Capítulo 1 - Erros e Aritmética Computacional

Carlos Balsa

balsa@ipb.pt

Departamento de Matemática
Escola Superior de Tecnologia e Gestão de Bragança

2º Ano - Eng. Civil e Electrotécnica

Sumário

Métodos Numéricos *versus* Métodos Analíticos

- Métodos Analíticos
  - Solução exacta (não havendo arredondamentos)
  - Solução geral (normalmente uma expressão matemática) que permite obter soluções particulares em função das variáveis independentes
  - Soluções contínua (permite obter soluções particulares para qualquer valor da variável independente)
- Métodos Numéricos
  - Solução aproximada (contêm um erro associado)
  - Soluções particular na forma de números
  - Solução discreta (apenas calculada para alguns valores da variável independente)
- Métodos numéricos são necessários porque
  - Problemas reais nem sempre têm solução analítica
  - Métodos numéricos permitem quantificar o erro na solução
  - Maior parte dos problemas envolvem aproximações

Origem das aproximações

- Antes da computação
  - Modelação
  - Medições empíricas
  - Computações anteriores
- Durante a computação
  - Truncatura ou discretização
  - Arredondamentos
- A exactidão dos resultados finais reflecte todas as aproximações
- A incerteza dos dados introduzidos (*input*) pode ser amplificada pelo problema
- Perturbações durante a computação podem ser amplificadas pelo algoritmo

Exemplo 1: aproximações

- Calcular a superfície terrestre através da formula utilizando a formula $A = 4\pi r^2$ envolve várias aproximações
  - A Terra é modelada como uma esfera, idealizando a sua forma ideal
  - O valor do raio é baseado em medidas empíricas e em computações anteriores
  - O valor de π requer a truncatura de processos infinitos
  - O valor dos *inputs* assim como das operações aritméticas são arredondadas no computador

Quantificação dos Erros

Seja x um número real e $\hat{x}$ um valor aproximado de x:

- Erro absoluto = valor aproximado - valor exacto = $x - \hat{x}$
- Erro relativo = $\frac{\text{erro absoluto}}{\text{valor exacto}} = \frac{x-\hat{x}}{x}$
- Valor aproximado = valor exacto $\times$ (1+ erro rel.)
- É comum trabalhar com o valor absoluto dos erros:

$$\Delta_x = |x - \hat{x}| \quad \text{e} \quad r_x = \frac{|x - \hat{x}|}{|x|}$$

Exemplo 2: Quantificação dos Erros

Seja $x = 1/3$ e $\hat{x} = 0.333$

- $\Delta_x = |x - \hat{x}| = |1/3 - 0.333| = 0.0003(3) \leq 0.00034$
- $r_x = \frac{|x-\hat{x}|}{|x|} = \frac{|1/3-0.333|}{|1/3|} = 0.0009(9) \leq 0.0010 = 0.1\%$

Erros de dados e erros de computação

- Problema típico: calcular o valor da função $f : \mathbb{R} \rightarrow \mathbb{R}$ para os argumentos
  - x valor exacto do argumento
  - $f(x)$ valor pretendido
  - $\hat{x}$ valor aproximado do input
  - $\hat{f}$ valor aproximado da função a calcular
- Erro total = $\hat{f}(\hat{x}) - f(x)$

  $= (\hat{f}(\hat{x}) - f(\hat{x})) + (f(\hat{x}) - f(x))$

  = erro computacional + erro propagado
- Erro computacional depende do algoritmo e o erro propagado depende do condicionamento do problema

Erro de truncatura e erro de arredondamento

- Os erros computacionais são a soma dos erros de truncatura e dos erros de arredondamento, normalmente, um destes é dominante
- Erro de truncatura: diferença entre o resultado exacto (para o input actual) e o resultado produzido pelo algoritmo usando uma aritmética exacta
  - → Devido a aproximações tais como a truncatura de séries infinitas ou fins de processos iterativos antes de se verificar a convergência
- Erro de arredondamento: diferença entre o resultado produzido pelo algoritmo usando aritmética infinita e o resultado produzido pelo mesmo algoritmo usando uma aritmética de precisão limitada
  - → Devido a representação inexacta de números reais e às operações inexactas sobre esses números

Erros de Truncatura

Um exemplo de erro de truncatura é o desenvolvimento de uma função através da série de Taylor truncada

Série de Taylor:

Uma função $f(x)$, com x próximo de a, em que $f(a)$ é conhecido e f admite infinitas derivadas pode ser calculada através de

