Capítulo 6 - Equações Não-Lineares

Carlos Balsa

balsa@ipb.pt

Departamento de Matemática
Escola Superior de Tecnologia e Gestão de Bragança

2º Ano - Eng. Civil e Electrotécnica

INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e de Gestão

Sumário

Equações Não-Lineares

- Dada uma função f, procuramos x, tal que

$$f(x) = 0$$

- Solução x é raiz da equação, ou zero da função f
- Pelo que o problema é conhecido como encontrar a raiz da equação ou encontrar o zero da função

Equações Não-Lineares

Dois casos importantes

- Equação não-linear única sobre uma única incógnita, em que

  $$f : \mathbb{R} \rightarrow \mathbb{R}$$

  Solução é o escalar x para o qual $f(x) = 0$

- Sistema de n equações simultâneas em n incógnitas, em que

  $$F : \mathbb{R}^n \rightarrow \mathbb{R}^n$$

  Solução é o vector x para o qual todas as componentes de F são nulas simultâneamente, $F(x) = 0$

Existência e Unicidade da Solução

- Existência e unicidade da solução são mais difíceis de averiguar para equações não-lineares em comparação com as equações lineares
- Se f é contínua e sinal $(f(a)) \neq$ sinal$(f(b))$, então o Teorema do Valor Médio implica que exista $x^* \in [a,\ b]$ tal que $f(x^*) = 0$
- Não existe um resultado análogo tão simples para o caso de n dimensões (sistema de n equações não-lineares)

Exemplos 1: Uma Dimensão

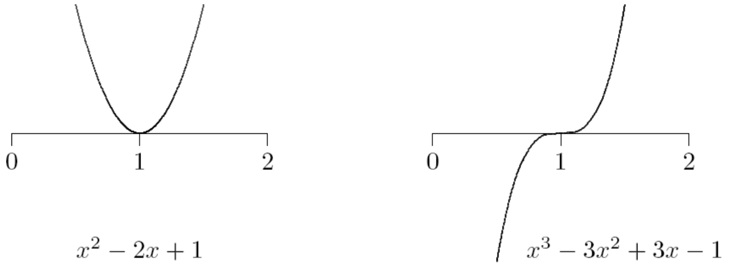
Equações não-lineares podem ter um numero variado de soluções

- $\exp(x) + 1 = 0$ não tem solução
- $\exp(-x) - x = 0$ tem uma solução
- $x^2 - 4\sin(x) = 0$ tem duas solução
- $x^3 - 6x^2 + 11x - 6 = 0$ tem três solução
- $\sin(x) = 0$ tem infinitas solução

Multiplicidade

- Se $f(x^*) = 0$ e $f'(x^*) \neq 0$, então x^* é uma raiz simples
- Se $f(x^*) = f'(x^*) = f''(x^*) = \ldots = f^{(m-1)}(x^*) = 0$ mas $f^{(m)}(x^*) \neq 0$, então a raiz x^* tem multiplicidade m

Exemplos 2: Multiplicidade de uma Raiz



- $x^2 - 2x + 1$ tem raiz $x = 1$ de multiplicidade 2
- $x^3 - 3x^2 + 3x - 1$ tem raiz $x = 1$ de multiplicidade 3

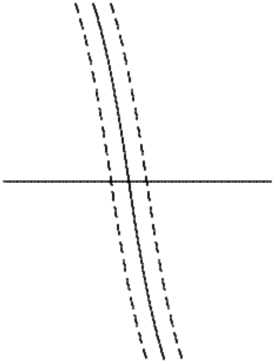
Sensibilidade e Condicionamento

- Numero de condição do problema de cálculo da raízes x^* de $f : \mathbb{R} \rightarrow \mathbb{R}$ é $1/\left|f'(x^*)\right|$
- Raiz é mal condicionada se a linha tangente for aproximadamente horizontal
- Em particular, raízes múltiplas ($m > 1$) são mal condicionadas
- Numero de condição do problema de cálculo da raízes x^* de $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ é $\left\|J_F^{-1}(x^*)\right\|$, com J_F a matriz Jacobiana de j,

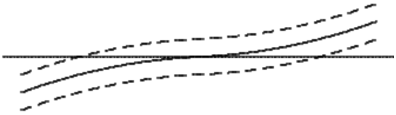
$$\{J_f(x)\}_{ij} = \partial f_i(x)/\partial x_j$$

