados do seguinte sistema linear $Va = y$

$$\begin{aligned} a_1 \operatorname{sen}(\pi x_1) + a_2 \cos(\pi x_1) &= y_1 \\ a_1 \operatorname{sen}(\pi x_2) + a_2 \cos(\pi x_2) &= y_2 \\ a_1 \operatorname{sen}(\pi x_3) + a_2 \cos(\pi x_3) &= y_3 \\ a_1 \operatorname{sen}(\pi x_4) + a_2 \cos(\pi x_4) &= y_4 \\ a_1 \operatorname{sen}(\pi x_5) + a_2 \cos(\pi x_5) &= y_5 \end{aligned}$$

cuja matriz de coeficientes V é:

$$V = \begin{bmatrix} \sin(0) & \cos(0) \\ \sin(0{,}25\pi) & \cos(0{,}25\pi) \\ \sin(0{,}5\pi) & \cos(0{,}5\pi) \\ \sin(0{,}75\pi) & \cos(0{,}75\pi) \\ \sin(\pi) & \cos(\pi) \end{bmatrix}$$

Então, a solução por mínimos quadrados é

$$a = (V^T V)^{-1} V^T y = \begin{bmatrix} 244{,}03658 \\ -161{,}18783 \end{bmatrix}.$$

Ou seja, $f(x) = 244{,}03658 \operatorname{sen}(\pi x) - 161{,}18783 \cos(\pi x)$ é a função ajustada ao conjunto de pontos dados. A Figura 7.3 apresenta o gráfica de $f(x)$ e dos pontos dados.

Em `Python`, podemos computar os coeficientes da função $f(x)$ da seguinte forma:

```
>>> xi = np.array([0,0.25,0.5,0.75,1])
>>> yi = np.array([-153,64,242,284,175])
>>> V = np.array([np.sin(np.pi*xi),np.cos(np.pi*xi)]).transpose()
>>> a = ((np.linalg.inv((V.transpose()).dot(V))).dot(V.transpose())).dot(yi)
```

$\Diamond$

7.2.1 Ajuste polinomial

O **ajuste polinomial** é o caso particular do ajuste linear para **funções polinomiais**, isto é, funções do tipo

$$p(x) = a_1 + a_2 x + \cdots + a_m x^{m-1}.$$

Neste caso, a matriz V associada ao ajuste dos pontos $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)\}$ é dada por:

$$V = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{m-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{m-1} \\ 1 & x_3 & x_3^2 & \cdots & x_3^{m-1} \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{m-1} \end{bmatrix}$$

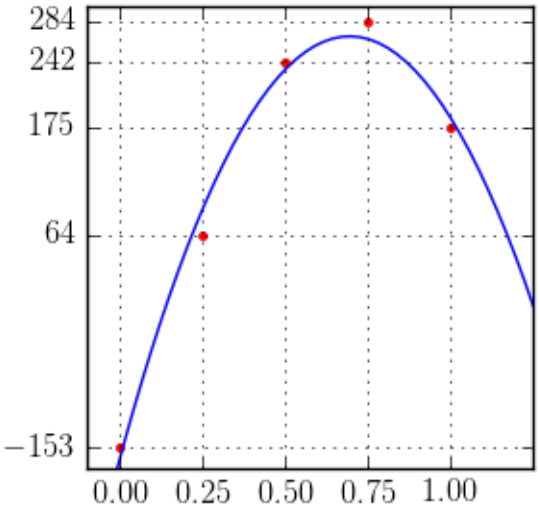


Figura 7.4: Gráfico da solução do problema apresentado no Exemplo 7.2.3.

Então, os coeficientes a_i, $i = 1, 2, \ldots, m$, são dados pela solução do sistema linear $V^T V a = v^T y$:

$$\underbrace{\begin{bmatrix} n & \sum_{j=1}^{n} x_j & \cdots & \sum_{j=1}^{n} x_j^{m-1} \\ \sum_{j=1}^{n} x_j & \sum_{j=1}^{n} x_j^2 & & \sum_{j=1}^{n} x_j^m \\ \vdots & & \ddots & \vdots \\ \sum_{j=1}^{n} x_j^{m-1} & \sum_{j=1}^{n} x_j^m & \cdots & \sum_{j=1}^{n} x_j^{2m-1} \end{bmatrix}}_{V^T V} \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{p+1} \end{bmatrix}}_{a} = \underbrace{\begin{bmatrix} \sum_{j=1}^{n} y_j \\ \sum_{j=1}^{n} x_j y_j \\ \vdots \\ \sum_{j=1}^{n} x_j^{m-1} y_j \end{bmatrix}}_{V^T y}$$

Exemplo 7.2.3. Entre o polinômio de grau 2 que melhor se ajusta aos pontos dados na seguinte tabela:

i	1	2	3	4	5
x_i	0,00	0,25	0,50	0,75	1,00
y_i	−153	64	242	284	175

Solução. Um polinômio de grau 2 pode ser escrito na seguinte forma:

$$p(x) = a_1 + a_2 x + a_3 x^2.$$

Assim, o problema se resume em encontrarmos a solução por mínimos quadrados do seguinte sistema linear:

$$\begin{aligned} a_1 + a_2x_1 + a_3x_1^2 &= y_1 \\ a_2 + a_2x_2 + a_3x_2^2 &= y_2 \\ a_3 + a_2x_3 + a_3x_3^2 &= y_3 \\ a_4 + a_2x_4 + a_3x_4^2 &= y_4 \\ a_5 + a_2x_5 + a_3x_5^2 &= y_5 \end{aligned}$$

Ou, escrita na forma matricial, $Va = y$, onde:

$$V = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ 1 & x_4 & x_4^2 \\ 1 & x_5 & x_5^2 \end{bmatrix}$$

A solução por mínimos quadrados é, então:

$$a = (V^TV)^{-1}V^Ty = \begin{bmatrix} -165{,}37143 \\ 1250{,}9714 \\ -900{,}57143 \end{bmatrix}$$

Ou seja, o polinômio de grau 2 que melhor ajusta os pontos dados no sentido de mínimos quadrados é $p(x) = -165{,}37143 + 1250{,}9714x - 900{,}57143x^2$. A Figura 7.4 mostra o gráfico do polinômio ajustado e os pontos dados.

Em `Python`, podemos computar os coeficientes do polinômio $p(x)$ da seguinte forma:

```
>>> xi = np.array([0,0.25,0.5,0.75,1])
>>> yi = np.array([-153,64,242,284,175])
>>> V = np.array([xi**2,xi**1,xi**0]).transpose()
>>> a = ((np.linalg.inv((V.transpose()).dot(V))).dot(V.transpose())).dot(yi)
```

Para fazermos o gráfico do polinômio e dos pontos, digitamos:

```
>>> xx = np.linspace(-0.25,1.25)
>>> plt.plot(xi,yi,'ro',xx,np.polyval(a,xx),'b-')
>>> plt.grid();plt.show()
```

$\Diamond$

Exercícios

E 7.2.1. Encontre o polinômio $p(x) = a_1 + a_2x + a_3x^2$ que melhor se ajusta no sentido de mínimos quadrados aos pontos:

i	1	2	3	4
x_i	$-1{,}50$	$-0{,}50$	1,25	1,50
y_i	1,15	$-0{,}37$	0,17	0,94

E 7.2.2. Encontrar a parábola $y = ax^2+bx+c$ que melhor aproxima o seguinte conjunto de dados:

i	1	2	3	4	5
x_i	0,01	1,02	2,04	2,95	3,55
y_i	1,99	4,55	7,20	9,51	10,82

E 7.2.3. Dado o seguinte conjunto de dados

x_i	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
y_i	31	35	37	33	28	20	16	15	18	23	31

a) Encontre a função do tipo $f(x) = a + b\sin(2\pi x) + c\cos(2\pi x)$ que melhor aproxima os valores dados.

b) Encontre a função do tipo $f(x) = a + bx + cx^2 + dx^3$ que melhor aproxima os valores dados.

7.3 Aproximando problemas não lineares por problemas lineares

Eventualmente, problemas de ajuste de curvas podem recair num sistema não linear. Por exemplo, para ajustar função $y = Ae^{bx}$ ao conjunto de pontos (x_1,y_1), (x_2,y_2) e (x_3,y_3), temos que minimizar o resíduo[2]

$$R = (Ae^{x_1b} - y_1)^2 + (Ae^{x_2b} - y_2)^2 + (Ae^{x_3b} - y_3)^2$$

[2] A soma do quadrado dos resíduos.

ou seja, resolver o sistema

$$\begin{aligned}\frac{\partial R}{\partial A} &= 2(Ae^{x_1 b} - y_1)e^{x_1 b} + 2(Ae^{x_2 b} - y_2)e^{x_2 b} + 2(Ae^{x_3 b} - y_3)e^{x_3 b} = 0\\ \frac{\partial R}{\partial b} &= 2Ax_1(Ae^{x_1 b} - y_1)e^{x_1 b} + 2Ax_2(Ae^{x_2 b} - y_2)e^{x_2 b}\\ &+ 2Ax_3(Ae^{x_3 b} - y_3)e^{x_3 b} = 0\end{aligned}$$

que é não linear em A e b. Esse sistema pode ser resolvido pelo método de Newton-Raphson, o que pode se tornar custoso, ou mesmo inviável quando não dispomos de uma boa aproximação da solução para inicializar o método.

Felizmente, algumas famílias de curvas admitem uma transformação que nos leva a um problema linear. No caso da curva $y = Ae^{bx}$, observe que $\ln y = \ln A + bx$. Assim, em vez de ajustar a curva original $y = Ae^{bx}$ a tabela de pontos, ajustamos a curva submetida a transformação logarítmica

$$\tilde{y} := a_1 + a_2\tilde{x} = \ln A + bx.$$

Usamos os pontos $(\tilde{x}_j,\tilde{y}_j) := (x_j, \ln y_j)$, $j = 1,2,3$ e resolvemos o sistema linear

$$V^T V \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = V^T \begin{bmatrix} \tilde{y}_1 \\ \tilde{y}_2 \\ \tilde{y}_3 \end{bmatrix},$$

onde

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{bmatrix}$$

Exemplo 7.3.1. Encontre uma curva da forma $y = Ae^x$ que melhor ajusta os pontos $(1, 2)$, $(2, 3)$ e $(3, 5)$.

Temos

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}$$

e a solução do sistema leva em $B = 0{,}217442$ e $b = 0{,}458145$. Portanto, $A = e^{0{,}217442} = 1{,}24289$.

Observação 7.3.1. Os coeficientes obtidos a partir dessa linearização são aproximados, ou seja, são diferentes daqueles obtidos quando aplicamos mínimos quadrados não linear. Observe que estamos minimizando $\sum_i [\ln y_i - \ln(f(x_i))]^2$ em vez de $\sum_i [y_i - f(x_i)]^2$. No exemplo resolvido, a solução do sistema não linear original seria $A = 1{,}19789$ e $B = 0{,}474348$

Observação 7.3.2. Mesmo quando se deseja resolver o sistema não linear, a solução do problema linearizado pode ser usada para construir condições iniciais.

A próxima tabela apresenta algumas curvas e transformações que linearizam o problema de ajuste.

curva	transformação	problema linearizado
$y = ae^{bx}$	$\tilde{y} = \ln y$	$\tilde{y} = \ln a + bx$
$y = ax^b$	$\tilde{y} = \ln y$	$\tilde{y} = \ln a + b \ln x$
$y = ax^b e^{cx}$	$\tilde{y} = \ln y$	$\tilde{y} = \ln a + b \ln x + cx$
$y = ae^{(b+cx)^2}$	$\tilde{y} = \ln y$	$\tilde{y} = \ln a + b^2 + bcx + c^2x^2$
$y = \frac{a}{b+x}$	$\tilde{y} = \frac{1}{y}$	$\tilde{y} = \frac{b}{a} + \frac{1}{a}x$
$y = A\cos(\omega x + \phi)$ ω conhecido	–	$y = a\cos(\omega x) - b\sin(\omega x)$ $a = A\cos(\phi)$, $b = A\sin(\phi)$

Exemplo 7.3.2. Encontre a função f da forma $y = f(x) = A\cos(2\pi x + \phi)$ que

ajusta a tabela de pontos

x_i	y_i
0,0	9,12
0,1	1,42
0,2	- 7,76
0,3	- 11,13
0,4	- 11,6
0,5	- 6,44
0,6	1,41
0,7	11,01
0,8	14,73
0,9	13,22
1,0	9,93

Solução. Usando o fato que $y = A\cos(2\pi x + \phi) = a\cos(2\pi x) - b\sin(2\pi x)$, onde $a = A\cos(\phi)$ e $b = A\sin(\phi)$, $z = [\ a \quad b\ ]^T$ é solução do problema

$$B^T Bz = B^T y,$$

onde

$$B = \begin{bmatrix} \cos(2\pi x_0) & -\sin(2\pi x_0) \\ \cos(2\pi x_1) & -\sin(2\pi x_1) \\ \vdots & \\ \cos(2\pi x_{10}) & -\sin(2\pi x_{10}) \end{bmatrix} = \begin{bmatrix} 1. & 0. \\ 0{,}8090170 & -0{,}5877853 \\ 0{,}3090170 & -0{,}9510565 \\ -0{,}3090170 & -0{,}9510565 \\ -0{,}8090170 & -0{,}5877853 \\ -1{,}0000000 & 0{,}0000000 \\ -0{,}8090170 & 0{,}5877853 \\ -0{,}3090170 & 0{,}9510565 \\ 0{,}3090170 & 0{,}9510565 \\ 0{,}8090170 & 0{,}5877853 \\ 1{,}0000000 & 0{,}0000000 \end{bmatrix}.$$

Assim, $a = 7{,}9614704$ e $b = 11{,}405721$ e obtemos o seguinte sistema:

$$\begin{cases} A\cos(\phi) = 7{,}9614704 \\ A\sin(\phi) = 11{,}405721 \end{cases}.$$

Observe que

$$A^2 = 7{,}9614704^2 + 11{,}405721^2$$

e, escolhendo $A > 0$, $A = 13{,}909546$ e

$$\sin(\phi) = \frac{11{,}405721}{13{,}909546} = 0{,}8199923$$

Assim, como $\cos\phi$ também é positivo, ϕ é um ângulo do primeiro quadrante:

$$\phi = 0{,}9613976$$

Portanto $f(x) = 13{,}909546\cos(2\pi x + 0{,}9613976)$. Observe que nesse exemplo a solução do problema linear é a mesma do problema não linear. ◊

Exemplo 7.3.3. Encontre a função f da forma $y = f(x) = \frac{a}{b+x}$ que ajusta a

tabela de pontos

x_i	y_i
0,0	101
0,2	85
0,4	75
0,6	66
0,8	60
1,0	55

usando uma das transformações tabeladas.

Solução. Usando o fato que $Y = \frac{1}{y} = \frac{b}{a} + \frac{1}{a}x$, $z = \begin{bmatrix} \frac{b}{a} & \frac{1}{a} \end{bmatrix}^T$ é solução do problema

$$A^T A z = A^T Y,$$

onde

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \\ 1 & x_5 \\ 1 & x_6 \end{bmatrix} = \begin{bmatrix} 1 & 0{,}0 \\ 1 & 0{,}2 \\ 1 & 0{,}4 \\ 1 & 0{,}6 \\ 1 & 0{,}8 \\ 1 & 1{,}0 \end{bmatrix}$$

e

$$Y = \begin{bmatrix} 1/y_1 \\ 1/y_2 \\ 1/y_3 \\ 1/y_4 \\ 1/y_5 \\ 1/y_6 \end{bmatrix} = \begin{bmatrix} 0{,}0099010 \\ 0{,}0117647 \\ 0{,}0133333 \\ 0{,}0151515 \\ 0{,}0166667 \\ 0{,}0181818 \end{bmatrix}$$

Assim, $\frac{1}{a} = 0{,}0082755$ e $\frac{b}{a} = 0{,}0100288$ e, então, $a = 120{,}83924$ e $b = 1{,}2118696$, ou seja, $f(x) = \frac{120{,}83924}{1{,}2118696 + x}$. ♢

Capítulo 8

Derivação Numérica

Nesta seção, discutiremos sobre estratégias numéricas de aproximação de derivadas de funções reais. Com as técnicas que abordaremos é possível o cálculo aproximado da derivada de uma função a partir de um conjunto de pontos discretos $\{x_i, y_i\}_{i=1}^n$. Começamos discutindo sobre as chamadas **aproximações por diferenças finitas** e, então, discutimos sobre aproximações de derivadas via ajuste ou interpolação.

8.1 Diferenças finitas

A técnica de **diferenças finitas** consiste em aproximar a derivada de uma função via fórmulas discretas que requerem apenas um conjunto finito de pares ordenados $\{(x_i, y_i = f(x_i))\}_{i=1}^n$. As chamadas fórmulas de diferenças finitas podem ser obtidas de várias formas. Começamos discutindo a mais básica delas, a chamada fórmula de diferenças progressiva de ordem 1.

Seja dada uma função diferenciável $y = f(x)$. A derivada $f'(x_0)$ da função $f(x)$ no ponto x_0 é, por definição,

$$f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

Deste limite, tomando $h \neq 0$ pequeno (não muito pequeno para evitar o cancelamento catastrófico), é esperado que possamos obter uma aproximação razoável para $f'(x_0)$ calculando:

$$D_{+,h} f(x_0) := \frac{f(x_0 + h) - f(x_0)}{h} \approx f'(x_0). \tag{8.1}$$

Aqui, $D_{+,h} f(x_0)$ é a chamada fórmula de diferenças progressiva de ordem 1 (ou de dois pontos).

Exemplo 8.1.1. Use a fórmula de diferenças finitas progressiva de ordem 1, calcule aproximações da derivada de $f(x) = \cos(x)$ no ponto $x = 1$ usando $h = 10^{-1}$, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-12} e 10^{-14}. Então, compute o erro $|D_{+,h}f(1) - f'(1)|$ obtido com cada valor de h.

Solução. Usando a fórmula de diferenças dada na equação (8.1), devemos calcular:

$$D_{+,h}f(1) = \frac{\cos(1+h) - \cos(1)}{h}$$

para cada valor de h solicitado. Fazendo isso, obtemos:

h	$Df(1)$	$\lvert f'(1) - D_{+,h}F(1)\rvert$
10^{-1}	−8,67062E−01	2,55909E−02
10^{-2}	−8,44158E−01	2,68746E−03
10^{-3}	−8,41741E−01	2,70011E−04
10^{-4}	−8,41498E−01	2,70137E−05
10^{-12}	−8,41549E−01	7,80679E−05
10^{-14}	−8,43769E−01	2,29851E−03

Em `Python`, podemos calcular a aproximação da derivada $f'(1)$ com $h = 0{,}1$ usando as seguintes linhas de código:

```
>>> def f(x):
...     return np.cos(x)
...
>>> x0=1
>>> h=0.1
>>> df = (f(x0+h)-f(x0))/h
```

E, similarmente, para outros valores de x_0 e h. ◊

Exploremos o Exemplo 8.1.1 um pouco mais. Observamos que, para valores moderados de h, o erro $|f'(1) - D_{+,h}f(1)|$ diminui linearmente com h (veja Figura 8.1). Isto é consequência da ordem de truncamento da fórmula de diferenças finitas aplicada (que é de ordem 1). Porém, para valores muito pequenos de $h < 10^{-8}$, o erro passa a aumentar quando diminuímos h. Isto é devido ao efeito de cancelamento catastrófico.

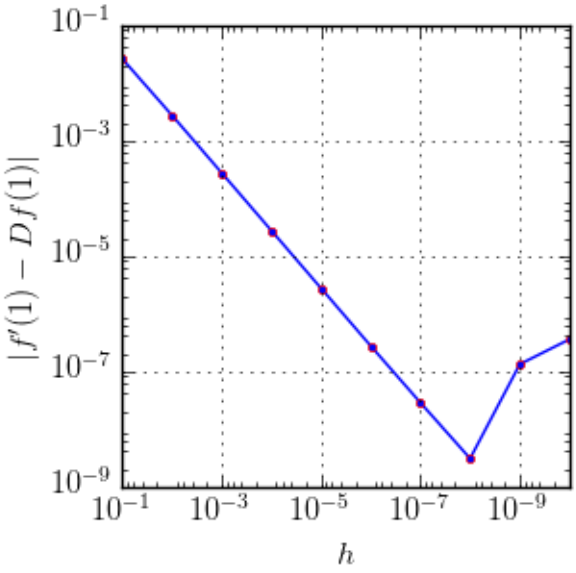


Figura 8.1: Erro absoluto das derivadas numéricas no Exemplo 8.1.1.

8.1.1 Obtenção de fórmulas de diferenças via série de Taylor

Podemos construir fórmulas de diferenças finitas para uma função $f(x)$ (suave[1]) no ponto $x = x_0$ a partir de sua expansão em série de Taylor. Em alguns casos, este procedimento acaba por nos fornecer, também, a ordem de truncamento da fórmula.

Fórmula de diferenças finitas progressiva de ordem 1

A fórmula de diferenças finitas progressiva pode ser obtida fazendo a seguinte expansão em série de Taylor:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + h^2 \frac{f''(\xi)}{2}, \quad h > 0, \xi \in (x_0, x_0 + h).$$

Então, isolando $f'(x_0)$, obtemos:

$$f'(x_0) = \underbrace{\frac{f(x_0 + h) - f(x_0)}{h}}_{D_{+,h}} - \underbrace{h\frac{f''(\xi)}{2}}_{O(h)}, \tag{8.2}$$

[1]Uma função suave é uma função infinitamente continuamente diferenciável, i.e. $f \in C^\infty(\mathbb{R})$. Uma análise mais cuidadosa, rapidamente revela que hipóteses mais fracas podem ser assumidas.

o que corrobora que o erro de truncamento da fórmula de diferença finitas progressiva[2]:

$$D_{+,h}f(x_0) := \frac{f(x_0+h)-f(x_0)}{h}$$

é de ordem h.

Fórmula de diferenças finitas regressiva de ordem 1

A fórmula de diferenças finitas regressiva também pode ser obtida fazendo, agora, a seguinte expansão em série de Taylor:

$$f(x_0-h) = f(x_0) - hf'(x_0) + h^2\frac{f''(\xi)}{2}, \quad h>0, \xi \in (x_0, x_0+h).$$

Então, isolando $f'(x_0)$, obtemos:

$$f'(x_0) = \underbrace{\frac{f(x_0)-f(x_0-h)}{h}}_{D_{-,h}} + \underbrace{h\frac{f''(\xi)}{2}}_{O(h)}.$$

Desta equação, temos que a fórmula:

$$D_{-,h}f(x_0) := \frac{f(x_0)-f(x_0-h)}{h},$$

a qual é chamada de fórmula de diferenças finitas regressiva[3] tem erro de truncamento da ordem h.

Fórmula de diferenças finitas central de ordem 2

A fórmula de diferenças finitas central[4] pode-se obter de duas expansões em série de Taylor: uma progressiva e outra regressiva. Seguem as expansões:

$$f(x_0+h) = f(x_0) + hf'(x_0) + h^2f''(x_0) + h^3\frac{f'''(\xi_+)}{3!},$$
$$f(x_0-h) = f(x_0) - hf'(x_0) + h^2f''(x_0) + h^3\frac{f'''(\xi_-)}{3!}$$

Fazendo a primeira equação menos a segunda, obtemos:

$$f(x_0+h) - f(x_0-h) = 2hf'(x_0) + h^3\left(\frac{f'''(\xi_+)-f'''(\xi_-)}{3!}\right).$$

[2]Também chamada de fórmula de diferenças finitas progressiva de dois pontos.
[3]Também chamada de fórmula de diferenças finitas regressiva de dois pontos.
[4]Também chamada de fórmula de diferenças finitas central de dois pontos.

Então, isolando $f'(x_0)$ obtemos:

$$f'(x_0) = \underbrace{\frac{f(x_0+h)-f(x_0-h)}{2h}}_{D_{0,h}} - \underbrace{h^2\left(\frac{f'''(\xi_+)-f'''(\xi_-)}{3!}\right)}_{O(h^2)}.$$

Desta equação, temos que a fórmula:

$$D_{0,h}f(x_0) := \frac{f(x_0+h)-f(x_0-h)}{2h},$$

a qual é chamada de fórmula de diferenças finitas central[5] e tem erro de truncamento da ordem 2.

Exemplo 8.1.2. Calcule a derivada numérica da função $f(x) = e^{\frac{1}{2}x}$ no ponto $x = 2$ usando diferenças finitas progressivas, diferenças regressivas e diferenças centrais com $h = 10^{-1}$, $h = 10^{-2}$ e $h = 10^{-4}$. Também, compute o valor do erro absoluto da aproximação obtida em cada caso.

Solução. Usando diferenças finitas progressiva, devemos computar:

$$D_{+,h} = \frac{f(x+h)-f(x)}{h} = \frac{e^{\frac{1}{2}(x+h)}-e^{\frac{1}{2}x}}{h}.$$

Com a fórmula de diferenças finitas regressiva, computamos:

$$D_{-,h} = \frac{f(x)-f(x-h)}{h} = \frac{e^{\frac{1}{2}x}-e^{\frac{1}{2}(x-h)}}{h}.$$

Então, usando diferenças finitas central temos:

$$D_{0,h} = \frac{f(x+h)-f(x-h)}{2h} = \frac{e^{\frac{1}{2}(h+h)}-e^{\frac{1}{2}(x-h)}}{2h}.$$

As aproximações e os erros absolutos computados em cada caso estão apresentados na seguinte tabela:

h	$D_{+,h}f(2)$	Erro	$D_{-,h}$	Erro	$D_{0,h}$	Erro
10^{-1}	1,39369	3,5E−02	1,32572	3,3E−02	1,35971	5,7E−04
10^{-2}	1,36254	3,4E−03	1,35575	3,4E−03	1,35915	5,7E−06
10^{-4}	1,35917	3,4E−05	1,35911	3,4E−05	1,35914	5,7E−10

◊

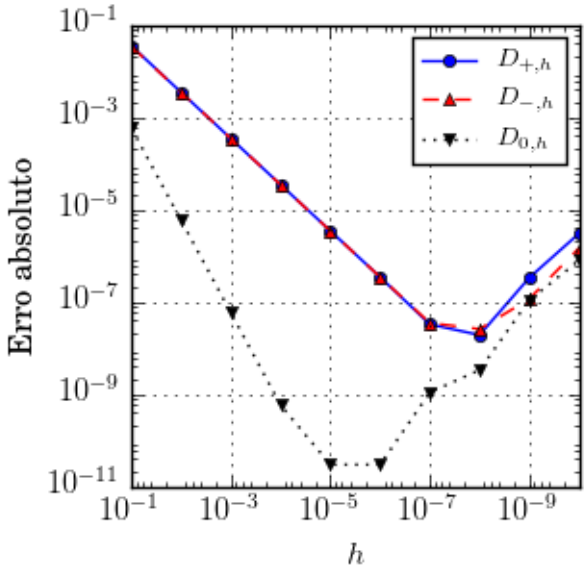


Figura 8.2: Erro absoluto das derivadas numéricas no Exemplo 8.1.2.

