CCI-22

Matemática Computacional

Erros

Erros de arredondamento, representação e de cálculo

CCI-22

- Tipos de erros
- Sistemas de ponto flutuante
- Arredondamentos
- Erros absolutos e relativos
- Dígitos significativos
- Épsilon da máquina

Tipos de erros

- Erro inerente: sempre presente na incerteza das medidas experimentais e nas simplificações dos modelos utilizados
  - Exemplo: (50,3 ± 0,2) cm
- Erro de aproximação (ou de discretização): proveniente das limitações dos métodos numéricos (número de iterações, etc.) durante a determinação de um valor
  - Exemplo: para $0 \leq x \leq \pi/4$, $\operatorname{sen} x = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$
- Erro de representação (de arredondamento e de truncamento): limitação de um número real ao valor representado com uma quantidade finita de dígitos

Um caso real

- Considere as seguintes equações:
  - H = 1,0/2,0
  - X = 2,0/3,0 – H
  - Y = 3,0/5,0 – H
  - E = (X+X+X) – H
  - F = (Y+Y+Y+Y+Y) – H
  - G = E/F

Cálculo analítico:

H = 1/2

X = 1/6

Y = 1/10

E = 0

F = 0

indeterminado

- Resultados em diferentes máquinas:

HP 25	*Texas SR50*	*IBM 4341*
H = 0,5	H = 0,5	H = 0,5
X = 0,166666667	X = 0,1666666667	X = 0,166666
Y = 0,1	Y = 0,1	Y = 0,1
E = 1,0 E-10	E = 2,0 E-13	E = -0,119209 E-6
F = 0	F = 0	F = -0,178813 E-6
G = inválido	G = inválido	G = 0,666...

CCI-22

- Tipos de erros
- Sistemas de ponto flutuante
- Arredondamentos
- Erros absolutos e relativos
- Dígitos significativos
- Épsilon da máquina

Definições

- Um número real x está representado como *float* normalizado se:
  - $x = \pm 0,d_1d_2...d_n.b^e$, $n \in N$
  - $1 \leq d_1 < b$, $0 \leq d_i < b$ para $2 \leq i \leq n$
  - $e_1 \leq e \leq e_2$, com $e_1 \leq 0$, $e_2 \geq 1$, onde $e_1, e_2 \in Z$
- Um *sistema de ponto flutuante* é definido pela quádrupla $F(b,n,e_1,e_2)$, e contém todos os *floats* normalizados representados por esses valores, incluindo também o número zero
- Observação: o número zero é representado como $0,00...0.b^{e_1}$

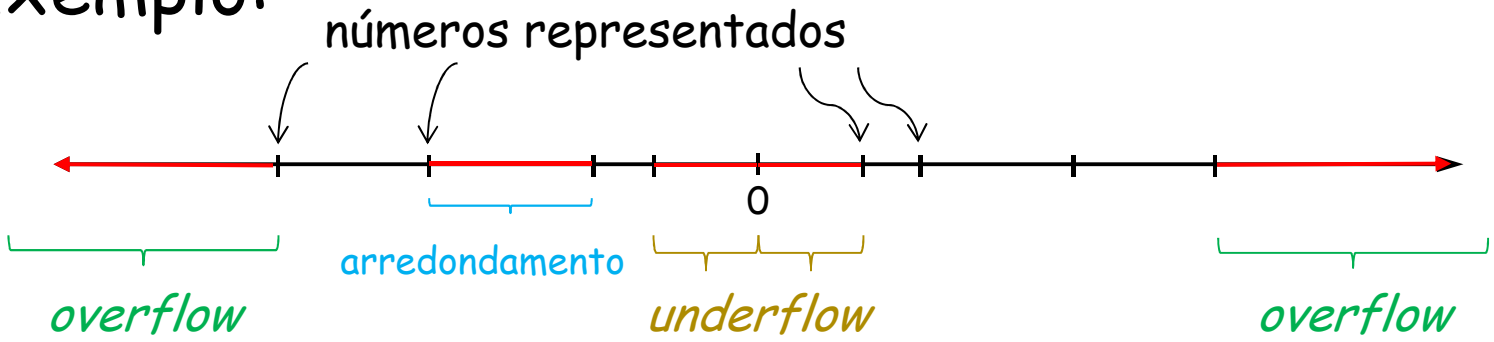
n vezes

Sistemas de ponto flutuante

- Como vimos, um sistema de ponto flutuante é uma quádrupla $F(b,n,e_1,e_2)$
- Alguns exemplos reais:
  - HP 25: F(10,9,-98,100)
  - Texas SR50: F(10,10,-98,100)
  - IBM 370: F(16,6,-64,63)
  - B6700: F(8,13,-51,77)
- No sistema $F(b,n,e_1,e_2)$:
  - Menor número não nulo representável (em módulo):
    - $N_{min} = 0,100...0.b^{e_1}$
  - Maior número representável:
    - $N_{max} = 0,[b-1][b-1]...[b-1].b^{e_2}$
  - Total de representações:
    - $\#F = 2.(b-1).b^{n-1}.(e_2-e_1+1)+1$

