MS211 - Cálculo Numérico

Apresentação da Disciplina, O Sistema de Ponto Flutuante

Introdução

Atualmente, os computadores fazem parte do nosso cotidiano.

Apesar dos computadores serem usados para uma grande variedade de tarefas, inicialmente eles foram concebidos para a resolução de problemas que surgem na ciência e na engenharia.

De fato, praticamente não existe uma área da ciência ou da engenharia que não usa computadores para encontrar uma solução de problemas matemáticos!

As técnicas usadas para encontrar tais soluções fazem parte da chamada **computação científica**.

Formalmente, **computação científica** refere-se a coleção de algoritmos, técnicas e teorias usadas para resolver num computador problemas da ciência e da engenharia.

Muitos dos algoritmos, técnicas e teorias da computação científica foram propostos bem antes dos primeiros computadores.

Contudo, com o avanço dos computadores eletrônicos, muitos métodos que foram desenvolvidos para o cálculo em papel e caneta precisaram ser revistos ou abandonados.

Muitas considerações irrelevantes para o cálculo manual tornaram-se indispensáveis em um grande sistema computacional.

O estudo dos algoritmos para resolver num computador modelos matemáticos (contínuos) define o que chamamos **análise numérica**.

O curso de cálculo numérico pode ser visto como uma introdução à **computação científica** e à **análise numérica**.

Nesse curso, veremos como a matemática e um pouco da ciência da computação são usadas para resolver problemas da ciência e da engenharia.

Lembre-se que um **computador** representa números reais na **base 2** enquanto uma **calculadora** representa números reais na **base 10**.

Representação de Números na Base β

Números Inteiros

Seja $x \in \mathbb{N} = \{1, 2, \ldots\}$ (note que para $0 \neq x \in \mathbb{Z} \backslash \mathbb{N}$ tem-se $-x \in \mathbb{N}$. Note que $\exists\ \{d_0, d_1, \ldots, d_n\} \subseteq \{0, 1, \ldots, \beta - 1\}$ tal que

$$x = d_0 + d_1\beta^1 + \ldots + d_n\beta^n = d_0 + \beta \cdot (d_1 + \beta \cdot (d_2 \ldots + \beta \cdot d_n) \ldots)$$

O lado direito facilita trocar a base.

Números Fracionários

Seja $x \in \mathbb{R}$ tal que $0 \leqslant x < 1$ (ou $0 \leqslant -x < 1$). Note que existe $\{a_1, a_2, \ldots, \} \subseteq \{0, 1, \ldots, \beta - 1\}$ tal que

$$x = a_1\beta^{-1} + a_2\beta^{-2} + \ldots = \beta^{-1} \cdot (a_1 + \beta^{-1} \cdot (a_2 + \ldots) \ldots))$$

O lado direito facilita trocar a base.

Conversão Sistema Decimal → Sistema Binário

Números Inteiros

Seja $x \in \mathbb{N} = \{1, 2, \ldots\}$. Segundo o slide anterior, $\exists$ $\{b_0, b_1, \ldots, b_n\} \subseteq \{0, 1\}$ tal que

$$x = b_0 + b_1 2^1 + \ldots + b_n 2^n = b_0 + 2 \cdot (b_1 + 2 \cdot (b_2 \ldots + 2 \cdot b_n) \ldots)$$

Portanto, $25 \in \mathbb{N}$ pode ser convertido como segue:

- $25 = \mathbf{1} + 2 \cdot 12 \Rightarrow b_0 = 1$;
- $12 = \mathbf{0} + 2 \cdot 6 \Rightarrow b_1 = 0$;
- $6 = \mathbf{0} + 2 \cdot 3 \Rightarrow b_2 = 0$;
- $3 = \mathbf{1} + 2 \cdot 1 \Rightarrow b_3 = 1$;
- $1 = \mathbf{1} + 2 \cdot 0 \Rightarrow b_4 = 1$.

Assim, $25 = (25)_{10} = (11001)_2$.

Conversão Sistema Decimal → Sistema Binário

Números Fracionários

Seja $x \in \mathbb{R}$ tal que $0 \leqslant x < 1$ (ou $0 \leqslant -x < 1$). Note que existe $\{b_1, b_2, \ldots,\} \subseteq \{0, 1\}$ tal que

$$x = b_1 2^{-1} + b_2 2^{-2} + \ldots = 2^{-1} \cdot (b_1 + 2^{-1} \cdot (b_2 + \ldots) \ldots))$$

Portanto, $0.2 = (0.2)_{10}$ pode ser convertido como segue:

- $2 \cdot 0.2 = \mathbf{0} + 0.4 \Rightarrow b_1 = 0$;
- $2 \cdot 0.4 = \mathbf{0} + 0.8 \Rightarrow b_2 = 0$;
- $2 \cdot 0.8 = \mathbf{1} + 0.6 \Rightarrow b_3 = 1$;
- $2 \cdot 0.6 = \mathbf{1} + 0.2 \Rightarrow b_4 = 1$.