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \cdots$$

Se considerarmos apenas os n primeiros termos

$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Exemplo 3: erro de truncatura

Aproximar $f(x) = cos(x)$ utilizando os três primeiros termos da série de Taylor em torno de $a = 0$

$$\begin{aligned}\cos(x) &\approx \cos(a) + \frac{(\cos(a))'}{1!}(x-a) + \frac{(\cos(a))''}{2!}(x-a)^2 \\ &\approx \cos(0) + \frac{-\text{sen}(0)'}{1}(x) + \frac{\cos(0)}{2}(x)^2 \\ &\approx 1 - x^2/2\end{aligned}$$

- Se $x = 0.1$ temos $\cos(0.1) \approx 1 - 0.1^2/2 = 0.995$
- Erro absoluto: $\Delta_f = |\cos(0.1) - 0.995| \approx 0.0000042$
- Erro relativo: $r_f = \frac{|\cos(0.1) - 0.995|}{|\cos(0.1)|} \approx 0.0000042 = 0.00042\%$

Erros de Arredondamento

Arredondamento de $x \in \mathbb{R}$ pode ser feito de três maneiras diferentes:

- Arredondamento simétrico:
  - Eliminam-se todos os algarismos (dígitos) situados à direita do último número que queremos manter.
  - Se o primeiro dos algarismo eliminados for maior ou igual a 5 adiciona-se 1 ao último algarismo da parte não eliminada.
  - Se o primeiro dos algarismo eliminados for inferior a 5 mantêm-se a parte não eliminada sem alterações.
- Arredondamento por excesso: adiciona-se sempre 1 ao último dígito da parte mantida
- Arredondamento por defeito: último algarismo da parte mantida não é alterado

Nota: Arredondamento simétrico é normalmente utilizado porque minimiza o erro cometido. Arredondamento por excesso utilizado no arredondamento dos erros de forma a obter um majorante.

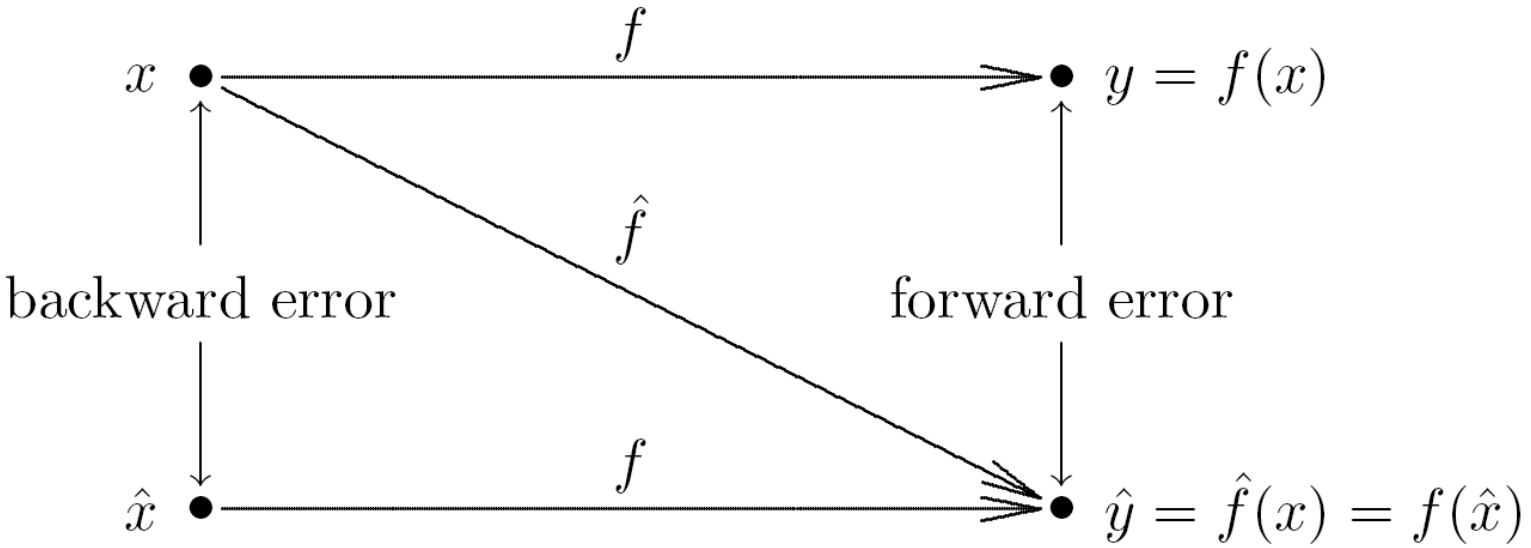
Exemplo 4: erro de arredondamento

Arredondar $x = 0.0012870012$ a quatro posições decimais ($m = 4$) pelos três métodos anteriores e calcular os respectivos erros