Observação 8.1.1. O experimento numérico realizado no Exemplo 8.1.2, nos mostra que a erro absoluto na derivação numérica não é da ordem do erro de truncamento. Entretanto, este erro tende a variar com h na mesma ordem do erro de truncamento. A Figura 8.1.2 apresenta o erro absoluto das derivadas numéricas computadas para o Exemplo 8.1.2. Note que, devido ao efeito de cancelamento catastrófico, o erro absoluto deixa de variar na ordem do erro de truncamento para valores muito pequenos de h.

Exemplo 8.1.3. Estime o erro absoluto no cálculo da derivada de $f(x) = e^{-x}$ para $x > 0$ pela fórmula de diferença progressiva.

Solução. Da equação 8.2, temos:

$$f'(x) = D_{+,h}f(x) - h\frac{f''(\xi)}{2}, \quad \xi > 0,$$

ou seja:

$$|f'(x) - D_{+,h}f(x)| = \left|\frac{f''(\xi)}{2}\right| h, \quad \xi > 0.$$

Agora, como $|f''(x)| = |e^{-x}| < 1$ para $x > 0$, concluímos que:

$$|f'(x) - D_{+,h}f(x)| \leq \frac{1}{2}h, \quad x > 0.$$

$\Diamond$

[5]Também chamada de fórmula de diferenças finitas central de três pontos.

8.1.2 Erros de arredondamento

Para entender como os erros de arredondamento se propagam ao calcular as derivadas numéricas vamos analisar a fórmula de diferenças finitas progressiva

$$D_{+,h}f(x) = \frac{f(x+h) - f(x)}{h}.$$

Nesse contexto temos o valor exato $f'(x)$ para a derivada, a sua aproximação numérica $D_{+,h}f(x)$ e a representação em número de máquina do operador $D_{+,h}f(x)$ que denotaremos por $\overline{D_{+,h}f(x)}$. Denotando por $\varepsilon(x,h)$ o erro de arredondamento ao calcularmos a derivada, vamos assumimos que

$$\overline{D_{+,h}f(x)} = D_{+,h}f(x)(1+\varepsilon(x,h)) = \frac{\overline{f(x+h)} - \overline{f(x)}}{h}(1+\varepsilon(x,h)). \tag{8.3}$$

Também, consideremos

$$|\overline{f(x+h)} - f(x+h)| = \delta(x,h) \leq \delta$$

e

$$|\overline{f(x)} - f(x)| = \delta(x,0) \leq \delta,$$

onde $\overline{f(x+h)}$ e $\overline{f(x)}$ são as representações em ponto flutuante dos números $f(x+h)$ e $f(x)$, respectivamente.

Então, da equação (8.3), a diferença do valor da derivada e sua aproximação representada em ponto flutuante pode ser estimada por:

$$\left|f'(x) - \overline{D_{+,h}f(x)}\right| = \left|f'(x) - \frac{\overline{f(x+h)} - \overline{f(x)}}{h}(1+\varepsilon(x,h))\right|.$$

Podemos reescrever o lado direito desta equação, da seguinte forma

$$\begin{aligned}
\left|f'(x) - \overline{D_{+,h}f(x)}\right| &= \left|f'(x) - \left(\frac{\overline{f(x+h)} - \overline{f(x)}}{h} + \frac{f(x+h) - f(x+h)}{h}\right.\right. \\
&+ \left.\left.\frac{f(x) - f(x)}{h}\right)(1+\varepsilon)\right| \\
&= \left|f'(x) + \left(-\frac{f(x+h) - f(x)}{h} - \frac{\overline{f(x+h)} - f(x+h)}{h}\right.\right. \\
&+ \left.\left.\frac{\overline{f(x)} - f(x)}{h}\right)(1+\varepsilon)\right|.
\end{aligned}$$

Então, separando os termos e estimando, obtemos:

$$\begin{aligned}\left|f'(x) - \overline{D_{+,h}f(x)}\right| &\leq \left|f'(x) - \frac{f(x+h)-f(x)}{h}\right| + \left(\left|\frac{\overline{f(x+h)} - f(x+h)}{h}\right| \right.\\ &+ \left.\left|\frac{\overline{f(x)} - f(x)}{h}\right|\right)|1+\varepsilon| + \left|\frac{f(x+h)-f(x)}{h}\right|\varepsilon\\ &\leq Mh + \left(\left|\frac{\delta}{h}\right| + \left|\frac{\delta}{h}\right|\right)|1+\varepsilon| + |f'(x)|\varepsilon\\ &\leq Mh + \left(\frac{2\delta}{h}\right)|1+\varepsilon| + |f'(x)|\varepsilon\end{aligned}$$

onde

$$M = \frac{1}{2}\max_{x\leq y\leq x+h}|f''(y)|$$

está relacionado com o erro de truncamento.

Por fim, obtemos a seguinte estimativa para o erro absoluto na computação da derivada numérica:

$$\left|f'(x) - \overline{D_{+,h}f(x)}\right| \leq Mh + \left(\frac{2\delta}{h}\right)|1+\varepsilon| + |f'(x)|\varepsilon. \tag{8.4}$$

Esta estimativa mostra que se o valor de h for muito pequeno o erro ao calcular a aproximação numérica cresce. Isso nos motiva a procurar o valor ótimo de h que minimiza o erro.

Exemplo 8.1.4. No Exemplo 8.1.2, computamos a derivada numérica da função $f(x) = e^{\frac{1}{2}x}$ no ponto $x = 2$ usando as fórmulas de diferenças finitas progressiva, regressiva e central. A Figura 8.2, mostra que, para valores h muito pequenos, os erros de arredondamento passam a dominar os cálculos e, por consequência, o erro da derivada numérica passa a aumentar. Pela figura, podemos inferir que a escolha ótima de h para as fórmulas progressiva e regressiva é $h \approx 10^{-7}$. Agora, para a fórmula central, $h \approx 10^{-5}$ parece ser a melhor escolha.

Observação 8.1.2. Note que a estimativa (8.4), mostra que o erro na computação da derivada numérica depende da função que está sendo derivada. Assim, o h ótimo depende não somente da fórmula de diferenças finitas, mas também da função a ser derivada.

Exercícios Resolvidos

ER 8.1.1. Aproxime a derivada de $f(x) = \operatorname{sen}(2x) - x^2$ no ponto $x = 2$ usando a fórmula de diferenças finitas progressiva de ordem 1 com: a) $h = 0{,}1$ e b) $h = 0{,}01$. Compute, também, o erro absoluto de cada aproximação computada.

Solução. A fórmula de diferenças finitas de ordem 1 para uma função $y = f(x)$ em um ponto $x = x_0$ é dada por:

$$D_{+,h}f(x_0) = \frac{f(x_0+h) - f(x_0)}{h}.$$

Substituindo $f(x) = \operatorname{sen}(2x) - x^2$ e $x_0 = 2$, obtemos:

$$\begin{aligned} D_{+,h}f(x_0) &= \frac{(\operatorname{sen}(2(x_0+h)) - (x_0+h)^2) - (\operatorname{sen}(2x_0) - x_0^2)}{h} \\ &= \frac{\operatorname{sen}(2(x_0+h)) - x_0^2 + 2x_0h + h^2 - \operatorname{sen}(2x_0) + x_0^2)}{h} \\ &= \frac{\operatorname{sen}(4+2h) + 4h + h^2 - \operatorname{sen}(4))}{h}. \end{aligned}$$

Então, tomando $h = 0{,}1$, podemos computar a derivada numérica e o erro associado:

$$D_{+,0,1}f(2) = -5{,}247733, \quad |f'(2) - D_{+,0,1}f(2)| = 5{,}96 \times 10^{-2},$$

onde $f'(x) = 2\operatorname{sen}(2x) - 2x$ é a derivada analítica. Tomando $h = 0{,}01$ temos:

$$D_{+,0,1}f(2) = -5{,}302065, \quad |f'(2) - D_{+,0,1}f(2)| = 5{,}22 \times 10^{-3}.$$

Em `Python`, podemos fazer os cálculos com o seguinte código:

```
from __future__ import division
import numpy as np

#funcao
def f(x):
    return np.sin(2*x) - x**2

#derivada analitica
def fl(x):
    return 2*np.cos(2*x) - 2*x

#d.f. progressiva de ordem 1
def dp1(f,x,h=0.1):
    return (f(x+h)-f(x))/h

#h=0.1
dy = dp1(f,2)
print("D.F. Progressiva de ordem 1 com h = %f" % 1e-1)
print("Df = %f" % dy)
```

```
print("Erro = %1.2e" % np.abs(fl(2)-dy))

#h=0.01
dy = dp1(f,2,1e-2)
print("D.F. Progressiva de ordem 1 com h = %f" % 1e-2)
print("Df = %f" % dy)
print("Erro = %1.2e" % np.abs(fl(2)-dy))
```

$\Diamond$

Exercícios

E 8.1.1. Use os esquemas numéricos do exercício 8.1.2 para aproximar as seguintes derivadas:

a) $f'(x)$ onde $f(x) = \text{sen}\,(x)$ e $x = 2$.

b) $f'(x)$ onde $f(x) = e^{-x}$ e $x = 1$.

Use $h = 10^{-2}$ e $h = 10^{-3}$ e compare com os valores obtidos através da avaliação numérica das derivadas exatas.

E 8.1.2. Expanda a função suave $f(x)$ em um polinômio de Taylor adequado para obter as seguintes aproximações:

a) $f'(x) = \frac{f(x+h)-f(x)}{h} + O(h)$

b) $f'(x) = \frac{f(x)-f(x-h)}{h} + O(h)$

c) $f'(x) = \frac{f(x+h)-f(x-h)}{2h} + O(h^2)$

E 8.1.3. Use a expansão da função $f(x)$ em torno de $x = 0$ em polinômios de Taylor para encontrar os coeficientes a_1, a_2 e a_3 tais que

a) $f'(0) = a_1 f(0) + a_2 f(h) + a_3 f(2h) + O(h^2)$

b) $f'(0) = a_1 f(0) + a_2 f(-h) + a_3 f(-2h) + O(h^2)$

c) $f'(0) = a_1 f(-h_1) + a_2 f(0) + a_3 f(h_2) + O(h^2), \;\; |h_1|, |h_2| = O(h)$

E 8.1.4. As tensões na entrada, v_i, e saída, v_o, de um amplificador foram medidas em regime estacionário conforme tabela abaixo.

0.	0.5	1.	1.5	2.	2.5	3.	3.5	4.	4.5	5.
0.	1.05	1.83	2.69	3.83	4.56	5.49	6.56	6.11	7.06	8.29

onde a primeira linha é a tensão de entrada em volts e a segunda linha é tensão de saída em volts. Sabendo que o ganho é definido como

$$\frac{\partial v_o}{\partial v_i}.$$

Calcule o ganho quando $v_i = 1$ e $v_i = 4.5$ usando as seguintes técnicas:

a) Derivada primeira numérica de primeira ordem usando o próprio ponto e o próximo.

b) Derivada primeira numérica de primeira ordem usando o próprio ponto e o anterior.

c) Derivada primeira numérica de segunda ordem usando o ponto anterior e o próximo.

d) Derivada primeira analítica da função do tipo $v_0 = a_1 v_i + a_3 v_i^3$ que melhor se ajusta aos pontos pelo critério dos mínimos quadrados.

Caso	*a*	*b*	*c*	*d*
$v_i = 1$				
$v_i = 4.5$				

E 8.1.5. Estude o comportamento da derivada de $f(x) = e^{-x^2}$ no ponto $x = 1{,}5$ quando h fica pequeno.

8.2 Diferenças finitas de ordem mais alta

Para aproximar a derivada de uma função $f(x)$ em x_0, x_1 ou x_2 usaremos os três pontos vizinhos $(x_0,f(x_0))$, $(x_1,f(x_1))$ e $(x_2,f(x_2))$. Uma interpolação usando

polinômios de Lagrange para esses três pontos é da forma:

$$\begin{aligned} f(x) &= f(x_0)\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \\ &+ f(x_2)\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} + \frac{f'''(\xi(x))}{6}(x-x_0)(x-x_1)(x-x_2). \end{aligned}$$

A derivada de $f(x)$ é

$$\begin{aligned} f'(x) = f(x_0)&\frac{2x-x_1-x_2}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{2x-x_0-x_2}{(x_1-x_0)(x_1-x_2)} \\ &+ f(x_2)\frac{2x-x_0-x_1}{(x_2-x_0)(x_2-x_1)} \\ &+ \frac{f'''(\xi(x))}{6}\left((x-x_1)(x-x_2)+(x-x_0)(2x-x_1-x_2)\right) \\ &+ D_x\left(\frac{f'''(\xi(x))}{6}\right)(x-x_0)(x-x_1)(x-x_2). \end{aligned} \tag{8.5}$$

Trocando x por x_0, temos

$$\begin{aligned} f'(x_0) = f(x_0)&\frac{2x_0-x_1-x_2}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{2x_0-x_0-x_2}{(x_1-x_0)(x_1-x_2)} \\ &+ f(x_2)\frac{2x_0-x_0-x_1}{(x_2-x_0)(x_2-x_1)} \\ &+ \frac{f'''(\xi(x_0))}{6}\left((x_0-x_1)(x_0-x_2)+(x_0-x_0)(2x_0-x_1-x_2)\right) \\ &+ D_x\left(\frac{f'''(\xi(x_0))}{6}\right)(x_0-x_0)(x_0-x_1)(x_0-x_2). \end{aligned}$$

Considerando uma malha equiespaçada onde $x_1 = x_0 + h$ e $x_2 = x_0 + 2h$, temos:

$$\begin{aligned} f'(x_0) &= f(x_0)\frac{-3h}{(-h)(-2h)} + f(x_1)\frac{-2h}{(h)(-h)} \\ &+ f(x_2)\frac{-h}{(2h)(h)} + \frac{f'''(\xi(x_0))}{6}\left((-h)(-2h)\right) \\ &= \frac{1}{h}\left[-\frac{3}{2}f(x_0) + 2f(x_1) - \frac{1}{2}f(x_2)\right] + h^2\frac{f'''(\xi(x_0))}{3} \end{aligned}$$

Similarmente, trocando x por x_1 ou trocando x por x_2 na expressão (8.5), temos outras duas expressões

$$\begin{aligned} f'(x_1) &= \frac{1}{h}\left[-\frac{1}{2}f(x_0) + \frac{1}{2}f(x_2)\right] + h^2\frac{f'''(\xi(x_1))}{6} \\ f'(x_2) &= \frac{1}{h}\left[\frac{1}{2}f(x_0) - 2f(x_1) + \frac{3}{2}f(x_2)\right] + h^2\frac{f'''(\xi(x_2))}{3} \end{aligned}$$

Podemos reescrever as três fórmulas da seguinte forma:

$$\begin{aligned} f'(x_0) &= \frac{1}{h}\left[-\frac{3}{2}f(x_0)+2f(x_0+h)-\frac{1}{2}f(x_0+2h)\right]+h^2\frac{f'''(\xi(x_0))}{3} \\ f'(x_0+h) &= \frac{1}{h}\left[-\frac{1}{2}f(x_0)+\frac{1}{2}f(x_0+2h)\right]+h^2\frac{f'''(\xi(x_0+h))}{6} \\ f'(x_0+2h) &= \frac{1}{h}\left[\frac{1}{2}f(x_0)-2f(x_0+h)+\frac{3}{2}f(x_0+2h)\right]+h^2\frac{f'''(\xi(x_0+2h))}{3} \end{aligned}$$

ou ainda

$$f'(x_0) = \frac{1}{2h}\left[-3f(x_0)+4f(x_0+h)-f(x_0+2h)\right]+h^2\frac{f'''(\xi(x_0))}{3} \quad (8.6)$$

$$f'(x_0) = \frac{1}{2h}\left[f(x_0+h)-f(x_0-h)\right]+h^2\frac{f'''(\xi(x_0))}{6} \quad (8.7)$$

$$f'(x_0) = \frac{1}{2h}\left[f(x_0-2h)-4f(x_0-h)+3f(x_0)\right]+h^2\frac{f'''(\xi(x_0))}{3} \quad (8.8)$$

Observe que uma das fórmulas é exatamente as diferenças centrais obtida anteriormente.

Analogamente, para construir as fórmulas de cinco pontos tomamos o polinômio de Lagrange para cinco pontos e chegamos a cinco fórmulas, sendo uma delas a seguinte:

$$f'(x_0) = \frac{1}{12h}\left[f(x_0-2h)-8f(x_0-h)+8f(x_0+h)-f(x_0+2h)\right]+\frac{h^4}{30}f^{(5)}(\xi(x_0)) \quad (8.9)$$

Exemplo 8.2.1. Calcule a derivada numérica de $f(x) = e^{-x^2}$ em $x = 1{,}5$ pelas fórmulas de três e cinco pontos para $h = 0{,}1$, $h = 0{,}01$ e $h = 0{,}001$.

Solução. Em `Python`, podemos computar estas derivadas numéricas com $h = 0.1$ da seguinte forma:

```
>>> def f(x):
>>> ...     return np.exp(-x**2)
>>> x=1.5
>>> h=0.1
>>> #progressiva de ordem 1
>>> dp1 = (f(x+h)-f(x))/h
>>> #regressiva de ordem 1
>>> dr1 = (f(x)-f(x-h))/h
>>> #central de ordem 2
>>> dc2 = (f(x+h)-f(x-h))/(2*h)
```

Diferenças Finitas	$h = 0{,}1$	0,01	0,001
Progressiva $O(h)$	$-0{,}2809448$	$-0{,}3125246$	$-0{,}3158289$
Regressiva $O(h)$	$-0{,}3545920$	$-0{,}3199024$	$-0{,}3165667$
Progressiva $O(h^2)$	$-0{,}3127746$	$-0{,}3161657$	$-0{,}3161974$
Central $O(h^2)$	$-0{,}3177684$	$-0{,}3162135$	$-0{,}3161978$
Regressiva $O(h^2)$	$-0{,}3135824$	$-0{,}3161665$	$-0{,}3161974$
Central $O(h^4)$	$-0{,}3162384$	$-0{,}3161977$	$-0{,}31619767$

Tabela 8.1: Derivadas numéricas de $f(x) = e^{-x^2}$ em $x = 1{,}5$. Veja o Exemplo 8.2.1.

```
>>> #progressiva de ordem 2
>>> dp2 = (-3*f(x)+4*f(x+h)-f(x+2*h))/(2*h)
>>> #regressiva de ordem 2
>>> dr2 = (f(x-2*h)-4*f(x-h)+3*f(x))/(2*h)
>>> #central de ordem 4
>>> dc4 = (f(x-2*h)-8*f(x-h)+8*f(x+h)-f(x+2*h))/(12*h)
```

e, análogo, para $h = 0.01$ e $h = 0.001$. O valor analítico da derivada é $f'(1{,}5) \approx -0{,}3161976736856$. A Tabela 8.1 mostra os resultados computados com as derivadas numéricas.

$\Diamond$

Exercícios

Em construção ... Gostaria de colaborar na escrita deste livro? Veja como em:

http://www.ufrgs.br/numerico

8.3 Diferenças finitas para derivadas de ordem mais alta

Para aproximar a derivada segunda, considere as expansões em série de Taylor

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0) + O(h^4)$$

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) - \frac{h^3}{6}f'''(x_0) + O(h^4).$$

Somando as duas expressões, temos:

$$f(x_0+h)+f(x_0-h)=2f(x_0)+h^2f''(x_0)+O(h^4)$$

ou seja, uma aproximação de segunda ordem para a derivada segunda em x_0 é

$$f''(x_0)=\frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2}+O(h^2):=D^2_{0,h}f(x_0)+O(h^2),$$

onde

$$D^2_{0,h}f(x_0)=\frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2}.$$

Exemplo 8.3.1. Calcule a derivada segunda numérica de $f(x)=e^{-x^2}$ em $x=1{,}5$ para $h=0{,}1$, $h=0{,}01$ e $h=0{,}001$.

Solução. A tabela mostra os resultados:

h	$h=0{,}1$	$h=0{,}01$	$h=0{,}001$
$D^2_{0,h}f(1{,}5)$	0,7364712	0,7377814	0,7377944

Observe que $f''(x)=(4x^2-2)e^{-x^2}$ e $f''(1{,}5)=0{,}7377946$. ◊

Exercícios

E 8.3.1. Use a expansão da função $f(x)$ em torno de $x=0$ em polinômios de Taylor para encontrar os coeficientes a_1, a_2 e a_3 tais que

a) $f''(0)=a_1f(0)+a_2f(h)+a_3f(2h)+O(h)$

b) $f''(0)=a_1f(0)+a_2f(-h)+a_3f(-2h)+O(h)$

8.4 Derivada via ajuste ou interpolação

Dado os valores de uma função em pontos $\{(x_i,y_i)\}_{i=1}^N$, as derivadas $\left(\frac{dy}{dx}\right)_i$ podem ser obtidas através da derivada de uma curva que melhor ajusta ou interpola os pontos. Esse tipo de técnica é necessário quando os pontos são muito espaçados entre si ou quando a função oscila muito. Por exemplo, dado os pontos (0,1), (1,2), (2,5), (3,9), a parábola que melhor ajusta os pontos é

$$Q(x)=0{,}95+0{,}45x+0{,}75x^2.$$

Usando esse ajuste para calcular as derivadas, temos:

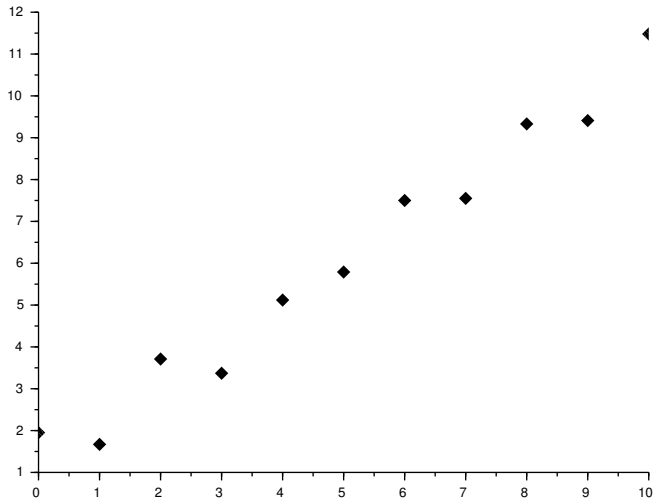
$$Q'(x) = 0{,}45 + 1{,}5x$$

e

$$y'(x_1) \approx Q'(x_1) = 0{,}45, \qquad y'(x_2) \approx Q'(x_2) = 1{,}95,$$
$$y'(x_3) \approx Q'(x_3) = 3{,}45 \qquad \text{e} \qquad y'(x_4) \approx Q'(x_4) = 4{,}95$$

Agora olhe o gráfico da seguinte tabela de pontos.

x	y
0	1,95
1	1,67
2	3,71
3	3,37
4	5,12
5	5,79
6	7,50
7	7,55
8	9,33
9	9,41
10	11,48



Observe que as derivadas calculadas por diferenças finitas oscilam entre um valor pequeno e um grande em cada intervalo e além disso, a fórmula progressiva difere da regressiva significantemente. Por exemplo, por diferenças regressivas $f'(7) \approx \frac{(7{,}55-7{,}50)}{1} = 0{,}05$ e por diferenças progressivas $f'(7) \approx \frac{(9{,}33-7{,}55)}{1} = 1{,}78$. A melhor forma de calcular a derivada aqui é fazer um ajuste de curva. A reta que melhor ajusta os dados da tabela é $y = f(x) = 1{,}2522727 + 0{,}9655455x$. Usando esse ajuste, temos $f'(7) \approx 0{,}9655455$.

Exercícios

Em construção ... Gostaria de colaborar na escrita deste livro? Veja como em:

http://www.ufrgs.br/numerico

8.5 Exercícios finais

Em construção ... Gostaria de colaborar na escrita deste livro? Veja como em:

http://www.ufrgs.br/numerico

Capítulo 9

Integração Numérica

Neste capítulo discutiremos técnicas numéricas para aproximar **integrais** definidas de funções reais.

Considere o problema de calcular (ou estimar) a integral de $f(x)$ no intervalo $[a,b]$, ou seja,

$$I = \int_a^b f(x)\, dx.$$

Uma maneira de estimar esta integral numericamente consiste em subdividir o intervalo $[a,b]$ em $n-1$ intervalos a partir de um conjunto ordenado de pontos $a = x_1 < x_2 < ... < x_n = b$. Em cada intervalo i, a integral será aproximada por ΔS_i e a integral será aproximada por

$$I \approx S = \sum_{i=1}^{n-1} \Delta S_i$$

O tamanho de cada intervalo é dado por $h_i = x_{i+1} - x_i$. No caso uniforme, todos os intervalos possuem o mesmo tamanho $h = h_i = \frac{b-a}{n}$.