- Raiz mal condicionada se a matriz Jacobiana for aproximadamente singular

Sensibilidade e Condicionamento



Bem Condicionado



Mal Condicionado

Sensibilidade e Condicionamento

- Que entendemos por solução aproximada de um sistema não-linear,

$$\|F(\hat{x})\| \approx 0 \qquad \text{ou} \qquad \|\hat{x} - x^*\| \approx 0?$$

- Primeira medida corresponde a um "resíduo pequeno", segunda mede a proximidade em relação à (geralmente desconhecida) solução verdadeira x^*
- Critérios de solução não são necessariamente pequenos em simultâneo
- Resíduo pequeno implica solução exacta apenas se o problema for bem condicionado

Taxa de Convergência

- Para um método iterativo genérico, define-se o erro na iteração k por
$$e_k = x_k - x^*$$
em que x_k é a solução aproximada e x^* a solução verdadeira
- Para métodos que mantêm o intervalo onde se situa a solução conhecido, em vez de se utilizar uma aproximação especifica à solução verdadeira, considera-se que o erro é igual ao comprimento do intervalo que contém a solução
- Sequência dos erros converge com uma taxa r se
$$\lim_{k\to\infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C$$
para alguma constante finita e não-nula C

Taxa de Convergência, continuação

Alguns casos particulares com interesse

- $r = 1$: linear ($C < 1$)
- $r > 1$: superlinear
- $r = 2$: quadrática

Taxa de convergência	Dígitos ganhos por iteração
linear	constante
superlinear	aumenta
quadrática	duplica

Método da Bissecção

- Método da bissecção consiste em dividir sucessivamente a meio o intervalo onde está situada a raiz até que a solução seja isolada com a correcção pretendida

ALGORITMO: MÉTODO DA BISSECÇÃO

Input: a **e** b **tal que** $x^* \in [a,\ b]$
Output: $\hat{x}$ **(solução aproximada)**
while $((b - a) > tol)$
    $m = (a + b)/2$
        se $f(a)*f(m) > 0$
            $a = m$
        else
            $b = m$
        end
    end

Exemplo 3: Método da Bissecção

- Aproxime, com uma exactidão de duas casas decimais ($tol \leq 0.5e-2$), a raiz da equação

$$f(x) = x^2 - 4\sin(x) = 0$$

sabendo que $x^* \in [1,\ 3]$, pois $f(1) = -2.365884$ e $f(3) = 8.435520$

k	a	b	m	$f(m)$	$\Delta_x = b - a$	
0	1	3	2	0.362810		
1	1	2	1.5	-1.739980		
2	1.5	2	1.75	-0.873444		
3	1.75	2	1.875	-0.300718		
4	1.875	2	1.9375	0.019849		
5	1.875	1.9375	1.906250	-0.143255	0.125	
6	1.906250	1.9375	1.929688	-0.143255	0.0625	
7	1.921875	1.9375	1.929688	-0.021454	0.0313	
7	1.929688	1.9375	1.933594	-0.000846	0.0156	
8	1.933594	1.9375	1.935547	0.009491	0.0079	
9	1.933594	1.935547			0.0040	$< tol$

Método da Bissecção, continuação

- Método da bissecção converge de certeza, mas é lento
- Em cada iteração o cumprimento do intervalo contendo a solução é reduzido a metade, taxa de convergência é linear, com $r = 1$ e $C = 0.5$
- Dado um intervalo de partida $[a, \ b]$, o cumprimento do intervalo depois de k iterações é $(b - a)/2^k$, pelo que a redução do erro abaixo de certo valor *tol* implica que

$$k \geq \log_2 \left(\frac{b - a}{tol} \right)$$

independentemente da função f envolvida

Método de Newton-Raphson

- Desenvolvimento de uma função em Série de Taylor

$$f(x+h) = f(x) + f'(x)\,h + f''(x)\,\frac{h^2}{2!} + f'''(x)\,\frac{h^3}{3!} + \ldots$$

- Truncando a série de Taylor a partir do termo de primeira ordem

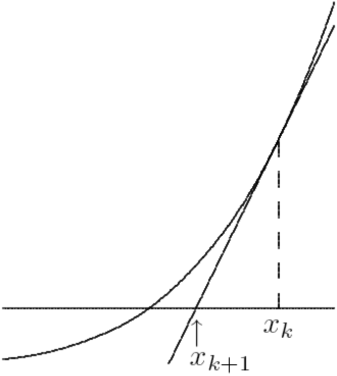
$$f(x+h) \approx f(x) + f'(x)\,h$$

  obtemos uma função linear em h que aproxima f em torno de x
- Substituindo a função não-linear pela função linear, cujo zero é $h = -f(x)/f'(x)$, obtemos uma aproximação do zero de f
- Como os zeros das duas funções não são exactamente os mesmo repete-se este processo sucessivamente, originando o método de Newton-Raphson

