MATLAB EM

DINÂMICA DE MÁQUINAS

O QUE É O MATLAB?

O MATLAB ("MATrix LABoratory") é um pacote de programas computacionais que pode ser usado para a resolução de uma variedade de problemas científicos e de engenharia, tais como:

diferenciação e integração numérica, equações diferenciais ordinárias e parciais, ajuste de curvas, equações não lineares e otimização;

→ também pode resolver muitos tipos de problemas simbolicamente.

O MATLAB proporciona um excelente ambiente para cálculo e programação bem como para geração de gráficos.

Pode-se executar uma única declaração ou uma lista de declarações, reunidas num arquivo de programa ("script file").

DADOS E OPERAÇÕES ELEMENTARES

O único tipo de dado no MATLAB é uma matriz de valores complexos. Assim, escalares, vetores com elementos inteiros e matrizes com valores reais são tratados como casos especiais de matrizes complexas.

Os símbolos usados nas operações aritméticas básicas de adição, subtração, multiplicação, divisão e exponenciação são, respectivamente,

+, –, *, / e ^ .

Em qualquer expressão, os cálculos são executados da esquerda para a direita, sendo que a exponenciação tem a prioridade mais alta, seguida pela multiplicação e pela divisão (com prioridades iguais) e então pela adição e pela subtração (também com prioridades iguais).

PRECISÃO E VARIÁVEIS

PRECISÃO

O MATLAB usa precisão dupla durante os cálculos, mas fornece os resultados na tela em formato mais reduzido. Essa característica pode ser alterada com a utilização do comando *format*.

VARIÁVEIS

Quando o MATLAB encontra um novo nome de variável, ele automaticamente cria a variável e aloca o espaço apropriado.

Os nomes das variáveis devem começar com uma letra, seguida por qualquer combinação de letras, dígitos e “underscores”, sendo usados os primeiros 63 caracteres. Letras maiúsculas e minúsculas são distinguidas.

CRIAÇÃO DE VETORES E MATRIZES

Antes de realizar operações aritméticas como adição, subtração e multiplicação de matrizes, essas devem ser criadas, da seguinte forma:

VETOR LINHA

> A = [1 2 3]

Um vetor linha é tratado como uma matriz (1 x n), ficando seus elementos entre colchetes e separados por espaços ou vírgulas.

Se não for acrescentado um ponto e vírgula no final da linha, o MATLAB apresenta os resultados da linha na tela, após ela ter sido lida.

CRIAÇÃO DE VETORES E MATRIZES (cont.)

VETOR COLUNA

> A = [1; 2; 3] ou > A = [1 2 3]’

Um vetor coluna é tratado como uma matriz (n x 1). Seus elementos podem ser digitados em uma única linha, usando ponto e vírgula para separá-los.

Alternativamente, pode ser usado um vetor linha, com um apóstrofo no colchete da direita, que representa a operação de transposição.

CRIAÇÃO DE VETORES E MATRIZES (cont.)

MATRIZ

Para inserir a matriz

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

a seguinte especificação pode ser usada:

> A = [1 2 3; 4 5 6; 7 8 9]

VETORES COM ESTRUTURA ESPECIAL

Em certos casos, a estrutura especial de um vetor é usada para especificá-lo de uma maneira mais simples. Por exemplo,

```
> A = 1:10
```

representa o vetor linha

A = [1 2 3 4 5 6 7 8 9 10],

enquanto

```
> A = 2:0.5:4
```

representa o vetor linha

A = [2,0 2,5 3,0 3,5 4,0].

MATRIZES ESPECIAIS

Algumas matrizes especiais também são referenciadas de maneira especial.

Por exemplo, o comando

> A = eye(3)

implica uma matriz identidade de ordem 3, qual seja

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

MATRIZES ESPECIAIS (cont.)

Já os comandos

> A = ones(3)

> B = zeros(2,3)

implicam, respectivamente, uma matriz 3x3 e uma matriz 2x3, tais que

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

OPERAÇÕES ELEMENTARES COM MATRIZES

Matrizes podem somadas, subtraídas e multiplicadas pelo uso dos sinais

+, – e * ,

respectivamente.

