Meu Cálculo Numérico

Um livro colaborativo

Arquimedes Macedo REAMAT

28 December 2024

Table of contents

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 materias pode se dar através deste site.

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

Licença

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 Python.

0.0.2 Octave

A utilização da linguagem computacional GNU Octave.

0.0.3 Scilab

A utilização do software livre Scilab.

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:

$$\cos(x) = x \quad \text{ou} \quad xe^x = 10 \tag{1}$$

Para resolver o problema de valor inicial:

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

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:

$$\begin{aligned} y' + xy &= e^{-y}, \\ y(0) &= 2. \end{aligned} \tag{3}$$

Da mesma forma, resolvemos a integral:

$$\int_1^2 xe^{-x^2} dx \tag{4}$$

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

$$\int_1^2 e^{-x^2} dx \tag{5}$$

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.

Chapter 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:

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

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

1.0.1 Sistema de numeração de base b

Dado um número natural $b > 1$ e o conjunto de símbolos $\pm, \mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots, \boldsymbol{b-1}$ [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 $-$ à esquerda do numeral[2].

[1] Para $b > 10$, veja Tip 1.

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

Note 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,

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

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

1.0.2 Python

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

1.0.3 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
```

1.0.4 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.

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 $b = 16$

Resposta

a) $\sim(100101, 001)_2$
b) $\sim(11, 4)_{16}$
c) $\sim(11, 5)_8$
d) $\sim(9, A)_{16}$

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.

Appendix A

Installing R and RStudio

To get started with R, you need to acquire your own copy. This appendix will show you how to download R as well as RStudio, a software application that makes R easier to use. You'll go from downloading R to opening your first R session.

Both R and RStudio are free and easy to download.

> **Binaries Versus Source**
>
> R can be installed from precompiled binaries or built from source on any operating system. For Windows and Mac machines, installing R from binaries is extremely easy. The binary comes preloaded in its own installer. Although you can build R from source on these platforms, the process is much more complicated and won't provide much benefit for most users. For Linux systems, the opposite is true. Precompiled binaries can be found for some systems, but it is much more common to build R from source files when installing on Linux. The download pages on CRAN's website provide information about building R from source for the Windows, Mac, and Linux platforms.

A.0.1 Linux

R comes preinstalled on many Linux systems, but you'll want the newest version of R if yours is out of date. The CRAN website provides files to build R from source on Debian, Redhat, SUSE, and Ubuntu systems under the link "Download R for Linux." Click the link and then follow the directory trail to the version of Linux you wish to install on. The exact installation procedure will vary depending on the Linux system you use. CRAN guides the process by grouping each set of source files with documentation or README files that explain how to install on your system.

> **32-bit Versus 64-bit**
>
> R comes in both 32-bit and 64-bit versions. Which should you use? In most cases, it won't matter. Both versions use 32-bit integers, which means they compute

numbers to the same numerical precision. The difference occurs in the way each version manages memory. 64-bit R uses 64-bit memory pointers, and 32-bit R uses 32-bit memory pointers. This means 64-bit R has a larger memory space to use (and search through).
As a rule of thumb, 32-bit builds of R are faster than 64-bit builds, though not always. On the other hand, 64-bit builds can handle larger files and data sets with fewer memory management problems. In either version, the maximum allowable vector size tops out at around 2 billion elements. If your operating system doesn't support 64-bit programs, or your RAM is less than 4 GB, 32-bit R is for you. The Windows and Mac installers will automatically install both versions if your system supports 64-bit R.

A.1 Using R

R isn't a program that you can open and start using, like Microsoft Word or Internet Explorer. Instead, R is a computer language, like C, C++, or UNIX. You use R by writing commands in the R language and asking your computer to interpret them. In the old days, people ran R code in a UNIX terminal window—as if they were hackers in a movie from the 1980s. Now almost everyone uses R with an application called RStudio, and I recommend that you do, too.

R and UNIX

You can still run R in a UNIX or BASH window by typing the command:

```
$ R
```

which opens an R interpreter. You can then do your work and close the interpreter by running `q()` when you are finished.

Do I still need to download R?

Even if you use RStudio, you'll still need to download R to your computer. RStudio helps you use the version of R that lives on your computer, but it doesn't come with a version of R on its own.

A.2 Opening R

Now that you have both R and RStudio on your computer, you can begin using R by opening the RStudio program. Open RStudio just as you would any program, by clicking on its icon or by typing "RStudio" at the Windows Run prompt.

Press, W. H. 2007. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press. https://books.google.com.br/books?id=1aAOdzK3FegC.