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \qquad k = 0, 1, 2, \ldots$$

Método de Newton-Raphson, continuação

- Método de Newton-Raphson aproxima a função não-linear f, na vizinhança de x_k, pela recta tangente em $f(x_k)$



Exemplo 4: Método de Newton-Raphson

- Aproximar com uma exactidão de duas casas decimais ($tol \leq 0.5e-2$) a raiz da equação
$$f(x) = x^2 - 4\sin(x) = 0$$
sabendo que $x^* \in [1,\ 3]$
- Derivada é
$$f'(x) = 2x - 4\cos(x)$$
pelo que o esquema iterativo é
$$x_{k+1} = x_k - \frac{x_k^2 - 4\sin(x_k)}{2x_k - 4\cos(x_k)} \quad k = 0, 1, 2, \ldots$$
Escolhendo $x_0 = 3$ como valor de partida, obtemos

k	x	$f(x)$	$f'(x)$	h	$\Delta_x = \lvert x_{k+1} - x_k \rvert$
0	3.000000	8.435520	9.959970	-0.846942	
1	2.153058	1.294772	6.505771	-0.199019	0.846942
2	1.954039	0.108438	5.403795	-0.020067	0.199019
3	1.933972	0.001152	5.288919	-0.000218	0.020067
4					$0.000218 < tol$

Convergência do Método de Newton-Raphson

- Para raízes simples ($f(x^*) = 0$ e $f'(x^*) \neq 0$) a convergência do método de Newton-Raphson é quadrática ($r = 2$)
- Mas as iterações tem de ser iniciadas suficientemente próximas da raiz para convergir
- No caso de raízes múltiplas, a convergência é apenas linear, com constante $C = 1 - (1/m)$, em que m é multiplicidade da raiz

Exemplo:

k	(Raiz simples $x = 1$) $f(x) = x^2 - 1$	(Raiz dupla $x = 1$) $f(x) = x^2 - 2x + 1$
0	2.0	2.0
1	1.25	1.5
2	1.025	1.25
3	1.0003	1.125
4	1.00000005	1.0625
5	1.0	1.03125

Outros Métodos para Equações Não-Lineares Simples

- Existe uma grande variedade de métodos iterativos alternativos, como por exemplo:
  - Método da falsa posição
  - Método do ponto fixo
  - Método da secante
- Consultar bibliografia para mais informações

Sistemas de Equações Não-Lineares

Resolução de sistemas de equações não-lineares é mais difícil do que resolver uma única equação porque

- Existe uma maior variedade de comportamento, pelo que a determinação da existência do numero de soluções ou de uma boa estimativa inicial é muito mais complicado
- Em geral, não existe uma maneira simples de garantir a convergência para a solução pretendida ou simplesmente de a localizar a solução
- Numero de cálculos a efectuar cresce rapidamente com a dimensão do problema

formulação do Problema

- Dado um sistema de equações não lineares

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0 \\ f_2(x_1, x_2, \dots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) = 0 \end{cases}$$

- Queremos determina $x = (x_1, x_2, \dots, x_n)$ de tal forma que

$$F(x) = 0 \Leftrightarrow \begin{bmatrix} f_1(x_1, x_2, \dots, x_n) \\ f_2(x_1, x_2, \dots, x_n) \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Método de Newton

- Para n dimensões, o método de Newton tem a forma

$$x_{k+1} = x_k - J_{F(x_k)}^{-1} F(x_k)$$

em que $J_{F(x_k)}$ é a matriz Jacobiana de F

$$J_{F(x)} = \begin{bmatrix} \partial f_1(x)/\partial x_1 & \partial f_1(x)/\partial x_2 & \cdots & \partial f_1(x)/\partial x_n \\ \partial f_2(x)/\partial x_1 & \partial f_2(x)/\partial x_2 & \cdots & \partial f_2(x)/\partial x_n \\ \vdots & \vdots & \ddots & \vdots \\ \partial f_n(x)/\partial x_1 & \partial f_n(x)/\partial x_2 & \cdots & \partial f_n(x)/\partial x_n \end{bmatrix}$$

