

Meu Cálculo Numérico

Um livro colaborativo

Arquimedes Macedo REAMAT

31/12/2024

Índice

Bem-Vindo	3
Licença	3
Prefácio	4
Linguagens Computacionais	4
0.0.1 Python	4
0.0.2 Octave	4
0.0.3 Scilab	4
Introdução	5
I Capítulo 1: Representação de números e aritmética de máquina	7
1 Sistema de numeração e mudança de base	9
1.0.1 Python	10
1.0.2 Scilab	10
1.0.3 Octave	10
1.1 Exercícios	10
1.1.1 Exercício	10
1.1.2 Exercício	10
Apêndices	12
A Rápida introdução ao Python	12
A.1 Sobre a linguagem Python	12
A.1.1 Instalação e execução	12
A.1.2 Usando Python	12
A.1.3 Exercício	13

Bem-Vindo

REAMAT é um projeto de escrita colaborativa de recursos educacionais abertos (REA) sobre tópicos de matemática e suas aplicações.

Nosso objetivo é de fomentar o desenvolvimento de materiais didáticos pela colaboração entre professores e alunos de universidades, institutos de educação e demais interessados no estudo e na aplicação da matemática nos mais diversos ramos da ciência e da tecnologia.

O sucesso do projeto depende da colaboração! Participe diretamente da escrita dos recursos educacionais, dê sugestões ou avise-nos de erros e imprecisões. Toda a colaboração é bem-vinda. Veja como participar aqui.

Nós preparamos uma série de ações para ajudá-lo a participar. Em primeiro lugar, o acesso irrestrito aos materiais pode se dar através deste site.

i Os códigos fontes e a documentação dos recursos estão disponíveis em repositórios GitHub públicos.

Licença

Nada disso estaria completo sem uma licença apropriada à colaboração. Por isso, escolhemos disponibilizar os materiais 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 mais informações.

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. Conhecimento prévio em linguagem de computadores é fortemente recomendável, embora apenas técnicas elementares de programação sejam realmente necessárias.

Os códigos computacionais dos métodos numéricos apresentados no livro são implementados em uma abordagem didática. Isto é, temos o objetivo de que a implementação em linguagem computacional auxilie o leitor no aprendizado das técnicas numéricas apresentadas no livro. Implementações computacionais eficientes de técnicas de cálculo numérico podem ser obtidas na série de livros “Numerical Recipes”, veja ([Press 2007](#)).

Linguagens Computacionais

0.0.1 Python

A utilização da linguagem computacional Python¹.

¹ <https://www.python.org/>

0.0.2 Octave

A utilização da linguagem computacional GNU Octave².

² <https://www.gnu.org/software/octave/>

0.0.3 Scilab

A utilização do software livre Scilab³.

³ <https://www.scilab.org/>

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 desempenho.

Aliado ao aumento contínuo da capacidade de computação disponível, o desenvolvimento de métodos numéricos tornou a simulação computacional de problemas 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 de equações lineares, interpolar e ajustar pontos, calcular derivadas e integrais, resolver equações diferenciais ordinárias, etc. Neste livro, abordamos o desenvolvimento, a implementação, a utilização e os aspectos teóricos de métodos numéricos para a resolução desses problemas.

Trabalharemos com problemas que abordam aspectos teóricos e de utilização dos métodos estudados, bem como com 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:

$$\cos(x) = x \quad \text{ou} \quad xe^x = 10 \quad (0.1)$$

Para resolver o problema de valor inicial:

$$\begin{aligned}y' + xy &= x, \\ y(0) &= 2,\end{aligned}\tag{0.2}$$

podemos usar o método de fator integrante e obtemos $y = 1 + e^{-x^2/2}$. No entanto, não parece possível encontrar uma expressão fechada em termos de funções elementares para a solução exata do problema de valor inicial dado por:

$$\begin{aligned}y' + xy &= e^{-y}, \\ y(0) &= 2,\end{aligned}\tag{0.3}$$

Da mesma forma, resolvemos a integral:

$$\int_1^2 xe^{-x^2} dx \tag{0.4}$$

pelo método da substituição e obtemos $\frac{1}{2}(e^{-1} - e^{-4})$. Porém a integral:

$$\int_1^2 e^{-x^2} dx \tag{0.5}$$

não pode ser expressa analiticamente em termos de funções elementares, como uma consequência do teorema de Liouville.