Erros na representação de *floats*

- A representação de um número real com uma quantidade finita de *bits* geralmente acarreta a necessidade do *arredondamento*, que é um erro em relação ao seu valor exato
- Casos especiais:
  - *Overflow* : número a ser representado tem módulo maior que N_{max}
  - *Underflow* : número a ser representado tem módulo menor que N_{min}
- Exemplo:



Exemplo

- Seja F(2,3,-1,2)
- Possíveis mantissas: 0,100; 0,101; 0,110; 0,111
- Expoentes: -1, 0, 1, 2
- Valores positivos:

e	b^e	0,100	0,101	0,110	0,111
-1	1/2	1/4	5/16	3/8	7/16
0	1	1/2	5/8	3/4	7/8
1	2	1	5/4	3/2	7/4
2	4	2	5/2	3	7/2

- #F = $2.(2-1).2^{3-1}.(2-(-1)+1)+1 = 2.1.4.4+1 = 33$
- Região de *underflow*: (-1/4, 0) U (0, 1/4)
- Região de *overflow*: (-∞, -7/2) U (7/2, ∞)
- Sejam x = 5/4 e y = 3/8. Portanto, x+y = 13/8 não está em F: terá que ser arredondado...

Outro exemplo

- Seja F(3,2,-1,2)
- Possíveis mantissas: 0,10; 0,11; 0,12; 0,20; 0,21; 0,22
- Expoentes: -1, 0, 1, 2
- Valores positivos:

e	b^e	0,10	0,11	0,12	0,20	0,21	0,22
-1	1/3	1/9	4/27	5/27	2/9	7/27	8/27
0	1	1/3	4/9	5/9	2/3	7/9	8/9
1	3	1	4/3	5/3	2	7/3	8/3
2	9	3	4	5	6	7	8

- $\#F = 2.(3-1).3^{2-1}.(2-(-1)+1)+1 = 2.2.3.4+1 = 49$
- Região de *underflow*: (-1/9, 0) U (0, 1/9)
- Região de *overflow*: (-∞, -8) U (8, ∞)
- Exercício: neste sistema, encontre dois valores cuja soma terá que ser arredondada

Padrão IEEE

- Em 1985, foi definido o padrão IEEE754 para representação de *floats*
- Possui 3 níveis de precisão (simples, dupla e estendida), que utilizam respectivamente 32, 64 e 80 *bits*
- É um sistema normalizado, mas o *bit* não nulo não é armazenado
- Os *bits* do expoente contêm também informação sobre seu sinal
- Precisão simples (8 *bits* no expoente e 23 *bits* na mantissa):
  - ao invés do expoente e, armazena-se o resultado de e + $(01111111)_2$
  - basta subtrair $(01111111)_2 = (127)_{10}$ para se obter o valor de e (evita-se um teste)
  - os expoentes $(00000000)_2$ e $(11111111)_2$ são reservados
  - expoente mínimo e_1: $(00000001)_2 - (01111111)_2 = (1)_{10} - (127)_{10} = (-126)_{10}$
  - expoente máximo e_2: $(11111110)_2 - (01111111)_2 = (254)_{10} - (127)_{10} = (127)_{10}$
  - valor infinito (Inf): mantissa nula e expoente $(11111111)_2$
  - valor inválido (NaN): mantissa não nula e expoente $(11111111)_2$
  - número zero: mantissa e expoente nulos
  - expoente nulo e mantissa não nula: números menores que N_{min} em formato não normalizado
  - N_{min}: $(1{,}000...00)_2.2^{-126} = 2^{-126} \approx 1{,}18.10^{-38}$
  - N_{max}: $(1{,}111...11)_2.2^{127} = (2 - 2^{-23}).\ 2^{127} \approx 3{,}40.10^{38}$

bit escondido

Padrão IEEE 754

		Precisão		
		Simples	Dupla	Estendida
Comprimento total		32	64	80
Bits na mantissa	n	23	52	64
Bits no expoente		8	11	15
Base	b	2	2	2
Expoente mínimo	e_1	-126	-1022	-16382
Expoente máximo	e_2	127	1023	16383
Menor número (em módulo)	N_{min}	2^{-126} $\approx 1{,}18.10^{-38}$	2^{-1022} $\approx 2{,}23.10^{-308}$	2^{-16382} $\approx 3{,}36.10^{-4932}$
Maior número	N_{max}	$(2-2^{-23}).2^{127}$ $\approx 3{,}40.10^{38}$	$(2-2^{-52}).2^{1023}$ $\approx 1{,}80.10^{308}$	$(2-2^{-64}).2^{16383}$ $\approx 1{,}19.10^{4932}$
Dígitos decimais		7	16	19