Por exemplo, para somar as matrizes A e B, de modo a obter a matriz C, usa-se a declaração

> C = A + B

FUNÇÕES MATLAB

O MATLAB tem um grande número de funções embutidas, tais como

sqrt(x) – raiz quadrada de x

sin(x) – seno de x

cos(x) – cosseno de x

tan(x) – tangente de x

log10(x) – logaritmo de x na base 10

ln(x) – logaritmo neperiano de x

exp(x) – função exponencial de x

FUNÇÕES MATLAB (cont.)

Para gerar um vetor y que contenha 11 valores associados com a função

$y = e^{-2x} \cos x$,

com

$x = 0,\ 0.1, 0.2, ..., 1.0$,

digita-se o seguinte:

```
> x = 0:0.1:1

> y = exp(-2*x).*cos(x)
```

O ponto antes do sinal de multiplicação permite que os valores das funções exponencial e cosseno sejam multiplicados em correspondência.

NÚMEROS COMPLEXOS

O MATLAB considera a álgebra de números complexos automaticamente. O símbolo i, ou o j, pode ser usado para representar a parte imaginária, sem necessidade de um asterisco entre o i (ou o j) e um número. Por exemplo,

```
> a = 1 - 3i
```

gera a variável a, que é igual ao número complexo com parte real igual a 1 e parte imaginária igual a –3, respectivamente.

O módulo e o argumento (em radianos) de um número complexo podem ser determinados da seguinte forma:

```
> ma = abs(a)
```

```
> aa = angle(a)
```

ARQUIVOS M

O MATLAB pode ser usado em modo interativo, com a digitação imediata de cada comando pelo teclado. Nesse modo, ele executa as operações como se fosse uma calculadora mais avançada.

Contudo, se o mesmo conjunto de comandos tiver que ser repetido várias vezes, com valores diferentes dos parâmetros de entrada, desenvolver um programa será mais rápido e eficiente.

Um programa MATLAB consiste em uma sequência de instruções de interesse, escritas em ambiente próprio e então executadas como um único bloco de comandos.

Um arquivo de programa é dito “script file” ou “m-file” (arquivo m).

ARQUIVOS M (cont.)

É necessário dar um nome ao arquivo de programa, que deverá terminar com a extensão .m (um ponto seguido da letra m). Um arquivo m típico, denominado fibo.m, é dado a seguir:

```
% m-file to compute Fibonacci numbers
f=[1 1];
i=1;
while f(i) + f(i+1) < 1000
   f(i+2)=f(i)+f(i+1);
   i=i+1;
end
```

ARQUIVOS M (cont.)

Após digitar as linhas do arquivo, ele pode ser gravado na pasta de trabalho, com o nome escolhido, e dali executado.

Ele pode também ser gravado numa pasta específica, designada pelo usuário. Nesse caso, contudo, no momento de se rodar o programa, será solicitada a alteração da pasta de trabalho.

Um arquivo m também pode ser usado para escrever funções (subrotinas), que serão utilizadas por outros programas.

Por exemplo, a solução de uma equação quadrática

$$ax^2 + bx + c = 0$$

pode ser determinada com a utilização da seguinte função:

ARQUIVOS M (cont.)

```
function [x1, x2]=raizes_quadraticas(a,b,c)
% função para encontrar as raízes de uma equação quadrática
dis=b^2-4*a*c;  % dis = discriminante
if (dis < 0.0)
  x1=(-b+i*sqrt(-dis))/(2*a);
  x2=(-b-i*sqrt(-dis))/(2*a);
  disp('As raízes são complexas conjugadas.');
elseif (abs(dis) < 1e-8)  % dis = 0.0
  x1=-b/(2*a);    x2=-b/(2*a);
  disp('As raízes são idênticas.');
```

```
else (dis > 0.0)
  x1=(-b+sqrt(dis))/(2*a);
  x2=(-b-sqrt(dis))/(2*a);
  disp('As raízes são reais e distintas.');
end
```

Para a = 2, b = 2 e c = 1, essa função, através do comando

```
> [x1,x2] = raizes_quadraticas(2,2,1)
```

fornecerá o seguinte resultado:

As raízes são complexas conjugadas.