Assim, $0.2 = (0.2)_{10} = (0.00110011\ldots)_2 = (0.\overline{0011})_2$.

O Sistema de Ponto Flutuante

Definição 1

Numa maquina um número $x \neq 0$ é representado na forma

$$\pm 0.a_1 a_2 a_3 \ldots a_t \cdot \beta^e,$$

em que

- β é a base;
- t é o número de dígitos na mantissa, com $a_1 \neq 0$ e $0 \leqslant a_i \leqslant \beta - 1$, para todo $i = 1, \ldots, t$.
- e é o exponente em um intervalo $[a, b] \subset \mathbb{Z}$.

O número 0 é representado por $0.0 \ldots 0 \cdot \beta^a$.

Seja $F(\beta, t, [a, b]) \subset \mathbb{R}$ o conjunto dos números que podem ser representados exatamente.

Representação no Sistema de Ponto Flutuante

Observação Referente a Representação na Maquina

A representação interna $+0.a_1a_2a_3\ldots a_t \cdot \beta^e$ corresponde a $(0.a_1a_2a_3\ldots a_t)_\beta \cdot \beta^e = (a_1\beta^{-1} + a_2\beta^{-2}\ldots a_t\beta^{-t}) \cdot \beta^e$.

Perguntas

1. Qual é o menor número positivo $m \in F(\beta, t, [a, b])$?
2. Qual é o maior número $M \in F(\beta, t, [a, b])$?

Respostas

1. $m = (0.1)_\beta \cdot \beta^a = \beta^{-1}\beta^a = \beta^{a-1}$.
2. $M = (0.(\beta-1)(\beta-1)\ldots(\beta-1))_\beta \cdot \beta^b = (1-\beta^{-t}) \cdot \beta^b = \beta^b - \beta^{b-t}$

Exemplo 2

Considere o sistema $F(10, 3, [-2, 2])$. Represente nesse sistema, se possível, os números:

$$x_1 = 0.35, \quad x_2 = -5.17, \quad x_3 = 0.0123, \tag{1}$$

$$x_4 = 5390, \quad x_5 = 0.0003. \tag{2}$$

Exemplo 2

Considere o sistema $F(10, 3, [-2, 2])$. Represente nesse sistema, se possível, os números:

$$x_1 = 0.35, \quad x_2 = -5.17, \quad x_3 = 0.0123, \tag{1}$$

$$x_4 = 5390, \quad x_5 = 0.0003. \tag{2}$$

Resposta:

$$x_1 = 0.350 \cdot 10^0, \quad x_2 = -0.517 \cdot 10^1, \quad x_3 = 0.123 \cdot 10^{-1}.$$

O número $5390 = 0.539 \cdot 10^4$ não pode ser representado porque seu expoente é maior que 2. Tem-se *overflow*.
O número $0.0003 = 0.300 \cdot 10^{-3}$ não pode ser representado porque seu exponente é menor que -2. Tem-se um *underflow*.

A maioria dos computadores e das calculadoras científicas trabalham respectivamente com a base $\beta = 2$ e a base $\beta = 10$.

Muitos *softwares* científicos usam o formato IEEE **binary64** (antes conhecido como **precisão dupla**) com 64 bits: 1 para o sinal, 11 para o expoente (no formato “biased"), 53 (dos quais 52 são efetivamente armazenados) para a mantissa.

O padrão IEEE precisão dupla é capaz de representar números positivos entre $1.7977 \cdot 10^{308}$ e $2.2251 \cdot 10^{-308}$, aproximadamente. Ao comprometer a precisão, a representação subnormal permite valores positivos ainda menores.

Arredondamento em Ponto Flutuante

O *arredondamento* em ponto flutuante é usado para representar um número real x, dentro dos limites de representação do sistema, que não pertence a $F(\beta, t, [a, b])$.

Especificamente, arredondar um número x em ponto flutuante consiste em encontrar $\bar{x} \in F(\beta, t, [a, b])$ tal que $|x - \bar{x}|$ seja o menor possível. Se $\exists\ \hat{x}, \check{x}$ com esta propriedade, arredonda-se para cima se $x \geqslant 0$. Caso contrário, arredonda-se para baixo.

Denotaremos por $\texttt{fl}$ a função que associa $x \in \mathbb{R}$ ao seu arredondamento em ponto flutuante, ou seja, $\bar{x} = \texttt{fl}(x)$.

Quando se usa *truncamento* em ponto flutuante, $x = \pm(0.a_1 a_2 a_3 \ldots)_\beta \cdot \beta^e$ com $a_1 \neq 0$ é aproximado por $\bar{x} = \pm(0.a_1 a_2 a_3 \ldots a_t)_\beta \cdot \beta^e$.