Nas próximas seções apresentaremos formas diferentes de aproximar ΔS_i iniciando com o caso mais simples que é um retângulo. Cada uma das regras obtidas também é chamada de quadratura.

Exemplo 9.0.1. A figura 9.1 mostra um exemplo quando $f(x) = x^2 + 1$, $0 \leq x \leq 2$. Temos a aproximação por um retângulo com base $h_1 = 2$, depois com dois retângulos de base $h_2 = 1$ e, finalmente com quatro retângulo de bases $h_3 = 0{,}5$.

Os valores aproximados para a integral são dados na seguinte tabela:

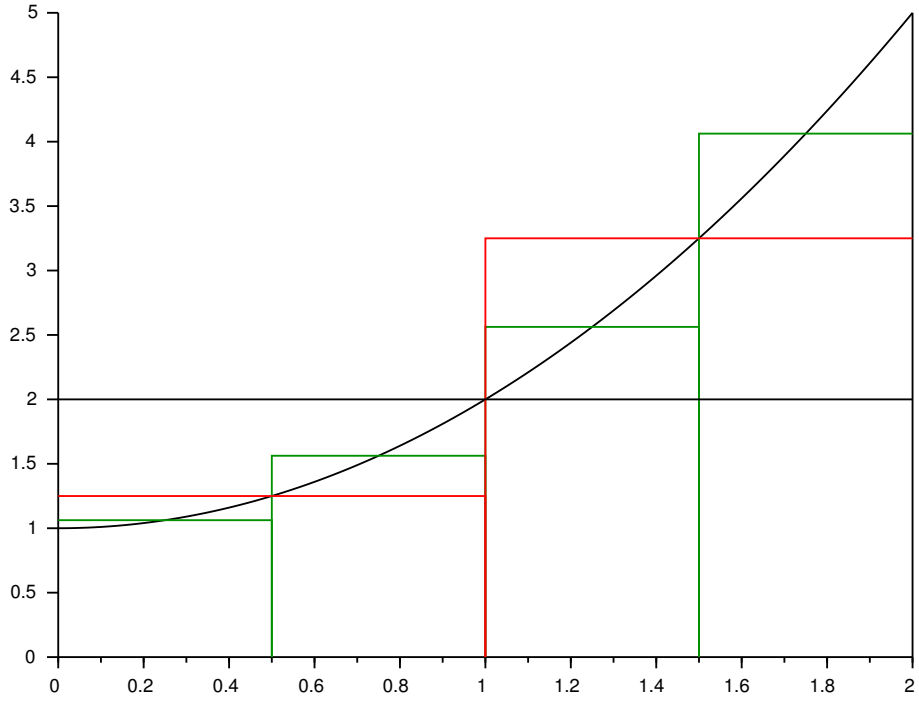


Figura 9.1: Aproximação por retângulos.

	$\int_0^2 (x^2+1)dx$
$h_1 = 2$	$h_1 f(1) = 4$
$h_2 = 1$	$h_2 f(0{,}5) + h_2 f(1{,}5) = 4{,}5$
$h_3 = 0{,}5$	$4{,}625$
$h_4 = 0{,}25$	$4{,}65625$

Observe que:

$$\int_0^2 (x^2+1)\,dx = \left[\frac{x^3}{3} + x\right]_0^2 = \frac{8}{3} + 2 = 4{,}6666667$$

Nos códigos `Python` apresentados ao longo deste capítulo, estaremos assumindo:

```
>>> from __future__ import division
>>> import numpy as np
```

9.1 Regras de Newton-Cotes

O método básico para encontrar as regras de integração consiste em aproximar a integral de f por uma combinação linear de n valores[1] de $f_i := f(x_i)$, ou seja,

$$I = \int_a^b f(x)\,dx \approx \sum_{i=1}^{n} A_i f_i.$$

Podemos obter os coeficientes A_i aproximando a função f pelo polinômio de Lagrange p_n que interpola $\{(x_i,f_i)\}_{i=1}^n$, tal que,

$$\begin{aligned} f(x) &= p_n(x) + E^n_{LAG}(x) && (9.1)\\ &= \sum_{i=1}^{n} f_i L_i(x) + E^n_{LAG}(x) && (9.2)\end{aligned}$$

onde o erro na interpolação de Lagrange é

$$E^n_{LAG}(x) = \frac{f^{(n)}(\xi(x))}{n!}\prod_{i=1}^{n}(x - x_i). \qquad (9.3)$$

Substituindo na integral obtemos

$$\int_a^b f(x)dx = \sum_{i=1}^{n}\left[f_i \int_a^b L_i(x)dx\right] + \int_a^b E^n_{LAG}(x)\,dx. \qquad (9.4)$$

[1] Utilizaremos neste capítulo a notação f_i para indicar $f(x_i)$.

A fórmula de quadratura é então

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} A_i f_i, \tag{9.5}$$

onde

$$A_i = \int_a^b L_i(x)\ dx. \tag{9.6}$$

9.1.1 Somas de Riemann

O método mais simples de aproximar

$$I = \int_a^b f(x)\ dx.$$

com apenas um intervalo, é aproximar $f(x)$ por um polinômio constante no intervalo [a,b], ou seja, $f(x) = c$. Se aproximarmos $f(x)$ pelo ponto a esquerda do intervalo temos que $f(x) \approx f(a)$ e

$$I = \int_a^b f(x)\ dx \approx \int_a^b f(a)\ dx \tag{9.7}$$

$$= f(a)\int_a^b dx = f(a)(b-a) \tag{9.8}$$

Esta é a regra de quadratura local para 1 intervalo.

Quando subdividimos [a,b] em n intervalos com tamanho $h = (b-a)/n$ nos pontos $x_i = a + (i-1)h$, em cada intervalo i aproximamos a área por

$$\Delta S_i \approx f(x_i)h$$

tal que a área total será aproximada pelas **somas de Riemann à esquerda**

$$S = \sum_{i=1}^{n-1} \Delta S_i = \sum_{i=1}^{n-1} f(x_i)h$$

Podemos obter uma fórmula similar se usarmos os pontos a direita do intervalo, ou seja, as **somas de Riemann à direita**

$$S = \sum_{i=1}^{n-1} f(x_{i+1})h$$

Uma terceira opção é utilizar o ponto médio do intervalo $[x_i,x_{i+1}]$ o qual fornece a **regra do ponto médio**

$$S = \sum_{i=1}^{n-1} f(\xi_i)h, \quad \xi_i = \frac{x_i + x_{i+1}}{2}. \tag{9.9}$$

9.1.2 Regra do Trapézio

A regra do trapézio consiste em aproximar a função $f(x)$ por um polinômio de grau 1. Se utilizarmos uma reta ligando extremos do intervalo obtemos um trapézio que fornece o nome da regra.

Desta forma, utilizando $x_1 := a$, $x_2 := b$, $h = x_2 - x_1$ e a notação $f_i = f(x_i)$ obtemos através da interpolação de Lagrange o polinômio

$$p_1(x) = f_1 L_1(x) + f_2 L_2(x) \tag{9.10}$$

Aproximando $f(x)$ por $p_1(x)$ e integrando obtemos

$$\begin{aligned}\int_a^b f(x)\,dx &\approx \int_a^b p_1(x)\,dx \\ &= \int_a^b f_1 L_1(x) + f_2 L_2(x)\,dx \\ &= f_1 \int_a^b L_1(x)\,dx + f_2 \int_a^b L_2(x)\,dx \\ &= A_1 f_1 + A_2 f_2\end{aligned}$$

onde

$$\begin{aligned}A_1 &= \int_a^b \frac{x - x_1}{x_2 - x_1} dx = \left[\frac{(x-x_1)^2}{2h}\right]_{x_1}^{x_2} \\ &= \frac{(x_2 - x_1)^2}{2h} = \frac{h^2}{2h} = \frac{1}{2}h\end{aligned}$$

Da mesma forma,

$$A_2 = \int_a^b \frac{(x - x_2)}{(x_1 - x_2)} dx = \frac{1}{2}h$$

de onde obtemos a **regra do trapézio** dada por

$$\int_a^b f(x)\,dx \approx \left(\frac{1}{2}f_1 + \frac{1}{2}f_2\right) h \tag{9.11}$$

Erro na regra do trapézio

O *erro na regra do trapézio* pode ser obtida integrando o erro da interpolação de Lagrange,

$$E_{TRAP} = \int_a^b E_{LAG}^2(x)\,dx = \int_a^b \frac{f''(\xi(x))}{2!}(x - x_1)(x - x_2)\,dx$$

Pelo teorema do valor médio, existe $a \leq \eta \leq b$ tal que

$$E_{TRAP} = \frac{f''(\eta)}{2!} \int_a^b (x - x_1)(x - x_2)\, dx,$$

portanto

$$\begin{aligned}
E_{TRAP} &= \frac{f''(\eta)}{2} \left[\frac{x^3}{3} - \frac{x^2}{2}(x_2 + x_1) + x_1 x_2 x \right]_{x_1}^{x_2} \\
&= \frac{f''(\eta)}{2} \left(\frac{x_2^3}{3} - \frac{x_2^2}{2}(x_2 + x_1) + x_1 x_2 x_2 - \frac{x_1^3}{3} + \frac{x_1^2}{2}(x_2 + x_1) - x_1 x_2 x_1 \right) \\
&= \frac{f''(\eta)}{2} \frac{2x_2^3 - 3x_2^2(x_2 + x_1) + 6x_2^2 x_1 - 2x_1^3 + 3x_1^2(x_2 + x_1) - 6x_2 x_1^2}{6} \\
&= \frac{f''(\eta)}{12} \left(x_1^3 - 3x_1^2 x_2 + 3x_2^2 x_1 - x_2^3 \right) = \frac{f''(\eta)}{12}(x_1 - x_2)^3 \\
&= -\frac{f''(\eta)}{12} h^3.
\end{aligned}$$

Assim o erro na regra do trapézio é

$$E_{TRAP} = -\frac{f''(\eta)}{12} h^3 = \mathcal{O}(h^3).$$

Exemplo 9.1.1. Use a regra do trapézio para aproximar a integral

$$\int_0^1 e^{-x^2} dx.$$

Depois divida a integral em duas

$$\int_0^{1/2} e^{-x^2} dx + \int_{1/2}^1 e^{-x^2} dx.$$

e aplique a regra do trapézio em cada uma delas. Finalmente, repita o processo dividindo em quatro integrais.

Usando o intervalo [0,1], temos $h = 1$, $x_0 = 0$ e $x_1 = 1$. A regra do trapézio resulta em

$$\int_0^1 e^{-x^2} dx \approx \frac{1}{2}(e^0 + e^{-1}) = 0{,}6839397$$

Usando dois intervalos, [0,1/2] e [1/2,1] e usando a regra do trapézio em cada um dos intervalos, temos:

$$\begin{aligned}
\int_0^1 e^{-x^2} dx &\approx \frac{0{,}5}{2} \left(e^0 + e^{-1/4} \right) + \frac{0{,}5}{2} \left(e^{-1/4} + e^{-1} \right) \\
&= 0{,}4447002 + 0{,}2866701 = 0{,}7313703.
\end{aligned}$$

Agora, usando quatro intervalos, temos

$$\begin{aligned}\int_0^1 e^{-x^2}dx &\approx \frac{0{,}25}{2}\left(e^0+e^{-1/16}\right)+\frac{0{,}25}{2}\left(e^{-1/16}+e^{-1/4}\right)\\ &+ \frac{0{,}25}{2}\left(e^{-1/4}+e^{-9/16}\right)+\frac{0{,}25}{2}\left(e^{-9/16}+e^{-1}\right)\\ &= 0{,}7429841\end{aligned}$$

9.1.3 Regra de Simpson

Na regra de Simpson aproximamos f por um polinômio de grau 2, portanto precisamos três pontos do intervalo $[a,b]$. Utilizando, por definição,

$$x_1 := a, \qquad x_2 := \frac{a+b}{2} \qquad \text{e} \qquad x_3 := b$$

com $h = x_3 - x_1$, podemos obter o polinômio de Lagrange

$$p_2(x) = f_1L_1(x) + f_2L_2(x) + f_3L_3(x)$$

Aproximando f por p_2 e integrando temos

$$\begin{aligned}\int_a^b f(x)dx &\approx \int_a^b p_2(x)\,dx &(9.12)\\ &= \int_a^b f_1L_1(x)+f_2L_2(x)+f_3L_3(x)\,dx &(9.13)\\ &= f_1A_1+f_2A_2+f_3A_3 &(9.14)\end{aligned}$$

onde

$$A_i = \int_a^b L_i(x)\,dx \tag{9.15}$$

Calculando essas integrais obtemos **a regra de Simpson**

$$\int_a^b f(x)dx = \left(\frac{1}{6}f(x_1)+\frac{4}{6}f(x_2)+\frac{1}{6}f(x_3)\right)h.$$

Exemplo 9.1.2. Obtenha os coeficientes A_i do método de Simpson integrando os polinômios de Lagrange $L_i(x)$.

Fazendo uma translação para a origem (subtraindo x_1 de x_2 e x_3)

$$\begin{aligned} A_1 &= \int_{x_1}^{x_3} \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)}\,dx \\ &= \int_0^h \frac{(x-h/2)(x-h)}{(0-h/2)(0-h)}\,dx = \frac{2}{h^2}\int_0^h (x-h/2)(x-h)\,dx \\ &= \frac{2}{h^2}\int_0^h x^2 - \frac{3}{2}hx + \frac{h^2}{2}\,dx = \frac{2}{h^2}(x^3/3 - \frac{3}{4}hx^2 + \frac{h^2x}{2})_0^h \\ &= \frac{2}{h^2}(h^3/3 - \frac{3}{4}h^3 + \frac{h^3}{2}) = (\frac{2}{3} - \frac{3}{2} + 1)h \\ &= \frac{1}{6}h \end{aligned}$$

Apesar de longa, é apenas a integral de um polinômio de grau 2. De forma semelhante podemos obter

$$A_2 = \frac{4}{6}h, \quad A_3 = \frac{1}{6}h$$

Erro na regra de Simpson

Se usarmos a mesma metodologia da regra dos trapézios, teremos

$$\int_a^b f(x)dx = \int_a^b p_2(x)dx + \int_a^b \frac{(x-x_1)(x-x_2)(x-x_3)}{6} f'''(\xi(x))dx$$

e obteremos o fórmula de Simpson com um erro de quarta ordem. O fato é que a regra de Simpson tem ordem cinco e, para isso, usaremos uma abordagem alternativa.

Considere o polinômio de Taylor em x_2,

$$f(x) = f(x_2) + f'(x_2)(x-x_2) + \frac{f''(x_2)}{2}(x-x_2)^2 + \frac{f'''(x_2)}{6}(x-x_2)^3 + \frac{f^{(4)}(\xi(x))}{24}(x-x_2)^4,$$

onde $x_1 \leq \xi(x) \leq x_3$ e integre no intervalo $[a,b] = [x_1,x_3]$:

$$\begin{aligned} \int_a^b f(x)dx = \Bigg[& f(x_2)(x-x_2) + f'(x_2)\frac{(x-x_2)^2}{2} + \frac{f''(x_2)}{6}(x-x_2)^3 \\ & + \frac{f'''(x_2)}{24}(x-x_2)^4 \Bigg]_{x_1}^{x_3} \\ & + \frac{1}{24}\int_{x_1}^{x_3} f^{(4)}(\xi(x))(x-x_2)^4 dx, \end{aligned}$$

Pelo teorema do valor médio, existe $x_1 \leq \eta \leq x_3$ tal que

$$\begin{aligned}\int_a^b f(x)dx &= \left[f(x_2)(x-x_2) + f'(x_2)\frac{(x-x_2)^2}{2} + \frac{f''(x_2)}{6}(x-x_2)^3\right.\\ &\left. + \frac{f'''(x_2)}{24}(x-x_2)^4\right]_{x_1}^{x_3}\\ &+ \frac{f^{(4)}(\eta)}{24}\int_{x_1}^{x_3}(x-x_2)^4dx\\ &= \left[f(x_2)(x-x_2) + f'(x_2)\frac{(x-x_2)^2}{2} + \frac{f''(x_2)}{6}(x-x_2)^3\right.\\ &\left. + \frac{f'''(x_2)}{24}(x-x_2)^4\right]_{x_1}^{x_3}\\ &+ \frac{f^{(4)}(\eta)}{120}\left[(x-x_2)^5\right]_{x_1}^{x_3}\end{aligned}$$

Usando o fato que

$$\begin{aligned}(x_3-x_2)^3 - (x_1-x_2)^3 &= 2h^3,\\ (x_3-x_2)^4 - (x_1-x_2)^4 &= 0\end{aligned}$$

e

$$(x_3-x_2)^5 - (x_1-x_2)^5 = 2h^5,$$

temos

$$\int_a^b f(x)dx = hf(x_2) + \frac{h^3}{3}f''(x_2) + \frac{h^5 f^{(4)}(\eta)}{60}.$$

Usando a fórmula de diferenças finitas centrais para a derivada segunda:

$$f''(x_2) = \frac{f(x_1) - 2f(x_2) + f(x_3)}{h^2} + \frac{h^2}{12}f^{(4)}(\eta_2),$$

$x_1 \leq \eta_2 \leq x_3$, temos

$$\begin{aligned}\int_a^b f(x)dx &= 2hf(x_2) + \frac{h^3}{3}\left(\frac{f(x_1) - 2f(x_2) + f(x_3)}{h^2} + \frac{h^2}{12}f^{(4)}(\eta_2)\right)\\ &+ \frac{h^5 f^{(4)}(\eta)}{60}\\ &= \frac{h}{3}\left(f(x_1) + 4f(x_2) + f(x_3)\right) - \frac{h^5}{12}\left(\frac{1}{3}f^{(4)}(\eta_2) - \frac{1}{5}f^{(4)}(\eta)\right).\end{aligned}$$

Pode-se mostrar que é possível escolher η_3 que substitua η e η_2 com a seguinte estimativa

$$\int_a^b f(x)dx = \frac{h}{3}\left(f(x_1) + 4f(x_2) + f(x_3)\right) - \frac{h^5}{90}f^{(4)}(\eta_3).$$

Exemplo 9.1.3. Use a regra de Simpson para aproximar a integral

$$\int_0^1 e^{-x^2} dx.$$

Depois divida a integral em duas

$$\int_0^{1/2} e^{-x^2} dx + \int_{1/2}^1 e^{-x^2} dx.$$

e aplica a regra de Simpson em cada uma delas.

Usando o intervalo [0,1], temos $h = 1/2$, $x_0 = 0$, $x_1 = 1/2$ e $x_2 = 1$. A regra de Simpson resulta em

$$\int_0^1 e^{-x^2} dx \approx \frac{0{,}5}{3}(e^0 + 4e^{-1/4} + e^{-1}) = 0{,}7471804$$

Usando dois intervalos, [0,1/2] e [1/2,1] e usando a regra do trapézio em cada um dos intervalos, temos:

$$\int_0^1 e^{-x^2} dx \approx \frac{0{,}25}{3}(e^0 + 4e^{-1/16} + e^{-1/4}) + \frac{0{,}25}{3}(e^{-1/4} + 4e^{-9/16} + e^{-1}) = 0{,}7468554$$

Exercícios

E 9.1.1. Calcule numericamente as seguintes integrais:

$$\text{a) } \int_0^1 e^{-x}\,dx \qquad \text{b) } \int_0^1 x^2\,dx$$

$$\text{c) } \int_0^1 x^3\,dx \qquad \text{d) } \int_0^1 xe^{-x^2}\,dx$$

$$\text{e) } \int_0^1 \frac{1}{x^2+1}\,dx \qquad \text{e) } \int_0^1 \frac{x}{x^2+1} dx$$

usando os métodos simples do Ponto médio, Trapézio e Simpson. Calcule, também, o valor analítico destas integrais e o erro nas aproximações dadas pelas quadraturas numérica.

E 9.1.2. Dê a interpretação geométrica dos métodos do ponto médio, trapézio e Simpson. A partir desta construção geométrica, deduza as fórmulas para aproximar

$$\int_a^b f(x)dx.$$

Verifique o método de Simpson pode ser entendido como uma média aritmética ponderada entre os métodos de trapézio e ponto médio. Encontre os pesos envolvidos. Explique o que são os métodos compostos.

E 9.1.3. Calcule numericamente o valor de $\int_2^5 e^{4-x^2} dx$ usando os métodos compostos do ponto médio, trapézio e Simpson. Obtenha os resultados utilizando, em cada quadratura, o número de pontos indicado.

n	Ponto médio	Trapézios	Simpson
3			
5			
7			
9			

9.2 Obtenção das regras de quadratura

Na seção anterior, obtivemos as regras de quadraturas pela aproximação do integrando por polinômios interpoladores de Lagrange. Aqui, veremos um outro método para obter regras de quadratura, que torna-se bastante útil para quando temos muitos pontos ou quando o intervalo entre os pontos não é uniforme.

Dados n pontos $[t_1,t_2,\ldots,t_n]$, queremos obter uma aproximação para

$$\int_a^b f(t)\, dt \approx w_1 f(t_1) + w_2 f(t_2) + \ldots + w_n f(t_n) \tag{9.16}$$

que seja exata para polinômios[2] até ordem $n-1$.

Aproxime $f(t)$ pelo polinômio $p(t) = w_1\phi_1(t) + \ldots + w_n\phi_n(t)$ de ordem $n-1$. Escolha uma base, como por exemplo $\phi_k(t) = t^{k-1}$. Como a regra de quadratura deve ser exata para qualquer polinômio até ordem $n-1$, então também deve ser exata para qualquer função da base. Substituindo $f(t)$ por $\phi_1(t) = 1$ em (9.16) obtemos

$$\int_a^b \phi_1(t)\, dt = t|_a^b = w_1\phi_1(t_1) + w_2\phi_1(t_2) + \ldots + w_n\phi_n(t_n) \tag{9.17}$$

$$b - a = w_1 + w_2 + \ldots + w_n \tag{9.18}$$

[2]Por exemplo, se $n = 2$, então a regra é exata para retas.

Da mesma forma para $\phi_k(t)$, $k = 2,\ldots,n$, obtemos

$$(t^2/2)|_a^b = \frac{b^2-a^2}{2} = w_1t_1 + w_2t_2 + \ldots + w_nt_n \tag{9.19}$$

$$(t^3/3)|_a^b = \frac{b^3-a^3}{3} = w_1t_1^2 + w_2t_2^2 + \ldots + w_nt_n^2 \tag{9.20}$$

$$\vdots = \vdots \tag{9.21}$$

$$\frac{b^n-a^n}{n} = w_1t_1^{n-1} + w_2t_2^{n-1} + \ldots + w_nt_n^{n-1} \tag{9.22}$$

que pode ser escrito na forma matricial

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ t_1 & t_2 & \ldots & t_n \\ t_1^2 & t_2^2 & \ldots & t_n^2 \\ \vdots & \vdots & & \vdots \\ t_1^{n-1} & t_2^{n-1} & \ldots & t_n^{n-1} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} b-a \\ \frac{b^2-a^2}{2} \\ \frac{b^3-a^3}{3} \\ \vdots \\ \frac{b^n-a^n}{n} \end{bmatrix} \tag{9.23}$$

Resolvendo o sistema obtemos os coeficientes w_k para a regra de integração.

Exemplo 9.2.1. Seja $n = 3$, $[a,b] = [0,h]$, onde $[t_1,t_2,t_3] = [0,h/2,h]$. Obtenha uma regra de integração para aproximar $\int_a^b f(t)\,dt$.

Solução. A regra terá a forma

$$\int_a^b f(t)\,dt \approx w_1f(t_1) + w_2f(t_2) + w_3f(t_3) \tag{9.24}$$

$$\approx w_1f_1 + w_2f_2 + w_3f_3 \tag{9.25}$$

Considere a base polinomial $[\phi_1(t),\phi_2(t),\phi_3(t)] = [1,t,t^2]$ e substitua $f(t)$ por $\phi_k(t)$ obtendo

$$\int_0^h 1\,dt = h = w_1(1) + w_2(1) + w_3(1) \tag{9.26}$$

$$\int_0^h t\,dt = h^2/2 = w_1(0) + w_2(h/2) + w_3(h) \tag{9.27}$$

$$\int_0^h t^2\,dt = h^3/3 = w_1(0)^2 + w_2(h/2)^2 + w_3(h)^2 \tag{9.28}$$

que pode ser escrito na forma matricial

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & h/2 & h \\ 0 & h^2/4 & h^2 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} h \\ h^2/2 \\ h^3/3 \end{bmatrix} \tag{9.29}$$

Note que podemos simplificar h tal que o sistema fique

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1/2 & 1 \\ 0 & 1/4 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = h \begin{bmatrix} 1 \\ 1/2 \\ 1/3 \end{bmatrix} \tag{9.30}$$

Resolvendo o sistema obtemos $[w_1,w_2,w_3] = h[\frac{1}{6},\frac{4}{6},\frac{1}{6}]$ fornecendo a regra de Simpson

$$\int_0^h f(t)\, dt \approx \frac{h}{6} f_0 + \frac{4h}{6} f_1 + \frac{h}{6} f_2 \tag{9.31}$$

$\Diamond$

»»»> 070aee735edae39b9603913dda7d0ba033e9e218

9.3 Regras compostas

Vimos que em todas as estimativas de erro que derivamos, o erro depende do tamanho do intervalo de integração. Uma estratégia para reduzir o erro consiste em particionar o intervalo de integração em diversos subintervalos menores tal que

$$\int_a^b f(x)dx = \sum_{i=1}^{n} \int_{x_i}^{x_{i+1}} f(x)\, dx$$

onde $a = x_1 < ... < x_{n+1} = b$, sendo n o número de subintervalos da partição do intervalo de integração. No caso uniforme $x_i = a + (i-1)h$, $h = (b-a)/n$.