$x1 = -0.5000 + 0.5000i$

$x2 = -0.5000 - 0.5000i$

GERAÇÃO DE GRÁFICOS

Para gerar um gráfico em MATLAB, define-se um vetor de valores da variável independente x (vetor x) e um vetor de valores da variável dependente y, correspondente aos valores de x (vetor y).

Então, o gráfico x-y pode ser gerado com o comando $\text{plot}(x, y)$.

Para gerar, por exemplo, o gráfico da função $y = x^2 + 1$, na faixa $0 \leq x \leq 3$, os seguintes comandos podem ser utilizados:

```
> x = 0:0.2:3;

> y = x.^2 + 1;

> plot(x,y)
```

GERAÇÃO DE GRÁFICOS (cont.)

Observe que as duas primeiras linhas acima geram os vetores x e y (com incrementos de 0,2 para x), enquanto a terceira linha acima gera o gráfico (utilizando linhas retas entre os pontos).

Já as cinco linhas abaixo permitem a representação gráfica dos eixos x e y, junto com a montagem da grade.

```
> hold on
> x1 = [0 3]; y1 = [0 0]; x2 = [0 0]; y2 = [0 10];
> plot(x1,y1,x2,y2,'-b')
> grid on
> hold off
```

SOLUÇÃO DE EQUAÇÕES LINEARES

Sistemas de equações lineares são resolvidos de forma expedita no MATLAB. Considere-se, por exemplo, o seguinte sistema:

$$x_1 - x_2 + x_3 = 0$$

$$-x_1 + x_2 - x_3 = 0$$

$$10x_2 + 25x_3 = 90$$

$$20x_1 + 10x_2 = 80$$

Trata-se de um sistema de 4 equações, com 3 incógnitas, cuja resposta será $x_1 = 2;\ x_2 = 4;\ x_3 = 2.$

Para a resolução desejada, recorre-se, inicialmente, à formulação matricial.

SOLUÇÃO DE EQUAÇÕES LINEARES

Em forma matricial, o problema passa a ser

$$\begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 0 & 10 & 25 \\ 20 & 10 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 90 \\ 80 \end{bmatrix} \quad \text{ou} \quad Ax = b$$

Para a solução em MATLAB, usa-se o seguinte programa:

```
% solução numérica de sistema de equações lineares
clc
clear all
A=[1 -1 1; -1 1 -1; 0 10 25; 20 10 0];
b=[0 0 90 80]';
x=A\b
```

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES

Na análise cinemática de mecanismos, surgem equações não lineares transcendentais que requerem, via de regra, solução numérica. O método de Newton-Raphson é amplamente empregado nesses casos.

Trata-se de um método iterativo que parte de estimativas iniciais para as incógnitas e melhora essas estimativas progressivamente até que a solução seja próxima o suficiente da solução correta.

Sem perda de generalidade, o problema é apresentado para o caso de 2 equações com 2 incógnitas. A expansão para outros casos pode ser feita sem maiores dificuldades.

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Expressa-se o problema da seguinte forma:

$$f_1(q_1, q_2) = 0$$

$$f_2(q_1, q_2) = 0$$

onde q_1 e q_2 são as incógnitas.

O primeiro passo é expressar as incógnitas em termos de estimativas da solução e de fatores de correção correspondentes, de forma que

$$q_i = \overline{q}_i + \Delta q_i$$

Emprega-se, então, a expansão em série de Taylor das funções acima, em torno das estimativas de interesse.

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Para a primeira função, tem-se que

$$f_1(q_1, q_2) = f_1(\bar{q}_1, \bar{q}_2) + \left.\frac{\partial f_1}{\partial q_1}\right|_{\bar{q}_1, \bar{q}_2} \Delta q_1 + \left.\frac{\partial f_1}{\partial q_2}\right|_{\bar{q}_1, \bar{q}_2} \Delta q_2 + \text{ tos}$$

Os tos (termos de ordem superior) são relevados, sem dano para o método.