CCI-22

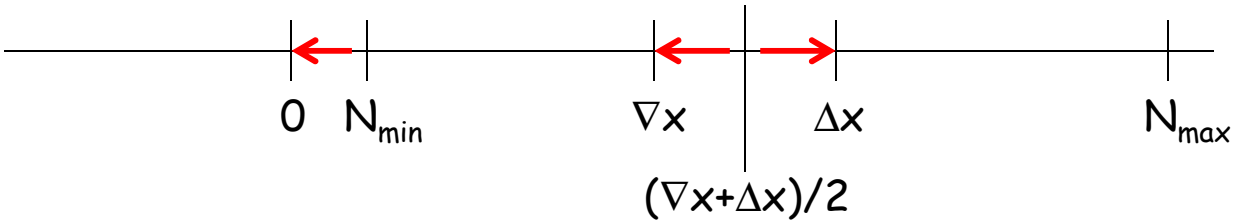
- Tipos de erros
- Sistemas de ponto flutuante
- Arredondamentos
- Erros absolutos e relativos
- Dígitos significativos
- Épsilon da máquina

Definição de arredondamento

- Seja um sistema de ponto flutuante $F(b,n,e_1,e_2)$. Uma função $\xi: R \rightarrow F$ é de <u>arredondamento</u> se:
  - $\forall x \in F, \xi x = x$ (não modifica os valores de F)
  - Além disso, essa função será:
    - De <u>arredondamento para baixo</u> (ou por falta, ou truncamento) se, $\forall x \in R, \xi x \leq x$ (nesse caso, $\xi x = \nabla x$)
    - De <u>arredondamento para cima</u> (ou por excesso, ou simplesmente arredondamento) se, $\forall x \in R, \xi x \geq x$ (nesse caso, $\xi x = \Delta x$)
    - <u>Monotônica</u> se, $\forall x,y \in R, x \leq y \Rightarrow \xi x \leq \xi y$

Para o *float* mais próximo

- Se o sistema de ponto flutuante $F(b,n,e_1,e_2)$ tem a base b par, o arredondamento para o *float* mais próximo, denotado por Ox, é definido como:
  - $\forall x \in [0, N_{min})$, $Ox = 0$
  - $\forall x \in [N_{min}, N_{max}]$:
    - $Ox = \nabla x$, para $x \in [\nabla x, (\nabla x+\Delta x)/2)$
    - $Ox = \Delta x$, para $x \in [(\nabla x+\Delta x)/2, \Delta x]$
  - Se $x < 0$, $Ox = -O(-x)$



Exemplo

- No sistema de ponto flutuante F(10, 4, -98, 10):

$x = 0{,}333333$	$y = 0{,}348436$	$z = 0{,}666666\ldots$	$w = 0{,}12345$
$\nabla x = 0{,}3333$	$\nabla y = 0{,}3484$	$\nabla z = 0{,}6666$	$\nabla w = 0{,}1234$
$\Delta x = 0{,}3334$	$\Delta y = 0{,}3485$	$\Delta z = 0{,}6667$	$\Delta w = 0{,}1235$
$Ox = 0{,}3333$	$Oy = 0{,}3484$	$Oz = 0{,}6667$	$Ow = 0{,}1235$

CCI-22

- Tipos de erros
- Sistemas de ponto flutuante
- Arredondamentos
- Erros absolutos e relativos
- Dígitos significativos
- Épsilon da máquina

Erros absolutos e relativos

- A diferença entre o valor exato e o arredondado pode ser medida por dois parâmetros: erro absoluto e erro relativo
- Erro absoluto: $E_A x = |\xi x - x|$
- Erro relativo: $E_R x = E_A x / |\xi x|$
- Pode ser demonstrado que, em um sistema de ponto flutuante $F(b,n,e_1,e_2)$, $\forall x \in R$, se $|x| \in [N_{min}, N_{max}]$, então $E_R x \leq \mu$, onde:

$$\mu = \begin{cases} b^{1-n}/2, \text{ se } \xi x = \bigcirc x \\ b^{1-n}, \text{ se } \xi x = \nabla x \end{cases}$$

- Portanto, no padrão IEEE, o erro relativo de arredondamento para o *float* mais próximo nos casos de precisão simples, dupla e estendida, é 2^{-23}, 2^{-52} e 2^{-64}, respectivamente, ou seja, é muito pequeno...