Depois, aplica-se um método simples de integração em cada subintervalo,

$$\int_{x_i}^{x_{i+1}} f(x)\, dx \approx \Delta S_i$$

e a integral será aproximada por

$$\int_a^b f(x)\, dx \approx S = \sum_{i=1}^{n} \Delta S_i$$

9.3.1 Método composto dos trapézios

A **regra composta dos trapézios** assume a seguinte forma:

$$\begin{aligned} \int_a^b f(x)dx &= \sum_{i=1}^{n} \int_{x_i}^{x_{i+1}} f(x)\, dx \\ &\approx \sum_{i=1}^{n} \frac{x_{i+1} - x_i}{2} \left[f(x_i) + f(x_{i+1}) \right] \end{aligned}$$

Como $h = x_{i+1} - x_i$, temos:

$$\begin{aligned}\int_a^b f(x)\,dx &\approx \frac{h}{2}\sum_{k=1}^{N_i}[f(x_k)+f(x_{k+1})]\\ &= \frac{h}{2}[f(x_1)+2f(x_2)+2f(x_3)+\cdots+2f(x_{N_i})+f(x_{N_i+1})]\\ &= \frac{h}{2}[f(x_1)+f(x_{N_i+1})]+h\sum_{i=2}^{N_i}f(x_i)\end{aligned}$$

9.3.2 Método composto de Simpson

Já a regra composta de Simpson assume a seguinte forma:

$$\begin{aligned}\int_a^b f(x)\,dx &= \sum_{k=1}^{n}\int_{x_k}^{x_{k+1}} f(x)dx\\ &\approx \sum_{k=1}^{n}\frac{x_{x+1}-x_k}{6}\left[f(x_k)+4f\left(\frac{x_{k+1}+x_k}{2}\right)+f(x_{k+1})\right]\end{aligned}$$

onde, como anteriormente, $x_k = a + (k-1)h$, $h = (b-a)/n$ e $i = 1,2,\ldots,n+1$, sendo n o número de subintervalos da partição do intervalo de integração. Podemos simplificar o somatório acima, escrevendo:

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[f(x_1)+2\sum_{i=1}^{n-1}f(x_{2i+1})+4\sum_{i=1}^{n}f(x_{2i})+f(x_{2n+1})\right]+O(h^5)$$

onde, agora, $h = (b-a)/(2n)$, $x_i = a + (i-1)h$, $i = 1,2,\ldots,2n+1$.

Exemplo 9.3.1. Calcule numericamente a integral

$$\int_0^2 x^2 e^{x^2} dx$$

pelas regras compostas do ponto médio, trapézio e Simpson variando o número de intervalos
$N_i = 1$, 2, 3, 6, 12, 24, 48, 96.

Solução.

n	Ponto Médio	Trapézios	Simpson
1	5,4365637	218,3926	76,421909
2	21,668412	111,91458	51,750469
3	31,678746	80,272022	47,876505
6	41,755985	55,975384	46,495785
12	45,137529	48,865685	46,380248
24	46,057757	47,001607	46,372373
48	46,292964	46,529682	46,37187
96	46,352096	46,411323	46,371838

$\Diamond$

Exercícios

E 9.3.1. Use as rotinas construídas em aula e calcule numericamente o valor das seguintes integrais usando o método composto dos trapézios para os seguintes números de pontos:

n	h	$\int_0^1 e^{-4x^2}dx$	$\int_0^1 \frac{1}{1+x^2}dx$	$\int_0^1 x^4(1-x)^4dx$	$\int_0^1 e^{-\frac{1}{x^2+1}}dx$
17		0.4409931			
33		0.4410288			
65		0.4410377			
129		0.4410400			
257		0.4410405			
513		0.4410406			
1025		0.4410407	0.7853981	$1.5873015873016 \cdot 10^{-3}$	$4.6191723776309 \cdot 10^{-1}$

E 9.3.2. O valor exato da integral imprópria $\int_0^1 x\ln(x)dx$ é dado por

$$\int_0^1 x\ln(x)dx = \left(\frac{x^2}{2}\ln x - \frac{x^2}{4}\right)\Big|_0^1 = -1/4$$

Aproxime o valor desta integral usando a regra de Simpson para $n = 3$, $n = 5$ e $n = 7$. Como você avalia a qualidade do resultado obtido? Por que isso acontece.

E 9.3.3. O valor exato da integral imprópria $\int_0^\infty e^{-x^2}dx$ é dado por $\frac{\sqrt{\pi}}{2}$. Escreva esta integral como

$$I = \int_0^1 e^{-x^2}dx + \int_0^1 u^{-2}e^{-1/u^2}du = \int_0^1 \left(e^{-x^2} + x^{-2}e^{-1/x^2}\right)dx$$

e aproxime seu valor usando o esquema de trapézios e Simpson para $n = 5$, $n = 7$ e $n = 9$.

E 9.3.4. Estamos interessados em avaliar numericamente a seguinte integral:

$$\int_0^1 \ln(x)\sin(x)dx$$

cujo valor com 10 casas decimais corretas é $-.2398117420$.

a) Aproxime esta integral via Gauss-Legendre com $n = 2$,$n = 3$, $n = 4$, $n = 5$, $n = 6$ e $n = 7$.

b) Use a identidade

$$\begin{aligned}\int_0^1 \ln(x)\sin(x)dx &= \int_0^1 \ln(x)xdx + \int_0^1 \ln(x)\left[\sin(x) - x\right]dx \\ &= \left(\frac{x^2}{2}\ln x - \frac{x^2}{4}\right)\Big|_0^1 + \int_0^1 \ln(x)\left[\sin(x) - x\right]dx \\ &= -\frac{1}{4} + \int_0^1 \ln(x)\left[\sin(x) - x\right]dx\end{aligned}$$

e aproxime a integral $\int_0^1 \ln(x)\left[\sin(x) - x\right]dx$ numericamente via Gauss-Legendre com $n = 2$, $n = 3$, $n = 4$, $n = 5$, $n = 6$ e $n = 7$.

c) Compare os resultados e discuta levando em consideração as respostas às seguintes perguntas: 1)Qual função é mais bem-comportada na origem? 2)Na segunda formulação, qual porção da solução foi obtida analiticamente e, portanto, sem erro de truncamento?

9.4 O método de Romberg

O método de Romberg é um método simplificado para construir quadraturas de alta ordem.

Considere o método de trapézios composto aplicado à integral

$$\int_a^b f(x)dx$$

Defina $I(h)$ a aproximação desta integral pelo método dos trapézios composto com malha de largura constante igual a h. Aqui $h = \frac{b-a}{N_i}$ para algum N_i inteiro, i.e.:

$$I(h) = \frac{h}{2}\left[f(a) + 2\sum_{j=2}^{N_i} f(x_j) + f(b)\right], \quad N_i = \frac{b-a}{h}$$

Teorema 9.4.1. *Se $f(x)$ é uma função analítica no intervalo (a,b), então a função $I(h)$ admite uma representação na forma*

$$I(h) = I_0 + I_2h^2 + I_4h^4 + I_6h^6 + \dots$$

Para um demonstração, veja [4]. Em especial observamos que

$$\int_a^b f(x)dx = \lim_{h\to 0} I(h) = I_0$$

Ou seja, o valor exato da integral procurada é dado pelo coeficiente I_0.

A ideia central do método de Romberg, agora, consiste em usar a extrapolação de Richardson para construir métodos de maior ordem a partir do métodos dos trapézios para o intervalo (a,b)

Exemplo 9.4.1. Construção do método de quarta ordem.

$$\begin{aligned} I(h) &= I_0 + I_2h^2 + I_4h^4 + I_6h^6 + \dots \\ I\left(\frac{h}{2}\right) &= I_0 + I_2\frac{h^2}{4} + I_4\frac{h^4}{16} + I_6\frac{h^6}{64} + \dots \end{aligned}$$

Usamos agora uma eliminação gaussiana para obter o termo I_0:

$$\frac{4I(h/2) - I(h)}{3} = I_0 - \frac{1}{4}I_4h^4 - \frac{5}{16}I_6h^6 + \dots$$

Vamos agora aplicar a fórmula para $h = b - a$,

$$\begin{aligned} I(h) &= \frac{h}{2}\left[f(a) + f(b)\right] \\ I(h/2) &= \frac{h}{4}\left[f(a) + 2f\left(c\right) + f(b)\right], \quad c = \frac{a+b}{2} \end{aligned}$$

$$\begin{aligned} \frac{4I(h/2) - I(h)}{3} &= \frac{h}{3}\left[f(a) + 2f\left(c\right) + f(b)\right] - \frac{h}{6}\left[f(a) + f(b)\right] \\ &= \frac{h}{6}\left[f(a) + 4f\left(c\right) + f(b)\right] \end{aligned}$$

Observe que esquema coincide com o método de Simpson.

A partir de agora, usaremos a seguinte notação

$$\begin{aligned} R_{1,1} &= I(h) \\ R_{2,1} &= I(h/2) \\ R_{3,1} &= I(h/4) \\ &\vdots \\ R_{n,1} &= I(h/2^{n-1}) \end{aligned}$$

Observamos que os pontos envolvidos na quadratura $R_{k,1}$ são os mesmos pontos envolvidos na quadratura $R(k-1,1)$ acrescidos dos pontos centrais, assim, temos a seguinte fórmula de recorrência:

$$R_{k,1} = \frac{1}{2}R_{k-1,1} + \frac{h}{2^{k-1}}\sum_{i=1}^{2^{k-2}} f\left(a + (2i-1)\frac{h}{2^{k-1}}\right)$$

Definimos $R_{k,2}$ para $k \geq 2$ como o esquema de ordem quatro obtido da fórmula do exemplo 9.4.1:

$$R_{k,2} = \frac{4R_{k,1} - R_{k-1,1}}{3}$$

Os valores $R_{k,2}$ representam então os valores obtidos pelo método de Simpson composto aplicado a uma malha composta de $2^{k-1} + 1$ pontos.

Similarmente os valores de $R_{k,j}$ são os valores obtidos pela quadratura de ordem $2j$ obtida via extrapolação de Richardson. Pode-se mostrar que

$$R_{k,j} = R_{k,j-1} + \frac{R_{k,j-1} - R_{k-1,j-1}}{4^{j-1} - 1}.$$

Exemplo 9.4.2. Construa o esquema de Romberg para aproximar o valor de $\int_0^2 e^{-x^2}dx$ com erro de ordem 8.

O que nos fornece os seguintes resultados:

55,59815	0,000000	0,000000	0,000000
30,517357	22,157092	0,000000	0,000000
20,644559	17,353626	17,033395	0,000000
17,565086	16,538595	16,484259	**16,475543**

Ou seja, temos:

$$\int_0^2 e^{x^2}dx \approx 16{,}475543$$

usando uma aproximação de ordem 8.

Exemplo 9.4.3. Construa o esquema de Romberg para aproximar o valor de $\int_0^2 x^2e^{x^2}dx$ com erro de ordem 12.

O que nos fornece:

218,3926					
111,91458	76,421909				
66,791497	51,750469	50,105706			
51,892538	46,926218	46,604601	46,549028		
47,782846	46,412949	46,378731	46,375146	46,374464	
46,72661	46,374531	46,37197	46,371863	46,37185	**46,371847**

Ou seja, temos:

$$\int_0^2 x^2e^{x^2}dx \approx 46{,}371847$$

com uma aproximação de ordem 12.

Exercícios

E 9.4.1. Para cada integrando encontre o função $I(h) = a_0 + a_1h + a_2h^2 + a_3h^3 + a_4h^4$ que melhor se ajusta aos dados, onde $h = \frac{1}{n-1}$. Discuta os resultados com base no teorema envolvido na construção do método de Romberg.

E 9.4.2. Calcule os valores da quadratura de Romberg de $R_{1,1}$ até $R_{4,4}$ para $\int_0^\pi \sin(x)dx$. Não use rotinas prontas neste problema.

E 9.4.3. Sem usar rotinas prontas, use o método de integração de Romberg para obter a aproximação $R_{3,3}$ das seguintes integrais:

a) $\int_0^1 e^{-x^2} dx$

b) $\int_0^2 \sqrt{2 - \cos(x)} dx$

c) $\int_0^2 \frac{1}{\sqrt{2-\cos(x)}} dx$

E 9.4.4. Encontre uma expressão para $R_{2,2}$ em termos de $f(x)$ e verifique o método de Romberg $R_{2,2}$ é equivalente ao método de Simpson.

E 9.4.5. Considere o problema de aproximar numericamente o valor de

$$\int_0^{100} \left(e^{\frac{1}{2}\cos(x)} - 1 \right) dx$$

pelo método de Romberg. Usando rotinas prontas, faça o que se pede.

- Calcule $R(6,k)$, $k = 1, \ldots, 6$ e observe os valores obtidos.
- Calcule $R(7,k)$, $k = 1, \ldots, 6$ e observe os valores obtidos.
- Calcule $R(8,k)$, $k = 1, \ldots, 6$ e observe os valores obtidos.
- Discuta os resultados anteriores e proponha uma estratégia mais eficiente para calcular o valor da integral.

9.5 Ordem de precisão

Todos os métodos de quadratura que vimos até o momento são da forma

$$\int_a^b f(x)dx \approx \sum_{j=1}^{N} w_j f(x_j)$$

Exemplo 9.5.1. (a) Método do trapézio

$$\begin{aligned}\int_a^b f(x)dx &\approx [f(a)+f(b)]\,\frac{b-a}{2}\\ &= \frac{b-a}{2}f(a)+\frac{b-a}{2}f(b)\\ &:= w_1f(x_1)+w_2f(x_2)=\sum_{j=1}^{2}w_jf(x_j)\end{aligned}$$

(b) Método do trapézio com dois intervalos

$$\begin{aligned}\int_a^b f(x)dx &\approx \left[f(a)+2f\left(\frac{a+b}{2}\right)+f(b)\right]\frac{b-a}{4}\\ &= \frac{b-a}{4}f(a)+\frac{b-a}{2}f\left(\frac{a+b}{2}\right)+\frac{b-a}{4}f(b)\\ &:= w_1f(x_1)+w_2f(x_2)+w_3f(x_3)=\sum_{j=1}^{3}w_jf(x_j)\end{aligned}$$

(c) Método de Simpson

$$\begin{aligned}\int_a^b f(x)dx &\approx \left[f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right]\frac{b-a}{6}\\ &= \frac{b-a}{6}f(a)+\frac{2(b-a)}{3}f\left(\frac{a+b}{2}\right)+\frac{b-a}{6}f(b)\\ &:= \sum_{j=1}^{3}w_jf(x_j)\end{aligned}$$

(d) Método de Simpson com dois intervalos

$$\begin{aligned}\int_a^b f(x)dx &\approx \left[f(a)+4f\left(\frac{3a+b}{4}\right)+2f\left(\frac{a+b}{2}\right)\right.\\ &+ \left.4f\left(\frac{a+3b}{4}\right)+f(b)\right]\frac{b-a}{12}\\ &= \frac{b-a}{12}f(a)+\frac{b-a}{3}f\left(\frac{3a+b}{4}\right)+\frac{b-a}{6}f\left(\frac{a+b}{2}\right)\\ &+ \frac{b-a}{3}f\left(\frac{a+3b}{4}\right)+\frac{b-a}{12}f(b)\\ &:= \sum_{j=1}^{5}w_jf(x_j)\end{aligned}$$

A principal técnica que temos usado para desenvolver os métodos numéricos é o **polinômio de Taylor**:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n + R_n(x)$$

Integrando termo a termo, temos:

$$\begin{aligned}\int_a^b f(x)dx &= \int_a^b a_0 dx + \int_a^b a_1 x dx + \int_a^b a_2 x^2 dx + \ldots + \\ &\quad \int_a^b a_n x^n dx + \int_a^b R_n(x)dx \\ &= a_0(b-a) + a_1\frac{b^2-a^2}{2} + a_2\frac{b^3-a^3}{3} + \ldots + \\ &\quad a_n\frac{b^{n+1}-a^{n+1}}{n+1} + \int_a^b R_n(x)dx\end{aligned}$$

Neste momento, é natural investigar o desempenho de um esquema numérico aplicado a funções do tipo $f(x) = x^n$.

Definição 9.5.1. *A* **ordem de precisão** *ou* **ordem de exatidão** *de um esquema de quadratura numérica como o maior inteiro positivo* **n** *para o qual o esquema é exato para todas as funções do tipo* x^k *com* $0 \leq k \leq n$*, ou seja, Um esquema é dito de ordem* n *se*

$$\sum_{j=1}^{n} w_j f(x_j) = \int_a^b f(x)dx, \quad f(x) = x^k,\ k = 0,1,\ldots n$$

ou, equivalentemente:

$$\sum_{j=1}^{n} w_j x_j^k = \int_a^b x^k dx = \frac{b^{k+1}-a^{k+1}}{k+1}, \quad k = 0,1,\ldots n$$

Observação 9.5.1. Se o método tem ordem 0 ou mais, então

$$\sum_{j=1}^{n} w_j = b - a$$

Exemplo 9.5.2. A ordem de precisão do esquema de trapézios é 1:

$$\int_a^b f(x)dx \approx [f(a) + f(b)]\,\frac{b-a}{2} = \sum_{j=1}^{2} w_j f(x_j)$$

onde $w_j = \frac{b-a}{2}$, $x_1 = a$ e $x_2 = b$.

$$\begin{aligned}(k=0): &\quad \textstyle\sum_{j=1}^{n} w_j = b - a \\ (k=1): &\quad \textstyle\sum_{j=1}^{n} w_j x_j = (a+b)\frac{b-a}{2} = \frac{b^2-a^2}{2} \\ (k=2): &\quad \textstyle\sum_{j=1}^{n} w_j x_j^2 = (a^2+b^2)\frac{b-a}{2} \neq \frac{b^3-a^3}{3}\end{aligned}$$

Exemplo 9.5.3. A ordem de precisão do esquema de Simpson é 3:

$$\int_a^b f(x)dx \approx \left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right]\frac{b-a}{6} = \sum_{j=1}^{3} w_j f(x_j)$$

onde $w_1 = w_3 = \frac{b-a}{6}$,$w_2 = 4\frac{b-a}{6}$, $x_1 = a$, $x_2 = \frac{a+b}{2}$ e $x_3 = b$

$$\begin{aligned}
(k=0): &\quad \textstyle\sum_{j=1}^{n} w_j = (1+4+1)\frac{b-a}{6} = b-a \\
(k=1): &\quad \textstyle\sum_{j=1}^{n} w_j x_j = (a+4\frac{a+b}{2}+b)\frac{b-a}{6} = (a+b)\frac{b-a}{2} = \frac{b^2-a^2}{2} \\
(k=2): &\quad \textstyle\sum_{j=1}^{n} w_j x_j^2 = (a^2+4\left(\frac{a+b}{2}\right)^2+b^2)\frac{b-a}{6} = \frac{b^3-a^3}{3} \\
(k=3): &\quad \textstyle\sum_{j=1}^{n} w_j x_j^3 = (a^3+4\left(\frac{a+b}{2}\right)^3+b^3)\frac{b-a}{6} = \frac{b^4-a^4}{4} \\
(k=4): &\quad \textstyle\sum_{j=1}^{n} w_j x_j^4 = (a^4+4\left(\frac{a+b}{2}\right)^4+b^4)\frac{b-a}{6} \neq \frac{b^5-a^5}{4}
\end{aligned}$$

Exemplo 9.5.4. Encontre os pesos w_j e as abscissas x_j tais que o esquema de dois pontos

$$\int_{-1}^{1} f(x)dx = w_1 f(x_1) + w_2 f(x_2)$$

é de ordem 3.

Solução. Temos um sistema de quatro equações e quatro incógnitas dado por:

$$\begin{aligned}
w_1 + w_2 &= 2 \\
x_1 w_1 + x_2 w_2 &= 0 \\
x_1^2 w_1 + x_2^2 w_2 &= \frac{2}{3} \\
x_1^3 w_1 + x_2^3 w_2 &= 0
\end{aligned}$$

Da segunda e quarta equação, temos:

$$\frac{w_1}{w_2} = -\frac{x_2}{x_1} = -\frac{x_2^3}{x_1^3}$$

Como $x_1 \neq x_2$, temos $x_1 = -x_2$ e $w_1 = w_2$. Da primeira equação, temos $w_1 = w_2 = 1$. Da terceira equação, temos $-x_1 = x_2 = \frac{\sqrt{3}}{3}$.

Esse esquema de ordem de precisão três e dois pontos chama-se quadratura de Gauss-Legendre com dois pontos:

$$\int_{-1}^{1} f(x)dx = f\left(\frac{\sqrt{3}}{3}\right) + f\left(-\frac{\sqrt{3}}{3}\right)$$

$\Diamond$

Exemplo 9.5.5. Comparação

$f(x)$	Exato	Trapézio	Simpson	Gauss-Legendre (2)
e^x	$e - e^{-1}$ $\approx 2{,}35040$	$e^{-1} + e$ $\approx 3{,}08616$	$\frac{e^{-1} + 4e^0 + e^1}{3}$ $\approx 2{,}36205$	$e^{-\frac{-\sqrt{3}}{3}} + e^{\frac{\sqrt{3}}{3}}$ $\approx 2{,}34270$
$x^2\sqrt{3+x^3}$	$\frac{16}{9} - \frac{4}{9}\sqrt{2}$ $\approx 1{,}14924$	3,41421	1,13807	1,15411
$x^2e^{x^3}$	$\frac{e-e^{-1}}{3} \approx 0{,}78347$	3,08616	1,02872	0,67905

Exercícios

E 9.5.1. Encontre os pesos w_1, w_2 e w_3 tais que o esquema de quadratura dado por

$$\int_0^1 f(x)dx \approx w_1 f(0) + w_2 f(1/2) + w_3 f(1)$$

apresente máxima ordem de exatidão. Qual a ordem obtida?

E 9.5.2. Encontre a ordem de exatidão do seguinte método de integração:

$$\int_{-1}^1 f(x)dx \approx \frac{2}{3}\left[f\left(\frac{-\sqrt{2}}{2}\right) + f(0) + f\left(\frac{\sqrt{2}}{2}\right)\right]$$

E 9.5.3. Encontre a ordem de exatidão do seguinte método de integração:

$$\int_{-1}^1 f(x)dx = -\frac{1}{210}f'(-1) + \frac{136}{105}f(-1/2) - \frac{62}{105}f(0) + \frac{136}{105}f(1/2) + \frac{1}{210}f'(1)$$

E 9.5.4. Encontre os pesos w_1, w_2 e w_3 tal que o método de integração

$$\int_0^1 f(x)dx \approx w_1 f(1/3) + w_2 f(1/2) + w_3 f(2/3)$$

tenha ordem de exatidão máxima. Qual é ordem obtida?

E 9.5.5. Quantos pontos são envolvidos no esquema de quadratura $R_{3,2}$? Qual a ordem do erro deste esquema de quadratura? Qual a ordem de exatidão desta quadratura?

9.6 Quadratura de Gauss-Legendre

Utilizando n pontos para aproximar a integral de $f(x)$ em $[-1,1]$ podemos encontrar a regra de quadratura de Gauss-Legendre

$$\int_{-1}^{1} f(t)\, dt \approx \sum_{j=1}^{n} w_j f(t_j)$$

cuja ordem de exatidão é $2n-1$.

- Note que temos n coeficientes w_j e n pontos t_j para determinar. O problema de encontrar os n pesos e n abscissas é equivalente a um sistema não linear com $2n$ equações e $2n$ incógnitas.

- Pode-se mostrar que este problema sempre tem solução e que a solução é única se $t_1 < t_2 < \ldots < t_n$

- Os nós x_j são dados pelos zeros do polinômio de Legendre, $P_n(t)$.

- Os pesos são dados por

$$w_j = \frac{2}{\left(1-t_j^2\right)\left[P_n'(t_j)\right]^2}.$$

A tabela abaixo lista os nós e os pesos da quadratura de Gauss-Legendre para $n = 1,\ldots,4$.

n	t_j	w_j
1	0	2
2	$\pm\frac{\sqrt{3}}{3}$	1
3	0 $\pm\sqrt{\frac{3}{5}}$	$\frac{8}{9}$ $\frac{5}{9}$
4	$\pm\sqrt{\left(3-2\sqrt{6/5}\right)/7}$ $\pm\sqrt{\left(3+2\sqrt{6/5}\right)/7}$	$\frac{18+\sqrt{30}}{36}$ $\frac{18-\sqrt{30}}{36}$

Mudança de intervalo

Os coeficientes da quadratura de Gauss-Legendre forma obtidos no intervalo $[-1,1]$. Para aproximar a integral de $f(x)$ no intervalo $[a,b]$ devemos fazer a mudança de variável

$$\bar{x}_i = \alpha t_i + \beta, \quad \alpha = (b-a)/2, \quad \beta = (b+a)/2$$

tal que

$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{n} w_i f(\bar{x}_i)(b-a)/2$$

Quando subdividimos o intervalo inicial $[a,b]$ em N intervalos com extremos $[x_i,x_{i+1}]$ a transformação torna-se

$$\bar{x}_i = \alpha t_i + \beta, \quad \alpha = (x_{i+1}-x_i)/2, \quad \beta = (x_{i+1}+x_i)/2$$

e

$$\int_{x_i}^{x_{i+1}} f(x)\,dx \approx \sum_{i=1}^{n} w_i f(\bar{x}_i)(x_{i+1}-x_i)/2$$

Exemplo 9.6.1. Aproximar

$$\int_{-1}^{1} \sqrt{1+x^2}dx$$

pelo método de Gauss-Legendre com 3 pontos.

Solução.

$$I_3 = \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right) \approx 2{,}2943456$$

$\Diamond$

Exemplo 9.6.2. Aproximar

$$\int_0^1 \sqrt{1+x^2}dx$$

pelo método de Gauss-Legendre com 3, 4 e 5 pontos.