Com a expansão para as duas funções, tem-se, em forma matricial, que

$$\begin{bmatrix} f_1(\bar{q}_1, \bar{q}_2) \\ f_2(\bar{q}_1, \bar{q}_2) \end{bmatrix} + \begin{bmatrix} \left.\frac{\partial f_1}{\partial q_1}\right|_{\bar{q}_1, \bar{q}_2} & \left.\frac{\partial f_1}{\partial q_2}\right|_{\bar{q}_1, \bar{q}_2} \\ \left.\frac{\partial f_2}{\partial q_1}\right|_{\bar{q}_1, \bar{q}_2} & \left.\frac{\partial f_2}{\partial q_2}\right|_{\bar{q}_1, \bar{q}_2} \end{bmatrix} \begin{bmatrix} \Delta q_1 \\ \Delta q_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

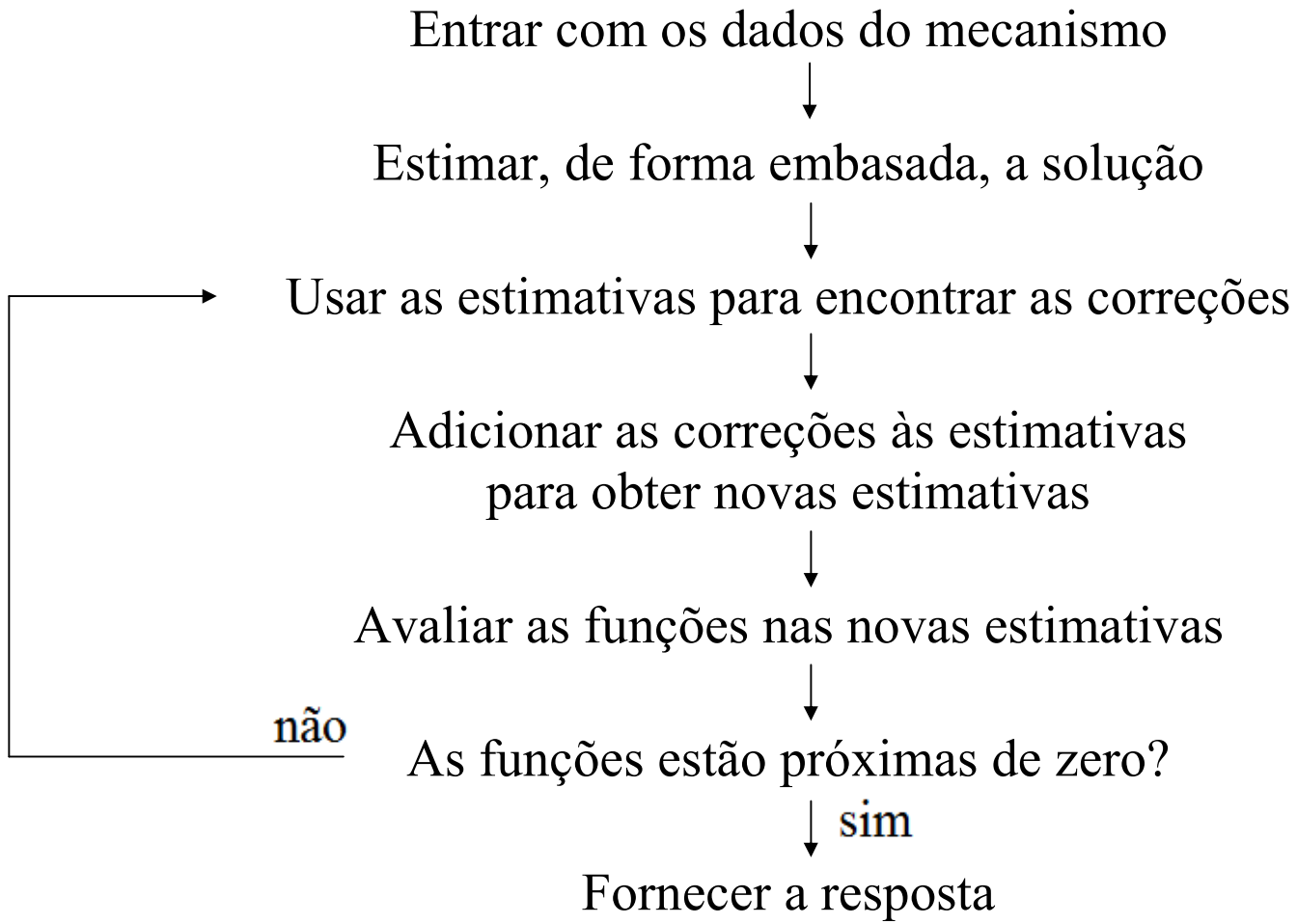
SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Resolvendo a equação matricial acima, decorre que