- Na pratica, não se inverte explicitamente a matriz $J_F(x_k)$, em vez disso resolve-se o sistema linear

$$J_{F(x_k)}\delta_k = -F(x_k)$$

em ordem ao passo δ_k e definimos a nova iteração como

$$x_{k+1} = x_k + \delta_k$$

Exemplo 5: Método de Newton

- Aproximar a solução do sistema não-linear

$$\left\{ \begin{array}{l} x_1 + 2x_2 - 2 = 0 \\ x_1^2 + 4x_2^2 - 4 = 0 \end{array} \right. \Leftrightarrow \left[\begin{array}{c} x_1 + 2x_2 - 2 \\ x_1^2 + 4x_2^2 - 4 \end{array} \right] = \left[\begin{array}{c} 0 \\ 0 \end{array} \right] \Leftrightarrow F(x) = 0$$

  efectuando duas iterações do método de Newton

- Matriz Jacobiana é

$$J_{F(x_k)} = \left[\begin{array}{cc} \frac{\partial f_1(x_1,x_2)}{\partial x_1} & \frac{\partial f_1(x_1,x_2)}{\partial x_2} \\ \frac{\partial f_2(x_1,x_2)}{\partial x_1} & \frac{\partial f_2(x_1,x_2)}{\partial x_2} \end{array} \right] = \left[\begin{array}{cc} 1 & 2 \\ 2x_1 & 8x_2 \end{array} \right]$$

- PRIMEIRA ITERAÇÃO: Escolhendo $x_0 = [1 \;\; 2]^T$ como aproximação inicial obtemos

$$F(x_0) = \left[\begin{array}{c} 3 \\ 13 \end{array} \right], \quad J_{F(x_0)} = \left[\begin{array}{cc} 1 & 2 \\ 2 & 16 \end{array} \right]$$

Exemplo 5, continuação

- Resolução do sistema linear $J_{F(x_0)}\delta_0 = -F(x_0)$

$$\begin{bmatrix} 1 & 2 \\ 2 & 16 \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} = \begin{bmatrix} -3 \\ -13 \end{bmatrix}$$

origina $\delta_0 = \begin{bmatrix} -1.83 \\ -0.58 \end{bmatrix}$, pelo que $x_1 = x_0 + \delta_0 = \begin{bmatrix} -0.83 \\ 1.42 \end{bmatrix}$

- SEGUNDA ITERAÇÃO: Recalculando para o ponto x_1

$$F(x_1) = \begin{bmatrix} 0 \\ 4.72 \end{bmatrix}, \quad J_{F(x_1)} = \begin{bmatrix} 1 & 2 \\ -1.67 & 11.3 \end{bmatrix}$$

- Resolução do sistema linear $J_{F(x_1)}\delta_1 = -F(x_1)$
$\begin{bmatrix} 1 & 2 \\ -1.67 & 11.3 \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} = \begin{bmatrix} 0 \\ -4.72 \end{bmatrix}$ obtemos
$\delta_1 = [0.64 \quad -0.32]^T$, pelo que $x_2 = x_1 + \delta_1 = [-0.19 \ \ 1.10]^T$
(continuando a iterar iriamos aproximar-nos de $x^* = [0 \ \ 1]^T$)

Critério de paragem

- Na prática, os dois critérios de paragem mais usuais são:
  - Erro: impondo que uma certa aproximação do erro absoluto seja inferior a um valor tolerado

    $$\|x_{k+1} - x_k\| = \|\delta_k\| < tol$$

    ou então impondo o mesmo critério ao erro relativo aproximado

    $$\frac{\|x_{k+1} - x_k\|}{\|x_{k+1}\|} = \frac{\|\delta_k\|}{\|x_{k+1}\|} < tol$$

  - Resíduo: em vez de obter uma aproximação do erro, verifica-se a proximidade de zero da norma da função

    $$\|F(x_{k+1})\| < tol$$

    sabendo que este critério é um bom indicador da proximidade da solução apenas quando o problema é bem condicionado

Considerações Finais

→ *Métodos Disponíveis na NMLibforOctave:*

- Bissecção: `[x,res,k] = bisection(fun,a,b,itmax,tol)`
- Newton-Raphson: `[x,res,k]=nle_newtraph(fun,dfun,x0,itmax,tol)`
- Newton: `[x,res,k] = nle_newtsys(fun,jfun,x0,itmax,tol)`

→ *BIBLIOGRAFIA: Exposição baseada essencialmente no capítulo 5 de*

- Michael T. Heath. "Scientific Computing an Introductory Survey". McGraw-Hill, 2002, New York.

E nos capítulo 5, 6 e 7 de

- Steven C. Chapra e Raymond P. Canale. "Métodos Numéricos para Engenharia". McGraw-Hill, 2008, São Paulo.