Solução. Para tanto, fazemos a mudança de variáveis $u = 2x - 1$:

$$\int_0^1 \sqrt{1+x^2}dx = \frac{1}{2}\int_{-1}^1 \sqrt{1+\left(\frac{u+1}{2}\right)^2}du$$

E, então aplicamos a quadratura gaussiana nesta última integral. ◊

Exercícios

E 9.6.1. Encontre aproximações para a seguinte integral via Gauss-Legendre com 2, 3, 4, 5, 6 e 7 pontos e compare com o valor exato

$$\int_{-1}^1 x^4 e^{x^5}dx.$$

E 9.6.2. Encontre aproximações para as seguintes integrais via Gauss-Legendre com 4 e 5 pontos:

a) $\int_0^1 e^{-x^4}dx$

b) $\int_1^4 \log(x+e^x)dx$

c) $\int_0^1 e^{-x^2}dx$

E 9.6.3. Calcule numericamente o valor das seguintes integrais usando a quadratura de Gauss-Legendre para os seguintes valores de n:

n	$\int_0^1 e^{-4x^2}dx$	$\int_0^1 \frac{1}{1+x^2}dx$	$\int_0^1 x^4(1-x)^4dx$	$\int_0^1 e^{-\frac{1}{x^2+1}}dx$
2				
3				
4				
5				
8				
10				
12				
14				
16	0.4410407	0.7853982	0.0015873	0.4619172

9.7 Exercícios finais

E 9.7.1. Considere o problema de calcular numericamente a integral $I = \int_{-1}^1 f(x)dx$ quando $f(x) = \frac{\cos(x)}{\sqrt{|x|}}$.

a) O que acontece quando se aplica diretamente a quadratura gaussiana com um número impar de abscissas?

b) Calcule o valor aproximado por quadratura gaussiana com $n = 2$, $n = 4$, $n = 6$ e $n = 8$.

c) Calcule o valor aproximado da integral removendo a singularidade

$$\begin{aligned} I &= \int_{-1}^1 \frac{\cos(x)}{\sqrt{|x|}}dx = \int_{-1}^1 \frac{\cos(x)-1}{\sqrt{|x|}}dx + \int_{-1}^1 \frac{1}{\sqrt{|x|}}dx \\ &= \int_{-1}^1 \frac{\cos(x)-1}{\sqrt{|x|}}dx + 2\int_0^1 \frac{1}{\sqrt{x}}dx = \int_{-1}^1 \frac{\cos(x)-1}{\sqrt{|x|}}dx + 4 \end{aligned}$$

e aplicando quadratura gaussiana com $n = 2$, $n = 4$, $n = 6$ e $n = 8$.

d) Calcule o valor aproximado da integral removendo a singularidade, considerando a paridade da função

$$I = 4 + \int_{-1}^1 \frac{\cos(x)-1}{\sqrt{|x|}}dx = 4 + 2\int_0^1 \frac{\cos(x)-1}{\sqrt{x}}dx = 4 + \sqrt{2}\int_{-1}^1 \frac{\cos\left(\frac{1+u}{2}\right)-1}{\sqrt{1+u}}du$$

e aplicando quadratura gaussiana com $n = 2$, $n = 4$, $n = 6$ e $n = 8$.

e) Expandindo a função $\cos(x)$ em série de Taylor, truncando a série depois do n-ésimo termos não nulo e integrando analiticamente.

f) Aproximando a função $\cos(x)$ pelo polinômio de Taylor de grau 4 dado por

$$P_4(x) = 1 - \frac{x^2}{2} + \frac{x^4}{24}$$

e escrevendo

$$\begin{aligned} I &= \int_{-1}^{1} \frac{\cos(x)}{\sqrt{|x|}}dx = \int_{-1}^{1} \frac{\cos(x) - P_4(x)}{\sqrt{|x|}}dx + \int_{-1}^{1} \frac{P_4(x)}{\sqrt{|x|}}dx \\ &= 2 \underbrace{\int_0^1 \frac{\cos(x) - P_4(x)}{\sqrt{x}}dx}_{\text{Resolver numericamente}} + 2 \underbrace{\int_0^1 \left(x^{-1/2} - \frac{x^{3/2}}{2} + \frac{x^{7/2}}{24}\right)dx}_{\text{Resolver analiticamente}} \end{aligned}$$

E 9.7.2. Calcule numericamente o valor das seguintes integrais com um erro relativo inferior a 10^{-4}.

a) $\displaystyle\int_0^1 \frac{\sin(\pi x)}{x}dx$

b) $\displaystyle\int_0^1 \frac{\sin(\pi x)}{x(1-x)}dx$

c) $\displaystyle\int_0^1 \frac{\sin\left(\frac{\pi}{2}x\right)}{\sqrt{x(1-x)}}dx$

d) $\displaystyle\int_0^1 \ln(x)\cos(x)dx$

E 9.7.3. Calcule as integrais $\int_0^1 \frac{e^x}{|x|^{1/4}}dx$ e $\int_0^1 \frac{e^{-x}}{|x|^{4/5}}dx$ usando procedimentos analíticos e numéricos.

E 9.7.4. Use a técnica de integração por partes para obter a seguinte identidade envolvendo integrais impróprias:

$$I = \int_0^\infty \frac{\cos(x)}{1+x}dx = \int_0^\infty \frac{\sin(x)}{(1+x)^2}dx.$$

Aplique as técnicas estudadas para aproximar o valor de I e explique por que a integral da direita é mais bem comportada.

E 9.7.5. Resolva a equação

$$x + \int_0^x e^{-y^2} dy = 5$$

com 5 dígitos significativos.

E 9.7.6. O calor específico (molar) de um sólido pode ser aproximado pela teoria de Debye usando a seguinte expressão

$$C_V = 9Nk_B \left(\frac{T}{T_D}\right)^3 \int_0^{T_D/T} \frac{y^4 e^y}{(e^y - 1)^2} dy$$

onde N é a constante de Avogrado dado por $N = 6.022 \times 10^{23}$ e k_B é a constante de Boltzmann dada por $k_B = 1.38 \times 10^{-23}$. T_D é temperatura de Debye do sólido.

a) Calcule o calor específico do ferro em quando $T = 200K$, $T = 300K$ e $T = 400K$ supondo $T_D = 470K$.

b) Calcule a temperatura de Debye de um sólido cujo calor específico a temperatura de $300K$ é $24J/K/mol$. Dica: aproxime a integral por um esquema numérico com um número fixo de pontos.

c) Melhore sua cultura geral: A lei de Dulong-Petit para o calor específico dos sólidos precede a teoria de Debye. Verifique que a equação de Debye é consistente com Dulong-Petit, ou seja:

$$\lim_{T \to \infty} C_v = 3Nk_B.$$

Dica: use $e^y \approx 1 + y$ quando $y \approx 0$

E 9.7.7. Explique por quê quando um método simples tem estimativa de erro de truncamento local de ordem h^n, então o método composto associado tem estimativa de erro de ordem h^{n-1}.

E 9.7.8. Encontre os pesos w_1 e w_2 e as abcissas x_1 e x_2 tais que

$$\int_{-1}^{1} f(x) = w_1 f(x_1) + w_2 f(x_2)$$

quando $f(x) = x^k$, $k = 0,1,2,3$, isto é o método que apresente máxima ordem de exatidão possível com dois pontos.

Use esse método para avaliar o valor da integral das seguintes integrais e compare com os valores obtidos para Simpson e trapézio, bom como com o valor exato.

a) $\int_{-1}^{1} (2 + x - 5x^2 + x^3)\, dx$

b) $\int_{-1}^{1} e^x dx$

c) $\int_{-1}^{1} \frac{dx}{\sqrt{x^2+1}}$

E 9.7.9. Encontre os pesos w_1, w_2 e w_3 tal que o método de integração

$$\int_{-1}^{1} f(x)dx \approx w_1 f\left(-\frac{\sqrt{3}}{3}\right) + w_2 f(0) + w_3 f\left(\frac{\sqrt{3}}{3}\right)$$

tenha ordem de exatidão máxima. Qual é ordem obtida?

Capítulo 10

Problemas de valor inicial

Neste capítulo, desenvolveremos técnicas numérica para aproximar a solução de problemas de valor inical da forma

$$\begin{aligned} y'(t) &= f(y(t),t) && \text{(10.1a)} \\ y(t_0) &= y_0 \ \text{(condição inicial).} && \text{(10.1b)} \end{aligned}$$

A ingógnita de um problema de valor inicial é uma função que satisfaz a equação diferencial (10.1a) e a condição inicial (10.1b).

Exemplo 10.0.1. Considere o seguinte problema de valor inicial

$$\begin{aligned} y'(t) &= 2y(t), && \text{(10.2a)} \\ y(t_0) &= 1. && \text{(10.2b)} \end{aligned}$$

A solução desta equação é dada pela função $y(t) = e^{2t}$ pois $y'(t) = 2e^{2t} = 2y(t)$ e $y(0) = e^0 = 1$.

Muitos problemas de valor inicial da forma (10.1) não podem ser resolvidos exatamente, ou seja, sabe-se que a solução existe e é única, porém não podemos expressá-la em termos de funções elementares. Por isso é necessário calcular aproximações numéricas. Diversos métodos completamente diferentes estão disponíveis para aproximar uma função real.

Aqui nos limitaremos a estudar métodos que se fundamentam em tentar calcular $y(t)$ em um conjunto finito de valores de t. Esse conjunto de valores para t será denotado por $\{t_i\}_{i=1}^N$, isto é $\{t_1, t_2, t_3, \ldots, t_N\}$ e calculamos o valor aproximado da função solução $y(t_i)$ em cada ponto da malha usando esquemas numéricos.

10.1 Método de Euler

Retornemos ao problema de valor inicial (10.1) dado por:

$$\begin{aligned} y'(t) &= f(y(t),t) && \text{(10.3a)} \\ y(0) &= y_0 \ \text{(condição inicial)} && \text{(10.3b)} \end{aligned}$$

O Método de Euler aplicado à solução desse problema consiste em aproximar a derivada $y'(t)$ por um esquema de primeira ordem do tipo

$$y'(t) = \frac{y(t+h) - y(t)}{h} + O(h), \ \ h > 0.$$

Aqui h é o passo do método, que consideraremos uma constante. Assim temos (10.3) se transforma em:

$$\begin{aligned} \frac{y(t+h) - y(t)}{h} &= f(y(t),t) + O(h) \\ y(t+h) &= y(t) + hf(y(t),t) + O(h^2). && \text{(10.4)} \end{aligned}$$

Definimos, então, $t^{(k)} = (k-1)h$ e $y^{(k)}$ como a aproximação para $y\left(t^{(k)}\right)$ produzida pelo Método de Euler. Assim, obtemos

$$\begin{aligned} y^{(k+1)} &= y^{(k)} + hf(y^{(k)},t^{(k)}) \ \text{(aproximação da EDO)}, && \text{(10.5)} \\ y^{(1)} &= y_0 \ \text{(condição inicial)}. && \text{(10.6)} \end{aligned}$$

O problema (10.5) consiste em um esquema iterativo, isto é, $y^{(1)}$ é a condição inicial; $y^{(2)}$ pode ser obtido de $y^{(1)}$; $y^{(3)}$, de $y^{(2)}$ e assim por diante, calculamos o termo $y^{(n)}$ apartir do anterior $y^{(n-1)}$.

Exemplo 10.1.1. Retornemos ao o problema de valor inicial do exemplo (10.2):

$$\begin{aligned} y'(t) &= 2y(t) \\ y(0) &= 1 \end{aligned}$$

Cuja solução é $y(t) = e^{2t}$. O método de Euler aplicado a este problema produz o seguinte esquema:

$$\begin{aligned} y^{(k+1)} &= y^{(k)} + 2hy^{(k)} = (1+2h)y^{(k)} \\ y^{(1)} &= 1, \end{aligned}$$

cuja solução é dada por

$$y^{(k)} = (1+2h)^{k-1}.$$

Como $t = (k-1)h$, a solução aproximada pelo Método de Euler é

$$y(t) \approx \tilde{y}(t) = (1+2h)^{\frac{t}{h}}.$$

Observe que $\tilde{y}(t) \neq y(t)$, mas se h é pequeno, a aproximação é boa, pois

$$\lim_{h \to 0+} (1+2h)^{\frac{t}{h}} = e^{2t}.$$

Vamos agora, analisar o desempenho do Método de Euler usando um exemplo mais complicado, porém ainda simples suficiente para que possamos obter a solução exata:

Exemplo 10.1.2. Considere o problema de valor inicial relacionado à equação logística:

$$\begin{aligned} y'(t) &= y(t)(1-y(t)) \\ y(0) &= 1/2 \end{aligned}$$

Podemos obter a solução exata desta equação usando o método de separação de variáveis e o método das frações parciais. Para tal escrevemos:

$$\frac{dy(t)}{y(t)(1-y(t))} = dt$$

O termo $\frac{1}{y(1-y)}$ pode ser decomposto em frações parciais como $\frac{1}{y} - \frac{1}{1-y}$ e chegamos na seguinte equação diferencial:

$$\left(\frac{1}{y} + \frac{1}{1-y}\right) dy = dt.$$

Integrando termo-a-termo, temos a seguinte equação algébrica relacionando $y(t)$ e t:

$$\ln(y) - \ln(1-y) = t + C$$

Onde C é a constante de integração, que é definida pela condição inicial, isto é, $y = 1/2$ em $t = 0$. Substituindo, temos $C = 0$. O que resulta em:

$$\ln\left(\frac{y}{1-y}\right) = t$$

Equivalente a

$$\frac{y}{1-y} = e^t$$

e

$$y = (1-y)e^t$$

Tabela 10.1: Tabela comparativa enter Método de Euler e solução exata para problema 10.1.2.

t	Exato	Euler $h = 0{,}1$	Euler $h = 0{,}01$
0	1/2	0,5	0,5
1/2	$\frac{e^{1/2}}{1+e^{1/2}} \approx 0{,}6224593$	0,6231476	0,6225316
1	$\frac{e}{1+e} \approx 0{,}7310586$	0,7334030	0,7312946
2	$\frac{e^2}{1+e^2} \approx 0{,}8807971$	0,8854273	0,8812533
3	$\frac{e^3}{1+e^3} \approx 0{,}9525741$	0,9564754	0,9529609

Colocando o termo y em evidência, encontramos:

$$(1 + e^t)y = e^t \tag{10.7}$$

E, finalmente, encontramos a solução exata dada por $y(t) = \frac{e^t}{1+e^t}$.

Vejamos, agora, o esquema iterativo produzido pelo método de Euler:

$$\begin{aligned} y^{(k+1)} &= y^{(k)} + hy^{(k)}(1 - y^{(k)}), \\ y^{(1)} &= 1/2. \end{aligned}$$

Para fins de comparação, calculamos a solução de 10.1.2 e de (**??**) para alguns valores de t e de passo h e resumimos na Tabela 10.1.

No exemplo a seguir, apresentamos um problema envolvendo uma equação não-autônoma, isto é, quando a função $f(y,t)$ depende explicitamente do tempo.

Exemplo 10.1.3. Resolva o problema de valor inicial

$$\begin{aligned} y' &= -y + t \\ y(0) &= 1, \end{aligned}$$

cuja solução exata é $y(t) = 2e^{-t} + t - 1$.

O esquema recursivo de Euler fica:

$$\begin{aligned} y^{(k+1)} &= y(k) - hy(k) + ht^{(k)} \\ y(0) &= 1 \end{aligned}$$

Comparação

t	Exato	Euler $h = 0{,}1$	Euler $h = 0{,}01$
0	1	1	1
1	$2e^{-1} \approx 0{,}7357589$	0,6973569	0,7320647
2	$2e^{-2} + 1 \approx 1{,}2706706$	1,2431533	1,2679593
3	$2e^{-3} + 2 \approx 2{,}0995741$	2,0847823	2,0980818

No exemplo 10.1.4, mostramos como o Método de Euler pode ser facilmente estendido para problemas envolvendo sistemas de equações diferenciais..

Exemplo 10.1.4. Escreva o processo iterativo de Euler para resolver numericamente o seguinte sistema de equações diferenciais

$$\begin{aligned} x' &= -y \\ y' &= x \\ x(0) &= 1 \\ y(0) &= 0, \end{aligned}$$

cuja solução exata é $x(t) = \cos(t)$ e $y(t) = \sin(t)$.

Para aplicar o Método de Euler a um sistema, devemos encarar as diversas incógnitas do sistema como formando um vetor, neste caso, escrevemos:

$$z(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}.$$

O sistema é igualmente escrito na forma vetorial:

$$\begin{bmatrix} x^{(k+1)} \\ y^{(k+1)} \end{bmatrix} = \begin{bmatrix} x^{(k)} \\ y^{(k)} \end{bmatrix} + h \begin{bmatrix} -y^{(k)} \\ x^{(k)} \end{bmatrix}.$$

Observe que este processo iterativo é equivalente a:

$$\begin{aligned} x^{(k+1)} &= x^{(k)} - hy^{(k)} \\ y^{(k+1)} &= y^{(k)} + hx^{(k)}. \end{aligned}$$

Exemplo 10.1.5. Escreva o problema de valor inicial de segunda ordem dado por

$$\begin{aligned} y'' + y' + y &= \cos(t), \\ y(0) &= 1, \\ y'(0) &= 0, \end{aligned}$$

como um problema envolvendo um sistema de primeira ordem.

A fim de transformar a equação diferencial dada em um sistema de equações de primeira ordem, introduzimos a substituição $w = y'$, de forma que obteremos o sistema:

$$\begin{aligned} y' &= w \\ w' &= -w - y + \cos(t) \\ y(0) &= 1 \\ w(0) &= 0 \end{aligned}$$

Portanto, o Método de Euler produz o seguinte processo iterativo:

$$\begin{aligned} y^{(k+1)} &= y^{(k)} + hw^{(k)}, \\ w^{(k+1)} &= w^{(k)} - hw^{(k)} - hy^{(k)} + h\cos(t^{(k)}), \\ y^{(1)} &= 1, \\ w^{(1)} &= 0. \end{aligned}$$

Exercícios

E 10.1.1. Resolva o problema de valor inicial dado por

$$\begin{aligned} y' &= -2y + \sqrt{y} \\ y(0) &= 1 \end{aligned}$$

com passo $h = 0{,}1$ e $h = 0{,}01$ para obter aproximações para $y(1)$. Compare com a solução exata dada por $y(t) = \left(1 + 2e^{-t} + e^{-2t}\right)/4$

E 10.1.2. Resolva o problema de valor inicial dado por

$$\begin{aligned} y' &= -2y + \sqrt{z} \\ z' &= -z + y \\ y(0) &= 0 \\ z(0) &= 2 \end{aligned}$$

com passo $h = 0{,}2$, $h = 0{,}02$, $h = 0{,}002$ e $h = 0{,}0002$ para obter aproximações para $y(2)$ e $z(2)$.

E 10.1.3. Resolva o problema de valor inicial dado por

$$\begin{aligned} y' &= \cos(ty(t)) \\ y(0) &= 1 \end{aligned}$$

com passo $h = 0{,}1$, $h = 0.01$, $h = 0{,}001$, $h = 0{,}0001$ e $0{,}00001$ para obter aproximações para $y(2)$.

10.2 Método de Euler melhorado

O método de Euler foi o primeiro método que estudamos e sua principal virtude é a simplicidade. Outros métodos, no entanto, podem apresentar resultados superiores. Vamos apresentar agora uma pequena modificação ao Método de Euler, dando origem a um novo método chamado de Método de Euler Modificado ou Método de Euler Melhorado.

No método de Euler, usamos a seguinte iteração:

$$\begin{aligned} y^{(k+1)} &= y^{(k)} + hf(y^{(k)},t^{(k)}) \\ y^{(1)} &= y_0 \ \text{(condição inicial)} \end{aligned}$$

A ideia do método de Euler Melhorado é substituir a declividade $f(y^{(k)},t^{(k)})$ pela média aritmética entre $f(y^{(k)},t^{(k)})$ e $f(y^{(k+1)},t^{(k+1)})$, isto é, as declividades avaliadas no início e no fim do intervalo $[t^{(k)},t^{(k+1)}]$.

No entanto, não dispomos do valor de $y^{(k+1)}$ antes de executar o passo. Assim aproximamos esta grandeza pelo valor produzido pelo Método de Euler original:

$$\tilde{y}^{(k+1)} = y^{(k)} + hf(y^{(k)},t^{(k)}).$$

De posse desta aproximação, calculamos a média aritmética e, finalmente, com esta média, realizamos o passo do Método de Euler Melhorado. O processo iterativo de Euler Melhorado é, portanto, dado por:

$$\begin{aligned} \tilde{y}^{(k+1)} &= y^{(k)} + hf(y^{(k)},t^{(k)}) \\ y^{(k+1)} &= y^{(k)} + \frac{h}{2}\left[f(y^{(k)},t^{(k)}) + f(\tilde{y}^{(k+1)},t^{(k+1)})\right] \\ y^{(1)} &= y_0 \ \text{(condição inicial)} \end{aligned}$$

Podemos reescrever este mesmo processo iterativo da seguinte forma:

$$\begin{aligned} k_1 &= hf(y^{(k)},t^{(k)}), \\ k_2 &= hf(y^{(k)}+k_1,t^{(k+1)}), \\ y^{(k+1)} &= y^{(k)} + \frac{k_1+k_2}{2}, \\ y^{(1)} &= y_0 \ \text{(condição inicial)}. \end{aligned}$$

Aqui k_1 e k_2 são variáveis auxiliares que representam as inclinações e devem ser calculadas a cada passo. Esta notação é compatível com a notação usada nos métodos de Runge-Kutta, uma família de esquemas iterativos para aproximar problemas de valor inicial, da qual o Método de Euler e o Método de Euler Melhorado são casos particulares. Veremos os métodos de Runge-Kutta na seção 10.5.

Exercícios

E 10.2.1. Use o Método de Euler melhorado para obter uma aproximação numérica do valor de $y(1)$ quando $y(t)$ satisfaz o seguinte problema de valor inicial

$$\begin{aligned} y'(t) &= -y(t) + e^{y(t)}, \\ y(0) &= 0, \end{aligned}$$

usando passos $h = 0{,}1$ e $h = 0{,}01$.

E 10.2.2. Use o Método de Euler e o Método de Euler melhorado para obter aproximações numéricas para a solução do seguinte problema de valor inicial para $t \in [0,1]$:

$$\begin{aligned} y'(t) &= -y(t) - y(t)^2, \\ y(0) &= 1, \end{aligned}$$

usando passo $h = 0{,}1$. Compare os valores da solução exata dada por $y(t) = \frac{1}{2e^t-1}$ com os numéricos nos pontos $t = 0$, $t = 0.1$, $t = 0.2$, $t = 0.3$, $t = 0.4$, $t = 0.5$, $t = 0.6$, $t = 0.7$, $t = 0.8$, $t = 0.9$, $t = 1.0$.

10.3 Ordem de precisão

Considere o problema de valor inicial dado por

$$\begin{aligned} y'(t) &= f(y(t),t), \\ y(0) &= y_0. \end{aligned}$$

Nessa seção vamos definir a precisão de um método numérico pela ordem do erro acumulado ao calcular o valor da função em um ponto t_N em função do espaçamento da malha h. Se $y(t_n)$ pode ser aproximado por uma expressão que depende de f, h, $y(t_0)$, $y(t_1)$, $\cdots$, $y(t_n)$, com erro da ordem de $O(h^{p+1})$, ou seja,

$$y(t_{n+1}) = \mathcal{F}(f, h, y(t_n), y(t_{n-1}), \cdots, y_0) + O(h^{p+1}) \tag{10.8}$$

para cada função analítica f, dizemos que o método tem erro de truncamento da ordem de $O(h^p)$ ou **ordem de precisão** p. Essa afirmação faz sentido quando fazemos a seguinte análise informal: para aproximar y_1, acumulamos erros da ordem $O(h^{p+1})$, para calcular y_2 acumulamos os erros de y_1 e novos erros $O(h^{p+1})$. Para calcular y_N, acumulamos todos os erros até t_N, ou seja, N vezes $O(h^{p+1})$. Como $N = O(1/h)$, temos que os erros ao calcular y_N são da ordem $O(h^p)$. É verdade que essa análise só vale quando impomos condições de suavidade para f e condições adequada para a expressão $\mathcal{F}(f, h, y(t_n), y(t_{n-1}), \cdots, y_0)$. Para explicar melhor esse pequeno texto, fazemos em detalhes essa operação para o método de Euler na seção 10.3.1.

10.3.1 Ordem de precisão do Método de Euler

Primeiro lembramos da expressão (10.4) que origina a seguinte relação de recorrência:

$$y(t_{n+1}) \;=\; y(t_n) + hf(y(t_n),t_n) + O(h^2). \tag{10.9}$$

Para entender melhor o motivo de na expressão (10.9) aparecer $O(h^2)$ e o método ser de precisão 1, vamos a seguinte análise informal: observemos que

$$\begin{aligned} y(t_1) &= y(t_0) + hf(y(t_0),t_0) + O(h^2) \\ &= y_0 + hf(y_0,t_0) + O(h^2) = y_1 + O(h^2) \end{aligned}$$

onde y_i é a aproximação pelo método de Euler para o valor exato $y(t_i)$. Subsequentemente, temos

$$\begin{aligned} y(t_2) &= y(t_1) + hf(y(t_1),t_1) + O(h^2) \\ &= y(t_1) + hf(y_1 + O(h^2),t_1) + O(h^2) \\ &= y(t_1) + hf(y_1,t_1) + O(h^2) \\ &= y_1 + O(h^2) + hf(y_1,t_1) + O(h^2) = y_2 + O(h^2) + O(h^2). \end{aligned}$$

onde usamos o primeiro termo da série de Taylor $hf(y_1 + O(h^2),t_1) = hf(y_1,t_1) + O(h^3)$ na passagem da segunda para terceira linha. Repetindo sucessivamente o

passo anterior, obtemos uma expressão geral para o valor exato $y(t_N)$ em termos do valor aproximado y_N:

$$y(t_N) = y_N + NO(h^2).$$

Como $N = (t_f - t_0)/h$, temos

$$y(t_N) \quad = \quad = y_N + \frac{t - t_0}{h} O(h^2) = y_N + O(h), \tag{10.10}$$

ou seja, o erro entre o valor exato e o aproximado é de ordem h. Uma demonstração mais formal que garante que o erro é limitado por uma expressão que é proporcional a h está discutido na seção 10.4.1.