$$\begin{bmatrix} \Delta q_1 \\ \Delta q_2 \end{bmatrix} = \begin{bmatrix} \left.\frac{\partial f_1}{\partial q_1}\right|_{\bar{q}_1,\bar{q}_2} & \left.\frac{\partial f_1}{\partial q_2}\right|_{\bar{q}_1,\bar{q}_2} \\ \left.\frac{\partial f_2}{\partial q_1}\right|_{\bar{q}_1,\bar{q}_2} & \left.\frac{\partial f_2}{\partial q_2}\right|_{\bar{q}_1,\bar{q}_2} \end{bmatrix}^{-1} \begin{bmatrix} -f_1(\bar{q}_1,\bar{q}_2) \\ -f_2(\bar{q}_1,\bar{q}_2) \end{bmatrix}$$

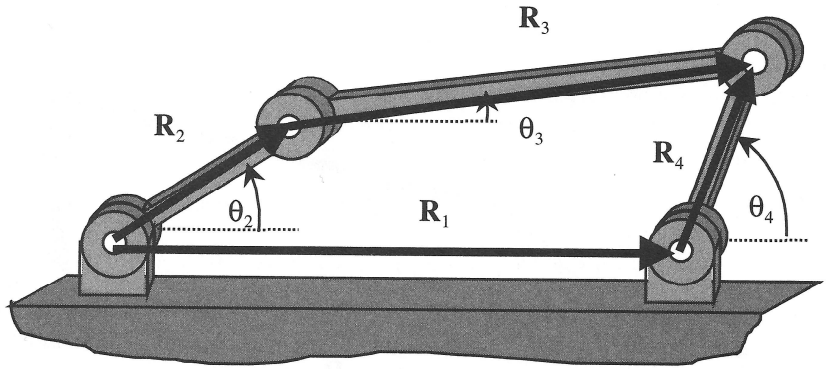
SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

O fluxograma correspondente ao método é apresentado abaixo.



SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Seja então o mecanismo de 4 barras da figura abaixo, em que se deseja, para um dado conjunto de comprimentos de elos e um dado valor de θ_2, encontrar valores de θ_3 e θ_4 para os quais as funções f_1 e f_2 sejam 0.



Para esse mecanismo, tem-se que

$$f_1(\theta_3, \theta_4) = R_2 \cos\theta_2 + R_3 \cos\theta_3 - R_1 - R_4 \cos\theta_4 = 0$$

$$f_2(\theta_3, \theta_4) = R_2 \operatorname{sen}\theta_2 + R_3 \operatorname{sen}\theta_3 - R_4 \operatorname{sen}\theta_4 = 0$$

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

O método de Newton-Raphson funciona bem na resolução numérica do sistema de equações não lineares transcendentais acima.

Nesse caso, tem-se

$$\theta_3 = \overline{\theta}_3 + \Delta\theta_3$$

$$\theta_4 = \overline{\theta}_4 + \Delta\theta_4$$

onde θ_3 e θ_4 representam a solução do problema.