CCI-22

- Tipos de erros
- Sistemas de ponto flutuante
- Arredondamentos
- Erros absolutos e relativos
- Dígitos significativos
- Épsilon da máquina

Dígitos significativos

- Em um sistema de numeração, um dígito é <u>significativo</u> se:
  - for diferente de zero
  - caso seja zero, se não for usado para fixar a vírgula ou preencher o lugar de dígitos descartados
- Exemplos de dígitos significativos (no sistema decimal):
  - 0,008735: 8, 7, 3 e 5
  - 30457: todos
  - $2,3.10^4 = 23000$: 2 e 3
  - $2,30.10^4 = 23000$: 2, 3 e o primeiro 0

Precisão e exatidão

- Como já foi definido, a precisão de um computador é o número de dígitos da sua mantissa
- A precisão é algo bem determinado e invariável em cada máquina
- Por outro lado, exatidão é uma medida da perfeição do resultado, e depende de três fatores:
  - precisão da máquina
  - método utilizado na sua obtenção
  - confiabilidade dos dados de entrada

Exemplos

- Em um computador com sistema F(16,6,-64,63), qual seria a sua precisão decimal?
  - Sabemos que sua precisão hexadecimal é 6
  - $16^{-6} = 10^{-p} \Leftrightarrow p = 6.\log 16 \approx 7,2$
  - Portanto, sua precisão decimal é 7
  - No entanto, nem todo número com representação exata de 7 casas decimais terá também uma representação exata nessa máquina. Exemplo: 0,1
- Seja o número irracional $2^{1/2}$ = 1,414214562... Com relação a ele, podemos dizer que:
  - 1,4142 é mais preciso e exato que 1,41, pois tem maior número de casas decimais e aproxima melhor $2^{1/2}$
  - 1,4149 é mais preciso que 1,414, pois tem mais casas decimais, mas é menos exato

CCI-22

- Tipos de erros
- Sistemas de ponto flutuante
- Arredondamentos
- Erros absolutos e relativos
- Dígitos significativos
- Épsilon da máquina

Épsilon da máquina

- Chamamos de ε (épsilon) de uma determinada máquina o menor *float* tal que 1 + ε > 1
- O programa abaixo imprime esse valor:

```
EpsilonMaq(){
    float epsilon, eps = 1;
    do {
        epsilon = eps;
        eps /= 2;
    }
    while (eps + 1 != 1);
    printf("%E\n",epsilon);
}
```

E x abaixo de epsilon??...

- O programa retorna:
- x eh maior que zero...
- Logo, 1+x<=1 e x>0...
- Estranho??

```
1.  int main()
2.  {
3.      float x=1;
4.      while(1+x>1) {
5.          x=x/2;
6.      }
7.      if(x>0)
8.          cout<<"x eh maior que zero";
9.      else
10.         cout<<"x NAO eh maior que
    zero";
11.     return 0;
12. }
```

Erro em somas

- Erro em operações podem ser gerado por erros nas parcelas e erros no armazenamento do resultado.
- Exemplo de erro no resultado (n=4 e base 10):

Dados $x = 0.937 \times 10^4$ e $y = 0.1272 \times 10^2$, obter $x + y$.

Alinhando os pontos decimais dos valores acima, temos

$x = 0.937 \times 10^4$ e $y = 0.001272 \times 10^4$.

Então,

$$x + y = (0.937 + 0.001272) \times 10^4 = 0.938272 \times 10^4.$$

$\overline{x + y} = 0.9383 \times 10^4$ no arredondamento

$\underline{x + y} = 0.9382 \times 10^4$ no truncamento.

X abaixo de epsilon?..

- Representação do número 1 em IEEE 754
  - 0.01111111.00000000000000000000000
  - Campo expoente igual a 127 (significa 0)
  - Ou $(0,1)_{2*}\ 2^{0}$
- Representação do x em IEEE 754
  - 0.01100111.00000000000000000000000
  - Campo expoente igual 103 (significa -24)
  - Ou $(0,1)_{2*}\ 2^{-24}$
- Expoentes diferentes. Como somar??
  - Colocar no mesmo expoente..0 ou -24?

Desloque a vírgula de X

- Representação do x em IEEE 754
  - 0.01100111.00000000000000000000000
  - Campo expoente igual 103 (significa -24)
  - Ou $(0,1)_{2*}\ 2^{-24}$
- Precisamos armazenar esse número mas com expoente igual a zero. Como?
- Armazenar $(0,0...01)_{2*}\ 2^{0}$ com vinte e tres zeros antes do 1...
- Logo, a mantissa que tem 23 posições será zero.