- Arredondamento simétrico: $\hat{x} = 0.0013$,
  $\Delta_x = |\hat{x} - x| = 0.0000129988$, $r_x = \frac{|\hat{x}-x|}{|x|} = 0.010100068 \approx 1\%$
- Arredondamento por excesso: $\hat{x} = 0.0013$,
  $\Delta_x = |\hat{x} - x| = 0.0000129988$, $r_x = \frac{|\hat{x}-x|}{|x|} = 0.010100068 \approx 1\%$
- Arredondamento por defeito: $\hat{x} = 0.0012$,
  $\Delta_x = |\hat{x} - x| = 0.0000870012$, $r_x = \frac{|\hat{x}-x|}{|x|} = 0.067599936 \approx 7\%$

Erro anterior (backward error) e erro posterior (forward error)

- Supondo que queremos calcular $y = f(x)$, com $f : \mathbb{R} \to \mathbb{R}$, mas obtemos o valor aproximado $\hat{y}$
  - Erro posterior (final) $\Delta y = |\hat{y} - y|$, com $\hat{y} = \hat{f}(x)$
  - Erro anterior (inicial) $\Delta x = |\hat{x} - x|$



Exemplo 5: erro anterior e erro posterior

- Como aproximação a $y = \sqrt{2}$, $\hat{y} = 1.4$ tem como erro absoluto posterior

$$\Delta_y = |\hat{y} - y| = |1.4 - 1.41421| \approx 0.0142$$

que corresponde a um erro relativo posterior de cerca de 1%

- Uma vez que $\sqrt{1.96} = 1.4$, o erro absoluto anterior é

$$\Delta_x = |\hat{x} - x| = |1.96 - 2| \approx 0.04$$

que corresponde a um erro relativo anterior de cerca de 2%

Análise do erro anterior

- Ideia: solução aproximada é a solução exacta do problema modificado
- De quanto deve ser modificado o problema original para originar o resultado obtido?
- Quanto é que o os erros nos inputs podem explicar todos os erros nos resultados calculados?
- A solução aproximada é boa se for a solução exacta de um problema próximo do original
- O erro anterior é por vezes mais fácil de estimar do que o erro à posterior

Exemplo 6: análise do erro anterior

- Vamos aproximar a função cosseno $f(x) = \cos(x)$ através da série de Taylor truncada a partir dos 3 primeiros termos

$$\hat{y} = \hat{f}(x) = 1 - x^2/2$$

- O erro posterior é dado por

$$\Delta y = |\hat{y} - y| = \left|\hat{f} - f\right| = \left|1 - x^2/2 - \cos(x)\right|$$

- Para determinar o erro anterior necessitamos do valor $\hat{x}$ tal que $f(\hat{x}) = \hat{f}(x)$
- Para a função cosseno, $\hat{x} = arccos(\hat{f}(x)) = arccos(\hat{y})$
- Tal como no exemplo 3, se $x = 0.1$ temos um erro posterior de $\Delta_y = 0.0000042$ e o erro anterior é $\Delta_x = |0.1 - arccos(0.995)| \approx 0.000042$

Sensibilidade e Condicionamento

- Um problema é *insensível* ou *bem condicionado* se mudanças relativas no input provocam mudanças relativas semelhantes na solução
- Um problema é *sensível* ou *mal condicionado* se mudanças relativas no input provocam muito maiores mudanças relativas na solução
- Número de condição

$$\text{Cond} = \frac{|\text{Mud. relativa na sol.}|}{|\text{Mud. relativa nos inputs}|}$$

$$= \frac{|[f(\hat{x}) - f(x)]/f(x)|}{|(\hat{x} - x)/x|} = \left|\frac{\Delta y/y}{\Delta x/x}\right|$$

- O problema é sensível ou mal condicionado se $\text{Cond} \gg 1$

Número de Condição

- O número de condição é um factor de ampliação do erro anterior em relação ao erro posterior

$$|\text{Erro relativo posterior}| = \text{cond} \times |\text{Erro relativo anterior}|$$

- Normalmente o numero de condição não é exactamente conhecido e pode variar com o input, pelo que se usa uma aproximação ou um limite máximo para o valor de **Cond**

$$|\text{Erro relativo posterior}| \leq \text{cond} \times |\text{Erro relativo anterior}|$$