Usando a expansão em série de Taylor, obtém-se

$$\begin{bmatrix} \Delta\theta_3 \\ \Delta\theta_4 \end{bmatrix} = \begin{bmatrix} -R_3 \operatorname{sen}\overline{\theta}_3 & R_4 \operatorname{sen}\overline{\theta}_4 \\ R_3 \cos\overline{\theta}_3 & -R_4 \cos\overline{\theta}_4 \end{bmatrix}^{-1} \begin{bmatrix} -f_1(\overline{\theta}_3, \overline{\theta}_4) \\ -f_2(\overline{\theta}_3, \overline{\theta}_4) \end{bmatrix}$$

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Para um mecanismo em que $R_1 = 12$, $R_2 = 4$, $R_3 = 10$ e $R_4 = 7$, e partindo de $\theta_2 = 0$, $\theta_3 = 45\pi/180$ e $\theta_4 = 135\pi/180$, o programa e a função correspondentes em MATLAB são os seguintes:

a) programa sol4bar

```
% solução numérica de mecanismo de 4 barras
clc
clear all
close all
R=[12 4 10 7];
teta=[0 0 45*pi/180 135*pi/180];
[teta(3), teta(4)]=m4bar(R,teta);
[teta(3) teta(4)]
```

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

b) função m4bar

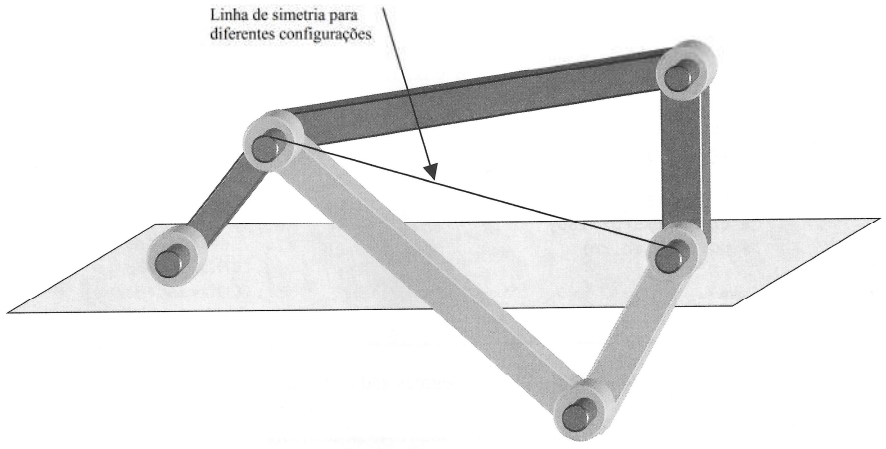
```
function [t3, t4] = m4bar(R,teta)
% posição de mecanismo de 4 barras via Newton-Raphson
t2=teta(2);
t3b=teta(3);
t4b=teta(4);
eps=1e-6; % condição de convergência
f=[R(3)*cos(t3b)-R(4)*cos(t4b)+R(2)*cos(t2)-R(1);
    R(3)*sin(t3b)-R(4)*sin(t4b)+R(2)*sin(t2)];
while norm(f) > eps
  J=[-R(3)*sin(t3b)   R(4)*sin(t4b);
       R(3)*cos(t3b)  -R(4)*cos(t4b)];
  dt=inv(J)*(-1.0*f);
  t3b=t3b+dt(1);
  t4b=t4b+dt(2);
```

```
   f=[R(3)*cos(t3b)-R(4)*cos(t4b)+R(2)*cos(t2)-R(1);
       R(3)*sin(t3b)-R(4)*sin(t4b)+R(2)*sin(t2)];
end;
t3=t3b;
t4=t4b;
end
```

A resposta associada será $\theta_3 = 0,7688$ (44,05°) e $\theta_4 = 1,6871$ (96,66°).

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Deve-se ter em mente que vários mecanismos podem ter múltiplas soluções. Por exemplo, a figura abaixo mostra duas formas distintas de se montar um mecanismo de 4 barras, com os mesmos comprimentos de elo e o mesmo ângulo de entrada.



Nota-se, nesse contexto, que a solução fornecida pelo método de Newton-Raphson depende das estimativas iniciais.

SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

Assim sendo, uma escolha apropriada das estimativas iniciais é que irá garantir a operação correta do algoritmo. A melhor forma de se obter estimativas iniciais é esboçar o mecanismo em escala de modo aproximado.

Frequentemente, requer-se que sejam obtidas soluções em várias posições, usualmente em intervalos iguais da rotação da manivela ao longo de um ciclo completo.

Para tanto, apenas as estimativas iniciais são necessárias, posto que cada solução obtida poderá servir como estimativa inicial para a seguinte.