Exemplo 3

Represente no sistema $F(10, 3, [-5, 5])$ os números

$$x_1 = 1234.56, \quad x_2 = -0.00054962, \quad x_3 = 0.9995,$$
$$x_4 = 123456.7, \quad x_5 = 0.0000001.$$

Exemplo 3

Represente no sistema $F(10, 3, [-5, 5])$ os números

$$x_1 = 1234.56, \quad x_2 = -0.00054962, \quad x_3 = 0.9995,$$
$$x_4 = 123456.7, \quad x_5 = 0.0000001.$$

Resposta:

$$\texttt{fl}(x_1) = 0.123 \cdot 10^4, \quad \texttt{fl}(x_2) = -0.550 \cdot 10^{-3},$$
$$\texttt{fl}(x_3) = 0.100 \cdot 10^1.$$

Para x_4 e x_5 tem-se *overflow* e *underflow*, respectivamente.

Para arredondar um número na base $\beta = 10$, devemos apenas observar o primeiro dígito a ser descartado. Se ele for menor que 5, deixamos os dígitos inalterados; Se ele é maior ou igual a 5, devemos somar 1 ao último dígito remanescente.

Épsilon da Máquina

Definição 4 (Épsilon da Máquina)

O épsilon da máquina, denotado por ε_{mach}, é a *distância entre 1 e o menor número no sistema de ponto flutuante estritamente maior que 1*. (Numa definição alternativa, não utilizado nesta aula, o épsilon da máquina é a metade disso.)

Em outras palavras, ε_{mach} é o menor x tal que

$$1 + x = \texttt{fl}(1 + x) > 1.$$

Pode-se mostrar que $\varepsilon_{mach} = \beta^{1-t}$ no sistema $F(\beta, t, [a, b])$.

No padrão IEEE precisão dupla, a precisão é

$$\varepsilon_{mach} = 2^{-52} \approx 2.22 \cdot 10^{-16}.$$

Épsilon da Máquina e Erros Relativos

Os valores ε_{mach} e $\frac{\varepsilon_{mach}}{2}$ fornecem limitantes superiores para o erro relativo do truncamento e a metade do erro relativo do arredondamento, respectivamente, de ponto flutuante.

Especificamente, para qualquer x dentro dos limites de representação do sistema, existe $\bar{x} \in F(\beta, t, [a, b])$ tal que

- $$\frac{|x - \bar{x}|}{|\bar{x}|} < \varepsilon_{mach} = \beta^{1-t}$$

  no caso de *truncamento*.

- $$\frac{|x - \bar{x}|}{|\bar{x}|} \leqslant \frac{\varepsilon_{mach}}{2} = \frac{1}{2}\beta^{1-t}$$

  no caso de *arredondamento*.

Exemplo 5 (Identidade Não-Nula)

Introduzimos ao `GNU Octave` os seguintes comandos:

```
» b = 1; while 1+b > 1; b = b/2; end
```

Estaríamos num *loop* infinito se fizéssemos as mesmas operações com números reais. Num computador, porém, encontramos

$$b = 1.1102 \cdot 10^{-16} \approx 2^{-53} = \frac{\varepsilon_{mach}}{2}.$$

Note que, no padrão *IEEE binary 64*, $1 + \frac{\varepsilon_{mach}}{2}$ é arredondado para baixo, o que não corresponde às regras usuais:
Round to the nearest value, if number falls exactly midway, it is rounded to the nearest value with an even (e.g. zero for binary FP) least significant bit.

Outros Detalhes do Padrão da IEEE

IEEE Standard for FP arithmetic (IEEE 754-1985, newer 2008)
Five rounding rules (1-3 directed rounding, 4-5 round to nearest)

1. Round towards 0 (truncamento);
2. Round towards +infinity (arredondamento para cima);
3. Round towards -infinity (arredondamento para baixo);
4. Round to nearest, ties away from 0 (typical for decimal FP);
5. Round to nearest, ties to even (default for binary FP, recommended for decimal FP): Round to the nearest value, if number falls exactly midway, it is rounded to the nearest value with an even (e.g. zero for binary FP) least significant bit.

Rule 5 prevents upwards or downwards bias in summing many #s that look like x.5, since round up and round down happen 50

Na aula de hoje, apresentamos a representação dos números nos computadores usuais e destacamos que podem surgir erros na aritmética de ponto flutuante.

Os erros de arredondamento, quando repetidos em algoritmos longos e complexos, podem ter efeitos cadastróficos.
Exemplos incluem:

- Fracasso do mìssil Patriot durante a Guerra do Golfo;
- Explosão do míssil Arianne em Junho de 1996 devido à *overflow* no computador de bordo.

Para chegar a um melhor entendimento destes assuntos, estudaremos conceitos de erros e o funcionamento da aritmética de ponto flutuante na próxima aula.

Muito grato pela atenção!