A maioria dos problemas envolvendo fenômenos reais produzem modelos matemáticos cuja solução analítica é difícil (ou impossível) de obter, mesmo quando provamos que a solução 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 como funciona a aritmética implementada nos computadores e erros de arredondamento. Como computadores, em geral, usam uma base binária para representar números, começaremos falando em mudança de base.

Part I

Capítulo 1: Representação de números e aritmética de máquina

Neste capítulo, abordaremos formas de representar números reais em computadores. Iniciamos com uma discussão sobre representação posicional e mudança de base. Então, enfatizaremos a representação de números com quantidade finita de dígitos, 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, abordaremos os efeitos do erro de arredondamento na computação científica.

1 Sistema de numeração e mudança de base

Usualmente, utilizamos o sistema de numeração decimal, isto é, base 10, para representar números. Esse é um sistema de numeração em que a posição do algarismo indica a potência de 10 pela qual seu valor é multiplicado.

Exemplo

O número 293 é decomposto como:

$$\begin{aligned} 293 &= 2 \text{ centenas} + 9 \text{ dezenas} + 3 \text{ unidades} \\ &= 2 \cdot 10^2 + 9 \cdot 10^1 + 3 \cdot 10^0 \end{aligned} \quad (1.1)$$

293 é igual a 2 centenas mais 9 dezenas mais 3 unidades. Que também é igual a 2 vezes 10 elevado à potência 2, mais 9 vezes 10 elevado à potência 1, mais 3 vezes 10 elevado à potência 0

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

Sistema de numeração de base b

Dado um número natural $b > 1$ e o conjunto de símbolos $\pm, 0, 1, 2, \dots, b-1$ ¹, a sequência de símbolos

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

representa o número positivo

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

Para representar números negativos usamos o símbolo $-$ à esquerda do numeral².

Nota 1: Observação

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,

¹Para $b > 10$, veja Tip 1.

²O uso do símbolo $+$ é opcional na representação de números positivos.

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

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

1.0.1 Python

```
>>> 1*2**3 + 0*2**2 + 0*2**1 + 1*2**0 + 1*2**-1 + 0*2**-2 + 1*2**-3
9.625
```

1.0.2 Scilab

```
--> 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0 + 1*2^-1 + 0*2^-2 + 1*2^-3
ans = 9.6250
```

1.0.3 Octave

```
>> 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0 + 1*2^-1 + 0*2^-2 + 1*2^-3
ans = 9.6250
```

1.1 Exercícios

1.1.1 Exercício

Escreva cada número dado para a base b .

- $(45, 1)_8$ para a base $b = 2$
- $(21, 2)_8$ para a base $b = 16$
- $(1001, 101)_2$ para a base $b = 8$
- $(1001, 101)_2$ para a base $b = 16$

Resposta

- $\sim (100101, 001)_2$
- $\sim (11, 4)_{16}$
- $\sim (11, 5)_8$
- $\sim (9, A)_{16}$

1.1.2 Exercício

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?



Resposta

50; 18.

A Rápida introdução ao Python

Neste apêndice, discutiremos os principais aspectos da linguagem computacional Python que são essenciais para uma boa leitura desta versão do livro. O material aqui apresentado, é uma adaptação livre do Apêndice A de (Todos os Colaboradores 2016).

A.1 Sobre a linguagem Python

Python⁴ é uma linguagem de programação de alto nível, interpretada e multi-paradigma. Lançada por Guido van Rossum^{5,6} 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 gratuito no site Codecademy⁷:

<https://www.codecademy.com/learn/python>

⁴ <https://www.python.org/>

⁵ <https://gvanrossum.github.io/>

⁶ Guido van Rossum, nascido em 1956, programador de computadores dos Países Baixos.

⁷ <https://www.codecademy.com/>

A.1.1 Instalação e execução

Para executar um código Python é necessário ter instalado um interpretador para a linguagem. No site oficial do Python⁸ estão disponíveis para *download* os interpretadores 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 esteja usando um computador rodando Linux. Para outros sistemas, pode ser necessário fazer algumas adaptações.

⁸ <https://www.python.org/>

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;

A.1.3 Exercício

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`.
- c) digitando seu código em um arquivo separado e executando-o em terminal com o comando Python.

Resposta

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`. No terminal, digite:

```
$ python ola.py
```

- Press, W. H. 2007. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press. <https://books.google.com.br/books?id=1aAOdzK3FegC>.
- Todos os Colaboradores. 2016. “Cálculo Numérico - Um Livro Colaborativo - Versão Com Scilab.”