Retomando o caso anterior e considerando intervalos de 5° no ângulo da manivela, apresentam-se o programa MATLAB a seguir.

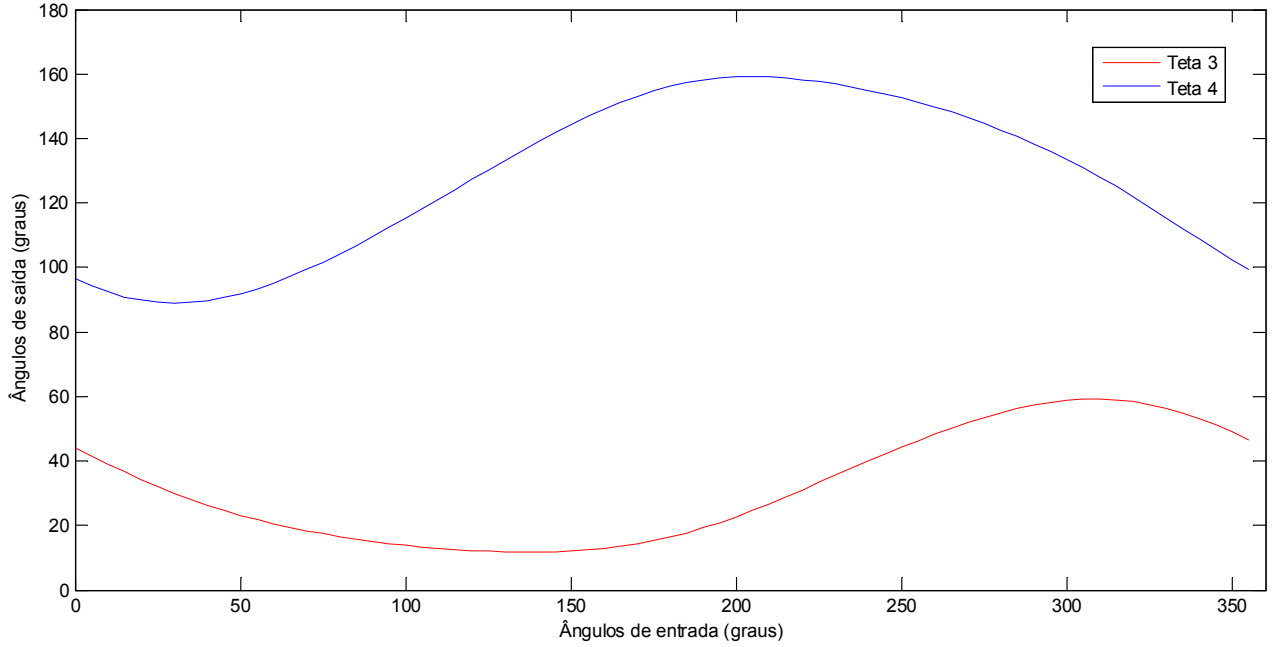
SOLUÇÃO DE EQUAÇÕES NÃO LINEARES (cont.)

programa sseq4bar

```
% solução numérica sequencial de mecanismo de 4 barras
clc
clear all
close all
R=[12 4 10 7];
teta=[0 0 45*pi/180 135*pi/180];
dt=5*pi/180;
for i=1:72
  [teta(3), teta(4)]=m4bar(R,teta);
  angs(i,:)=[teta(2) teta(3) teta(4)];
  teta(2)=teta(2)+dt;
end
angs=angs*180/pi; % conversão para graus
plot(angs(:,1),angs(:,2),'r-',angs(:,1),angs(:,3),'b-');
```

```
xlabel('Ângulos de entrada (graus)');
ylabel('Ângulos de saída (graus)');
axis([0 360 0 180])
legend('Teta 3','Teta 4')
```

Obtém-se, como resultado, o seguinte gráfico:



SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS

Para a solução de um sistema de equações diferenciais ordinárias de primeira ordem, o MATLAB tem diversas funções construídas com base nos métodos de Runge-Kutta.

A função MATLAB **ode23** implementa uma combinação de métodos de Runge-Kutta de segunda e terceira ordens, enquanto a função **ode45** tem como base uma combinação de métodos de quarta e quinta ordens.