10.3.2 Ordem de precisão do Método de Euler Melhorado

Para obter o erro de precisão do método de Euler Melhorado vamos calcular o erro de truncamento do método, ou seja, precisamos demonstrar que:

$$y(t+h) = y(t) + \frac{h}{2} f(y(t),t) + \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) + O(h^3) \tag{10.11}$$

De fato, tomando a diferença do termo da esquerda o os termos da direita, temos:

$$\begin{aligned}
& y(t+h) - \left(y(t) + \frac{h}{2} f(y(t),t) + \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) \right) \\
& = y(t) + hy'(t) + \frac{h^2}{2} y''(t) + O(h^3) \\
& - \left(y(t) + \frac{h}{2} y'(t) + \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) \right),
\end{aligned}$$

onde usamos uma expansão em série de Taylor para $y(t+h)$ e a equação diferencial $y'(t) = f(y(t),t)$. Portanto,

$$\begin{aligned}
& y(t+h) - \left(y(t) + \frac{h}{2} f(y(t),t) + \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) \right) \\
& = \frac{h}{2} y'(t) + \frac{h^2}{2} y''(t) - \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) + O(h^3).
\end{aligned}$$

Agora, usamos a série de Taylor de $f(y(t) + hf(y(t),t),t+h)$ e, torno de (y,t):

$$\begin{aligned}
& y(t+h) - \left(y(t) + \frac{h}{2} f(y(t),t) + \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) \right) \\
& = \frac{h}{2} y'(t) + \frac{h^2}{2} y''(t) + O(h^3) \\
& - \frac{h}{2} \left(f(y(t),t) + \frac{\partial f(y(t),t)}{\partial t} h + \frac{\partial f(t,y(t))}{\partial y} hf(t,y(t)) + O(h^2) \right).
\end{aligned}$$

Usando a equação diferencial $y'(t) = f(y(t),t)$ obtemos

$$y''(t) = \frac{f(y(t),t)}{\partial t} + \frac{f(y(t),t)}{\partial y} y'(t) = \frac{f(y(t),t)}{\partial t} + \frac{f(y(t),t)}{\partial y} f(y(t),t).$$

Logo,

$$\begin{aligned}
& y(t+h) - \left(y(t) + \frac{h}{2} f(y(t),t) + \frac{h}{2} f(y(t) + hf(t,y(t))),t+h) \right) \\
&= \frac{h}{2} y'(t) + \frac{h^2}{2} y''(t) + O(h^3) \\
&- \frac{h}{2} \left(f(y(t),t) + hy''(t) + O(h^2) \right) \\
&= \frac{h}{2} y'(t) + \frac{h^2}{2} y''(t) \\
&- \frac{h}{2} \left(y'(t) + hy''(t) \right) + O(h^3) = O(h^3)
\end{aligned}$$

Portanto, a expressão (10.11) é válida. Logo, usando uma discussão análoga aquela feita na seção 10.3.1 para o método de Euler, concluímos que o método de Euler Melhorado possui ordem de precisão 2.

10.4 Convergência

Em construção ... Gostaria de colaborar na escrita deste livro? Veja como em:

http://www.ufrgs.br/numerico

10.4.1 Convergência do método de Euler

Em construção ... Gostaria de colaborar na escrita deste livro? Veja como em:

http://www.ufrgs.br/numerico

10.4.2 Convergência do método de Euler Melhorado

Em construção ... Gostaria de colaborar na escrita deste livro? Veja como em:

http://www.ufrgs.br/numerico

10.5 Métodos de Runge-Kutta

Os métodos de Runge-Kutta consistem em iterações do tipo:

$$y^{(k+1)} = y^{(k)} + w_1 k_1 + \ldots + w_n k_n$$

onde

$$\begin{aligned} k_1 &= hf(y^{(k)},t^{(k)}) \\ k_2 &= hf(y^{(k)} + \alpha_{2,1}k_1,t^{(k)} + \beta_2 h) \\ k_3 &= hf(y^{(k)} + \alpha_{3,1}k_1 + \alpha_{3,2}k_2,t^{(k)} + \beta_3 h) \\ &\vdots \\ k_n &= hf(y^{(k)} + \alpha_{n,1}k_1 + \alpha_{n,2}k_2 + \ldots \alpha_{n,n-1}k_{n-1},t^{(k)} + \beta_n h) \end{aligned}$$

Os coeficientes são escolhidos de forma que a expansão em Taylor de $y^{(k+1)}$ e $y^{(k)} + w_1 k_1 + \ldots + w_n k_n$ coincidam até ordem $n+1$.

Exemplo 10.5.1. O método de Euler melhorado é um exemplo de Runge-Kutta de segunda ordem

$$y^{(n+1)} = y^{(n)} + \frac{k_1 + k_2}{2}$$

onde $k_1 = hf(y^{(n)},t^{(n)})$ e $k_2 = hf(y^{(n)} + k_1,t^{(n)} + h)$

10.5.1 Métodos de Runge-Kutta - Quarta ordem

$$y^{(n+1)} = y^{(n)} + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

onde

$$\begin{aligned} k_1 &= hf(y^{(n)},t^{(n)}) \\ k_2 &= hf(y^{(n)} + k_1/2,t^{(n)} + h/2) \\ k_3 &= hf(y^{(n)} + k_2/2,t^{(n)} + h/2) \\ k_4 &= hf(y^{(n)} + k_3,t^{(n)} + h) \end{aligned}$$

Este método tem ordem de precisão 4. Uma discussão heurística usando método de Simpson pode ajudar a compreender os estranhos coeficientes:

$$\begin{aligned} y(t^{(n+1)}) - y(t^{(n)}) &= \int_{t^{(n)}}^{t^{(n+1)}} f(y(s),s)ds \\ &\approx \frac{h}{6}\left[f\left(y(t^{(n)}),t^{(n)}\right) + 4f\left(y(t^{(n)}+h/2),t^{(n)}+h/2\right)\right. \\ &+ \left. f\left(y(t^{(n)}+h),t^{(n)}+h\right)\right] \\ &\approx \frac{k_1 + 4(\frac{k_2+k_3}{2}) + k_4}{6} \end{aligned}$$

onde k_1 e k_4 representam as inclinações nos extremos e k_2 e k_3 são duas aproximações diferentes para a inclinação no meio do intervalo.

10.6 Métodos de passo múltiplo - Adams-Bashforth

O método de Adams-Bashforth consiste de um esquema recursivo do tipo:

$$y^{(n+1)} = y^{(n)} + \sum_{j=0}^{k} w_j f(y^{(n-j)},t^{(n-j)})$$

Exemplo 10.6.1. Adams-Bashforth de segunda ordem

$$y^{(n+1)} = y^{(n)} + \frac{h}{2}\left[3f\left(y^{(n)},t^{(n)}\right) - f\left(y^{(n-1)},t^{(n-1)}\right)\right]$$

Exemplo 10.6.2. Adams-Bashforth de terceira ordem

$$y^{(n+1)} = y^{(n)} + \frac{h}{12}\left[23f\left(y^{(n)},t^{(n)}\right) - 16f\left(y^{(n-1)},t^{(n-1)}\right) + 5f\left(y^{(n-2)},t^{(n-2)}\right)\right]$$

Exemplo 10.6.3. Adams-Bashforth de quarta ordem

$$\begin{aligned} y^{(n+1)} = y^{(n)} + \frac{h}{24}\left[55f\left(y^{(n)},t^{(n)}\right) - 59f\left(y^{(n-1)},t^{(n-1)}\right)\right. \\ \left. + 37f\left(y^{(n-2)},t^{(n-2)}\right) - 9f\left(y^{(n-3)},t^{(n-3)}\right)\right] \end{aligned}$$

Os métodos de passo múltiplo evitam os múltiplos estágios do métodos de Runge-Kutta, mas exigem ser "iniciados"com suas condições iniciais.

10.7 Métodos de passo múltiplo - Adams-Moulton

O método de Adams-Moulton consiste de um esquema recursivo do tipo:

$$y^{(n+1)} = y^{(n)} + \sum_{j=-1}^{k} w_j f(y^{(n-j)}, t^{(n-j)})$$

Exemplo 10.7.1. Adams-Moulton de quarta ordem

$$y^{(n+1)} = y^{(n)} + \frac{h}{24}\left[9f\left(y^{(n+1)}, t^{(n+1)}\right) + 19f\left(y^{(n)}, t^{(n)}\right) - 5f\left(y^{(n-1)}, t^{(n-1)}\right) + f\left(y^{(n-2)}, t^{(n-2)}\right)\right]$$

O método de Adams-Moulton é implícito, ou seja, exige que a cada passo, uma equação em $y^{(n+1)}$ seja resolvida.

10.8 Estabilidade

Consideremos o seguinte problema de teste:

$$\begin{cases} y' &= -\alpha y \\ y(0) &= 1 \end{cases}$$

cuja solução exata é dada por $y(t) = e^{-\alpha t}$.

Considere agora o método de Euler aplicado a este problema com passa h:

$$\begin{cases} y^{(k+1)} &= y^{(k)} - \alpha h y^{(k)} \\ y^{(1)} &= 1 \end{cases}$$

A solução exata do esquema de Euler é dada por

$$y^{(k+1)} = (1 - \alpha h)^k$$

e, portanto,

$$\tilde{y}(t) = y^{(k+1)} = (1 - \alpha h)^{t/h}$$

Fixamos um $\alpha > 0$, de forma que $y(t) \to 0$. Mas observamos que $\tilde{y}(t) \to 0$ somente quando $|1 - \alpha h| < 1$ e solução positivas somente quando $\alpha h < 1$.

Conclusão: Se o passo h for muito grande, o método pode se tornar instável, produzindo solução espúrias.

Exercícios

E 10.8.1. Resolva o problema 1 pelos diversos métodos e verifique heuristicamente a estabilidade para diversos valores de h.

10.9 Exercícios finais

E 10.9.1. Considere o seguinte modelo para o crescimento de uma colônia de bactérias:

$$\frac{dy}{dt} = \alpha y(A - y)$$

onde y indica a densidade de bactérias em unidades arbitrárias na colônia e α e A são constantes positivas. Pergunta-se:

a) Qual a solução quando a condição inicial $y(0)$ é igual a 0 ou A?

b) O que acontece quando a condição inicial $y(0)$ é um número entre 0 e A?

c) O que acontece quando a condição inicial $y(0)$ é um número negativo?

d) O que acontece quando a condição inicial $y(0)$ é um número positivo maior que A?

e) Se $A = 10$ e $\alpha = 1$ e $y(0) = 1$, use métodos numéricos para obter tempo necessário para que a população dobre?

f) Se $A = 10$ e $\alpha = 1$ e $y(0) = 4$, use métodos numéricos para obter tempo necessário para que a população dobre?

E 10.9.2. Considere o seguinte modelo para a evolução da velocidade de um objeto em queda (unidades no SI):

$$v' = g - \alpha v^2$$

Sabendo que $g = 9{,}8$ e $\alpha = 10^{-2}$ e $v(0) = 0$. Pede-se a velocidade ao tocar o solo, sabendo que a altura inicial era 100.

E 10.9.3. Considere o seguinte modelo para o oscilador não-linear de Van der Pol:

$$y''(t) - \alpha(A - y(t)^2)y'(t) + w_0^2 y(t) = 0$$

onde A, α e w_0 são constantes positivas.

- Encontre a frequência e a amplitude de oscilações quando $w_0 = 1$, $\alpha = .1$ e $A = 10$. (Teste diversas condições iniciais)
- Estude a dependência da frequência e da amplitude com os parâmetros A, α e w_0. (Teste diversas condições iniciais)
- Que diferenças existem entre esse oscilador não-linear e o oscilador linear?

E 10.9.4. Considere o seguinte modelo para um oscilador não-linear:

$$\begin{aligned} y''(t) - \alpha(A - z(t))y'(t) + w_0^2 y(t) &= 0 \\ Cz'(t) + z(t) &= y(t)^2 \end{aligned}$$

onde A, α, w_0 e C são constantes positivas.

- Encontre a frequência e a amplitude de oscilações quando $w_0 = 1$, $\alpha = .1$, $A = 10$ e $C = 10$. (Teste diversas condições iniciais)
- Estude a dependência da frequência e da amplitude com os parâmetros A, α, w_0 e C. (Teste diversas condições iniciais)

E 10.9.5. Considere o seguinte modelo para o controle de temperatura em um processo químico:

$$\begin{aligned} CT'(t) + T(t) &= \kappa P(t) + T_{ext} \\ P'(t) &= \alpha(T_{set} - T(t)) \end{aligned}$$

onde C, α e κ são constantes positivas e $P(t)$ indica o potência do aquecedor. Sabendo que T_{set} é a temperatura desejada, interprete o funcionamento esse sistema de controle.

- Calcule a solução quando a temperatura externa $T_{ext} = 0$, $T_{set} = 1000$, $C = 10$, $\kappa = .1$ e $\alpha = .1$. Considere condições iniciais nulas.
- Quanto tempo demora o sistema para atingir a temperatura 900K?
- Refaça os dois primeiros itens com $\alpha = 0.2$ e $\alpha = 1$
- Faça testes para verificar a influência de T_{ext}, α e κ na temperatura final.

E 10.9.6. Considere a equação do pêndulo dada por:

$$\frac{d^2\theta(t)}{dt^2} + \frac{g}{l}\sin(\theta(t)) = 0$$

onde g é o módulo da aceleração da gravidade e l é o comprimento da haste.

- Mostre analiticamente que a energia total do sistema dada por

$$\frac{1}{2}\left(\frac{d\theta(t)}{dt}\right)^2 - \frac{g}{l}\cos(\theta(t))$$

é mantida constante.

- Resolva numericamente esta equação para $g = 9{,}8m/s^2$ e $l = 1m$ e as seguintes condições iniciais:

$\theta(0) = 0.5$ e $\theta'(0) = 0$.

$\theta(0) = 1.0$ e $\theta'(0) = 0$.

$\theta(0) = 1.5$ e $\theta'(0) = 0$.

$\theta(0) = 2.0$ e $\theta'(0) = 0$.

$\theta(0) = 2.5$ e $\theta'(0) = 0$.

$\theta(0) = 3.0$ e $\theta'(0) = 0$.

Em todos os casos, verifique se o método numérico reproduz a lei de conservação de energia e calcule período e amplitude.

E 10.9.7. Considere o modelo simplificado de FitzHugh-Nagumo para o potencial elétrico sobre a membrana de um neurônio:

$$\begin{aligned} \frac{dV}{dt} &= V - V^3/3 - W + I \\ \frac{dW}{dt} &= 0.08(V + 0.7 - 0.8W) \end{aligned}$$

onde I é a corrente de excitação.

- Encontre o único estado estacionário (V_0,W_0) com $I = 0$.

- Resolva numericamente o sistema com condições iniciais dadas por (V_0,W_0) e

$I = 0$

$I = 0.2$

$I = 0.4$

$I = 0.8$

$I = e^{-t/200}$

E 10.9.8. Considere o problema de valor inicial dado por

$$\begin{aligned} \frac{du(t)}{dt} &= -u(t) + e^{-t} \\ u(0) &= 0 \end{aligned}$$

Resolva analiticamente este problema usando as técnicas elementares de equações diferenciais ordinárias. A seguir encontre aproximações numéricas usando os métodos de Euler, Euler modificado, Runge-Kutta Clássico e Adams-Bashforth de ordem 4 conforme pedido nos itens.

a) Construa uma tabela apresentando valores com 7 algarismos significativos para comparar a solução analítica com as aproximações numéricas produzidas pelos métodos sugeridos. Construa também uma tabela para o erro absoluto obtido por cada método numérico em relação à solução analítica. Nesta última tabela, expresse o erro com 2 algarismos significativos em formato científico. Dica: $format('e',8)$ para a segunda tabela.

	0.5	1.0	1.5	2.0	2.5
Analítico					
Euler					
Euler modificado					
Runge-Kutta Clássico					
Adams-Bashforth ordem 4					

	0.5	1.0	1.5	2.0	2.5
Euler					
Euler modificado					
Runge-Kutta Clássico					
Adams-Bashforth ordem 4					

b) Calcule o valor produzido por cada um desses método para $u(1)$ com passo $h = 0.1$, $h = 0.05$, $h = 0.01$, $h = 0.005$ e $h = 0.001$. Complete a tabela com os valores para o erro absoluto encontrado.

	0.1	0.05	0.01	0.005	0.001
Euler					
Euler modificado					
Runge-Kutta Clássico					
Adams-Bashforth ordem 4					

Capítulo 11

Problemas de Valores de Controno

Neste capítulo, discutimos sobre métodos numéricos para resolver equações diferenciais ordinárias com condições de controno.

Nos códigos `Python` apresentados, assumimos que as seguintes bibliotecas e módulos estão carregados:

```
>>> from __future__ import division
>>> import numpy as np
>>> from numpy import linalg
>>> import matplotlib.pyplot as plt
```

11.1 Método de Diferenças Finitas

Nesta seção, discutimos os fundamentos do método de diferenças finitas (MDF) para problemas de valores de contorno. Este método consiste na reformulação do problema contínuo em um problema discreto definido sobre uma malha apropriada.

Para introduzir os conceitos principais, consideramos o seguinte problema de valor de contorno de Dirichlet:

$$\begin{aligned} -u_{xx} &= f(x,u), \quad a < x < b, && (11.1)\\ u(a) &= u_a, && (11.2)\\ u(b) &= u_b, && (11.3) \end{aligned}$$

Resolver numericamente o problema acima exige uma discretização do domínio $[a,b]$, ou seja, dividir o domínio em N partes iguais, definindo

$$h = \frac{b-a}{N}$$

O conjunto de abcissas x_i, $i = 1,...,N+1$ formam uma malha para o problema discreto. Nosso objetivo é encontrar as ordenadas $u_i = u(x_i)$ que satisfazem a versão discreta:

$$\begin{cases} -\frac{u_{i+1}-2u_i+u_{i-1}}{h^2} = f(x_i,u_i), \ \ 2 \leq i \leq N. \\ u_1 = u_a \\ u_{N+1} = u_b \end{cases}$$

O vetor solução $(u_i)_{i=1}^{N+1}$ do problema é solução do sistema acima, que é linear se f for linear em u e não linear caso contrário.

Exemplo 11.1.1. Encontre uma solução numérica para o problema de contorno:

$$\begin{cases} -u_{xx} + u = e^{-x}, \ \ 0 < x < 1. \\ u(0) = 1 \\ u(1) = 2 \end{cases}$$

Solução. Observe que

$$h = \frac{1}{N}$$

e a versão discreta da equação é

$$\begin{cases} -\frac{u_{i+1}-2u_i+u_{i-1}}{h^2} + u_i = e^{-x_i}, \ \ 2 \leq i \leq N. \\ u_1 = 1 \\ u_{N+1} = 2 \end{cases}$$

ou seja,

$$\begin{cases} u_1 = 1 \\ -u_{i+1} + (2+h^2)u_i - u_{i-1} = h^2 e^{-x_i}, \ \ 2 \leq i \leq N. \\ u_{N+1} = 2 \end{cases}$$

que é um sistema linear. A sua forma matricial é:

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 2+h^2 & -1 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 2+h^2 & \cdots & 0 & 0 & 0 \\ \vdots & & & & \ddots & & \\ 0 & 0 & 0 & \cdots & -1 & 2+h^2 & -1 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_N \\ u_{N+1} \end{bmatrix} = \begin{bmatrix} 1 \\ h^2 e^{-x_2} \\ h^2 e^{-x_3} \\ \vdots \\ h^2 e^{-x_N} \\ 2 \end{bmatrix}$$

Para $N = 10$, temos a seguinte solução:

$$\begin{bmatrix} 1{,}000000 \\ 1{,}0735083 \\ 1{,}1487032 \\ 1{,}2271979 \\ 1{,}3105564 \\ 1{,}4003172 \\ 1{,}4980159 \\ 1{,}6052067 \\ 1{,}7234836 \\ 1{,}8545022 \\ 2{,}000000 \end{bmatrix}$$

$\Diamond$

Exercícios

E 11.1.1. Considere o seguinte problema de valor de contorno para a equação de calor no estado estacionário:

$$\begin{cases} -u_{xx} = 32, \ \ 0 < x < 1. \\ u(0) = 5 \\ u(1) = 10 \end{cases}$$

Defina $u_j = u(x_j)$ onde $x_j = (j-1)h$ e $j = 1,\ldots,5$. Aproxime a derivada segunda por um esquema de segunda ordem e transforme a equação diferencial em um sistema de equações lineares. Escreva este sistema linear na forma matricial e resolva-o. Faça o mesmo com o dobro de subintervalos, isto é, com malha de 9 pontos.

E 11.1.2. Considere o seguinte problema de valor de contorno para a equação de calor no estado estacionário:

$$\begin{cases} -u_{xx} = 200e^{-(x-1)^2}, \ \ 0 < x < 2. \\ u(0) = 120 \\ u(2) = 100 \end{cases}$$

Defina $u_j = u(x_j)$ onde $x_j = (j-1)h$ e $j = 1,\ldots,21$. Aproxime a derivada segunda por um esquema de segunda ordem e transforme a equação diferencial em um sistema de equações lineares. Resolva o sistema linear obtido.

E 11.1.3. Considere o seguinte problema de valor de contorno para a equação de calor no estado estacionário:

$$\begin{cases} -u_{xx} = 200e^{-(x-1)^2}, \ \ 0 < x < 2. \\ u'(0) = 0 \\ u(2) = 100 \end{cases}$$

Defina $u_j = u(x_j)$ onde $x_j = (j-1)h$ e $j = 1,\ldots,21$. Aproxime a derivada segunda por um esquema de segunda ordem, a derivada primeira na fronteira por um esquema de primeira ordem e transforme a equação diferencial em um sistema de equações lineares. Resolva o sistema linear obtido.

E 11.1.4. Considere o seguinte problema de valor de contorno para a equação de calor no estado estacionário com um termo não linear de radiação:

$$\begin{cases} -u_{xx} = 100 - \frac{u^4}{10000}, \ \ 0 < x < 2. \\ u(0) = 0 \\ u(2) = 10 \end{cases}$$

Defina $u_j = u(x_j)$ onde $x_j = (j-1)h$ e $j = 1,\ldots,21$. Aproxime a derivada segunda por um esquema de segunda ordem e transforme a equação diferencial em um sistema de equações não lineares. Resolva o sistema obtido. Expresse a

solução com dois algarismos depois do separador decimal. Dica: Veja problema 38 da lista 2, seção de sistemas não lineares.

E 11.1.5. Considere o seguinte problema de valor de contorno para a equação de calor no estado estacionário com um termo não linear de radiação e um termo de convecção:

$$\begin{cases} -u_{xx} + 3u_x = 100 - \frac{u^4}{10000}, \ \ 0 < x < 2. \\ u'(0) = 0 \\ u(2) = 10 \end{cases}$$

Defina $u_j = u(x_j)$ onde $x_j = (j-1)h$ e $j = 1, \ldots, 21$. Aproxime a derivada segunda por um esquema de segunda ordem, a derivada primeira na fronteira por um esquema de primeira ordem, a derivada primeira no interior por um esquema de segunda ordem e transforme a equação diferencial em um sistema de equações não lineares. Resolva o sistema obtido.

E 11.1.6. Considere o seguinte problema de valor de contorno:

$$\begin{cases} -u'' + 2u' = e^{-x} - \frac{u^2}{100}, \ \ 1 < x < 4. \\ u'(1) + u(1) = 2 \\ u'(4) = -1 \end{cases}$$

Defina $u_j = u(x_j)$ onde $x_j = 1 + (j-1)h$ e $j = 1, \ldots, 101$. Aproxime a derivada segunda por um esquema de segunda ordem, a derivada primeira na fronteira por um esquema de primeira ordem, a derivada primeira no interior por um esquema de segunda ordem e transforme a equação diferencial em um sistema de equações não lineares. Resolva o sistema obtido.

Apêndice A

Rápida Introdução à Python

Neste apêndice, discutiremos os principais aspectos da linguagem computacional `Python` que são excenciais para uma boa leitura desta versão do livro. O material aqui apresetado, é uma adaptação livre do Apêndice A de [11].

A.1 Sobre a linguagem Python

Python é uma linguagem de programação de alto nível, interpretada e multiparadigma. Lançada por Guido van Rossum[1] em 1991 é, atualmente, mantida de forma colaborativa e aberta.

Para mais informações, consulte:

- Página oficial da linguagem Python: `https://www.python.org/`
- Comunidade Python Brasil: `http://wiki.python.org.br/`

Para iniciantes, recomendamos o curso EAD gratuíto no site Codecademy:

`https://www.codecademy.com/learn/python`

A.1.1 Instalação e Execução

Para executar um código `Python` é necessário ter instalado um interpretador para a linguagem. No site oficial do Python estão disponíveis para *download* os interpretadores `Python 2.7` e `Python 3` para vários sistemas operacionais, como `Linux`, `Mac OS` e `Windows`. Muitas distribuições de `Linux` (`Linux Mint`, `Ubuntu`, etc.) têm o `Python` no seu sistema de pacotes (incluindo documentação em várias línguas).

Ao longo do texto, assumiremos que o leitor estaja usando um computar rodando `Linux`. Para outros sistemas, pode ser necessário fazer algumas adaptações.

[1] Guido van Rossum, nascido em 1956, programador de computadores dos Países Baixos.

A.1.2 Usando Python

O uso do Python pode ser feito de três formas básicas:

- usando um **console Python** de modo iterativo;
- executando um código `codigo.py` no console `Python`;
- executando um código `Python codigo.py` diretamente em terminal;

Exemplo A.1.1. Considere o seguinte pseudocódigo:

```
s = "Olá Mundo!". (Sem imprimir na tela o resultado.)
saída(s). (Imprime na tela.)
```

Implemente este pseudocódigo em `Python`: a) usando diretamente um console; b) digitando seu código em um arquivo separado e executando-o no console Python com a função `execfile`.b) digitando seu código em um arquivo separado e executando-o em terminal com o comando `python`.