Exemplo 7: condicionamento de uma função

- Calcular uma função para o input aproximado $\hat{x} = x + \Delta x$ em vez de x
- Erro absoluto posterior: $f(x + \Delta x) - f(x) \approx f'(x)\Delta x$
- Erro relativo posterior: $\frac{f(x+\Delta x)-f(x)}{f(x)} \approx \frac{f'(x)\Delta x}{f(x)}$
- Número de condição: $\text{cond} \approx \left| \frac{f'(x)\Delta x/f(x)}{\Delta x/x} \right| = \left| \frac{xf'(x)}{f(x)} \right|$
- Condicionamento de uma função depende x e de f

Exemplo 8: sensibilidade da função tangente

- A função tangente é sensível para argumentos próximos de $\pi/2$
  - $\tan(1.57079) \approx 1.58058 \times 10^5$
  - $\tan(1.57078) \approx 6.12490 \times 10^4$
- Mudança relativa no output é um quarto de milhão maior do que a mudança relativa no input
  - Para $x = 1.57079$, $cond \approx 2.48275 \times 10^5$

Notação de Virgula Flutuante

- Nos computadores os números são representados por um sistema de números de vírgula (ou ponto) flutuante da forma

$$x = \pm f_x * b^e$$

em que

f_x: mantissa (fracção)
b: base
e: expoente

- Maior parte dos computadores modernos são concebidos de acordo o sistema de ponto flutuante do IEEE, em que a base é binária ($b = 2$)
- Os computadores convertem os inputs, na base decimal ($b = 10$), para a base binária antes de efectuar as operações pedidas, posteriormente convertem também os resultados para a base decimal antes de serem apresentados

- A forma padrão de representar um numero em computador é através da notação científica

$$x = \pm f_x * 10^e$$

em que $1 \leq f_x < 10$ (todos os dígitos de f_x são significativos)

- Na notação científica normalizada tem-se $0.1 \leq f_x < 1$

  Em análise de erros esta notação é útil pois verifica a relação $-m = e - t$, em que m é o numero de posições décimas, t é o número de dígitos significativos e e é o expoente na base 10

- Por exemplo $x = 0.0003450$

  $x = 3.450 * 10^{-4}$ ou $x = 3.450e-4$: notação científica
  $x = 0.3450 * 10^{-3}$ ou $x = 0.3450e-3$: not. normalizada

Precisão Máquina

- Conjunto dos números de ponto flutuante é discreto e finito; quando um $x \in \mathbb{R}$ não tem representação exacta neste conjunto, é aproximado pelo número de ponto flutuante mais próximo $fl(x)$
- Erro relativo devido ao arredondamento produzido quando um valor $x \neq 0$ é substituído por $fl(x)$ é majorado por

$$\frac{|x - fl(x)|}{|x|} \leq \frac{1}{2}\epsilon_{maq}$$

em que ϵ_{maq}, designada por unidade de arredondamento (ou precisão máquina), é um parâmetro interno que depende do computador e do software
- Expoente de ϵ_{maq} corresponde ao número de dígitos de precisão com que um número real é representado no sistema de ponto flutuante
- No sistema IEEE de precisão simples $\epsilon_{maq} \approx 10^{-7}$ e no de precisão dupla $\epsilon_{maq} \approx 10^{-16}$ (maior parte dos computadores)

Underflow e *Overflow*

- Menor valor (em valor absoluto), diferente de zero, que é possível representar no sistema de ponto flutuante é designado por *underflow* (no Octave cerca de $2.2250e-308$)
- Maior máximo que é possível representar no sistema de ponto flutuante é designado por *overflow* (no Octave cerca de $1.7977e+308$)
- No decorrer da execução de um algoritmo se o *overflow* ocorre verifica-se um erro fatal responsável pelo fim precipitado da execução
- Não confundir *underflow* com ϵ_{maq}, embora ambos sejam pequenos, a precisão máquina depende do número de dígitos na mantissa (f_x) enquanto que o *underflow* é determinado pelo número de dígitos no campo do expoente (e)
- Num sistema de ponto flutuante temos

$$0 < \mathit{underflow} < \epsilon_{maq} < \mathit{overflow}$$

Bibliografia

1. Michael T. Heath, "Scientific Computing an Introductory Survey". McGraw-Hill, 2005.
2. A. Quarteroni e F. Saleri, "Cálculo Científico com Matlab e Octave". Springer, 2006.
3. C. Balsa e A. Santos, "Texto de Apoio à Disciplina de Análise Numérica". ESTiG-IPB, 2006.