Assim, para usar essas funções na resolução de uma equação diferencial ordinária de ordem n, a equação deve ser antes convertida para um sistema de n equações diferenciais ordinárias de primeira ordem.

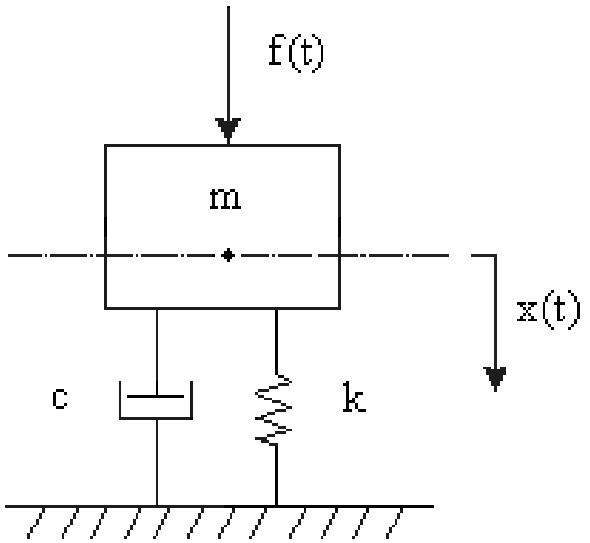
SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS (cont.)

Como exemplo, considere a resolução do problema de valor inicial composto pela equação diferencial ordinária

$\boxed{m\ddot{x} + c\dot{x} + kx = f}$ ou, na forma padrão, $\boxed{\ddot{x} + \frac{c}{m}\dot{x} + \frac{k}{m}x = \frac{1}{m}f}$,

junto com as condições iniciais $\boxed{x(0) = x_0 \text{ e } \dot{x}(0) = v_0}$.

Essa equação descreve o comportamento de um sistema mecânico com um grau de liberdade, como ilustrado na figura ao lado.



SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS (cont.)

Para resolver esse problema numericamente, transforma-se a equação num sistema de 2 equações de ordem, pela definição das variáveis de estado

$x_1(t) = x(t)$ e $x_2(t) = \dot{x}(t)$,

onde $x_1(t)$ representa o deslocamento e $x_2(t)$ a velocidade.

Com base nessas definições, tem-se que

$$\boxed{\dot{x}_1 = x_2 \quad \text{e} \quad \dot{x}_2 = -\frac{c}{m}x_2 - \frac{k}{m}x_1 + \frac{1}{m}f}$$, com $\boxed{x_1(0) = x_0 \text{ e } x_2(0) = v_0}$ (ci's).

Para um sistema com m = 100 kg, c = 200 kg/s, k = 2000 N/m, x_0 = 0,01 m, v_0 = 0,1 m/s e f(t) = 150sen(10t), o programa e a função correspondentes em MATLAB serão os seguintes:

SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS (cont.)

a) programa solumgdl

```
% solução numérica de sistema com 1 gdl sob excitação harmônica
clc
clear all
close all
ts=[0 5];
ci=[0.01  0.1];
[t,x]=ode45('seh',ts,ci);
plot(t,x(:,1))
xlabel('tempo (s) ')
ylabel('deslocamento (m)')
grid on
```

SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS (cont.)

b) função seh

```
function f=seh(t,x)
% sistema de 1 gdl sob excitação harmônica
m=100; c=200; k=2000; % sistema
f0=150; wf=10; % força
f=[x(2); -(c/m)*x(2)-(k/m)*x(1)+(1/m)*f0*sin(wf*t)];
```

A função denominada seh contém o sistema de equações de primeira ordem, correspondente à equação diferencial de interesse.

Nela também são incluídos os parâmetros do sistema e as características de amplitude e frequência da força atuante sobre o sistema.

FONTES

Vibrações Mecânicas (4ª edição), S. Rao, Pearson/Prentice-Hall, 2009;

Simulations of Machines using Matlab and Simulink, John F. Gardner, Thomson Engineering, 2001;

Engineering Vibration (3rd edition), D. J. Inman, Pearson/Prentice-Hall, 2008.