Solução. Seguem as soluções de cada item:

a) No console temos:

```
>>> s = "Olá, Mundo!"
>>> print(s)
Olá, Mundo!
```

Para sair do console, digite:

```
>>> quit()
```

b) Abra o editor de texto de sua preferência e digite o código:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

s = 'Olá'
print(s)
```

Salve o arquivo como, por exemplo, `ola.py`. No console `Python`, digite:

```
>>> execfile("ola.py")
```

c) Abra o editor de texto de sua preferência e digite o código:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

s = 'Olá'
print(s)
```

Salve o arquivo como, por exemplo, `ola.py`. Em um terminal, digite:

```
$ python ola.py
```

◊

A.2 Elementos da linguagem

`Python` é uma linguagem de alto nível, interpretada e dinâmica. Uma variável é criada quando um valor é atribuído a ela. Por exemplo:

```
>>> x=1
>>> y = x * 2.0
```

a variável `x` recebe o valor `int` 1 e, logo após, na segunda linha de comando, a variável `y` recebe o valor `double` 2. Observamos que o símbolo `=` significa o operador de atribuição não o de igualdade. O operador lógico de igualdade no `Python` é `==`. Veja os seguintes comandos:

```
>>> print(x,y)
(1, 2.0)
>>> type(x), type(y)
(<type 'int'>, <type 'float'>)
```

Comentários e continuação de linha de comando são usados como no seguinte exemplo:

```
>>> #isto é um comentário
...
>>> x = 1 \
... + 2
>>> print(x)
3
```

A.2.1 Operações matemáticas elementares

Em `Python`, os operadores matemáticos elementares são os seguintes:

```
+ adição
- subtração
* multiplicação
/ divisão
** potenciação
```

Atenção, a operação de divisão se comporta diferente nas versões `Python 2.7` e `Python 3`. Em `Python 3`, temos:

```
>>> 1/2
0.5
```

Já, em `Python 2.7`:

```
>>> 1/2
0
>>> from __future__ import division
>>> 1/2
0.5
```

A.2.2 Funções e constantes elementares

Várias funções e constantes elementares estão disponíveis no pacote módulo Python math. Por exemplo:

```
>>> import math as math
>>> math.cos(pi)
-1.0
>>> math.exp(1)
2.718281828459045
>>> math.log(math.exp(1))
1.0
```

Observamos que `math.log` é a função logarítmo natural, i.e. $f(x) = \ln(x)$, enquanto que a implementação Python de $f(x) = \log(x)$ é:

```
>>> math.log10(10)
1.0
```

Veja mais na documentação do módulo math:

```
>>> help(math)
```

A.2.3 Operadores lógicos

Em `Python`, o valor lógico verdadeiro é escrito como `True` e o valor lógico falso como `False`. Temos os seguintes operadores lógicos disponíveis:

```
and  e lógico
or  ou lógico
not  negação
== igualdade
!= diferente
<  menor que
>  maior que
<= menor ou igual que
>= maior ou igual que
```

Exemplo A.2.1. Se $x = 2$, então x é maior ou igual a 1 e menor que 3?

Solução. Em `Python`, temos:

```
>>> x=2
>>> (x >= 1) and (x < 3)
True
```

$\Diamond$

A.3 Matrizes

Em `Python`, temos um ótimo suporte para computação científica com o pacote numpy. Uma matriz $A = [a_{i,j}]_{i,j=1}^{m,n}$ em Python é definida usando-se a seguinte sintaxe:

```
>>> import numpy as np
>>> A = np.array([[ a11 , a12 , ... , a1n], [...]. [am1 , am2 , ... , amn]])
```

Exemplo A.3.1. Defina a matriz:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

Solução. Em `Python`, digitamos:

```
>>> import numpy as np
>>> A = np.array([[1,2,3],
...               [4,5,6]])
>>> print(A)
[[1 2 3]
 [4 5 6]]
```

◊

A seguinte lista contém uma série de funções que geram matrizes particulares:

```
numpy.eye      matrix identidade
numpy.linspace vetor de elementos linearmente espaçados
numpy.ones     matriz cheia de uns
numpy.zeros    matriz nula
```

A.3.1 Obtendo dados de uma matriz

A função `numpy.shape` retorna o tamanho de uma matriz, por exemplo:

```
>>> A = np.ones((3,2))
>>> print(A)
[[ 1.  1.]
 [ 1.  1.]
 [ 1.  1.]]
>>> nl, nc = np.shape(A)
>>> print(nl,nc)
(3, 2)
```

informando que a matriz `A` tem três linhas e duas colunas.

Existem vários métodos para se acessar os elementos de uma matriz dada `A`:

- a matriz inteira acessa-se com a sintaxe:

  `A`

- o elemento da i-ésima linha e j-ésima coluna acessa-se usando a sintaxe:

  `A[i,j]`

- o bloco formado pelas linhas i_1, i_2 e pelas colunas j_1, j_2 obtém-se usando a sintaxe:

  `A[i1:i2, j1:j2]`

Exemplo A.3.2. Veja as seguintes linhas de comando:

```
>>> from numpy import random
>>> A = np.random.random((3,4))
>>> A
array([[ 0.39235668,  0.30287204,  0.24379253,  0.98866709],
       [ 0.72049734,  0.99300252,  0.14232844,  0.25604346],
       [ 0.61553036,  0.80615392,  0.22418474,  0.13685148]])
>>> A[2,3]
0.13685147547025989
>>> A[1:3,1:4]
array([[ 0.99300252,  0.14232844,  0.25604346],
       [ 0.80615392,  0.22418474,  0.13685148]])
```

Definida uma matriz A em Python, as seguintes sintaxes são bastante úteis:

```
A[:,:]   toda a matriz
A[i:j,k] os elementos das linhas i até j (exclusive) da k-ésima coluna
A[i,j:k] os elementos da i-ésina linha das colunas j até k (exclusive)
A[i,:]   a i-ésima linha da matriz
A[:,j]   a j-ésima coluna da matriz
```

Atenção, os índices em `Python` iniciam-se em 0. Assim, o comando `A[1:3,1:4]` retorna o bloco da matriz A compreendido da segunda à terceira linha e da segunda a quarta coluna desta matriz.

Exemplo A.3.3. Veja as seguintes linhas de comando:

```
>>> B = np.random.random((4,4))
>>> B
array([[ 0.94313432,  0.72650883,  0.55487089,  0.18753526],
       [ 0.02094937,  0.45726099,  0.51925464,  0.8535878 ],
       [ 0.75948469,  0.95362926,  0.77942318,  0.06464183],
       [ 0.91243198,  0.22775889,  0.04061536,  0.14908227]])
>>> aux = np.copy(B[:,2])
>>> B[:,2] = np.copy(B[:,3])
>>> B[:,3] = np.copy(aux)
>>> B
array([[ 0.94313432,  0.72650883,  0.18753526,  0.55487089],
       [ 0.02094937,  0.45726099,  0.8535878 ,  0.51925464],
       [ 0.75948469,  0.95362926,  0.06464183,  0.77942318],
       [ 0.91243198,  0.22775889,  0.14908227,  0.04061536]])
```

A.3.2 Operações matriciais e elemento-a-elemento

Em `Python` com `numpy`, o operador `*` opera elemento a elemento. Por exemplo:

```
>>> A = np.array([[1,2],[2,1]]); print(A)
[[1 2]
 [2 1]]
>>> B = np.array([[2,1],[2,1]]); print(B)
[[2 1]
 [2 1]]
>>> print(A*B)
[[2 2]
 [4 1]]
```

A multiplicação matricial obtemos com:

```
>>> C = A.dot(B)
>>> print(C)
[[6 3]
 [6 3]]
```

Aqui, temos as sintaxes análogas entre operações elemento-a-elemento:

```
+ adição
- subtração
* multiplicação
/ divisão
** potenciação
```

Exemplo A.3.4. Veja as seguintes linhas de comando:

```
>>> A = np.ones((2,2))
>>> A
array([[ 1.,  1.],
       [ 1.,  1.]])
>>> B = 2 * np.ones((2,2))
>>> B
array([[ 2.,  2.],
       [ 2.,  2.]])
>>> A*B
array([[ 2.,  2.],
       [ 2.,  2.]])
>>> A.dot(B)
array([[ 4.,  4.],
```

```
       [ 4.,  4.]])
>>> A/B
array([[ 0.5,  0.5],
       [ 0.5,  0.5]])
```

A.4 Estruturas de ramificação e repetição

A linguagem `Python` contém estruturas de repetição e ramificação padrões de linguagens estruturadas.

A.4.1 A instrução de ramificação "if"

A instrução "if" permite executar um pedaço do código somente se uma dada condição for satisfeita.

Exemplo A.4.1. Veja o seguinte código Scilab:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

i = 2
if (i == 1):
    print("Olá!")
elif (i == 2):
    print("Hallo!")
elif (i == 3):
    print("Hello!")
else:
    print("Ça Va!")
```

Qual é a saída apresentada pelo código? Porquê?

Observamos que, em `Python`, a identação é obrigatória, pois é ela que defini o escopo da instrução.

A.4.2 A instrução de repetição "for"

A instrução `for` permite que um pedaço de código seja executado repetidamente.

Exemplo A.4.2. Veja o seguinte código:

```
for i in range(6):
    print(i)
```

Qual é a saída deste código? Porquê?

Exemplo A.4.3. Veja o seguinte código:

```
import numpy as np
for i in np.arange(1,8,2):
    print(i)
```

Qual é a saída deste código? Porquê?

Exemplo A.4.4. Veja o seguinte código:

```
for k = 10:-3:1
    disp(k)
end
```

O que é mostrado no console do Scilab?

Exemplo A.4.5. Veja o seguinte código:

```
import numpy as np
for i in np.arange(10,1,-3):
    print(i)
```

O que é mostrado no console do Scilab?

A.4.3 A instrução de repetição "while"

A instrução `while` permite que um pedaço de código seja executado repetidamente até que uma dada condição seja satisfeita.

Exemplo A.4.6. Veja o seguinte código Scilab:

```
s = 0
i = 1
while (i <= 10):
    s = s + i
    i = i + 1
```

Qual é o valor de `s` ao final da execução? Porquê?

A.5 Funções

Além das muitas funções disponíveis em `Python` (e os tantos muitos pacotes livers disponíveis), podemos definir nossas próprias funções. Para tanto, existe a instrução `def`. Veja os seguintes exemplos:

Exemplo A.5.1. O seguinte código:

```
def f(x):
    return x + np.sin(x)
```

define a função $f(x) = x + \text{sen } x$.

Observe que $f(\pi) = \pi$. Confirme isso computando:

```
>>> f(np.pi)
```

Exemplo A.5.2. O seguinte código em `Python`:

```
def h(x,y):
    if (x < y):
        return y - x
    else:
        return x - y
```

define a função:

$$h(x,y) = \begin{cases} y - x & , x < y \\ x - y & , x \geq y \end{cases}$$

Exemplo A.5.3. O seguinte código:

```
def J(x):
    y = np.zeros((2,2))
    y[0,0] = 2*x[0]
    y[0,1] = 2*x[1]

    y[1,0] = -x[1]*np.sin(x[0]*x[1])
    y[1,1] = -x[0]*np.sin(x[0]*x[1])

    return y
```

define a matriz jacobiana $J(x_1,x_2) := \frac{\partial(f_1,f_2)}{\partial(x_1,x_2)}$ da função:

$$\boldsymbol{f}(x_1,x_2) = (x_1^2 + x_2^2,\ \cos(x_1 x_2)).$$

A.6 Gráficos

Para criar um esboço do gráfico de uma função de uma variável real $y = f(x)$, podemos usar a biblioteca `Python` mathplotlib.A função `matplotlib.pyplot.plot` faz uma representação gráfica de um conjunto de pontos $\{(x_i, y_i)\}$ fornecidos. Existe uma série de opções para esta função de forma que o usuário pode ajustar várias questões de visualização. Veja a documentação.

Exemplo A.6.1. Veja as seguintes linhas de código:

```
>>> import numpy as np
>>> import matplotlib.pyplot as plt
>>> def f(x): return x**3 + 1
...
>>> x = np.linspace(-2,2)
>>> plt.plot(x, f(x))
[<matplotlib.lines.Line2D object at 0x7f4f6d153510>]
>>> plt.grid()
>>> plt.show()
```

Resposta dos Exercícios

Recomendamos ao leitor o uso criterioso das respostas aqui apresentadas. Devido a ainda muito constante atualização do livro, as respostas podem conter imprecisões e erros.

E 2.1.1.

a) 4; b) 9; c) b^2; d) 7; e) 170; f) 7,125; g) 3,28

E 2.1.5.

$(101{,}1)_2$

E 2.1.6.

$(11{,}1C)_{16}$

E 2.1.7.

50; 18

E 2.1.8.

10,5; $(1010{,}1)_2$

E 2.5.1.

2% , deve-se melhorar a medida na variável x, pois, por mais que o erro relativo seja maior para esta variável, a propagação de erros atraves desta variáveis é muito menos importante do que para a outra variável.

E 2.5.2.

3,2% pela aproximação ou 3,4% pela segundo método ($0{,}96758 \leq I \leq 1{,}0342$).

E 2.6.1.

Quando μ é pequeno, $e^{1/mu}$ é um número grande. A primeira expressão produz um "overflow" (número maior que o máximo representável) quando μ é pequeno. A segunda expressão, no entanto, reproduz o limite 1 quando $\mu \to 0+$.

E 2.6.2.

a) $\frac{1}{2} + \frac{x^2}{4!} + O(x^4)$; b) $x/2 + O(x^2)$; c) $5 \cdot 10^{-4}x + O(x^2)$; d) $\frac{\sqrt{2}}{4}y + O(y^2) = \frac{\sqrt{2}}{4}x + O(x^2)$

E 2.6.5.

$4{,}12451228 \times 10^{-16}$ J; 0,002%; $0{,}26654956 \times 10^{-14}$ J; 0,002%; $4{,}98497440 \times 10^{-13}$ J; 0,057%; $1{,}74927914 \times 10^{-12}$ J; 0,522%.

E 2.6.6.

Em ambos casos, temos a seguinte estrutura:

$$\begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} [A] \\ [B] \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

De forma que

$$\begin{bmatrix} [A] \\ [B] \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix}^{-1} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \frac{1}{S_{11}S_{22} - S_{12}S_{21}} \begin{bmatrix} S_{22} & -S_{12} \\ -S_{21} & S_{11} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Portanto

$$\begin{aligned} [A] &= \frac{S_{22}v_1 - S_{12}v_2}{S_{11}S_{22} - S_{12}S_{21}} \\ [B] &= \frac{-S_{21}v_1 + S_{11}v_2}{S_{11}S_{22} - S_{12}S_{21}} \end{aligned}$$

Usando derivação logarítmica, temos

$$\begin{aligned} \frac{1}{[A]}\frac{\partial [A]}{\partial S_{11}} &= -\frac{S_{22}}{S_{11}S_{22} - S_{12}S_{21}} \\ \frac{1}{[A]}\frac{\partial [A]}{\partial S_{12}} &= -\frac{v_2}{S_{22}v_1 - S_{12}v_2} + \frac{S_{21}}{S_{11}S_{22} - S_{12}S_{21}} = -\frac{[A]}{[B]} \cdot \frac{S_{22}}{S_{11}S_{22} - S_{12}S_{21}} \\ \frac{1}{[A]}\frac{\partial [A]}{\partial S_{21}} &= \frac{S_{12}}{S_{11}S_{22} - S_{12}S_{21}} \\ \frac{1}{[A]}\frac{\partial [A]}{\partial S_{22}} &= \frac{v_1}{S_{22}v_1 - S_{12}v_2} - \frac{S_{11}}{S_{11}S_{22} - S_{12}S_{21}} = \frac{[A]}{[B]} \cdot \frac{S_{12}}{S_{11}S_{22} - S_{12}S_{21}} \end{aligned}$$

e

$$\begin{aligned}
\frac{1}{[B]}\frac{\partial [B]}{\partial S_{11}} &= \frac{v_2}{-S_{21}v_1 + S_{11}v_2} - \frac{S_{22}}{S_{11}S_{22} - S_{12}S_{21}} = \frac{[B]}{[A]}\frac{S_{21}}{S_{11}S_{22} - S_{12}S_{21}} \\
\frac{1}{[B]}\frac{\partial [B]}{\partial S_{12}} &= \frac{S_{21}}{S_{11}S_{22} - S_{12}S_{21}} \\
\frac{1}{[B]}\frac{\partial [B]}{\partial S_{21}} &= -\frac{v_1}{-S_{21}v_1 + S_{11}v_2} + \frac{S_{21}}{S_{11}S_{22} - S_{12}S_{21}} = -\frac{[B]}{[A]}\frac{S_{11}}{S_{11}S_{22} - S_{12}S_{21}} \\
\frac{1}{[B]}\frac{\partial [B]}{\partial S_{22}} &= -\frac{S_{11}}{S_{11}S_{22} - S_{12}S_{21}}
\end{aligned}$$

E o erro associado às medidas pode ser aproximado por

$$\begin{aligned}
\frac{1}{[A]}\delta_{[A]} &= \left|\frac{1}{[A]}\frac{\partial [A]}{\partial S_{11}}\right|\delta_{S_{11}} + \left|\frac{1}{[A]}\frac{\partial [A]}{\partial S_{12}}\right|\delta_{S_{12}} + \left|\frac{1}{[A]}\frac{\partial [A]}{\partial S_{21}}\right|\delta_{S_{21}} + \left|\frac{1}{[A]}\frac{\partial [A]}{\partial S_{22}}\right|\delta_{S_{22}} \\
&= \frac{1}{|\det S|}\left[S_{22}\delta_{S_{11}} + \frac{[A]}{[B]}S_{22}\delta_{S_{12}} + S_{12}\delta_{S_{21}} + \frac{[A]}{[B]}S_{12}\delta_{S_{22}}\right]
\end{aligned}$$

Analogamente, temos:

$$\frac{1}{[B]}\delta_{[B]} = \frac{1}{|\det S|}\left[\frac{[B]}{[A]}S_{21}\delta_{S_{11}} + S_{21}\delta_{S_{11}} + \frac{[B]}{[A]}S_{11}\delta_{S_{21}} + S_{11}\delta_{S_{22}}\right]$$

onde não se indicou $|S_{ij}|$ nem $|[.]|$ pois são todos positivos.
Fazemos agora a aplicação numérica:
Caso do par 1-2:

$$\det S = \begin{vmatrix} 270 & 30 \\ 140 & 20 \end{vmatrix} = 1200$$

$$\begin{aligned}
\frac{1}{[A]}\delta_{[A]} &= \frac{1}{1200}\left[20\times 270\times 2\% + 20\times 30\times 2\% + 30\times 140\times 2\% + 30\times 20\times 2\%\right] \\
&= \frac{216}{1200} = 0.18 = 18\% \\
\frac{1}{[B]}\delta_{[B]} &= \frac{1}{1200}\left[140\times 270\times 2\% + 140\times 30\times 2\% + 270\times 140\times 2\% + 270\times 20\times 2\%\right] \\
&= \frac{426}{1200} = 0.355 = 35.5\%
\end{aligned}$$

Caso do par 1-3:

$$\det S = \begin{vmatrix} 270 & 30 \\ 15 & 200 \end{vmatrix} = 53550$$

$$\begin{aligned}
\frac{1}{[A]}\delta_{[A]} &= \frac{1}{53550}\left[200\times 270\times 2\% + 200\times 30\times 2\% + 30\times 15\times 10\% + 30\times 200\times 10\%\right] \\
&= \frac{1804{,}6}{52550} \approx 0.0337 = 3.37\% \\
\frac{1}{[B]}\delta_{[B]} &= \frac{1}{53550}\left[15\times 270\times 2\% + 15\times 30\times 2\% + 270\times 15\times 10\% + 270\times 200\times 10\%\right] \\
&= \frac{5895}{53550} \approx 0.11 = 11\%
\end{aligned}$$

Conclusão, apesar de o sensor 3 apresentar uma incerteza cinco vezes maior na sensibilidade, a escolha do sensor 3 para fazer par ao sensor 1 parece mais adequada.

E 3.1.1.

Observamos que a equação é equivalente a $\cos(x) - x = 0$. Tomando, então, $f(x) = \cos(x) - x$, temos que $f(x)$ é contínua em $[0, \pi/2]$, $f(0) = 1$ e $f(\pi/2) = -\pi/2 < 0$. Logo, do teorema de Bolzano 3.1.1, concluímos que a equação dada tem pelo menos uma solução no intervalo $(0, \pi/2)$.

E 3.1.2.

No Exercício 3.1.1, mostramos que a função $f(x) = \cos(x) - x$ tem um zero no intervalo $[0, \pi/2]$. Agora, observamos que $f'(x) = -\operatorname{sen}(x) - 1$. Como $0 < \operatorname{sen} x < 1$ para todo $x \in (0, \pi/2)$, temos que $f'(x) < 0$ em $(0, \pi/2)$, i.e. $f(x)$ é monotonicamente decrescente neste intervalo. Logo, da Proposição 3.1.1, temos que existe um único zero da função neste intervalo.

E 3.1.3.

$k \approx 0{,}161228$

E 3.1.5.

Escolhendo o intervalo $[a, b] = [-1{,}841 - 10^{-3}, -1{,}841 + 10^{-3}]$, temos $f(a) \approx 5 \times 10^{-4} > 0$ e $f(b) \approx -1{,}2 \times 10^{-3} < 0$, i.e. $f(a) \cdot f(b) < 0$. Então, o teorema de Bolzano nos garante que o zero exato x^* de $f(x)$ está no intervalo (a, b). Logo, da escolha feita, $|-1{,}841 - x^*| < 10^{-3}$.

E 3.1.6. Basta aplicar as ideias da solução do Exercício 3.1.5.

E 3.2.2.

A primeira raiz se encontra no intervalo $(0{,}4, 0{,}5)$. A segunda raiz no intervalo $(1{,}7, 1{,}8)$. A terceira raiz se encontra no intervalo $(2{,}5, 2{,}6)$.

E 3.2.3.

1,390054; 1,8913954; 2,4895673; 3,1641544; 3,8965468

E 3.2.4.

$k\theta = \frac{lP}{2}\cos(\theta)$ com $\theta \in (0, \pi/2)$; 1,030.

E 3.2.5.

19; 23; 26; 0,567143; 1,745528; 3,385630

E 3.2.7.

a) 0,623; b) 0,559; c) 0,500; d) 0,300; e) $-0{,}3$; f) -30; g) -30

E 3.2.8.

a) 0,0294; b) $2.44e-3$; c) $2.50e-4$; d) $1.09 \cdot 10^{-7}$; e) -10^{-12}; f) -10^{-12}; g) -10^{-12}

E 3.3.1.

$-1{,}8414057$

E 3.3.2.

0,7391

E 3.3.3.

Tomemos $x^{(1)} = 1$ como aproximação inicial para a solução deste problema, iterando a primeira sequência a), obtemos:

$$\begin{aligned}
x^{(1)} &= 1 \\
x^{(2)} &= \ln\left(\frac{10}{1}\right) = 2{,}3025851 \\
x^{(3)} &= \ln\left(\frac{10}{2{,}3025851}\right) = 1{,}4685526 \\
&\vdots \\
x^{(21)} &= 1{,}7455151 \\
x^{(31)} &= 1{,}745528 \\
x^{(32)} &= 1{,}745528
\end{aligned}$$

Iterando a segunda sequência b), obtemos:

$$\begin{aligned} x^{(1)} &= 1 \\ x^{(2)} &= 10e^{-1} = 3{,}6787944 \\ x^{(3)} &= 10e^{-3{,}6787944} = 0{,}2525340 \\ x^{(4)} &= 10e^{-0{,}2525340} = 7{,}7682979 \\ x^{(5)} &= 10e^{-7{,}7682979} = 0{,}0042293 \\ x^{(6)} &= 10e^{-0{,}0042293} = 9{,}9577961 \end{aligned}$$

Este experimento numérico sugere que a iteração a) converge para 1,745528 e a iteração b) não é convergente.

E 3.3.9.
$x > a$ com $a \approx 0{,}4193648$.

E 3.3.12.
0.0431266

E 3.4.2.
0.0198679; 0.533890; 0.735412; 1.13237; 1.38851.

E 3.4.4.
-99.99970, -0.3376513; -1.314006.

E 3.4.7.
$x_0 > 1$.

E 3.6.5.
Seja $f(x) \in C^2$ um função tal que $f(x^*) = 0$ e $f'(x^*) \neq 0$. Considere o processo iterativo do método das secantes:

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f(x^{(n)}) - f(x^{(n-1)})}(x^{(n)} - x^{(n-1)})$$

Esta expressão pode ser escrita como:

$$\begin{aligned} x^{(n+1)} &= x^{(n)} - \frac{f(x^{(n)})(x^{(n)} - x^{(n-1)})}{f(x^{(n)}) - f(x^{(n-1)})} \\ &= \frac{x^{(n)}\left(f(x^{(n)}) - f(x^{(n-1)})\right) - f(x^{(n)})(x^{(n)} - x^{(n-1)})}{f(x^{(n)}) - f(x^{(n-1)})} \\ &= \frac{x^{(n)} f(x^{(n-1)}) - x^{(n-1)} f(x^{(n)})}{f(x^{(n)}) - f(x^{(n-1)})} \end{aligned}$$

Subtraindo x^* de ambos os lados temos:

$$\begin{aligned} x^{(n+1)} - x^* &= \frac{x^{(n)} f(x^{(n-1)}) - x^{(n-1)} f(x^{(n)})}{f(x^{(n)}) - f(x^{(n-1)})} - x^* \\ &= \frac{x^{(n)} f(x^{(n-1)}) - x^{(n-1)} f(x^{(n)}) - x^*\left(f(x^{(n)}) - f(x^{(n-1)})\right)}{f(x^{(n)}) - f(x^{(n-1)})} \\ &= \frac{(x^{(n)} - x^*) f(x^{(n-1)}) - (x^{(n-1)} - x^*) f(x^{(n)})}{f(x^{(n)}) - f(x^{(n-1)})} \end{aligned}$$

Definimos $\epsilon_n = x_n - x^*$, equivalente a $x_n = x^* + \epsilon_n$

$$\epsilon_{n+1} = \frac{\epsilon_n f(x^* + \epsilon_{n-1}) - \epsilon_{n-1} f(x^* + \epsilon_n)}{f(x^* + \epsilon_n) - f(x^* + \epsilon_{n-1})}$$

Aproximamos a função $f(x)$ no numerador por

$$\begin{aligned} f(x^* + \epsilon) &\approx f(x^*) + \epsilon f'(x^*) + \epsilon^2 \frac{f''(x^*)}{2} \\ f(x^* + \epsilon) &\approx \epsilon f'(x^*) + \epsilon^2 \frac{f''(x^*)}{2} \end{aligned}$$

$$\begin{aligned}\epsilon_{n+1} &\approx \frac{\epsilon_n\left[\epsilon_{n-1}f'(x^*)+\epsilon_{n-1}^2\frac{f''(x^*)}{2}\right]-\epsilon_{n-1}\left[\epsilon_n f'(x^*)+\epsilon_n^2\frac{f''(x^*)}{2}\right]}{f(x^*+\epsilon_n)-f(x^*+\epsilon_{n-1})}\\ &= \frac{\frac{f''(x^*)}{2}\left(\epsilon_n\epsilon_{n-1}^2-\epsilon_{n-1}\epsilon_n^2\right)}{f(x^*+\epsilon_n)-f(x^*+\epsilon_{n-1})}\\ &= \frac{1}{2}f''(x^*)\frac{\epsilon_n\epsilon_{n-1}\left(\epsilon_{n-1}-\epsilon_n\right)}{f(x^*+\epsilon_n)-f(x^*+\epsilon_{n-1})}\end{aligned}$$

Observamos, agora, que

$$\begin{aligned}f(x^*+\epsilon_n)-f(x^*+\epsilon_{n-1}) &\approx \left[f(x^*)+f'(x^*)\epsilon_n\right]-\left[f(x^*)+f'(x^*)\epsilon_{n-1}\right]\\ &= f'(x^*)(\epsilon_n-\epsilon_{n-1})\end{aligned} \tag{3.7}$$

Portanto:

$$\epsilon_{n+1} \approx \frac{1}{2}\frac{f''(x^*)}{f'(x^*)}\epsilon_n\epsilon_{n-1} \tag{3.8}$$

ou, equivalentemente:

$$x^{(n+1)}-x^* \approx \frac{1}{2}\frac{f''(x^*)}{f'(x^*)}\left(x^{(n)}-x^*\right)\left(x^{(n-1)}-x^*\right) \tag{3.9}$$

E 3.7.1.
$z_1 \approx 0.3252768$, $z_2 \approx 1.5153738$, $z_3 \approx 2.497846$, $z_4 \approx 3.5002901$, $z_j \approx j-1/2-(-1)^j\frac{e^{-2j+1}}{\pi}$, $\quad j>4$

E 3.7.2.
150 W, 133 W, 87 W, 55 W, 6,5 W

E 3.7.3.
a) 42 s e 8 min2 s, b) 14 min56 s.

E 3.7.4.
118940992

E 3.7.5.
7,7 cm

E 3.7.6.
4,32 cm

E 3.7.7.
(0,652919, 0,426303)

E 3.7.8.
7,19% ao mês

E 3.7.9.
4,54% ao mês.

E 3.7.10.
500 K, 700 K em $t = 3\ln(2)$, 26 min, 4 h27 min.

E 3.7.11.
$(\pm 1{,}1101388, -0{,}7675919)$, $(\pm 1{,}5602111, 0{,}342585)$

E 3.7.12.
1,5318075

E 3.7.13.

Aproximadamente 2500 reais por hora.

E 3.7.14.

a) 332,74 K b) 359,33 K

E 3.7.15.

1,2285751, 4,76770758, 7,88704085

E 4.1.1.

Escrevemos o sistema na forma matricial e resolvemos:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 1 & 0 & 10 & -48 \\ 0 & 10 & 1 & 25 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 0 & -1 & 9 & -48 \\ 0 & 10 & 1 & 25 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 0 & 10 & 1 & 25 \\ 0 & -1 & 9 & -48 \end{array}\right] \sim$$

$$\sim \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 0 & 10 & 1 & 25 \\ 0 & 0 & 9.1 & -45.5 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 0 & 10 & 1 & 25 \\ 0 & 0 & 1 & -5 \end{array}\right] \sim$$

$$\sim \left[\begin{array}{ccc|c} 1 & 1 & 0 & 5 \\ 0 & 10 & 0 & 30 \\ 0 & 0 & 1 & -5 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 1 & 0 & 5 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -5 \end{array}\right] \sim$$

$$\sim \left[\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -5 \end{array}\right]$$

Portanto $x = 2$, $y = 3$, $z = -5$

E 4.5.1.

$\lambda = \frac{71\times 30}{41} \approx 51.95122$, para $\lambda = 51$: $k_1 = k_\infty = 350.4$, $k_2 = 262.1$. Para $\lambda = 52$: $k_1 = k_\infty = 6888$, $k_2 = 5163$.

E 4.5.2.

$k_1(A) = 36$, $k_2(A) = 18{,}26$, $K_\infty(A) = 20{,}8$.

E 4.5.3.

$k_1 = k_\infty = 6888, k_2 = \sqrt{26656567}$ e $k_1 = 180, k_2 = 128{,}40972$ e $k_\infty = 210$

E 4.5.4.

$\frac{18}{\varepsilon} + 3$. Quando $\varepsilon \to 0+$, a matriz converge para uma matriz singular e o número de condicionamento diverge para $+\infty$.

E 4.5.5.

As soluções são $[-0.0000990 \quad 0.0000098]^T$ e $[0.0098029 \quad 0.0990294]^T$. A grande variação na solução em função de pequena variação nos dados é devido ao mau condicionamento da matriz ($k_1 \approx 1186274.3$).

Exemplo de implementação:

```
A=[1e5 -1e4+1e-2; -1e4+1e-2 1000.1]
b1=[-10 1]'
b2=[-9.999 1.01]'
A\b1
A\b2
```

E 4.5.6.

0,695; 0,292; 0,188; 0,0237; 0,0123; 0,00967

E 4.6.1.

E 4.6.4.

0.324295, 0.324295, 0.317115, 0.305943, 0.291539, 0.274169, 0.253971, 0.230846, 0.203551, 0.165301, 0.082650

E 4.6.6.

Permute as linhas 1 e 2.

E 4.7.1.

$\lambda = 86.1785$ associado ao autovetor dado por $v_1 = [0.65968 \quad 0.66834 \quad 0.34372]^T$.

E 4.7.3.

158,726

E 4.8.2.

Dica: $P(-1) = -3$, $P(1) = -1$ e $P(2) = 9$ produzem três equações lineares para os coeficientes a, b e c. Resp: a) $P(x) = 3x^2 + x - 5$, b) $A \approx 2.49$ e $B \approx -1.29$ c)$A_1 \approx 1.2872058$, $A_2 \approx -4.3033034$, $B_1 \approx 2.051533$ e $B_2 \approx -0.9046921$.

E 6.1.1.

$p(x) = -3 + 2x + 5x^3$.

E 6.1.2.

$p(x) = 0{,}25 + x^2$.

E 6.1.3.

a) Uma parábola de equação $y = a_1 + a_2x + a_3x^2$ que interpola os pontos deve satisfazer o sistema:

$$\begin{aligned} a_1 + a_2x_1 + a_3x_1^2 &= y_1 \\ a_1 + a_2x_2 + a_3x_2^2 &= y_2 \end{aligned}$$

Sem perda de generalidade, para cada $a_3 \in \mathbb{R}$ dado, temos:

$$\begin{aligned} a_1 + a_2x_1 &= y_1 - a_3x_1^2 \\ a_1 + a_2x_2 &= y_2 - a_3x_2^2, \end{aligned}$$

o qual tem solução única, pois $x_1 \neq x_2$. Ou seja, para cada $a_3 \in \mathbb{R}$ dado, existem $a_1, a_2 \in \mathbb{R}$ tais que a parábola de equação $y = a_1 + a_2x + a_3x^2$ interpola os pontos dados.

b) Certamente não existem retas de equação $x = a$ que interpolam os pontos dados. Consideremos então retas de equação $y = a_1 + a_2x$. Para uma tal reta interpolar os pontos dados é necessário que:

$$\begin{aligned} a_1 + a_2 &= 1 \\ a_1 + 2a_2 &= 2{,}1, \\ a_1 + 3a_2 &= 3 \end{aligned}$$

o qual é um sistema impossível.

c) Não existe uma parábola de equação $y = a_1 + a_2x + a_3x^2$ que interpole os pontos dados, pois tal equação determina uma função de x em y. Agora, para mostrar que existem infinitas parábolas de equação $x = a_1 + a_2y + a_3y^2$ que interpolam os pontos dados, basta seguir um raciocínio análogo ao do item a), trocando x por y e y por x.

E 6.4.1.

$\int_0^1 P(x)dx = \frac{f(0)+f(1)}{2}$, $\frac{1}{12}\max_{x\in[0,1]}|f''(x)|$

E 7.1.1.

$f(x) = -0{,}55 - 0{,}01x$.

E 7.1.2.

$f(x) = 0{,}19 - 0{,}47x$.

E 7.1.3.

a) $-0{,}6025387$; b) $-0{,}5651848$; c) $0{,}2851848$; d) $0{,}1488041$.

E 7.1.4.

a) Basta observar que

$$V^T V = \begin{bmatrix} 1 & 1\cdots & 1 & \\ x_1 & x_2 & \cdots & x_N \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix} = \begin{bmatrix} N & \sum_{j=1}^N x_j \\ \sum_{j=1}^N x_j & \sum_{j=1}^N x_j^2 \end{bmatrix} = M$$

e

$$V^T y = \begin{bmatrix} 1 & 1\cdots & 1 & \\ x_1 & x_2 & \cdots & x_N \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^N y_j \\ \sum_{j=1}^N x_j y_j \end{bmatrix} = w.$$

b) Sejam $x_i \neq x_j$ duas abscissas diferentes. Então, a i-ésima e j-ésima linhas na matriz V são linearmente independentes e, portanto, o posto de V é igual a 2. Por fim, $V^T V$ é não singular, pois, se u é tal que $V^T V u = 0$, então

$$0 = u^T V^T V u = (Vu)^T (Vu) = (Vu) \cdot (Vu) \Rightarrow Vu = 0.$$

Agora, $Vu = 0$ é uma combinação linear das linhas de V igual a zero, logo $u = 0$, pois as linhas de V são linearmente independentes como mostrado antes. Concluímos que se $V^T V u = 0$, então $u = 0$, i.e. $V^T V$ é não singular.

E 7.2.1.

$a_1 = -0{,}1946029$, $a_2 = 0{,}585986$, $a_3 = -0{,}0112599$.

E 7.2.2.

$y = -0{,}0407898x^2 + 2{,}6613293x + 1{,}9364598$.

E 7.2.3.

a) $a = 25{,}638625$, $b = 9{,}8591874$, $c = 4{,}9751219$; b)$a = 31{,}475524$, $b = 65{,}691531$, $c = -272{,}84382$, $d = 208{,}23621$.

E 8.1.3.

a) $f'(0) = \frac{-3f(0)+4f(h)-f(2h)}{2h} + O(h^2)$

b) $f'(0) = \frac{3f(0)-4f(-h)+f(-2h)}{2h} + O(h^2)$

c) $f'(0) = \frac{1}{h_1+h_2} l \left[-\frac{h_2}{h_1} f(-h_1) + \left(\frac{h_2}{h_1} - \frac{h_1}{h_2} \right) f(0) + \frac{h_1}{h_2} f(h_2) \right]$

E 8.1.4.

$Caso$	a	b	c	d
$v_i = 1$	1.72	1.56	1.64	1.86
$v_i = 4.5$	2.46	1.90	2.18	1.14

E 8.1.5.

Segue a tabela com os valores da derivada para vários valores de h.

h	10^{-2}	10^{-4}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
$D_{+,h}f(1{,}5)$	$-0{,}3125246$	$-0{,}3161608$	$-0{,}3161973$	$-0{,}3161976$	$-0{,}3161977$	$-0{,}3161977$

h	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}	10^{-15}
$D_{+,h}f(1{,}5)$	$-0{,}3161976$	$-0{,}3161971$	$-0{,}3162332$	$-0{,}3158585$	$-0{,}3178013$	$-0{,}3747003$

Observe que o valor exato é $-0{,}3161977$ e o h ótimo é algo entre 10^{-8} e 10^{-9}.

E 8.3.1.

a) $f''(0) = \frac{f(0)-2f(h)+f(2h)}{h^2} + O(h)$

b) $f''(0) = \frac{f(0)-2f(-h)+f(-2h)}{h^2} + O(h)$

E 9.1.2.

$$I_{Simpson} = \frac{1}{3} I_{Trap} + \frac{2}{3} I_{PM}$$

E 9.1.3.

n	Ponto médio	Trapézios	Simpson
3	0.1056606	0.7503919	0.5005225
5	0.1726140	0.3964724	0.2784992
7	0.1973663	0.3062023	0.2393551
9	0.2084204	0.2721145	0.2306618

E 9.3.2.

-0.2310491, -0.2452073, - 0.2478649.

E 9.3.4.

a)-0.2472261, -0.2416451, -0.2404596, -0.2400968, -0.2399563, -0.2398928. b)-0.2393727, -0.2397994, -0.2398104, -0.2398115, -0.2398117, -0.2398117.

E 9.4.1.

$$a) I(h) = 4.41041 \cdot 10^{-1} - 8.49372 \cdot 10^{-12} h - 1.22104 \cdot 10^{-2} h^2 - 1.22376 \cdot 10^{-7} h^3 + 8.14294 \cdot 10^{-3} h^4$$

$$b) I(h) = 7.85398 \cdot 10^{-1} - 1.46294 \cdot 10^{-11} h - 4.16667 \cdot 10^{-2} h^2 - 2.16110 \cdot 10^{-7} h^3 + 4.65117 \cdot 10^{-6} h^4$$

$$c) I(h) = 1.58730 \cdot 10^{-3} - 9.68958 \cdot 10^{-10} h + 2.03315 \cdot 10^{-7} h^2 - 1.38695 \cdot 10^{-5} h^3 + 2.97262 \cdot 10^{-4} h^4$$

$$d) I(h) = 4.61917 \cdot 10^{-1} + 3.83229 \cdot 10^{-12} h + 2.52721 \cdot 10^{-2} h^2 + 5.48935 \cdot 10^{-8} h^3 + 5.25326 \cdot 10^{-4} h^4$$

E 9.4.2.

1.5707963	2.0943951		
1.8961189	2.0045598	1.9985707	
1.9742316	2.0002692	1.9999831	2.0000055

E 9.4.3.

0.7468337,2.4606311, 1.6595275.

E 9.4.5.

$R(6,6) = -10.772065$, $R(7,7) = 5.2677002$, $R(8,8) = 6.1884951$, $R(9,9) = 6.0554327$, $R(10,10) = 6.0574643$. O valor desta integral com oito dígitos corretos é aproximado por 6.0574613.

E 9.5.1.

$w_1 = 1/6$, $w_2 = 2/3$, $w_3 = 1/6$. O esquema construído é o de Simpson e a ordem de exatidão é 3.

E 9.5.2.

3

E 9.5.3.

5

E 9.5.4.

$\int_0^1 f(x)dx \approx \frac{3}{2}f(1/3) - 2f(1/2) + \frac{3}{2}f(2/3)$ com ordem 3.

E 9.5.5.

5, 4, 3

E 9.7.1.

n	b	c	d	e	f
2	2.205508	3.5733599	3.6191866	3.6185185	3.618146
4	2.5973554	3.6107456	3.6181465	3.6180970	3.6180970
6	2.7732372	3.6153069	3.6181044	3.6180970	3.6180970
8	2.880694	3.6166953	3.6180989	3.6180970	3.6180970

Solução do item e: Como

$$\cos(x) = 1 + \sum_{n=1}^{\infty}(-1)^n \frac{x^{2n}}{(2n)!}$$

temos

$$\frac{1-\cos(x)}{\sqrt{x}} = -\sum_{n=1}^{\infty}(-1)^n \frac{x^{2n-1/2}}{(2n)!}, \quad x \geq 0$$

Logo, podemos integrar

$$\begin{aligned} I &= 4 + 2\int_0^1 \frac{\cos(x)-1}{\sqrt{|x|}}dx = 4 - 2\sum_{n=1}^{\infty}(-1)^n \int_0^1 \frac{x^{2n-1/2}}{(2n)!}dx \\ &= 4 - 2\sum_{n=1}^{\infty}(-1)^n \frac{1}{(2n)!(2n+1/2)} \end{aligned}$$

Solução do item f)

$$2\int_0^1 \left(x^{-1/2} - \frac{x^{3/2}}{2} + \frac{x^{7/2}}{24}\right)dx = 2\left(2 - \frac{1}{5} + \frac{1}{54}\right) = \frac{977}{270}$$

$$2\int_0^1 \frac{\cos(x) - P_4(x)}{\sqrt{x}}dx = \sqrt{2}\int_{-1}^1 \frac{\cos\left(\frac{1+u}{2}\right) - P_4\left(\frac{1+u}{2}\right)}{\sqrt{1+u}}du$$

E 9.7.5.

4.1138

E 9.7.6.

a)19.2, 22.1, 23.3 b)513.67K

E 9.7.8.

$\int_{-1}^{1} f(x)dx = f\left(-\frac{\sqrt{3}}{3}\right) + f\left(\frac{\sqrt{3}}{3}\right)$

E 9.7.9.

$w_1 = w_3 = 1$ e $w_2 = 0$ com ordem 3.

E 10.1.1.

0,4496 com $h = 0{,}1$ e 0,4660 com $h = 0{,}01$. A solução exata vale $y(1) = \frac{1+2e^{-1}+e^{-2}}{4} = \left(\frac{1+e^{-1}}{2}\right)^2 \approx 0{,}4678$

E 10.1.2.

$y(2) \approx 0{,}430202$ e $z(2) = 0{,}617294$ com $h = 0{,}2$, $y(2) \approx 0{,}435506$ e $z(2) = 0{,}645776$ com $h = 0{,}02$, $y(2) \approx 0{,}435805$ e $z(2) = 0{,}648638$ com $h = 0{,}002$ e $y(2) \approx 0{,}435832$ e $z(2) = 0{,}648925$ com $h = 0{,}0002$.

E 10.1.3.

$y(2) \approx 1{,}161793$ com $h = 0{,}1$, $y(2) \approx 1{,}139573$ com $h = 0{,}01$, $y(2) \approx 1{,}137448$ com $h = 0{,}001$, $y(2) \approx 1{,}137237$ com $h = 0{,}0001$, $y(2) \approx 1{,}137216$ com $h = 0{,}00001$

E 10.2.1.

$y(1) \approx 1{,}317078$ quando $h = 0{,}1$ e $y(1) \approx 1{,}317045$.

E 10.2.2.

t	Exato	Euler	Euler Melhorado	Erro Euler	Erro Euler Melhorado
0.0	1.	1.	1.	0.	0.
0.1	0.826213	0.8	0.828	0.026213	0.001787
0.2	0.693094	0.656	0.695597	0.037094	0.002502
0.3	0.588333	0.547366	0.591057	0.040967	0.002724
0.4	0.504121	0.462669	0.506835	0.041453	0.002714
0.5	0.435267	0.394996	0.437861	0.040271	0.002594
0.6	0.378181	0.339894	0.380609	0.038287	0.002428
0.7	0.330305	0.294352	0.332551	0.035953	0.002246
0.8	0.289764	0.256252	0.291828	0.033512	0.002064
0.9	0.255154	0.224061	0.257043	0.031093	0.001889
1.0	0.225400	0.196634	0.227126	0.028766	0.001726

E 10.9.1.

Os valores exatos para os itens e e f são: $\frac{1}{10}\ln\left(\frac{9}{4}\right)$ e $\frac{1}{10}\ln(6)$

E 10.9.2.

O valor exato é $\sqrt{\frac{g}{\alpha}\left[1-e^{-200\alpha}\right]}$ em $t = \frac{1}{\sqrt{g\alpha}}\tanh^{-1}\left(\sqrt{1-e^{-200\alpha}}\right)$

E 10.9.8.

	0.5	1.0	1.5	2.0	2.5
Analítico	0.3032653	0.3678794	0.3346952	0.2706706	0.2052125
Euler	0.3315955	0.3969266	0.3563684	0.2844209	0.2128243
Euler modificado	0.3025634	0.3671929	0.3342207	0.2704083	0.2051058
Runge-Kutta Clássico	0.3032649	0.3678790	0.3346949	0.2706703	0.2052124
Adams-Bashforth ordem 4	0.3032421	0.3678319	0.3346486	0.2706329	0.2051848

	0.5	1.0	1.5	2.0	2.5
Euler	2.8D-02	2.9D-02	2.2D-02	1.4D-02	7.6D-03
Euler modificado	7.0D-04	6.9D-04	4.7D-04	2.6D-04	1.1D-04
Runge-Kutta Clássico	4.6D-07	4.7D-07	3.5D-07	2.2D-07	1.2D-07
Adams-Bashforth ordem 4	2.3D-05	4.8D-05	4.7D-05	3.8D-05	2.8D-05

	0.1	0.05	0.01	0.005	0.001
Euler	2.9D-02	5.6D-03	2.8D-03	5.5D-04	2.8D-04
Euler modificado	6.9D-04	2.5D-05	6.2D-06	2.5D-07	6.1D-08
Runge-Kutta Clássico	4.7D-07	6.9D-10	4.3D-11	6.8D-14	4.4D-15
Adams-Bashforth ordem 4	4.8D-05	9.0D-08	5.7D-09	9.2D-12	5.8D-13

E 11.1.1.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \\ 2 \\ 2 \\ 10 \end{bmatrix}$$

Solução: [5, 9.25, 11.5, 11.75, 10]

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{bmatrix} = \begin{bmatrix} 5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 10 \end{bmatrix}$$

Solução: $[5, 7.375, 9.25, 10.625, 11.5, 11.875, 11.75, 1.125, 10]$

E 11.1.2. 120. 133.56 146.22 157.83 168.22 177.21 184.65 190.38 194.28 196.26 196.26 194.26 190.28 184.38 176.65 167.21 156.22 143.83 130.22 115.56 100.

E 11.1.3. 391.13 391.13 390.24 388.29 385.12 380.56 374.44 366.61 356.95 345.38 331.82 316.27 298.73 279.27 257.99 234.99 210.45 184.5 157.34 129.11 100.

E 11.1.4. 0., 6.57, 12.14, 16.73, 20.4, 23.24, 25.38, 26.93 , 28, 28.7, 29.06, 29.15, 28.95, 28.46, 27.62 , 26.36, 24.59, 22.18, 19.02, 14.98, 10.

E 11.1.5. u(0)=31.62, u(1)=31.50, u(1.9)=18.17

E 11.1.6. u(1)=1.900362, u(2.5)=1.943681, u(4)=1.456517

Referências Bibliográficas

[1] Cecill and free sofware. http://www.cecill.info. Acessado em 30 de julho de 2015.

[2] M. Baudin. Introduction to scilab. http://forge.scilab.org/index.php/p/docintrotoscilab/. Acessado em 30 de julho de 2015.

[3] R.L. Burden and J.D. Faires. *Análise Numérica*. Cengage Learning, 8 edition, 2013.

[4] J. P. Demailly. *Analyse Numérique et Équations Differentielles*. EDP Sciences, Grenoble, nouvelle Édition edition, 2006.

[5] W Gautschi. Numerical analysis: An introduction birkhauser. *Barton, Mass, USA*, 1997.

[6] Walter Gautschi and Gabriele Inglese. Lower bounds for the condition number of vandermonde matrices. *Numerische Mathematik*, 52(3):241–250, 1987/1988.

[7] L.F. Guidi. Notas da disciplina cálculo numérico. http://www.mat.ufrgs.br/~guidi/grad/MAT01169/calculo_numerico.pdf. Acessado em julho de 2016.

[8] E. Isaacson and H.B. Keller. *Analysis of numerical methods*. Dover, Ontário, 1994.

[9] W.H. Press. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007.

[10] R. Rannacher. Einführung in die numerische mathematik (numerik 0). http://numerik.uni-hd.de/~lehre/notes/num0/numerik0.pdf. Acessado em 10.08.2014.

[11] Todos os Colaboradores. Cálculo nuérico - um lirvo colaborativo - versão com scilab. disponível em https://www.ufrgs.br/numerico/livro/main.html, Novembro 2016.

Colaboradores

Aqui você encontra a lista de colaboradores do livro. Esta lista contém somente aqueles que explicitamente se manifestaram a favor de terem seus nomes registrados aqui. A lista completa de colaborações pode ser obtida no repositório GitHub do livro:

https://github.com/livroscolaborativos/CalculoNumerico

Além das colaborações via GitHub, o livro também recebe colaborações via discussões, sugestõese avisos deixados em nossa lista de emails:

livro_colaborativo@googlegroups.com

Estas colaborações não estão listadas aqui, mas podem ser vistas no site do grupo de emails.

Caso encontre algum equívoco ou veja seu nome listado aqui por engano, por favor, entre em contato conosco por email:

livroscolaborativos@gmail.com

ou via o repositório GitHub.

Tabela A.1: Lista de colaboradores

Nome	Afiliação	E-Mail	1ª Contribuição
Debora Lidia Gisch	-x-	-x-	#63

Índice Remissivo